Diagnosis of faults in plant operations and scheduling under uncertain production times

Diagnosis of faults in plant operations and scheduling under uncertain production times

By
Junichi Mori, B.Eng., M.Eng.

A Thesis
Submitted to the School Of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree
Doctor of Philosophy

McMaster University

Ph.D. Candidate (2016)	McMaster University
Chemical Engineering	Hamilton, Ontario

TITLE: **Diagnosis of faults in plant operations and scheduling under uncertain production times**

AUTHOR:	Junichi Mori
	B.Eng. (Waseda University)
	M.Eng. (Waseda University)
SUPERVISOR:	Professor Vladimir Mahalec

NUMBER OF PAGES: 193

Abstract

Technical advances since 1970s in the areas of plant measurements and data archiving have enabled us to measure large number of process variables and record in the industrial plant historians. Early stages of plant data analysis via charting have been replaced by methods which enable us to extract more useful information from plant data, thereby providing valuable insight into plant operations. Monitoring of plant operations needs enables us to assess whether production targets are met according to the schedule. Both of these activities, monitoring and scheduling, are subject to uncertainties. Uncertainty in the monitoring arises from the possibility that some measurements may not be accurate or that the process equipment has developed a fault. Uncertainty in scheduling is caused by variations in the length of time required to complete individual tasks in the multi-step production system.

This Thesis deals with both of the above aspect. The first part introduces novel process monitoring and fault diagnosis methods which have been developed to extract useful information from highly correlated process variables. The second part focuses on modeling and optimizing manufacturing processes impacted by many uncertainties, which necessitates statistical techniques.

Multivariate statistical process monitoring (MSPM) techniques developed in this research extract useful information from a large number of highly correlated process variables and historical data sets. In order to capture the non-Gaussian features and relationships between input and output variables, a new quality relevant non-Gaussian latent subspace projection method is proposed by adopting the high-order statistics of mutual information for searching the latent directions within input and output spaces, respectively. Moreover, to monitor quality related operational performance of nonlinear batch processes, a novel multiway kernel based quality relevant non-Gaussian latent subspace projection method is developed. Furthermore, a new probabilistic graphical model based network process monitoring has been developed for the identification of the root-cause variable.

Modeling of uncertainties in production times and scheduling under such uncertainties are subject of the research in the second part of this Thesis. The Bayesian network models are proposed for accurate estimation of the production loads and the total production times in manufacturing processes. The proposed models are applied to schedules for steelmaking continuous casting production. Continuous casting scheduling is a difficult optimization problem due to a large number of binary variables that are needed to represent exactly process characteristics in the optimization model. To solve the scheduling problems, a new two-level algorithm and parallel simulated annealing methods are developed. Moreover, the proposed algorithm is extended to a multi-objective evolutionary algorithm in order to optimize simultaneously multiple objectives. Real life steel production process data have been used to examine the effectiveness of all the above proposed algorithms.

Acknowledgements

I wish to express my gratitude to my supervisor Dr. Vladimir Mahalec for his guidance throughout my research work. Even after returning to Japan, he supported me greatly and was always willing to help me. I have been extremely lucky to have a supervisor who cared so much about my work. I would also like to thank my former supervisor, Dr. Jie Yu for his valuable guidance to conduct my research.

Besides my supervisor I would also like to thank the rest of my thesis committee: Dr. Prashant Mhaskar, Dr. Reilly James and Dr. Antoine Deza, for their insightful comments and encouragement. I would like to thank Dr. Shiping Zhu for his gracious encouragement and support in the hard times. I thank all the office staff at Department of Chemical Engineering at McMaster University. In particular I would like to thank Cathie Roberts, Lynn Falkiner and Kathy Goodram for their assistance.

I'm also thankful to Nippon Steel & Sumitomo Metal Corporation for financial support which made it possible for me to study at McMaster University. I would like to thank all my friends at McMaster for their support.

Finally, I must express my gratitude to Yukiko, my wife. Without her continuous support and encouragement, I would not have finished this thesis.

Table of Contents

Chapter 1 Introduction

Recently, the physical systems of manufacturing plants have become increasingly complex and large-scale due to the complex combination of a large number of different components. Since these components are developed individually often by many different technicians, even for skilled persons it is extremely difficult to understand the behavior of such large and complex integrated systems. Meanwhile, rapid development of measurement, automation, computing and database technologies have enabled many companies to record a huge number of process variables in industrial plant historians, which makes it possible to conduct data-driven approach for automatically extracting useful information from such big data. In this thesis, several applications of data-driven methods in making sense of industrial process data are discussed.

The first part in this thesis focuses on process monitoring and fault diagnosis which have been developed to extract useful information from highly correlated process variables. Process monitoring and fault diagnosis techniques are becoming critically important in order to improve product quality, yields, energy efficiency, plant safety and eco sustainability [1][2]. The approaches to process monitoring, fault diagnosis and process analysis fall into the two categories, which are the model based and the data-driven techniques [3]. Model-based process monitoring methods may be applicable only if the accurate mechanistic models of processes can be developed [4][5]. However, those first-principle models require in-depth knowledge about processes and also it is difficult and time consuming to build precise mechanistic models for large-scale complex industrial plants. Conversely, the data-driven monitoring techniques have become increasingly attractive because they do not require in-depth fundamental knowledge and mechanistic models but instead depend on the historical process data only. Traditionally, univariate statistical process control (SPC) has been applied for process monitoring. Nevertheless, most SPC methods are based on control charts of individual or non-correlated process variables and thus the highly correlated process variables in industrial plants can cause the failure of conventional SPC methods. Multivariate statistical process monitoring (MSPM) techniques have been developed to extract useful information from a large number of highly correlated process variables and historical data sets. Two widely used methods in MSPM fileds are principal component analysis (PCA) and partial least square (PLS). These methods can identify the statistical model within the low-dimensional subspace that retains most of the variance or covariance structure. Then the statistics such as T^2 and SPE are developed for monitoring multivariate process data [6]. These PCA/PLS-based process monitoring techniques are based on second-order statistics of covariance and thus they may not effectively extract the non-Gaussian process features that are characterized by the higher-order statistics. In industrial processes, however, process data often obey non-Gaussian distribution so that the PCA/PLS-based process monitoring methods cannot efficiently extract the process features. In order to handle non-Gaussian processes, independent component analysis (ICA) has been applied to project multivariate process data into latent subspace of statistically independent component (IC). Since ICs are assumed to be non-Gaussian, they retain the non-Gaussian process features that may not be effectively extracted in PCA/PLS methods. Moreover, the ICA-based statistics like I^2 and SPE are developed for monitoring non-Gaussian process data [7]. Nevertheless, if the variations in the process measurement variables are most influential on product quality variables, the ICA-based process monitoring methods may not be well suited because they do not take into consideration the quality variables in the PLS-based monitoring techniques.

Once a fault is detected, all the above MSPM methods can make contribution plots to identify the major fault effect variables without process knowledge. However, the contribution methods are not able to detect the root causes of faulty operations, while they can find the variables which are affected by the root cause variables. Therefore, in order to identify the cause-effect relationship and the direction of the fault effect, signed directed graph (SDG) and improved version of SDG have been developed [8][9][10]. However, these methods are able to identify candidate faults that the prior fault database includes and thus it is challenging to diagnosis faulty events that the prior fault database does not contain. More recently, Granger causality methods are proposed in order to identify the cause-effect relationship among process variables and capture the root cause of plant oscillation [11]. Nevertheless, it cannot explain whether the calculated Granger causality comes from process upset or not. Alternatively, causal maps that are based on directed graphs are employed to identify the cause-effect relationships by using the Kullback-Leibner distance [12]. However, this method requires in-depth process knowledge to make the fault propagation and hence it may not work well when the propagation pathways are difficult to identify due to the lack of process knowledge. Furthermore, the cross-correlation function [13] and the transfer entropy [14] are utilized to identify the fault propagation pathways. While these two methods do not require any process knowledge, the cross-correlation and transfer entropy can express relationship between two process variables only and thus it cannot model more intricate dependencies including over three variables. For these reasons mentioned above, it is very challenging to identify the root-cause relationship and the fault propagation pathways without in-depth process knowledge.

The second part in this thesis addresses the scheduling under uncertain production times. Knowledge of the production times is crucial for optimal operations of real world industrial systems. The approaches to predict the production times can fall into the two categories, which are based on mechanistic model or data-driven techniques. Mechanistic models require in-depth process knowledge and cannot take into account uncertainties that exist in the process. Meanwhile, data-driven approaches do not require in-depth knowledge about processes and they are able to deal with the process uncertainties. While some advanced machine learning methods such as support vector regression and random forest regression may be able to predict the production time, these methods do not consider model uncertainties and cannot handle missing values and unobserved variables. Furthermore, it is mandatory to have multiple specific models for specific purpose, e.g. for production planning or for scheduling. In order to overcome these limitations, Bayesian network models are useful in terms of having single prediction model for predicting production time with uncertainties. Bayesian network allows us to handle the complicated interaction among process variables and uncertainties. However, real-world industrial production data often have large domain of discrete variables and continuous variables simultaneously, which makes it difficult to estimate probability distributions of the production time. In this paper, most likely steel-plate production times are computed via Bayesian network.

In addition, the computed production times can be utilized to optimize the schedules for steelmaking continuous casting (SCC) production. In general, SCC production scheduling problems should determine charge sequencing in each casting machine (first level scheduling) and timing of the charges on continuous casting machines (second level scheduling). While the second level scheduling has been studied by several research groups, few studies have dealt with the first level scheduling since it is intractable to solve the first level scheduling of the SCC processes for a long time horizon. Furthermore, scheduling of continuous casting requires simultaneously minimization of the contaminated cost between charges, tardiness with respect to the customer due dates and violation of process capacities. While various kinds of multi-objective optimization algorithms have been proposed, most of them are population based algorithms which may not be suitable for sequence optimization problems like SCC mainly because combinations of the solutions rarely generate better solutions particularly for sequence optimization problems.

1.1 Research outline

The main objective of this thesis is to develop the date-driven methods for process monitoring, fault detection and scheduling.

Chapter 2. The first achievement of this thesis is to propose a new Quality Relevant non-Gaussian Latent Subspace Projection (QNGLSP) method for industrial process monitoring and fault detection by taking into consideration both process measurement and quality variables. In order to capture the non-Gaussian features and relationships between input and output variables, the developed QNGLSP method employs mutual information for searching the latent directions within input and output spaces, respectively. The utility and performance of the new QNGLSP method are demonstrated through the application example of the Tennessee Eastman Chemical process. This work has lead to the publication in *AIChE Journal* [15].

Chapter 3. The second contribution is to extend the QNGLSP method to nonlinear kernel feature space for monitoring quality related operational performance of nonlinear batch processes. Since batch process data typically have three-dimensional structure of batches, sampling instances and measurement variables, they are unfolded into two-dimensional matrices through multi-way analysis for handling batch trajectories. Then, the kernel principal components are extracted from the unfolded data sets to characterize the nonlinear process dynamics in the high-dimensional kernel feature space. Finally, the multi dimensional latent directions in the kernel-principal-components subspaces are searched so that the mutual information between measurement and quality variables is maximized. The proposed method is applied to the fed-batch Penicillin fermentation process. This work has lead to the publication in *Journal of Process Control* [16].

Chapter 4. The next achievement of this work is the development of a data-driven structure learning algorithm that can construct the probabilistic graphical model for the identification of the fault propagation pathways and the root cause variables. The proposed method does not require in-depth process knowledge. Instead, the most probable probabilistic graphical model is constructed from historical process data as well as the process measurement incidence matrix, which requires very little knowledge about the process. First, the incidence matrix is created from the process flow diagram. Based on the constructed incidence matrix, a set of monitored variables is broken into smaller subsets. Then the subgraph corresponding to each subset is searched from the historical process data both by maximizing Bayesian scores and by implementing the Markov chain Monte Carlo simulation. The proposed probabilistic graphical model based process monitoring method is applied to the Tennessee Eastman Chemical process. This work has led to the publication in *Computers and Chemical Engineering* [17].

Chapter 5 and 6. In addition to process monitoring, fault detection and fault diagnosis, data-driven techniques for predicting the production times in manufacturing processes are developed. Accurate estimation of the production times is essential ingredient for optimal production plants and schedules. This work presents a new inference algorithm in Bayesian networks with large domain discrete variables that enables:
1 Estimation of the probability distributions of production times in order to handle various sources of uncertainties.
2 Dealing with unobservable variables, because it is desirable to have a single model and avoid multiple models that meet with specific problems.

The network inference algorithm presented in Chapter 4 is utilized to construct most likely structure of the Bayesian network representing a complex manufacturing process. Since Bayesian networks constructed in the real-world industrial processes often contain large domain discrete variables and continuous variables simultaneously, inference technique based on the decision-tree structured conditional probability tables (CPTs) is proposed. This work has led to a publication in *Expert Systems With Applications* [18] and *Computers and Chemical Engineering* [19].

Chapter 7. The next accomplishment is the development of the optimization algorithm for SCC scheduling of steel-plate production by making use of the most likely production times computed via Bayesian network mentioned in chapter 6. The main difficulty to solve this scheduling problem is a large number of binary variables that are needed to represent accurately the process when a full space model mixed-integer linear programming (MILP) is used. Therefore, I proposed a decomposition approach where the full space model is divided into two levels: (i) top level which determines the number of pots per grade for each day by solving the relaxed MILP that does not consider the sequence related penalties and (ii) lower level which optimizes the cast sequence by means of meta-heuristic methods. This work has led to a publication in *Computers and Chemical Engineering* [20]

Chapter 8. The final accomplishment deals with multi-objective evolutional algorithms derived from non-dominated sorting genetic algorithm NSGA-II for steel-plate production schedules mentioned in chapter 7. In order to generate a large diverse Pareto optimal set, the cut-and-paste and copy-and-paste tranposons are employed as genetic operators. Real world steel production process data have been used to examine the effectiveness of the proposed algorithm. This work is submitted to the *AIChE journal* for peer review.

Chapter 9. Finally, the main conclusions of this work and future research suggestions are described in chapter 9.

1.2 References

[1] Piovoso M, Hoo K, Multivariate statistics for process control. *IEEE Control Syst Mag*. 2002;22:8-9

[2] Venkatasubramanian V, Rengaswamy R, Yin K, Kavuri SN. A review of process fault detection and diagnosis: part I: quantitative model-based methods. *Comput Chem Eng.* 2003;27:293-311

[3] Gertler J. Survey of model-based failure detection and isolation in complex plants. *IEEE Control Syst Mag*. 1988;12:3-11

[4] Pons M, Rajab A, Flaus J, Engasser J, Cheruy A. Comparison of estimation methods for biotechnological processes. *Chem Eng Sci*. 1988;8:1909-1914

[5] Bastin G, Dochan D. On-line estimation and adaptive control of bioreactors. Amsterdam, Netherlands: Elsevier, 1990.

[6] Qin SJ. Statistical process monitoring: basics and beyond. *J Chemom*. 2003;17:480-502

[7] Lee JM, Yoo CK, Lee IB. Statistical process monitoring with independent component analysis. *J Process Control*. 2004;14:467-485

[8] Maurya MR, Rengaswamy R, Venkatasubramanian V. Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets. *Eng Appl ArtifIntell.* 2004;17:501–18.

[9] Maurya MR, Rengaswamy R, Venkatasubramanian V. A signed directed graph and qualitative trend analysis-based framework for incipient fault diagnosis. ChemEng Res Des. 2007;85:1407–22.

[10] Vedam H, Venkatasubramanian V. PCA-SDG based process monitoring and fault diagnosis. *Control Eng Pract*. 1999;7:903–17.

[11] Yuan T, Qin J. Root cause diagnosis of plant-wide oscillations using Granger causality. *J Process Control*. 2014;24:450–9.

[12] Chiang LH, Braatz RD. Process monitoring using causal map and multivariate statistics fault detection and identification. *Chemom Intell Lab Syst*. 2003;65:159–78.

[13] Bauer M, Thornhill NF. A practical method for identifying the propagation path of plant-wide disturbances. *J Process Control*. 2008;18:707–19.

[14] Bauer M, Thornhill NF. A practical method for identifying the propagation path ofplant-wide disturbances. *J Process Control*. 2008;18:707–19.

[15] Mori J, Yu J. A quality relevant non-Gaussian latent subspace projection method for chemical process monitoring and fault detection. *AIChE J*. 2014a;60:485–99.

[16] Mori J, Yu J. Quality relevant nonlinear batch process performance monitoring usinga kernel based multiway non-Gaussian latent subspace projection approach. *J Process Control*. 2014b;24:57–71.

[17] Mori J, Mahalec V, Yu J. Identification of probabilistic graphical network model for root-cause diagnosis in industrial processes. *Comput Chem Eng*. 2014;71:171-209

[18] Mori J, Mahalec V. Inference in hybrid Bayesian networks with large discrete and continuous domains. *Expert Systems with Applications*. 2016;46:1-19

[19] Mori J, Mahalec V. Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables. *Comput Chem Eng*. 2015;79:113-134

[20] Mori J, Mahalec V. Planning and scheduling of steel plates production.Part II: Scheduling of continuous casting. *Comput Chem Eng*. 2016

Chapter 2

A Quality Relevant Non-Gaussian Latent Subspace Projection Method for Chemical Process Monitoring and Fault Detection

A Quality Relevant Non-Gaussian Latent Subspace Projection Method for Chemical Process Monitoring and Fault Detection

Junichi Mori and Jie Yu
Dept. of Chemical Engineering, McMaster University, Hamilton, Ontario L8S 4L7, Canada

DOI 10.1002/aic.14261
Published online October 28, 2013 in Wiley Online Library (wileyonlinelibrary.com)

Partial least-squares (PLS) method has been widely used in multivariate statistical process monitoring field. The goal of traditional PLS is to find the multidimensional directions in the measurement-variable and quality-variable spaces that have the maximum covariances. Therefore, PLS method relies on the second-order statistics of covariance only but does not takes into account the higher-order statistics that may involve certain key features of non-Gaussian processes. Moreover, the derivations of control limits for T^2 and squared prediction error (SPE) indices in PLS-based monitoring method are based on the assumption that the process data follow a multivariate Gaussian distribution approximately. Meanwhile, independent component analysis (ICA) approach has recently been developed for process monitoring, where the goal is to find the independent components (ICs) that are assumed to be non-Gaussian and mutually independent by means of maximizing the high-order statistics such as negentropy instead of the second-order statistics including variance and covariance. Nevertheless, the IC directions do not take into account the contributions from quality variables and, thus, ICA may not work well for process monitoring in the situations when the quality variables have strong influence on process operations. To capture the non-Gaussian relationships between process measurement and quality variables, a novel projection-based monitoring method termed as quality relevant non-Gaussian latent subspace projection (QNGLSP) approach is proposed in this article. This new technique searches for the feature directions within the measurement-variable and quality-variable spaces concurrently so that the two sets of feature directions or subspaces have the maximized multidimensional mutual information. Further, the new monitoring indices including I^2 and SPE statistics are developed for quality relevant fault detection of non-Gaussian processes. The proposed QNGLSP approach is applied to the Tennessee Eastman Chemical process and the process monitoring results of the present method are demonstrated to be superior to those of the PLS-based monitoring method. © 2013 American Institute of Chemical Engineers *AIChE J*, 60: 485–499, 2014

Keywords: quality relevant process monitoring, fault detection, non-Gaussian latent subspace projection, partial least squares, independent component analysis, multidimensional mutual information

Introduction

Process monitoring, fault detection and diagnosis are gaining significant attention for the rapid detection of abnormal operation, process upsets, equipment malfunctions, sensor failures, and other special events in industrial plants to improve plant safety, product quality, energy efficiency, and profit margin.[1–3] Recently, owing to the fast development of measurement, automation, and advanced computing technologies, a huge number of process variables can be frequently measured and recorded in industrial plant historians, which make it possible to conduct data-driven large-scale process monitoring and fault diagnosis. Meanwhile, effective process monitoring plays a critical role in ensuring product quality, operation safety, and manufacturing sustainability.

The approaches to process monitoring, fault detection and diagnosis fall into the two categories, which are the model-based and the data-driven techniques.[4] Model-based process monitoring methods may be applicable only if the accurate mechanistic models of processes can be developed.[5–7] However, those first-principle models require in-depth knowledge about processes and also it is difficult and time consuming to build precise mechanistic models for large-scale complex industrial plants. Conversely, the data-driven monitoring techniques have become increasingly attractive because they do not require in-depth fundamental knowledge and mechanistic models but instead depend on historical process data only. Traditionally, univariate statistical process control (SPC) has been applied for process monitoring. Nevertheless, most SPC methods are based on control charts of individual or noncorrelated process variables and, thus, the highly correlated process variables in industrial plants can cause the failure of conventional SPC methods.[8]

Multivariate statistical process monitoring (MSPM) techniques have been developed to extract useful information from large number of highly correlated process variables and historical data sets.[9–15] Two popular latent variables methods in MSPM field are principal component analysis (PCA) and

Correspondence concerning this article should be addressed to J. Yu at jieyu@mcmaster.ca.

partial least squares or projection to latent structure (PLS).[16–24] The major advantages of these methods include their strong capability to deal with the colinearity among different process variables and identify the statistical model within a lower-dimensional latent subspace that well retains the multivariable correlation structure. Then the statistics such as T^2 and squared prediction error (SPE) are proposed for extracting the critical features of process data for fault detection and diagnosis.[25,26] Furthermore, the total projection to latent structures (T-PLS) method is proposed for process monitoring.[27] Compared to conventional PLS method that is based on regression models, T-PLS approach can separate the orthogonal and correlated parts to the quality variable and, thus, is proven to be more suitable for process monitoring and fault detection. The conventional PCA/PLS-based process monitoring techniques are based on the second-order statistics of covariance only but do not take into account the higher-order statistics. Hence, they may not effectively extract the non-Gaussian process features that are characterized by the higher-order statistics, even though PCA and PLS models do not require Gaussian distribution explicitly. Moreover, the derivations of T^2 and SPE control limits in PCA/PLS-based process monitoring methods are based on the assumption that the process data follow a multivariate Gaussian distribution approximately.[28] In industrial processes, however, operating condition shifts are often encountered due to the changes of various factors such as feedstock, product specification, set points, and manufacturing strategy.[29] Such operating condition changes often result in non-Gaussian probability distribution of process data. As an alternative solution, eigenvalue decomposition on the covariance matrices of process measurement variables is utilized to determine the dissimilarity factor between the normal and the monitored data sets.[30] Nevertheless, this method suffers from the same issue as PCA/PLS-based monitoring methods that only the second-order statistics are taken into account and, thus, the non-Gaussian process features may not be efficiently extracted.

To deal with non-Gaussian processes, independent component analysis (ICA) has been applied to project multivariate process data into latent subspace of statistically independent components (IC).[31–37] ICs are assumed to be non-Gaussian and mutually independent based on high-order statistics and they retain the non-Gaussian process features that may not be effectively extracted in traditional PCA/PLS methods. Moreover, the ICA-based statistics like I^2 and SPE are developed for detecting faulty operation.[38] More recently, multidimensional mutual information is adopted to measure the statistical independency between IC subspaces and further determine the dissimilarity factors between the normal benchmark and the monitored data sets for process monitoring and fault detection.[39] This method takes into consideration not only the high-order statistics but also the time-varying process dynamics. However, if the variations in the process measurement variables are most influential on product quality variables, the above ICA-based monitoring techniques may not be well suited because only the process measurement variables are utilized in the developed statistical models while the product quality variables are excluded. In other words, they do not take into account the quality variables as in the PLS-based monitoring methods. Another non-Gaussian process monitoring technique is based on Gaussian mixture models (GMM) that decompose the process data into multiple Gaussian components with different means and covariances corresponding to various operational conditions and modes.[40] In this way, the globally non-Gaussian process data can be characterized as mixture models of different Gaussian components and then the Bayesian inference strategy can be developed to incorporate multiple local models for fault detection.[37] Furthermore, Gaussian mixture model can be updated by adopting particle filter strategy to take into account the dynamic changes of operating scenarios.[41] In addition, an ensemble clustering-based process pattern construction method and multiple ICA-PCA model-based multimode process monitoring technique are developed for operating mode identification and fault detection.[42] Due to the clustering algorithm as well as the integration of PCA and ICA methods, both Gaussian and non-Gaussian process features in the multimode operating data can be captured. However, the non-Gaussian process monitoring methods still do not utilize output variables especially product quality variables and, thus, the detected abnormal operating events may not be relevant to any degradations of product quality or losses of other operational objectives such as energy efficiency and sustainability.

Alternately, supervised learning techniques such as Fisher discriminant analysis (FDA) and support vector machine (SVM) have been developed for chemical process monitoring.[43,44] FDA approach can identify multiple classes with both maximized between-class separation and minimized within-class scattering. FDA may become well suited only if the subset of data in each class do not have significant within-class multimodality. To overcome this limitation, the localized Fisher discriminant analysis (LFDA) has recently been proposed for process monitoring and fault detection.[45] Nevertheless, the performance of LFDA depends on the way of calculating a similarity matrix and, thus, the best selection of similarity matrix in LFDA algorithm is very important. Conversely, SVM can perform nonlinear classification by maximizing separating margin between support vector hyperplanes. However, all these methods are based on supervised learning models and, thus, require known class labels of all the training samples, which may not be realistic for industrial applications. To overcome this limitation, support vector clustering (SVC)-based probabilistic approach is proposed for unsupervised process monitoring.[46] Different from SVM, SVC has ability to classify the unlabeled training samples and, thus, known class labels are not needed in advance. Nevertheless, the above supervised and unsupervised monitoring methods typically do not include the output quality variables in the classification models and, thus, are not quality relevant either.

In this study, a novel output quality variable relevant non-Gaussian latent subspace projection method is proposed to monitor complex chemical processes that follow non-Gaussian distributions. Both the process measurement and product quality variables are used to extract the non-Gaussian subspaces for monitoring the abnormal behaviors in process operations that have significant influence on product quality. The basic idea is to estimate the non-Gaussian loading matrices of both process measurement and product quality variables, respectively, so that the mutual information between latent scores of measurement and quality variables is maximized. In this way, the proposed method can identify the feature directions in the measurement-variable and quality-variable spaces concurrently to retain the maximized statistical dependency between two latent subspaces. With the quality relevant non-Gaussian latent variable model, I^2 and SPE indices are further proposed and compared for process monitoring and fault detection. In contrast to PLS-based monitoring methods, the proposed approach utilizes the high-order statistics of mutual information instead of the second-order statistics of covariance and, thus, can well extract the non-Gaussian features from process data. Meanwhile, compared to ICA, GMM or supervised learning-based monitoring approaches, both process measurement and output quality variables are included in the non-Gaussian model of the presented method so that the latent directions within input and output spaces are concurrently searched for with optimized mutual information.

The remainder of the article is organized as follows. “Review of PLS and ICA-Based Process Monitoring Methods” section briefly reviews the PLS and ICA-based monitoring technique and discusses the issues of these conventional methods. “Quality Relevant non-Gaussian Latent Subspace Projection Approach for Process Monitoring” section describes the proposed quality relevant non-Gaussian latent subspace projection (QNGLSP) approach and the corresponding monitoring indices for capturing abnormal process operations. “Application Example” section demonstrates the utility and performance of the new process monitoring approach through the application example of the Tennessee Eastman Chemical process and its comparison to the PLS-based monitoring method. Finally, the conclusions of this work are summarized in “Conclusions” section.

Review of PLS and ICA-Based Process Monitoring Methods

PLS-*based process monitoring method*

PLS handles high-dimensional correlated data by finding the multidimensional latent directions in both the measurement-variable and quality-variable spaces with the maximum covariance. Given an input matrix $\mathbf{X}=[\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_m]\in\Re^{n\times m}$ consisting of n samples along m process measurement variables and an output matrix $\mathbf{Y}=[\mathbf{y}_1,\mathbf{y}_2,\ldots,\mathbf{y}_k]\in\Re^{n\times k}$ along k quality variables, they are decomposed onto low-dimensional subspaces as follows

$$\mathbf{X}=\mathbf{T}\mathbf{P}^T+\mathbf{E} \tag{1}$$

$$\mathbf{Y}=\mathbf{T}\mathbf{Q}^T+\mathbf{F} \tag{2}$$

where $\mathbf{T}=[\mathbf{t}_1,\mathbf{t}_2,\ldots,\mathbf{t}_d]\in\Re^{n\times d}$ is the score matrix, $\mathbf{P}=[\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_d]\in\Re^{m\times d}$ is the loading matrix for $\mathbf{X}$, $\mathbf{Q}=[\mathbf{q}_1,\mathbf{q}_2,\ldots,\mathbf{q}_d]\in\Re^{k\times d}$ is the loading matrix for $\mathbf{Y}$, $\mathbf{E}\in\Re^{n\times m}$ denotes the residual matrix for $\mathbf{X}$, $\mathbf{F}\in\Re^{n\times k}$ represents the residual matrix for $\mathbf{Y}$, and d is the selected number of latent variables in PLS model. The basic idea of PLS is to determine the score matrix $\mathbf{T}$ and the loading matrices $\mathbf{P}$ and $\mathbf{Q}$ from $\mathbf{X}$ and $\mathbf{Y}$ through the nonlinear iterative partial least-squares (NIPALS) algorithm.[47]

Given a new test sample $\mathbf{x}$, its corresponding prediction and residual vectors are given as follows

$$\text{Score}: \mathbf{t}=\mathbf{x}\mathbf{P} \tag{3}$$

$$\text{Prediction}: \hat{\mathbf{x}}=\mathbf{x}\mathbf{P}\mathbf{P}^T \tag{4}$$

$$\text{Residual}: \mathbf{e}=\mathbf{x}\left(\mathbf{I}_m-\mathbf{P}\mathbf{P}^T\right) \tag{5}$$

where $\mathbf{I}_m$ is a $m\times m$ identity matrix. The following PLS-based T^2 and SPE statistics are used as the measures of variations in the latent variable and residual subspaces for process monitoring

$$T^2=\mathbf{t}\left\{\frac{1}{n-1}\mathbf{T}^T\mathbf{T}\right\}^{-1}\mathbf{t}^T \tag{6}$$

$$\text{SPE}=\mathbf{e}\mathbf{e}^T \tag{7}$$

where the confidence limits for T^2 and SPE statistics can be estimated from F and χ^2 distributions, respectively.[14]

ICA-*based process monitoring method*

PLS is based on the covariance between the score vectors of the measurement and quality variables, respectively. However, it does not take into account the high-order statistics and, thus, may not be well suited in extracting the non-Gaussian features from process data. In contrast, ICA is developed for non-Gaussian process monitoring on the basis of high-order statistics. It is essentially a multivariate statistical technique for computing ICs that are assumed to be non-Gaussian and mutually independent.[48] Given the input matrix $\mathbf{X}=[\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_m]\in\Re^{n\times m}$, all the process measurement variables are assumed to be generated as linear combinations of m unknown ICs

$$\mathbf{X}^T=\mathbf{A}\mathbf{S}^T \tag{8}$$

where $\mathbf{A}=[\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_d]\in\Re^{m\times d}$ is unknown mixing matrix and $\mathbf{S}=[\mathbf{s}_1,\mathbf{s}_2,\ldots,\mathbf{s}_d]\in\Re^{n\times d}$ represents the IC matrix. The solution is equivalent to finding a demixing matrix $\mathbf{W}=[\mathbf{w}_1,\mathbf{w}_2,\ldots,\mathbf{w}_d]^T\in\Re^{d\times m}$ as follows

$$\mathbf{S}^T=\mathbf{W}\mathbf{X}^T \tag{9}$$

where the ICs $\mathbf{S}=[\mathbf{s}_1,\mathbf{s}_2,\ldots,\mathbf{s}_d]$ have the maximized statistical independency in terms of negentropy among each other.[49,50]

Given a new test sample vector $\mathbf{x}$, its corresponding IC score, prediction, and residual vectors are given below

$$\text{IC Score}: \mathbf{y}=\mathbf{x}\mathbf{W}^T \tag{10}$$

$$\text{Prediction}: \hat{\mathbf{x}}=\mathbf{x}\mathbf{W}^T\mathbf{A}^T \tag{11}$$

$$\text{Residual}: \mathbf{e}=\mathbf{x}\left(\mathbf{I}_m-\mathbf{W}^T\mathbf{A}^T\right) \tag{12}$$

Further, the I^2 and SPE statistics can be defined as follows for process monitoring[38]

$$I^2=\mathbf{y}\mathbf{y}^T \tag{13}$$

$$\text{SPE}=\mathbf{e}\mathbf{e}^T \tag{14}$$

where the confidence limits for I^2 and SPE statistics can be estimated through kernel density estimation (KDE).[51]

Quality Relevant Non-Gaussian Latent Subspace Projection Approach for Process Monitoring

The basic idea of PLS approach is to optimize the loading matrices $\mathbf{P}$ and $\mathbf{Q}$ from the input and output data matrices $\mathbf{X}$ and $\mathbf{Y}$ by means of the NIPALS algorithm. The embedded optimization problem in PLS is defined as

$$\begin{aligned}\max \quad & \psi\left(\mathbf{w}_i^{\text{PLS}},\mathbf{c}_i^{\text{pls}}\right)=\text{cov}\left(\mathbf{t}_i^{\text{PLS}},\mathbf{u}_i^{\text{PLS}}\right)=\text{cov}\left(\mathbf{X}_i\mathbf{w}_i^{\text{PLS}},\mathbf{Y}_i\mathbf{c}_i^{\text{PLS}}\right)\\ \text{s.t.} \quad & ||\mathbf{w}_i^{\text{PLS}}||=1,||\mathbf{c}_i^{\text{PLS}}||=1\end{aligned} \tag{15}$$

where $\psi\left(\mathbf{w}_i^{\text{PLS}},\mathbf{c}_i^{\text{PLS}}\right)$ is the objective function $\mathbf{w}_i^{\text{PLS}}$ and $\mathbf{c}_i^{\text{PLS}}$ are the weighting vectors, and $\text{cov}\left(\mathbf{t}_i^{\text{PLS}},\mathbf{u}_i^{\text{PLS}}\right)$ denotes the covariance between the score vectors $\mathbf{t}_i^{\text{PLS}}$ and $\mathbf{u}_i^{\text{PLS}}$. It should be noted that the weighting vector $\mathbf{w}_i^{\text{PLS}}$ also corresponds to the i-th eigenvector of the matrix $\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{X}$.[52] If process data follow Gaussian distribution, the joint Gaussian density function p_{G} of $\mathbf{v}=[v_1,\ldots,v_M]$ can be described as follows

$$p_{\text{G}}(\mathbf{v})=f(\mathbf{v}|\mu,\Sigma)=\frac{1}{(2\pi)^{M/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{v}-\mu)^T\Sigma^{-1}(\mathbf{v}-\mu)\right) \tag{16}$$

where $\mathbf{v}$ represents the combined input and output variables, μ denotes a M-dimensional mean vector, and Σ is a $M\times M$ covariance matrix. As Eq. 16 indicates, if data are normalized to zero-mean, $p_{\text{G}}(\mathbf{v})$ is parameterized by the second-order statistic of covariance only. Therefore, PLS is able to capture the characteristic relationship between process measurement and quality variables if process data follow Gaussian distribution approximately.

However, PLS-based monitoring methods may not be well suited if process data follow significantly non-Gaussian distribution because the joint density function cannot be adequately characterized by the second-order statistics of covariance only. Specifically, the joint density function $p(\mathbf{v})$ of non-Gaussian data with up to fifth-order statistics can be expressed through Edgeworth expansion as[53]

$$p(\mathbf{v})\approx p_{\text{G}}(\mathbf{v})\left(1+\frac{1}{3!}\sum_{i,j,k}\kappa^{i,j,k}h_{ijk}(\mathbf{v})+\frac{1}{4!}\sum_{i,j,k,l}\kappa^{i,j,k,l}h_{ijkl}(\mathbf{v}) + \frac{1}{72}\sum_{i,j,k,l,p,q}\kappa^{i,j,k}\kappa^{l,p,q}h_{ijklpq}(\mathbf{v})\right) \tag{17}$$

where $p_G(\mathbf{v})$ denotes the Gaussian density function with the same mean and covariance as $p(\mathbf{v})$, (i,j,k), (i,j,k,l), and $(i,j,k,l,p,q)\in\{1,\ldots,M\}$ are the input dimensions, h_{ijk}, h_{ijkl}, and h_{ijklpq} are the ijk-th, $ijkl$-th, and $ijklpq$-th Hermite polynomials $\kappa^{i,j,k}=\frac{\kappa^{ijk}}{\sigma_i\sigma_j\sigma_k}$ is the standardized cumulant with κ^{ijk} being the cumulant for input dimension (i,j,k), and $\kappa^{i,j,k,l}=\frac{\kappa^{ijkl}}{\sigma_i\sigma_j\sigma_k\sigma_l}$ is the standardized cumulant with κ^{ijkl} being the cumulant for input dimension (i,j,k,l). Since non-Gaussian probability density function $p(\mathbf{v})$ includes the higher-order statistics instead of covariance only, PLS may not efficiently extract the non-Gaussian process features of measurement variables that contain sufficient information on product quality variables. Thus, PLS-based monitoring methods may not be effective in detecting abnormal events of non-Gaussian processes.

In ICA method, ICs are calculated by using the mutual information between the measurement variables and the high-order statistics are taken into account for extracting non-Gaussian process features. However, it does not incorporate output quality variables in data analysis and, thus, may not specifically isolate the abnormal variations of process measurement variables that have significant influence on product quality variables.

Due to the above technical challenges, the new QNGLSP method is developed for non-Gaussian process monitoring with output quality variables incorporated. The basic idea of QNGLSP approach is to find the multidimensional latent directions in the measurement-variable and quality-variable spaces concurrently so that the maximized multidimensional mutual information between measurement and quality spaces is obtained. It should be noted that mutual information is a quantitative measure of statistical dependency between two random variables and can be estimated from information entropy. Compared to covariance, it is essentially high-order statistics and, thus, is able to extract the non-Gaussian process features.

Given an input matrix $\mathbf{X}=[\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_m]\in\Re^{n\times m}$ with n samples and m process measurement variables and an output matrix $\mathbf{Y}=\left[\mathbf{y}_1,\mathbf{y}_2,\ldots,\mathbf{y}_k\right]\in\Re^{n\times k}$ with k quality variables, the data matrices are first normalized to zero-mean and unit-variance and then decomposed onto low-dimensional subspaces as follows

$$\mathbf{X}=\mathbf{S}\mathbf{P}^T+\mathbf{E} \tag{18}$$

$$\mathbf{Y}=\mathbf{S}\mathbf{Q}^T+\mathbf{F} \tag{19}$$

where $\mathbf{S}=[\mathbf{s}_1,\mathbf{s}_2,\ldots,\mathbf{s}_d]\in\Re^{n\times d}$ denotes the score matrix, $\mathbf{P}=[\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_d]\in\Re^{m\times d}$ is the loading matrix for $\mathbf{X}$, $\mathbf{Q}=[\mathbf{q}_1,\mathbf{q}_2,\ldots,\mathbf{q}_d]\in\Re^{k\times d}$ is the loading matrix for $\mathbf{Y}$, $\mathbf{E}\in\Re^{n\times m}$ is the residual matrix for $\mathbf{X}$, $\mathbf{F}\in\Re^{n\times k}$ is the residual matrix for $\mathbf{Y}$, and d is the number of latent variables. The initial objective in the proposed QNGLSP algorithm is to find weighting vectors $\mathbf{w}$ and $\mathbf{c}$ from the deflated $\mathbf{X}$ and $\mathbf{Y}$ for each pair of score vectors through the following constrained optimization problem

$$\max I(\mathbf{s}_i,\mathbf{u}_i)=I(\mathbf{X}_i\mathbf{w}_i,\mathbf{Y}_i\mathbf{c}_i) \tag{20}$$

$$\text{subject to}\ ||\mathbf{w}||_i=1,\ ||\mathbf{c}||_i=1 \tag{21}$$

where $I(\mathbf{s}_i,\mathbf{u}_i)$ represents the mutual information between the score vectors $\mathbf{s}_i$ and $\mathbf{u}_i$. The mutual information $I(\mathbf{s}_i,\mathbf{u}_i)$ can be expressed as

$$\begin{aligned}I(\mathbf{s}_i,\mathbf{u}_i)&=H(\mathbf{u}_i)-H(\mathbf{u}_i|\mathbf{s}_i)\\&=H(\mathbf{s}_i)-H(\mathbf{s}_i|\mathbf{u}_i)\\&=H(\mathbf{s}_i,\mathbf{u}_i)-H(\mathbf{u}_i|\mathbf{s}_i)-H(\mathbf{s}_i|\mathbf{u}_i)\end{aligned} \tag{22}$$

where $H(\mathbf{u}_i)$ is the marginal entropy, $H(\mathbf{u}_i|\mathbf{s}_i)$ is the conditional entropy, and $H(\mathbf{s}_i,\mathbf{u}_i)$ is the joint entropy defined as

$$H(\mathbf{u}_i)=-\int_{\mathbf{u}_i}f(u)\log f(u)du \tag{23}$$

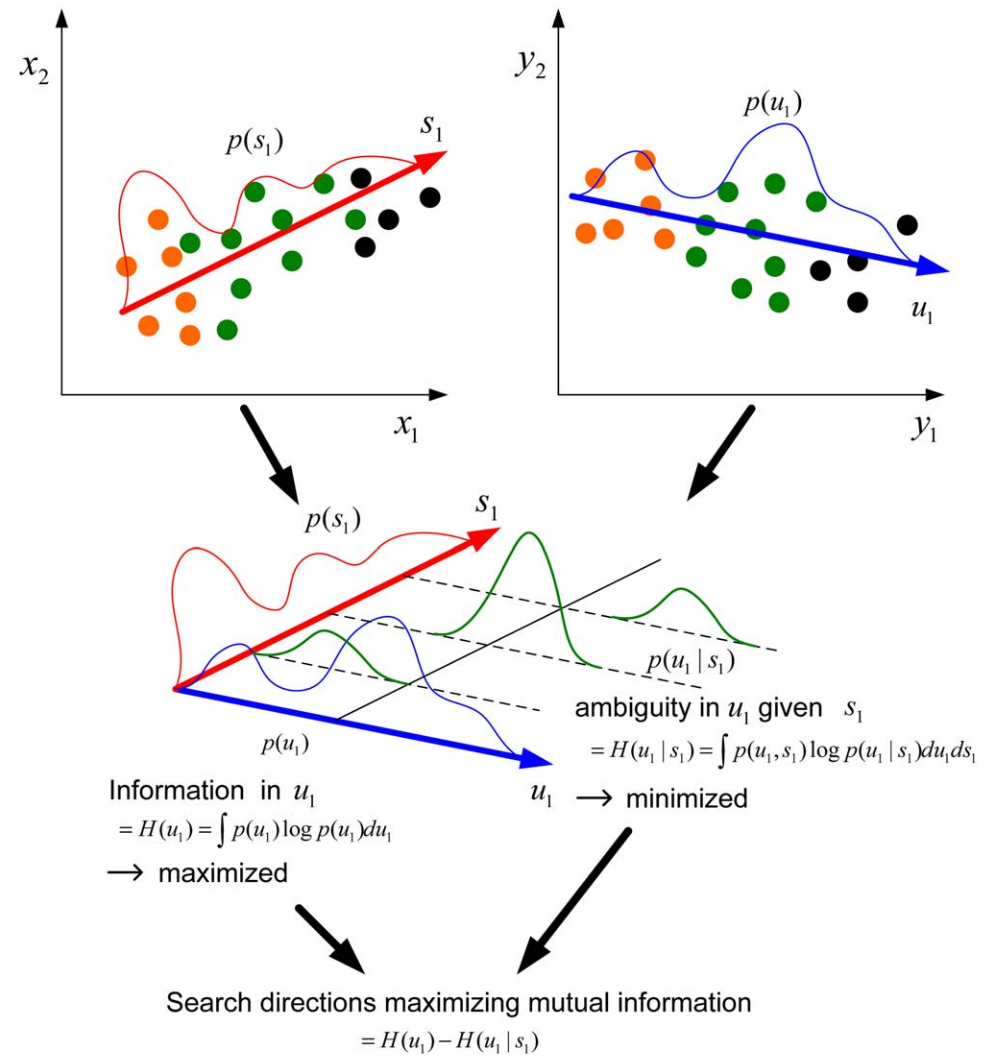
$$H(\mathbf{u}_i|\mathbf{s}_i)=-\int_{\mathbf{s}_i}\int_{\mathbf{u}_i}f(s,u)\log\frac{f(u|s)}{f(s,u)}dsdu \tag{24}$$

$$H(\mathbf{u}_i,\mathbf{s}_i)=-\int_{\mathbf{s}_i}\int_{\mathbf{u}_i}f(s,u)\log f(u,s)dsdu \tag{25}$$

The above complex integrals for mutual information are difficult to calculate analytically. Therefore, a numerical optimization method termed as Nelder–Mead algorithm is instead adopted to solve this problem through nonconstraint nonlinear heuristic optimization procedure.[54] It should be noted that the normalization step of $\mathbf{w}_i$ and $\mathbf{c}_i$ are added in the numerical iterations to handle the constraints in Eq. 21. Moreover, the objective function in the mutual information-based optimization problem may have strong nonlinearity and multipeak feature, which can potentially lead to local optimal solution instead of global optimum. To overcome this issue, the multistart optimization strategy is used.

After the extraction of the weighting vectors $\mathbf{w}_i$ and $\mathbf{c}_i$, the score vectors $\mathbf{s}_i$ and $\mathbf{u}_i$ can be computed as follows

$$\mathbf{s}_i=\mathbf{X}_i\mathbf{w}_i \tag{26}$$



x_2
x_1
$p(s_1)$
s_1
y_2
y_1
$p(u_1)$
u_1
$p(u_1 | s_1)$
ambiguity in u_1 given s_1
$= H(u_1 | s_1) = \int p(u_1, s_1) \log p(u_1 | s_1) du_1 ds_1$
$\longrightarrow$ minimized
Information in u_1
$= H(u_1) = \int p(u_1) \log p(u_1) du_1$
$\longrightarrow$ maximized
Search directions maximizing mutual information
$= H(u_1) - H(u_1 | s_1)$

Figure 1. Illustration of the proposed QNGLSP method.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

$$\mathbf{u}_i = \mathbf{Y}_i \mathbf{c}_i \tag{27}$$

Then, loading vectors $\mathbf{p}_i$ and $\mathbf{q}_i$ are estimated as

$$\mathbf{p}_i = \mathbf{X}_i^T \mathbf{s}_i / \mathbf{s}_i^T \mathbf{s}_i \tag{28}$$

$$\mathbf{q}_i = \mathbf{Y}_i^T \mathbf{s}_i / \mathbf{s}_i^T \mathbf{s}_i \tag{29}$$

With the obtained loading vectors, the matrices $\mathbf{X}$ and $\mathbf{Y}$ can be deflated as follows

$$\mathbf{X}_{i+1} = \mathbf{X}_i - \mathbf{s}_i \mathbf{p}_i^T \text{ and } \mathbf{Y}_{i+1} = \mathbf{Y}_i - \mathbf{s}_i \mathbf{q}_i^T \tag{30}$$

However, $\mathbf{w}_i$ does not relate $\mathbf{s}_i$ to the original input data matrix $\mathbf{X}$ directly. Thus, a decomposition matrix $\mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_d]$ is defined below

$$\mathbf{r}_1 = \mathbf{w}_1 \tag{31}$$

and

$$\mathbf{r}_i = \prod_{j=1}^{i-1} \left(\mathbf{I}_m - \mathbf{w}_j \mathbf{p}_j^T \right) \mathbf{w}_i \;\; i \geq 2 \tag{32}$$

Then, score matrix $\mathbf{S}$ can be computed from original input matrix $\mathbf{X}$ as follows

$$\mathbf{S} = \mathbf{X}\mathbf{R} \tag{33}$$

The searching strategy of latent directions in the proposed QNGLSP method is illustrated in Figure 1. It can be observed that both the input and output data are projected onto the first feature directions within the input and output spaces to obtain the scores $\mathbf{s}_1$ and $\mathbf{u}_1$. Both directions are searched in such a way that the marginal entropy $H(\mathbf{u}_1)$ that equals the amount of information in $\mathbf{u}_1$ is maximized while the conditional entropy that equals the amount of ambiguity in $\mathbf{u}_1$ given $\mathbf{s}_1$ is minimized. Equivalently, the mutual information $I(\mathbf{s}_1; \mathbf{u}_1)$ between the score vectors $\mathbf{s}_1$ and $\mathbf{u}_1$ is maximized. The remaining score vectors can be estimated in the same fashion through iterative procedure.

After all the loading and score vectors are obtained, it is necessary to sort the two sets of latent directions corresponding to the input and output spaces, respectively. The marginal entropies of score vectors are used to rearrange the column vectors of the score and decomposition matrices $\mathbf{S}$ and $\mathbf{R}$. With all the sorted latent variables, the number of components $\mathbf{p}_i$ and $\mathbf{q}_i$ for concurrent subspace projections needs to be selected to achieve the best monitoring performance. If the numbers of components are too small, the projected subspaces do not contain sufficient non-Gaussian features for quality relevant fault detection. On the contrary, if too many components are chosen, then the formed subspaces may include irrelevant or redundant information that can degrade the sensitivity of monitoring statistics to faults. In QNGLSP method, the numbers of latent variables d is chosen so that the first d column vectors of the full score and decomposition matrices $\mathbf{S}=[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m]$ and $\mathbf{R}=[\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_m]$ satisfy the following marginal entropy-based criteria

$$\frac{\sum_{i=1}^{d} H(\mathbf{s}_i)}{\sum_{i=1}^{m} H(\mathbf{s}_i)} > \epsilon \tag{34}$$

and

$$\frac{\sum_{i=1}^{d} H(\mathbf{r}_i)}{\sum_{i=1}^{m} H(\mathbf{r}_i)} > \epsilon \tag{35}$$

where ϵ is the predefined threshold value and set to 0.95 in this work. Thus, the loading matrices $\mathbf{P}$ and $\mathbf{Q}$, score matrix $\mathbf{S}$, and decomposition matrix $\mathbf{R}$ consisting of d latent variables can be extracted as

$$\mathbf{P}=[\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_d] \in \Re^{m\times d} \tag{36}$$

$$\mathbf{Q}=[\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_d] \in \Re^{k\times d} \tag{37}$$

$$\mathbf{R}=[\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_d] \in \Re^{m\times d} \tag{38}$$

$$\mathbf{S}=[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_d] \in \Re^{n\times d} \tag{39}$$

Given a new test sample vector $\mathbf{x}$, the corresponding score, prediction, and residual vectors are computed as follows

$$\text{Score} : \mathbf{s}=\mathbf{x}\mathbf{R} \tag{40}$$

$$\text{Prediction} : \hat{\mathbf{x}}=\mathbf{x}\mathbf{R}\mathbf{P}^T \tag{41}$$

$$\text{Residual} : \mathbf{e}=\mathbf{x}\left(\mathbf{I}-\mathbf{R}\mathbf{P}^T\right) \tag{42}$$

In multivariate statistical process monitoring, two types of statistics are widely used for fault detection. One is the D statistic for monitoring the systematic part of process variations, whereas the other is the Q statistic for monitoring the residual part of process variations. As described in "Review of PLS and ICA-Based Process Monitoring Methods," section PLS-based fault detection methods use T^2 and SPE indices, whereas ICA-based methods adopt I^2 and SPE statistics. In QNGLSP-based monitoring method, I^2 and SPE indices are proposed for quality relevant fault detection as follows

$$I^2=\mathbf{s}\mathbf{L}^{-1}\mathbf{s}^T \tag{43}$$

$$\begin{aligned}\text{SPE} &=\mathbf{e}\mathbf{e}^T\\ &=\mathbf{x}\left(\mathbf{I}-\mathbf{R}\mathbf{P}^T\right)\left(\mathbf{I}-\mathbf{P}\mathbf{R}^T\right)\mathbf{x}^T\end{aligned} \tag{44}$$

where $\mathbf{L}$ is the diagonal matrix with the variances of different column vectors of $\mathbf{S}$ being the diagonal entries.

As it is assumed that the latent variables may follow non-Gaussian distribution, the control limits of the proposed indices are estimated from kernel density estimation strategy.[51] Let $(D_1, D_2, \ldots, D_n)$ be a set of observations from an unknown probability density function f. Then f can be estimated by kernel density estimator as follows

Table 1. Step-by-Step Procedure of the Proposed Quality Relevant Non-Gaussian Latent Subspace Projection Method

```
(1)  Form X and Y by filling missing entries with zeros and then
     scale X and Y to zero mean and unit variance.
(2)  Set counter $i \Leftarrow 1$
(3)  Set $\mathbf{X}_i \Leftarrow \mathbf{X}$ and $\mathbf{Y}_i \Leftarrow \mathbf{Y}$
(4)  Take random initial vectors $\mathbf{w}_i$ and $\mathbf{c}_i$ of unit norm
(5)  Solve the following constrained nonlinear optimization problem
     (Details of optimization algorithm are given in Table 2)
         maximize    $I(\mathbf{s}_i, \mathbf{u}_i)=I(\mathbf{X}_i\mathbf{w}_i, \mathbf{Y}_i\mathbf{c}_i)$
         subject to  $||\mathbf{w}||_i=1, ||\mathbf{c}||_i=1$
(6)  Calculate $\mathbf{s}_i$ and $\mathbf{u}_i$
         $\mathbf{s}_i = \mathbf{X}_i\mathbf{w}_i$
         $\mathbf{u}_i = \mathbf{Y}_i\mathbf{c}_i$
(7)  Calculate $\mathbf{p}_i$ and $\mathbf{q}_i$
         $\mathbf{p}_i = \mathbf{X}_i^T\mathbf{s}_i/\mathbf{s}_i^T\mathbf{s}_i$
         $\mathbf{q}_i = \mathbf{Y}_i^T\mathbf{s}_i/\mathbf{s}_i^T\mathbf{s}_i$
(8)  Residual deflation for the available entries only:
         $\mathbf{X}_{i+1} \Leftarrow \mathbf{X}_i-\mathbf{s}_i\mathbf{p}_i^T$
         $\mathbf{Y}_{i+1} \Leftarrow \mathbf{Y}_i-\mathbf{s}_i\mathbf{q}_i^T$
(9)  Set i $\Leftarrow i+1$ and return to (4) until $i=i_{\max}$
(10) Calculate the decomposition vector $\mathbf{r}_i$
         $\mathbf{r}_i=\mathbf{w}_1 (i=1)$
         $\mathbf{r}_i=\prod_{j=1}^{i-1}\left(\mathbf{I}_m-\mathbf{w}_j\mathbf{p}_j^T\right)\mathbf{w}_i \quad (i \geq 2)$
```

$$\hat{f}(D)=\frac{1}{nh}\sum_{i=1}^{n} K\left\{\frac{D-D_i}{h}\right\} \tag{45}$$

where D represents the I^2 or SPE index, h is the kernel window width, and K denotes the Gaussian kernel function

$$K(u)=\frac{1}{\sqrt{2\pi}}e^{\left(-\frac{1}{2}u^2\right)} \tag{46}$$

with u being an arbitrary data point. After a probability density function is estimated, the corresponding point with cumulative density function value at $1-\alpha$ is the control limit under the confidence level of $(1-\alpha)\times 100\%$.

The step-by-step numerical procedure of the proposed QNGLSP method is listed in Table 1, whereas the constrained nonlinear optimization algorithm for maximizing the mutual information between input and output latent variables is shown in Table 2.

Application Example

Tennessee *Eastman chemical process*

In this study, the Tennessee Eastman Chemical process is used to examine the effectiveness of the proposed quality

Table 2. Step-by-Step Procedure of Constrained Nonlinear Optimization Algorithm for Searching Non-Gaussian Latent Directions

(1)	Set multi-loop counter $l \Leftarrow 1$
(2)	Make a simplex which is a special polytope of $m+k+1$ vertices corresponding to
(3)	$\begin{bmatrix} \mathbf{w}_i^{(1)(l)} & \mathbf{w}_i^{(2)(l)} & \ldots & \mathbf{w}_i^{(v)(l)} & \ldots & \mathbf{w}_i^{(m+k+1)(l)} \\ \mathbf{c}_i^{(1)(l)} & \mathbf{c}_i^{(2)(l)} & \ldots & \mathbf{c}_i^{(v)(l)} & \ldots & \mathbf{c}_i^{(m+k+1)(l)} \end{bmatrix}$
	Normalize $\mathbf{w}_i^{(v)(l)}$ and $\mathbf{c}_i^{(v)(l)}$
(3)	$\mathbf{w}_i^{(v)(l)} = \mathbf{w}_i^{(v)(l)}/\|\|\mathbf{w}_i^{(v)(l)}\|\| \quad \forall v$ $\mathbf{c}_i^{(v)(l)} = \mathbf{c}_i^{(v)(l)}/\|\|\mathbf{c}_i^{(v)(l)}\|\| \quad \forall v$
(4)	Determine the updated vertex $v\prime$ and its steps $\Delta\mathbf{w}$, $\Delta\mathbf{c}$ by the Nelder–Mead method Update $\mathbf{w}_i^{(v\prime)(l)}$ and $\mathbf{c}_i^{(v\prime)(l)}$
(5)	$\mathbf{w}_i^{(v\prime)(l)} = \mathbf{w}_i^{(v\prime)(l)} + \Delta\mathbf{w}$ $\mathbf{c}_i^{(v\prime)(l)} = \mathbf{c}_i^{(v\prime)(l)} + \Delta\mathbf{c}$
(6)	Return to (3) until $\mathbf{w}_i^{(v)(l)}$ and $\mathbf{c}_i^{(v)(l)}$ are converged. If converged, set $\mathbf{w}_i^{(l)} = \mathbf{w}_i^{(v)(l)}$, $\mathbf{c}_i^{(l)} = \mathbf{c}_i^{(v)(l)}$ and go to (6) Set l $\Leftarrow$ l + 1 and return to (2) until $l = l_{\max}$
(7)	Choose the optimal $\mathbf{w}_i$ and $\mathbf{c}_i$ as follows: $l_{\text{opt}} = \text{argmax}_{\,l}\, I\left(\mathbf{X}_i\mathbf{w}_i^{(l)}, \mathbf{Y}_i\mathbf{c}_i^{(l)}\right)$ $\mathbf{w}_i = \mathbf{w}^{l_{\text{opt}}}$ $\mathbf{c}_i = \mathbf{c}^{l_{\text{opt}}}$

relevant non-Gaussian process monitoring method. The process flow diagram of the Tennessee Eastman Chemical process is shown in Figure 2 and this process includes five major unit operations, which are a chemical reactor, a product condenser, a vapor-liquid separator, a recycle compressor, and a product stripper.[55] This process produces two liquid products, G and H, along with a byproduct of F from four gaseous reactants A, C, D, and E. An inert B is fed into the chemical reactor where G and H are formed. There are total 41 measurement variables and 12 manipulated variables

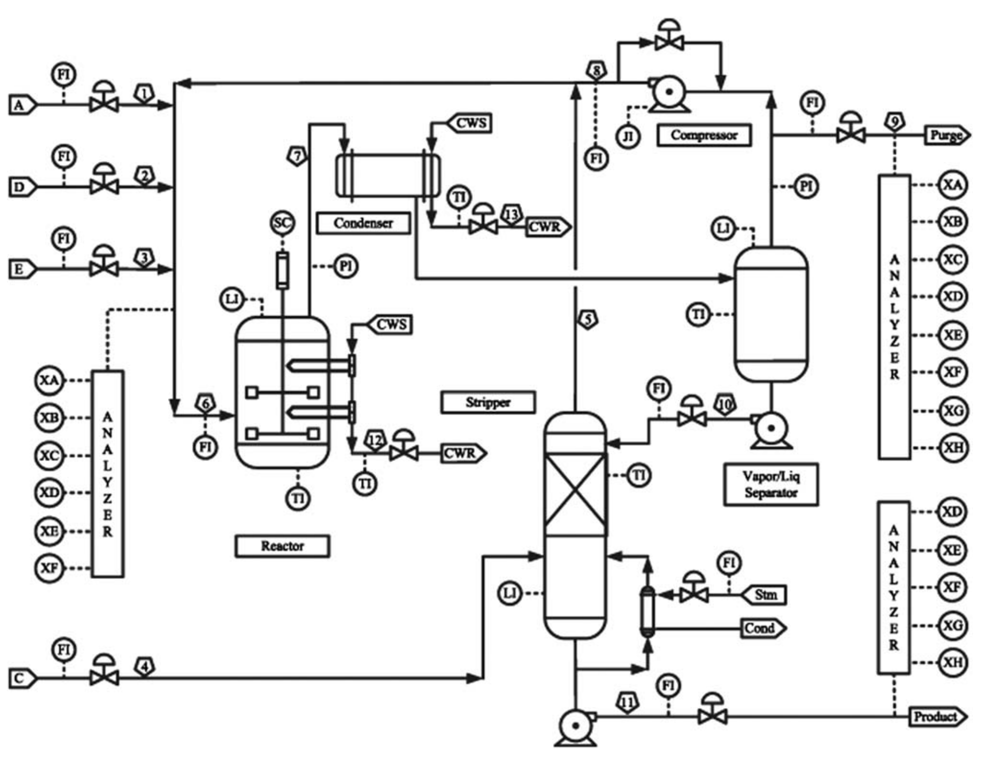


Figure 2. Process flow diagram of the Tennessee Eastman Chemical process.

Table 3. Input Variables of the Tennessee Eastman Chemical Process

Variable No.	Variable Description
1	A Feed (stream 1)
2	D feed (stream 2)
3	E feed (stream 3)
4	A and C feed (stream 4)
5	Recycle flow (stream 8)
6	Reactor feed RATE (stream 6)
7	Reactor pressure
8	Reactor level
9	Reactor temperature
10	Purge rate (stream 9)
11	Product Sep Temp
12	Product Sep level
13	Product Sep pressure
14	Product Sep underflow (stream 10)
15	Stripper level
16	Stripper pressure
17	Stripper underflow (stream 11)
18	Stripper Temp
19	Stripper steam flow
20	Compressor work
21	Reactor coolant Temp
22	Separator coolant Temp
23	D feed flow (stream 2)
24	E feed flow (stream 3)
25	A feed flow (stream 1)
26	A and C feed flow (stream 4)
27	Purge value (stream 9)
28	Separator pot liquid flow (stream 10)
29	Stripper liquid product flow (stream 11)
30	Reactor cooling water flow
31	Condenser cooling water flow

in the process. Moreover, there are 20 predefined abnormal operating events and six different operating conditions in the Tennessee Eastman Chemical process, as shown in Tables 5 and 6. The process involves a plant-wide decentralized control implementation with different feedback control loops.[56]

For process monitoring purpose, 22 continuous measurement variables and nine manipulated variables are selected as input variables, which are listed in Table 3. Meanwhile, as listed in Table 4, 19 composition variables that are measured through either off-line lab analysis or on-line analyzers are used as output quality variables in the process monitoring framework. The sampling time of both input and output variables are set to 0.25 h. Two subsets of training data with 1440 normal samples in each set are generated from operating modes 1 and 3, respectively. Then the combined normal data set of 2880 samples is used to build the QNGLSP model for process monitoring and fault detection. In this work, the normalized multivariate kurtosis of process data is used to quantitatively measure process non-Gaussianity. The value of normalized multivariate kurtosis of the training data set generated from different operating modes is 747, which is significantly larger than zero. Therefore, it can be inferred that the process data generated from two different operating modes follow a non-Gaussian probability distribution.

Furthermore, three test cases containing various types of predefined process faults are designed to compare the monitoring performance of the PLS and the proposed QNGLSP methods. It should be noted that the other existing techniques such as PCA, ICA, GMM, and supervised classification methods are not chosen for methodology comparison in the application example because all these monitoring methods do not include product quality variables as output variables in model development and data analysis while PLS and QNGLSP approaches take into account both process measurement and product quality variables concurrently. The detailed test scenarios are shown in Table 7 and all these

Table 4. Output Quality Variables of the Tennessee Eastman Chemical Process

Variable No.	Variable Description
1	Component to A to reactor
2	Component to B to reactor
3	Component to C to reactor
4	Component to D to reactor
5	Component to E to reactor
6	Component to F to reactor
7	Component A in purge
8	Component B in purge
9	Component C in purge
10	Component D in purge
11	Component E in purge
12	Component F in purge
13	Component G in purge
14	Component H in purge
15	Component D in product
16	Component E in product
17	Component F in product
18	Component G in product
19	Component H in product

Table 5. Predefined Faults of the Tennessee Eastman Chemical Process

Fault ID.	Fault Description
IDV(1)	Step in A/C feed ratio, B composition constant
IDV(2)	Step in B composition, A/C ratio constant
IDV(3)	Step in D feed temperature (stream 2)
IDV(4)	Step in reactor cooling water inlet temperature
IDV(5)	Step in condenser cooling water inlet temperature
IDV(6)	A feed loss (step change in stream 1)
IDV(7)	C header pressure loss (step change in stream 4)
IDV(8)	Random variation in A+C feed composition (stream 4)
IDV(9)	Random variation in D feed temperature (stream 2)
IDV(10)	Random variation in C feed temperature (stream 4)
IDV(11)	Random variation in reactor cooling water inlet temperature
IDV(12)	Random variation in condenser cooling water inlet temperature
IDV(13)	Slow drift in reaction kinetics
IDV(14)	Sticking reactor cooling water valve
IDV(15)	Sticking condenser cooling water valve
IDV(16)	Unknown disturbance
IDV(17)	Unknown disturbance
IDV(18)	Unknown disturbance
IDV(19)	Unknown disturbance
IDV(20)	Unknown disturbance

Table 6. Six Operation Modes of the Tennessee Eastman Chemical Process

Operating Mode	G/H Mass Ratio	Production Rate (stream 11)
1	50/50	7038 kg/h G and 7038 kg/h H
2	10/90	1408 kg/h G and 12669 kg/h H
3	90/10	10,000 kg/h G and 1111 kg/h H
4	50/50	Maximum
5	10/90	Maximum
6	90/10	Maximum

Table 7. Three Test Cases of the Tennessee Eastman Chemical Process

Case No.	Test Scenario
1	Normal operation from the 1-st to 40-th samples C header pressure loss from the 41-st to 100-th samples Normal operation from the 101-st to 160-th samples Random variation in condenser cooling water inlet temperature from the 161-st to 200-th samples
2	Normal operation from the 1-st to 55-th samples Step change in condenser cooling water inlet temperature from the 56-th to 100-th samples Normal operation from the 101-st to 130-th samples Sticking reactor cooling water valve from the 131-st to 200-th samples
3	Normal operation from the 1-st to 70-th samples C header pressure loss from the 71-st to 100-th samples Normal operation from the 101-st to 140-th samples Random variation in condenser cooling water inlet temperature from the 141-st to 200-th samples Normal operation from the 201-st to 250-th samples Sticking valve of reactor cooling water flow from the 251-st to 300-th samples

cases include both normal and different type of faulty operations. In the first test case, the process begins with normal operating condition from the first through the 40-th samples. Then the fault of C header pressure loss occurs at the 41-st sample and remains until the 100-th sample, after which the process returns to normal operation. From the 161-st sample, the fault of increased random variation in condenser cooling water inlet temperature takes place with the duration of 40 samples. For the second test scenario, the process fault of step change in condenser cooling water inlet temperature happens after the initial period of normal operation and lasts 45 samples. Then the process operation is back to normal condition and remains for 30 samples before a new fault of sticking valve of reactor cooling water flow occurs. In the more complex third test case, the faulty operation of C header pressure loss happens from the 71-st through 100-th samples. Then the operational status returns to normal before the second fault of increased random variations of condenser cooling water inlet temperature takes place from the 141-st through 200-th samples. After the process is operated under normal condition for another 50 samples, the third fault of sticking valve of reactor cooling water flow occurs between the 251-st and 300-th samples. The proposed I^2 and SPE indices in the QNGLSP method are compared to the T^2 and SPE statistics of the PLS method for fault detection in the three test cases.

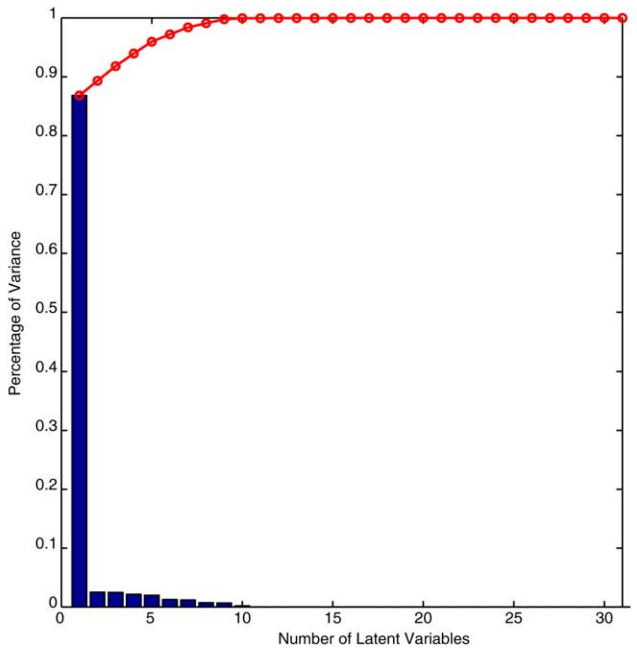


Figure 3. Trend plot of the individual percentage (bar) and cumulative percentage (solid line) of variance of PLS-based latent variables.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

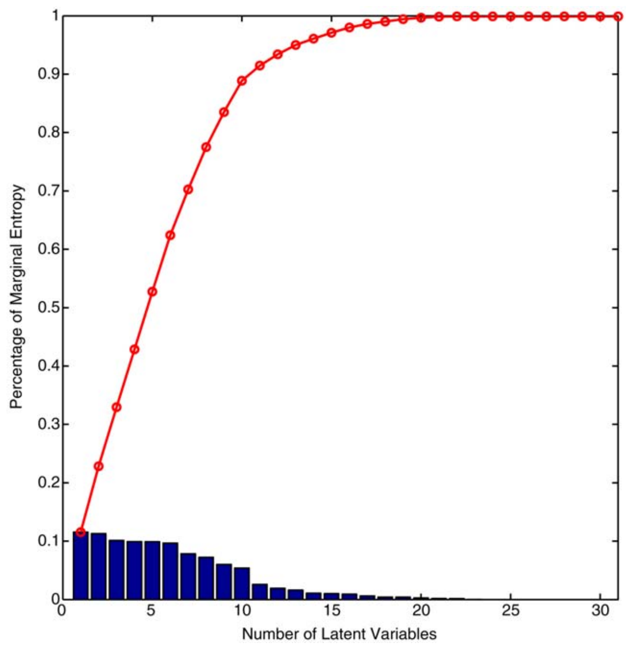


Figure 4. Trend plot of the individual percentage (bar) and cumulative percentage (solid line) of marginal entropy of QNGLSP-based latent variables.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Comparison of *process monitoring results*

After the QNGLSP model is built from normal training data with both input and output variables, it is important to select the number of non-Gaussian latent variables for input and output subspace projections. In the proposed approach, marginal entropy-based strategy is utilized to determine the best number of latent directions to be retained. For the training set, the individual and cumulative percentages of the marginal entropy of the sorted latent variables vs. the number of variables are shown Figure 4. Based on the proposed selection criteria, total 13 latent variables that contain over 95% of the marginal entropy is chosen. As shown in Figure 3, total 5 variables that cover over 95% of the variance are selected in PLS model. Figures 5 and 6 show the scatter plots of process data along the first vs. the second and the first vs. the third latent variables in the PLS and QNGLSP methods. It can be readily observed that the process data do not follow Gaussian distribution. In addition, as the training

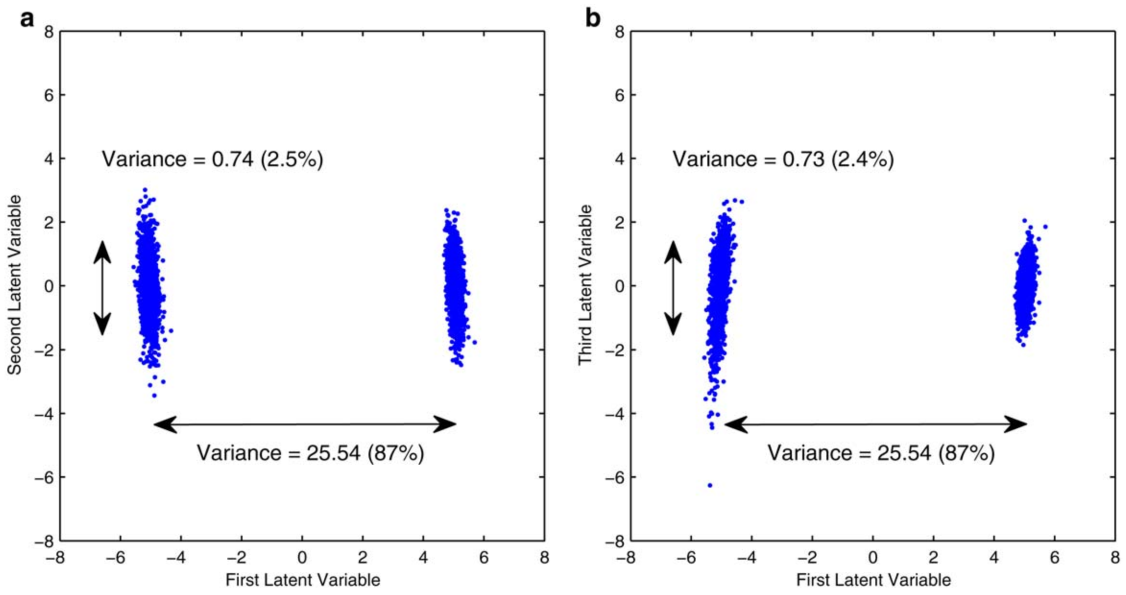


Figure 5. Scatter plots of process data along (a) the first vs. the second and (b) the first vs. the third latent variables of the PLS method.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

data are generated from multimode process operation, most of the variations of the training data are due to different operating modes and captured by the first latent variable. Therefore, the variances of the remaining latent variables become significantly smaller than that of the first latent variable, as shown in Figures 3 and 5.

In the first test case, the normal operation of the plant is mixed with two different types of faults, which are C header pressure loss from the 41-st to 100-th samples and increased random variation in condenser cooling water inlet temperature from the 161-st to 200-th samples. The process monitoring results of the proposed QNGLSP method including I^2 and SPE indices and the PLS method including T^2 and SPE statistics are compared in Figure 7. Meanwhile, the fault detection rates and false alarms rates of different indices in the two methods are listed in Tables 8 and 9. It can be seen that the QNGLSP-based I^2 and SPE indices detect abnormal operating events with fairly high fault detection rates of over 99.0% while low false alarm rates of only 1.25% and 0.50%, respectively. In contrast, the PLS-based monitoring method does not result in satisfactory performance as the T^2 index can capture only 39.75% of faulty samples, although the

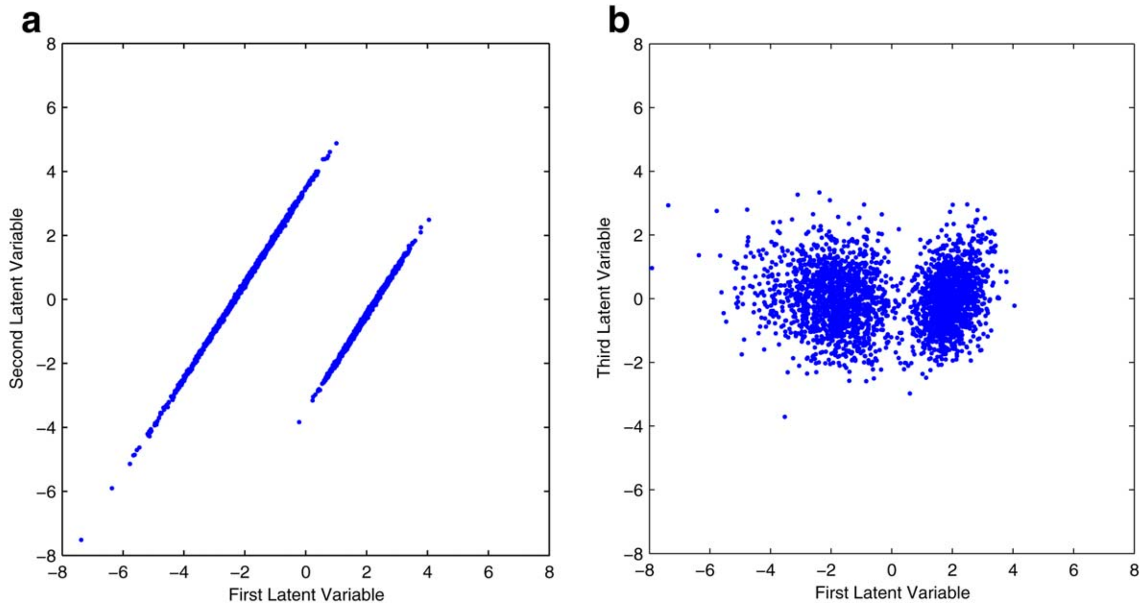


Figure 6. Scatter plots of process data along (a) the first vs. the second and (b) the first vs. the third latent variables of the proposed QNGLSP method.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

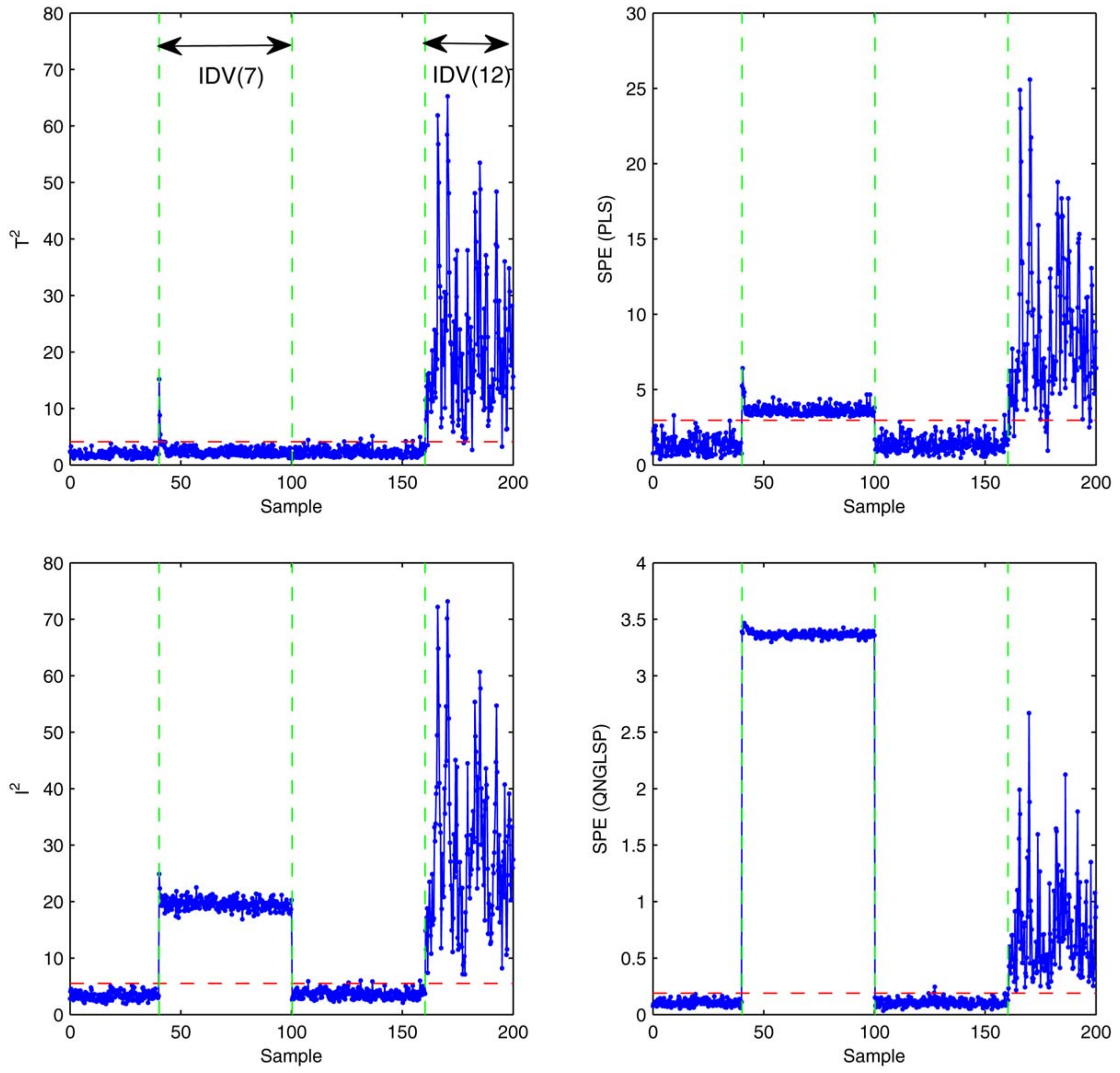


Figure 7. Monitoring results of PLS and QNGLSP methods in the first test case of the Tennessee Eastman Chemical process.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

SPE index leads to 97.75% fault detection rate. Such results reveal that the proposed QNGLSP method can well extract non-Gaussian process features from measurement and quality variables and, thus, the proposed I^2 and SPE indices are able to detect process faults accurately with minimum number of false alarms triggered. The specific comparison between PLS and QNGLSP monitoring results shows that the I^2 index has much higher fault detection accuracy than the T^2 index while the QNGLSP-based SPE statistic has a litter better fault detection rate than the PLS-based SPE index. As for the overall fault detection rates, both PLS and QNGLSP methods lead to the detection of over 99% of the abnormal operations with either T^2 or SPE index exceeds the corresponding control limit.

Table 8. Comparison of Fault Detection Rates (%) of Three Test Cases in the Tennessee Eastman Chemical Process

Method	Fault Detection Rate (%)					
	PLS				QNGLSP	
Number of Latent Variables	5 (95% of Variance)		8 (99% of Variance)		13 (95% of Entropy)	
Statistics	T^2	SPE	T^2	SPE	I^2	SPE
Case 1	39.75	97.75	41.25	87.50	99.75	99.75
Case 2	85.00	64.13	89.13	45.65	98.91	98.26
Case 3	64.64	83.57	71.79	78.39	98.57	96.79
Average	63.13	81.82	67.39	70.51	99.08	98.27

Table 9. Comparison of False Alarm Rates (%) of Three Test Cases in the Tennessee Eastman Chemical Process

Method	Fault Detection Rate (%)					
	PLS				QNGLSP	
Number of Latent Variables	5 (95% of Variance)		8 (99% of Variance)		13 (95% of Entropy)	
Statistics	T^2	SPE	T^2	SPE	I^2	SPE
Case 1	1.75	0.50	1.00	0.75	1.25	0.50
Case 2	0.59	0.59	0.29	1.17	1.17	0.59
Case 3	1.09	0.47	0.94	0.62	1.25	1.09
Average	1.14	0.52	0.74	0.85	1.01	0.73

In the second test case, the plant operation includes normal condition along with two different types of faults, which are a step change in condenser cooling water inlet temperature from the 56-th to 100-th samples and increased random variation in sticking valve of reactor cooling water flow from the 131-st to 200-th samples. The process monitoring results of PLS and QNGLSP methods are shown in Figure 8, and Tables 8 and 9. It can be seen that the QNGLSP-based I^2 and SPE indices are able to accurately alarm the faulty operations with the high fault detection rates of 98.91 and 98.26%, respectively. In comparison, the PLS-based T^2 and SPE statistics lead to the fault detection rates of only 85.00 and 64.13%, respectively. Therefore, the QNGLSP method has significantly stronger capability to capture different types of process faults than the PLS method. The main reason of the superior performance of QNGLSP method is due to its

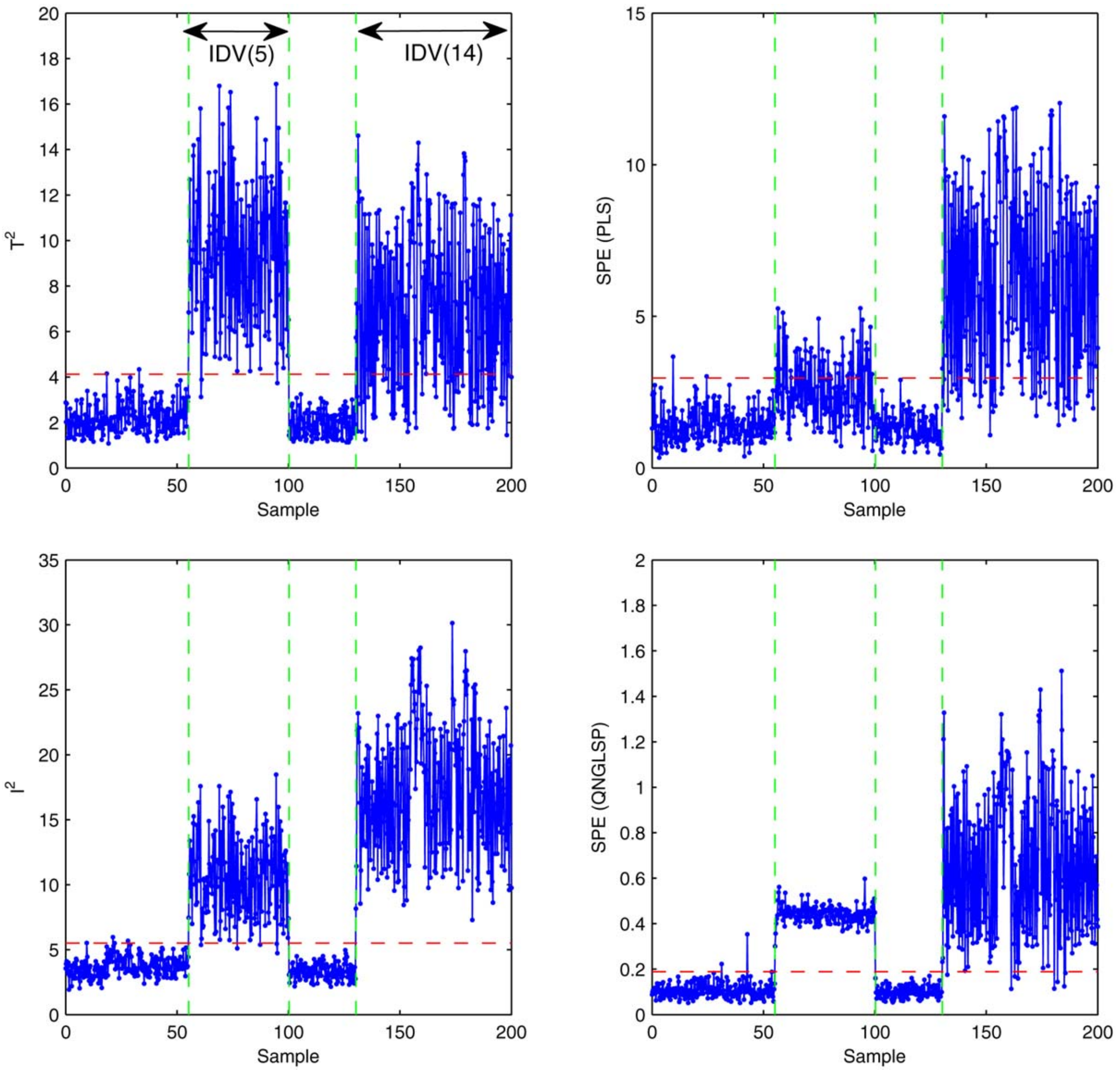


Figure 8. Monitoring results of PLS and QNGLSP methods in the second test case of the Tennessee Eastman Chemical process.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

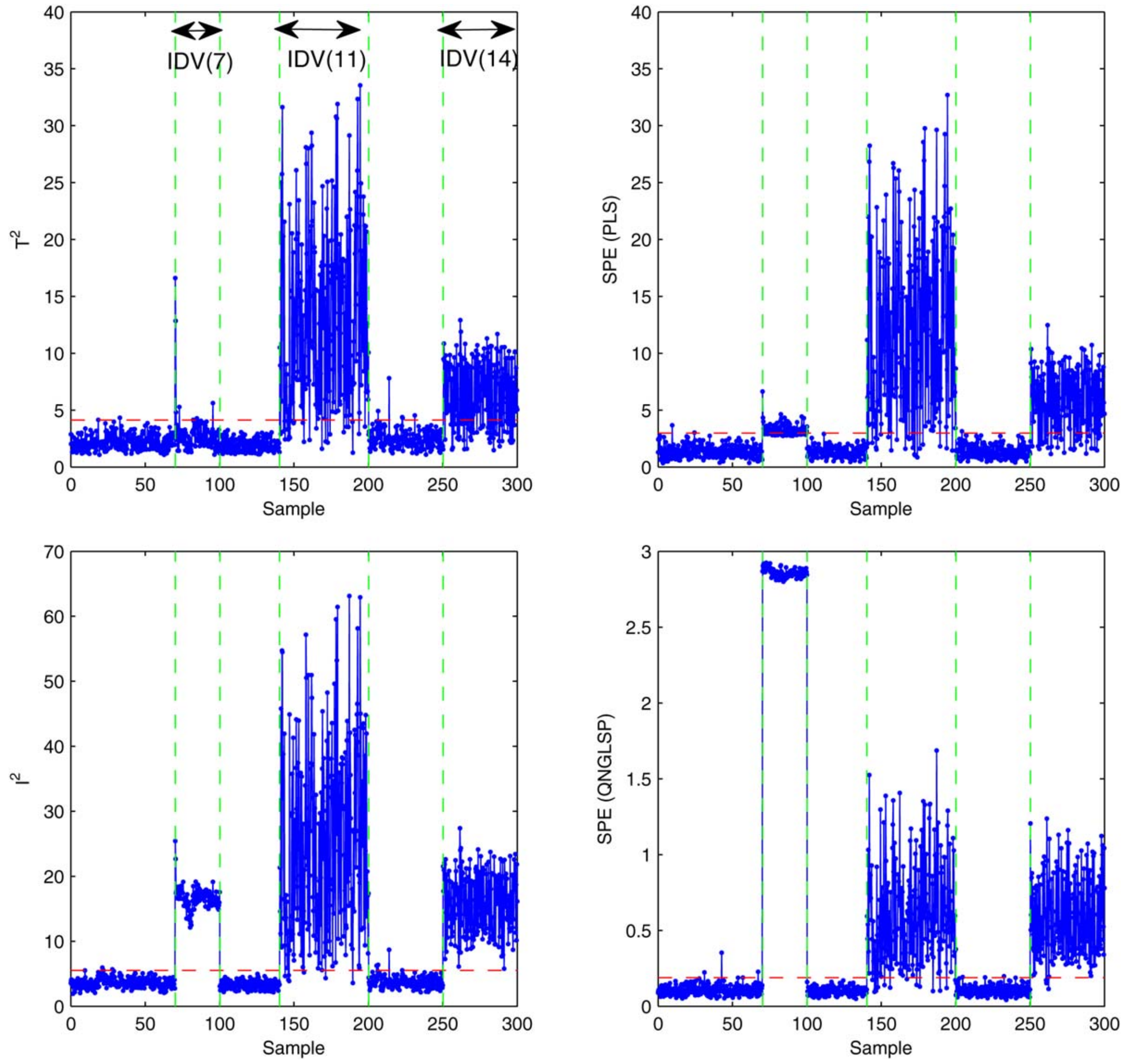


Figure 9. Monitoring results of PLS and QNGLSP methods in the third test case of the Tennessee Eastman Chemical process.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

mutual information rather than covariance-based objective function in searching for the optimal latent directions. In this way, the obtained latent subspace can efficiently extract and retain the non-Gaussian features for enhanced fault detection capacity. It should be noted that, compared to the first test case, the PLS-based T^2 statistic is more sensitive but the SPE index is less sensitive to the fault in this case. It implies that this fault affects the systematic part of process variations more than the residual part in the PLS model while it affects both the systematic and residual parts of process variations in the proposed QNGLSP model. In addition, the overall fault detection rate of the proposed QNGLSP method reaches 100.00% as either I^2 or SPE index exceeds its corresponding control limit, whereas the overall detection rate of PLS method is only 89.78%.

The last test case includes a complex operating scenario with three types of process faults that are mixed into normal operation. The abnormal operating events are C header pressure loss from the 71-st to the 100-th samples, increased random variation in condenser cooling water inlet temperature from the 141-st to 200-th samples and sticking valve of reactor cooling water flow from the 251-st to 300-th samples. The fault detection results of the various statistics from QNGLSP and PLS methods are shown in Figure 9. Meanwhile, the quantitative comparison of fault detection rates and false alarm rates are given in Tables 8 and 9. One can readily observe that the fault detection rate of T^2 statistic of PLS method is only 64.64%, which is lower than that of the QNGLSP-based I^2 index (98.57%). Meanwhile, the PLS-based SPE index also yields lower fault detection rate (83.57%) than that of the QNGLSP-based SPE statistic (96.79%). These comparisons verify that the proposed QNGLSP method has better monitoring performance and fault detection capability than the conventional PLS method. Furthermore, the overall fault detection rate of the PLS method is only 85.36%, whereas the proposed QNGLSP method leads to the overall detection rate of 99.11%, where either I^2 or SPE statistic exceeds its corresponding control limit.

In the above comparison, five latent variables that contain over 95% of the variance are selected in the PLS model. To investigate the effectiveness of an increase of the number of latent variables in PLS model, eight latent variables that contain over 99% of the variance are also selected and examined. The quantitative comparison of fault detection and false alarm rates are shown in Tables 8 and 9. It can be seen that the fault detection rates of T^2 and SPE statistics of PLS model that contains over 99% variance are still worse than those of the proposed QNGLSP method in all test cases.

Conclusions

In this article, a new QNGLSP method is proposed for chemical process monitoring and fault detection by taking into account both process measurement and quality variables. To capture the non-Gaussian features and relationships between input and output variables, the proposed QNGLSP method adopts the high-order statistics of mutual information instead of the second-order statistics of covariance for searching the latent directions within input and output spaces, respectively. Then, the multistart optimization procedure is designed to identify the optimal feature directions iteratively with nonlinear multipeak function handling capability. Furthermore, I^2 and SPE indices are developed to detect process faults within non-Gaussian latent variable and residual subspaces. Different from the PCA or ICA-based monitoring techniques, the presented QNGLSP method has the inherent model structure of combining process measurement and quality variables. Meanwhile, this new approach relies on mutual information-based objective function and, thus, can effectively extract the non-Gaussian features in latent subspaces, which cannot be achieved in the PLS-based monitoring method.

The proposed QNGLSP method is compared to the conventional PLS method in the three test cases of the Tennessee Eastman Chemical process with different operating modes. The monitoring results demonstrate that the QNGLSP-based I^2 and SPE indices are superior to the PLS-based T^2 and SPE indices in terms of more accurate fault detection. Future research will focus on extending the new QNGLSP approach for nonlinear batch or semibatch processes monitoring as well as taking into account the dynamic nature of process data.

Literature Cited

1. Piovoso M, Hoo K. Multivariate statistics for process control. *IEEE Control Syst Mag.* 2002;22:8–9.
2. Venkatasubramanian V, Rengaswamy R, Yin K, Kavuri SN. A review of process fault detection and diagnosis: part I: quantitative model-based methods. *Comput Chem Eng.* 2003;27:293–311.
3. Yu J. Online quality prediction of nonlinear and non-Gaussian chemical processes with shifting dynamics using finite mixture model based Gaussian process regression approach. *Chem Eng Sci.* 2012; 82:22–30.
4. Gertler J. Survey of model-based failure detection and isolation in complex plants. *IEEE Control Syst Mag.* 1988;12:3–11.
5. Pons M, Rajab A, Flaus J, Engasser J, Cheruy A. Comparison of estimation methods for biotechnological processes. *Chem Eng Sci.* 1988;8:1909–1914.
6. Bastin G, Dochain D. On-Line Estimation and Adaptive Control of Bioreactors. Amsterdam, Netherlands: Elsevier, 1990.
7. Doyle F. Nonlinear inferential control for process applications. *J Process Control.* 1998;8:339–353.
8. MacGregor JF, Kourti T. Statistical process control of multivariate processes. *Control Eng Practice.* 1995;3(3):403–414.
9. Nomikos P, MacGregor JF. Monitoring of batch processes using multi-way principal component analysis. *AIChE J.* 1994;40:1361–1375.
10. MacGregor JF, Jaeckle C, Kiparissides C, Koutoudi M. Process monitoring and diagnosis by multiblock PLS methods. *AIChE J.* 1994;40(5):826–838.
11. Kano M. Comparison of statistical process monitoring methods: application to the Eastman challenge problem. *Comput Chem Eng.* 2000;24:175–181.
12. Chiang L, Russell E, Braatz R. Fault Detection and Diagnosis in Industrial Systems. Advanced Textbooks in Control and Signal Processing. London, Great Britain: Springer-Verlag, 2001.
13. Ündey C, Ertunç S, Çinar A. Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis. *Ind Eng Chem Res.* 2003;42:4645–4658.
14. Qin SJ. Statistical process monitoring: basics and beyond. *J Chemom.* 2003;17:480–502.
15. Cinar A, Palazoglu A, Kayihan F. *Chemical Process Performance Evaluation.* Boca Raton: CRC Press, 2007.
16. Mejdell T, Skogestad S. Estimation of distillation compositions from multiple temperature measurements using partial least squares regression. *Ind Eng Chem Res.* 1991;30(12):2543–2555.
17. Kresta J, Marlin T, MacGregor J. Development of inferential process models using PLS. *Comput Chem Eng.* 1994;18:597–611.
18. Kosanovich K, Dahl K, Piovoso M. Improved process understanding using multiway principal component analysis. *Ind Eng Chem Res.* 1996;35:138–146.
19. Kano M, Miyazaki K, Hasebe S, Hashimoto I. Inferential control system of distillation compositions using dynamic partial least squares regression. *J Process Control.* 2000;10:157–166.
20. Lennox B, Montague G, Hiden H, Kornfeld G, Goulding P. Process monitoring of an industrial fed-batch fermentation. *Biotechnol Bioeng.* 2001;74:125–135.
21. Zhang H, Lennox B. Integrated condition monitoring and control of fed-batch fermentation processes. *J Process Control.* 2004;14:41–50.
22. Zamprogna E, Barolo M, Seborg D. Optimal selection of soft sensor inputs for batch distillation columns using principal component analysis. *J Process Control.* 2005;15:39–52.
23. Zhao SJ, Zhang J, Xu YM. Performance monitoring of processes with multiple operating modes through multiple PLS models. *J Process Control.* 2006;16:763–772.
24. Lin B, Recke B, Knudsen J, Jorgensen S. A systematic approach for soft sensor development. *Comput Chem Eng.* 2007;31:419–425.
25. Ku W, Storer R, Georgakis C. Disturbance detection and isolation by dynamic principal component analysis. *Chemom Intell Lab Syst.* 1995;30:179–196.
26. Bakshi BR. Multiscale PCA with application to multivariate statistical process monitoring. *AIChE J.* 1998;44(7):1596–1610.
27. Zhou D, Li G, Qin S. Total projection to latent structures for process monitoring. *AIChE J.* 2010;56(1):168–178.
28. Martin EB, Morris AJ. Non-parametric confidence bounds for process performance monitoring charts. *J Process Control.* 1996;6:349–358.
29. Hwang DH, Han C. Real-time monitoring for a process with multiple operating modes. *Control Eng Practice.* 1999;7:891–902.
30. Kano M, Nagao K, Hasebe S, Hashimoto I, Ohno H. Statistical process monitoring based on dissimilarity of process data. *AIChE J.* 2002;48(6):1231–1240.
31. Albazzaz H, Wang XZ. Statistical process control charts for batch operations based on independent component analysis. *Ind Eng Chem Res.* 2004;43:6731–6741.
32. Yan W, Shao H, Wang X. Soft sensing modeling based on support vector machine and Bayesian model selection. *Comput Chem Eng.* 2004;28:1489–1498.
33. Chen A, Song Z, Li P. Soft sensor modeling based on DICA-SVR. *Lect Notes Comput Sci.* 2005;3644:868–877.
34. Desai K, Badhe Y, Tambe S, Kulkarni B. Soft-sensor development for fed-batch bioreactors using support vector regression. *Biochem Eng J.* 2006;27(3):225–239.
35. Ge Z, Song Z. Process monitoring based on independent component analysis-principal component analysis (ICA-PCA) and similarity factors. *Ind Eng Chem Res.* 2007;46:2054–2063.
36. Kaneko H, Arakawa M, Funatsu K. Development of a new soft sensor method using independent component analysis and partial least squares. *AIChE J.* 2009;55:87–98.
37. Yu J. A nonlinear kernel Gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes. *Chem Eng Sci.* 2012;68:506–519.

38. Lee JM, Yoo CK, Lee IB. Statistical process monitoring with independent component analysis. *J Process Control.* 2004;14:467–485.
39. Rashid M, Yu J. A new dissimilarity method integrating multidimensional mutual information and independent component analysis for non-Gaussian dynamic process monitoring. *Chemom Intell Lab Syst.* 2012;115:44–58.
40. Yu J, Qin SJ. Multimode process monitoring with Bayesian inference-based finite Gaussian mixture models. *AIChE J.* 2008;54: 1811–1829.
41. Yu J. A particle filter driven dynamic Gaussian mixture model approach for complex process monitoring and fault diagnosis. *J Process Control.* 2012;22(4):778–788.
42. Zhu Z, Song Z, Palazoglu A. Process pattern construction and multimode monitoring. *J Process Control.* 2012;22:247–262.
43. Chiang L. Fault diagnosis based on Fisher discriminant analysis and support vector machines. *Comput Chem Eng.* 2004;28(8): 1389–1401.
44. Yu J. A Bayesian inference based two-stage support vector regression framework for soft sensor development in batch bioprocesses. *Comput Chem Eng.* 2012;41:134–144.
45. Yu J. Localized Fisher discriminant analysis based complex process monitoring. *AIChE J.* 2011;57:1817–1828.
46. Yu J. A support vector clustering-based probabilistic method for unsupervised fault detection and classification of complex chemical processes using unlabeled data. *AIChE J.* 2012;59(2):407–419.
47. Geladi P, Kowalski BR. Partial least-squares regresion: a tutorial. *Anal Chim Acta.* 1986;185:1–17.
48. Hyvarinen A, Karhunen J, Oja E. *Independent Component Analysis.* New York: John Wiley & Sons, 2001.
49. Hyvrinen A. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans Neural Netw.* 1999;10(3):626–634.
50. Hyvrinen A. Independent component analysis: algorithms and applications. *IEEE Trans Neural Netw.* 2000;13:411–430.
51. Bishop C. *Pattern Recognition and Machine Learning.* New York: Springer-Verlag, 2006.
52. Hoskuldsson A. PLS regression methods. *J Chemom.* 1988;2:211–228.
53. Hulle MV. Edgeworth approximation of multivariate differential entropy. *Neural Comput.* 2005;17(9):1903–1910.
54. Nelder A, Mead R. A simplex method for function minimization. *Comput J.* 1965;7(4):308–313.
55. Downs JJ, Vogel EF. Plant-wide industrial process control problem. *Comput Chem Eng.* 1993;17:245–255.
56. Ricker NL. Decentralized control of the Tennessee Eastman challenge process. *J Process Control.* 1996;6:205–221.

Manuscript received Aug. 15, 2013, and revision received Sept. 15, 2013.

Chapter 3

Quality relevant nonlinear batch process performance monitoring using a kernel based multiway non-Gaussian latent subspace projection approach

Journal of Process Control 24 (2014) 57–71

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

Quality relevant nonlinear batch process performance monitoring using a kernel based multiway non-Gaussian latent subspace projection approach

Junichi Mori, Jie Yu*

Department of Chemical Engineering, McMaster University, Hamilton, Ontario, Canada L8S 4L7

A R T I C L E I N F O

Article history:
Received 3 September 2013
Accepted 28 October 2013

Keywords:
Nonlinear batch process
Multidimensional mutual information
Nonlinear kernel feature space
Non-Gaussian latent subspace projection
Quality relevant batch process monitoring

A B S T R A C T

Multiway kernel partial least squares method (MKPLS) has recently been developed for monitoring the operational performance of nonlinear batch or semi-batch processes. It has strong capability to handle batch trajectories and nonlinear process dynamics, which cannot be effectively dealt with by traditional multiway partial least squares (MPLS) technique. However, MKPLS method may not be effective in capturing significant non-Gaussian features of batch processes because only the second-order statistics instead of higher-order statistics are taken into account in the underlying model. On the other hand, multiway kernel independent component analysis (MKICA) has been proposed for nonlinear batch process monitoring and fault detection. Different from MKPLS, MKICA can extract not only nonlinear but also non-Gaussian features through maximizing the higher-order statistic of negentropy instead of second-order statistic of covariance within the high-dimensional kernel space. Nevertheless, MKICA based process monitoring approaches may not be well suited in many batch processes because only process measurement variables are utilized while quality variables are not considered in the multivariate models. In this paper, a novel multiway kernel based quality relevant non-Gaussian latent subspace projection (MKQNGLSP) approach is proposed in order to monitor the operational performance of batch processes with nonlinear and non-Gaussian dynamics by combining measurement and quality variables. First, both process measurement and quality variables are projected onto high-dimensional nonlinear kernel feature spaces, respectively. Then, the multidimensional latent directions within kernel feature subspaces corresponding to measurement and quality variables are concurrently searched for so that the maximized mutual information between the measurement and quality spaces is obtained. The I^2 and SPE monitoring indices within the extracted latent subspaces are further defined to capture batch process faults resulting in abnormal product quality. The proposed MKQNGLSP method is applied to a fed-batch penicillin fermentation process and the operational performance monitoring results demonstrate the superiority of the developed method as apposed to the MKPLS based process monitoring approach.

1. Introduction

Batch and semi-batch processes have been widely used to produce low-volume but high-value-added products in polymer, pharmaceutical, semiconductor, materials, food, biotechnology, and agricultural industries. In order to improve product quality, plant safety, energy efficiency, environmental sustainability and economic profit, effective process performance monitoring is becoming critically important and have received significant attention in academia and industry. During batch process operation, even small process variations may have substantial impact on the final product quality, yields, production efficiency, environmental sustainability, etc. While it is possible to detect defective products in batch or semi-batch processes by tracking final product quality, it cannot prevent the abnormal operating events and avoid the wasted batches. Therefore, it is highly desirable to conduct on-line monitoring throughout batch operation so as to detect and resolve process faults in early stage [50,9,49,43]. On the other hand, nowadays most industrial plants are fully instrumented with large number of sensors and analyzers, which enable data-driven process performance monitoring and diagnosis.

The traditional process monitoring, fault detection and diagnosis methods are based on first-principle process models. However, the reliability of monitoring approaches heavily depends on the accuracy of the mechanistic models, which require in-depth process knowledge and analysis. Moreover, it can be very tedious and time-consuming to develop mechanistic models of complex processes [33,3,13]. Alternately, multivariate statistical process

* Corresponding author. Tel.: +1 9055259140x27702; fax: +1 9055211350.
E-mail address: jieyu@mcmaster.ca (J. Yu).

0959-1524/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jprocont.2013.10.017

monitoring (MSPM) techniques have been widely applied to both continuous and batch processes with many successful applications. Typical MSPM approaches can capture variable correlations and then transform the original process measurement space into the latent-variable subspace based on historical operating data [32,30,35,8,31,38,34,39–41]. Two popular MSPM methods, principal component analysis (PCA) and partial least squares (PLS), have been intensively used to monitor and diagnose the performance of different types of processes. Both methods can handle variable collinearity and project process data onto lower-dimensional latent subspace that retains most of the operational information based on variance or covariance. Then the T^2 and SPE indices are utilized to isolate abnormal events from normal process operation [20,26,2,7,54]. Nevertheless, batch process data typically have three-dimensional structure while the regular PCA and PLS methods can only handle two-dimensional data matrices. Therefore, multiway principal component analysis (MPCA) and multiway partial least squares (MPLS) techniques are developed to handle three-dimensional batch process data [30,42,27,14]. Furthermore, the dynamic model-based process monitoring technique is proposed to characterize variable dynamics of batch processes [10]. This method filters process data through a multivariate autoregressive model before performing PCA so as to capture the dynamic behavior of process variables. However, batch and semi-batch processes are often characterized with strong nonlinearity so that MPCA/MPLS based process monitoring approaches may not be effective in detecting abnormal operations because both methods are essentially based on multivariate linear models [51,44].

In order to tackle process nonlinearity, kernel principal component analysis (KPCA) is developed as an extension of regular PCA method for nonlinear process monitoring and diagnosis [21,22,25,28]. The high-dimensional kernel feature space enables KPCA method with the capability of extracting nonlinear process characteristics. Furthermore, multiway kernel partial least squares (MKPLS) approach is developed for batch process monitoring by capturing the covariance structure between measurement and quality variables in high-dimensional kernel feature space [12]. Different from KPCA, KPLS has the inherent input–output model structure through combining process measurement and quality variables. Nevertheless, KPCA and KPLS techniques rely only on second-order statistics within kernel feature space and thus may not effectively extract all non-Gaussian features from batch processes that are characterized by higher-order statistics such as entropy and mutual information. Meanwhile, the validity of the statistical confidence limits of T^2 and SPE indices in PLS model relies on the assumption that the process data follow Gaussian distribution approximately. Such requirement may not be satisfied when the normal operating data are of significant non-Gaussianity.

Alternately, independent component analysis (ICA) method is employed to project multivariate process data onto latent subspace consisting of statistically independent components (IC). [18,1,23,6,11,15,17]. The basic idea of ICA is to maximize the higher-order statistics such as negentropy, which can not only de-correlate the process data but also reduce statistical dependencies among the extracted latent variables. Thus, ICA can effectively capture non-Gaussian process features [24]. Moreover, kernel independent component analysis (KICA) method is developed to deal with nonlinear dynamics for process monitoring [53]. First, principal components are computed from process data in high-dimensional kernel feature space by performing KPCA. Then, ICA is utilized to search for the latent directions within the kernel-principal-components subspace so as to identify process nonlinearity and non-Gaussianity. KICA can be integrated with multiway analysis to handle three-dimensional data set for batch process monitoring and fault detection [36]. In addition, the kernel independent scores from KICA model can be directly utilized as the inputs of support vector machine (SVM) in order to diagnosis process faults [52]. However, the above ICA based techniques may not be appropriate for quality-relevant batch process monitoring because the underlying ICA models do not incorporate quality variables as outputs. Thus, the detected abnormal operating events may not cause any product quality degradations during batch operation and can trigger many unnecessary fault alarms in the monitoring systems. More recently, multiway Gaussian mixture model (MGMM) is developed to characterize non-Gaussian features and monitor operational performance of multi-phase batch processes [48,55]. Gaussian mixture model consists of multiple Gaussian components corresponding to different operating phases throughout batch operation and can be applied to batch processes with multiple phases and shifting dynamics [46,47]. Furthermore, the Gaussian mixture model is extended to nonlinear kernel feature space for monitoring complex processes that involve nonlinear dynamics within each local operating mode or phase [45]. Nevertheless, the above GMM based process monitoring approaches still do not take into consideration product quality variables and thereby they may not specifically identify abnormal operating events that cause deteriorated product quality throughout batch operation.

In this study, a novel multiway kernel based quality relevant non-Gaussian latent subspace projection (MKQNGLSP) method is developed to monitor quality related operational performance of nonlinear batch processes. First, three-dimensional batch process data are unfolded into two-dimensional matrices through multiway analysis for handling batch trajectories. Then, kernel principal components of measurement and quality variables are extracted to form high-dimensional feature spaces, respectively. The non-Gaussian multidimensional latent directions corresponding to measurement and quality variables within kernel-principal-component subspaces are further estimated through maximizing mutual information between measurement and quality variables. Thus, the MKQNGLSP based SPE and I^2 indices along with the corresponding confidence limits are derived to capture non-Gaussian process abnormalities of nonlinear batch operation. Different from MKPLS, the presented MKQNGLSP approach is based upon the high-order statistic of mutual information instead of the second-order statistic of covariance so that the non-Gaussian relationships between measurement and quality variables can be well identified. Moreover, the MKQNGLSP model includes measurement and quality variables so that the quality relevant process monitoring can be conducted to detect faults with significant influences on final product quality.

The organization of the rest of the article is as follows. Section 2 briefly reviews the multiway KPCA (MKPCA), multiway KICA (MKICA), MKPLS based batch process monitoring techniques. Section 3 describes the proposed multiway kernel based quality relevant non-Gaussian latent subspace projection approach for monitoring nonlinear batch processes. In 4, the presented method is applied to the example of fed-batch penicillin fermentation process and the superiority of the MKQNGLSP method as apposed to MKPCA, MKICA and MKPLS approaches is demonstrated. Finally, the conclusions are provided in Section 5.

2. Preliminary

KPCA, KICA and KPLS methods can deal with the nonlinearly correlated multi-dimensional data through the kernel projection of measurement data onto the high-dimensional feature space. While the purpose of KPCA and KICA is to maximize the second-order statistic of variance and higher-order statistic of negentropy among the projected measurement variables, respectively, KPLS aims to obtain the maximized covariance between the projected measurement and quality variables. Given b batches, s sampling instants for each batch, v_m measurement variables, and v_q quality variables, batch process data can be expressed as three-dimensional matrices $\underline{X} \in \Re^{b \times v_m \times s}$ and $\underline{Y} \in \Re^{b \times v_q \times s}$ for measurement and quality variables, respectively. Since KPCA, KICA and KPLS can deal with two-dimensional data matrices only, the batch process data need to be unfolded into two-dimensional matrices $\mathbf{X} \in \Re^{n \times v_m}$ and $\mathbf{Y} \in \Re^{n \times v_q}$ with $n = sb$. Alternatively, the batch process data can also be unfolded into two-dimensional matrices $\mathbf{X}' \in \Re^{b \times v_m s}$ and $\mathbf{Y}' \in \Re^{b \times v_q s}$ [21]. For on-line monitoring, however, the alternate unfolding strategy requires that test data should be completed until the end of each batch and hence variable trajectory needs to be estimated. In addition, this method may suffer from computational difficulty because the number of batches, b, is typically smaller than the product of the number of samples and variables $v_m s$. Therefore, the unfolded matrices $\mathbf{X} \in \Re^{n \times v_m}$ and $\mathbf{Y} \in \Re^{n \times v_q}$ are employed for KPCA, KICA, KPLS and the proposed monitoring method in this study.

2.1. Multiway kernel PCA

$\mathbf{X}$ can be mapped into high-dimensional feature space $\mathcal{F}$ through a nonlinear function $\boldsymbol{\phi} \in \Re^m$ as follows

$$\boldsymbol{\phi} : \mathbf{x}_j \in \Re^{v_m} \rightarrow \boldsymbol{\phi}(\mathbf{x}_j) \in \mathcal{F} \tag{1}$$

where $\mathbf{x}_j$ is the jth row vector of $\mathbf{X}$. Then, the covariance matrix of the mapped data can be expressed as follows

$$\mathbf{C} = \frac{1}{n}\sum_{j}^{n} \boldsymbol{\phi}(\mathbf{x}_j)^T \boldsymbol{\phi}(\mathbf{x}_j) \tag{2}$$

where $\boldsymbol{\phi}(\mathbf{x}_j)$ is assumed to be mean-centered. Thus the principal components in the kernel feature space can be computed by finding eigenvectors of $\mathbf{C}$. Instead of solving eigenvalue problem directly, the following eigenvalue decomposition is applied

$$\mathrm{K}\alpha_i = \lambda_i n \boldsymbol{\alpha}_i \tag{3}$$

where $\boldsymbol{\alpha}_i = [\alpha_{i1} \;\; \alpha_{i2} \ldots \alpha_{in}]^T \in \Re^n$ is the orthogonal eigenvector corresponding to the ith largest positive eigenvalues λ_i and $\mathbf{K} \in \Re^{n \times n}$ is the kernel gram matrix defined as

$$\mathbf{K} = \boldsymbol{\Phi}(\mathbf{X})\boldsymbol{\Phi}(\mathbf{X})^T \tag{4}$$

with $\boldsymbol{\Phi}(\mathbf{X}) = [\boldsymbol{\phi}(\mathbf{x}_1)^T \;\; \boldsymbol{\phi}(\mathbf{x}_2)^T \ldots \boldsymbol{\phi}(\mathbf{x}_n)^T]^T \in \Re^{n \times m}$. Given a new sample vector $\mathbf{x}_{new}$, its corresponding score vector is computed as follows

$$\mathbf{t}_{new}^{(\mathrm{PCA})} = \mathbf{k}_{new}\mathbf{A}_{(\mathrm{PCA})} \tag{5}$$

where $\mathbf{A}_{(\mathrm{PCA})} = [\alpha_1 \;\; \alpha_2 \ldots \alpha_a] \in \Re^{n \times a}$ and $\mathbf{k}_{new} = \boldsymbol{\Phi}(\mathbf{x}_{new})\boldsymbol{\Phi}(\mathbf{X})^T \in \Re^a$ denotes the normalized kernel vector for the new sample. The following MKPCA based T^2 and SPE statistics are used as the measures of variations in the latent variable and residual subspace for process monitoring purpose

$$T^2_{(\mathrm{PCA})} = \mathbf{t}_{new}^{(\mathrm{PCA})}\left\{\frac{1}{n-1}\mathbf{T}^T_{(\mathrm{PCA})}\mathrm{T}_{(\mathrm{PCA})}\right\}^{-1}\{\mathbf{t}_{new}^{(\mathrm{PCA})}\}^T \tag{6}$$

$$\mathrm{SPE}_{(\mathrm{PCA})} = 1 - 2\mathbf{k}_{new}\mathbf{A}_{(\mathrm{PCA})}\{\mathbf{t}_{new}^{(\mathrm{PCA})}\}^T + \mathbf{t}_{new}^{(\mathrm{PCA})}\mathbf{A}^T_{(\mathrm{PCA})}\mathbf{K}\mathbf{A}_{(\mathrm{PCA})}\{\mathbf{t}_{new}^{(\mathrm{PCA})}\}^T \tag{7}$$

where $\mathbf{T}_{(\mathrm{PCA})} = [\mathbf{t}_1^{(\mathrm{PCA})} \;\; \mathbf{t}_2^{(\mathrm{PCA})} \ldots \mathbf{t}_a^{(\mathrm{PCA})}] \in \Re^{n \times a}$ and the confidence limits of $T^2_{(\mathrm{PCA})}$ and $\mathrm{SPE}_{(\mathrm{PCA})}$ can be estimated from F and χ^2 distribution, respectively.

2.2. Multiway kernel ICA

MKPCA is based on the second-order statistic of variance among the measurement variables. However, it does not take into account the higher-order statistics and hence the non-Gaussian process features may not be efficiently extracted. In order to deal with non-Gaussian processes, MKICA is developed for nonlinearly projecting multivariate process data into latent subspace of statistically independent components. KICA algorithm needs whitening preprocessing by means of using KPCA [37]. The whitening data in the feature space $\mathcal{F}$ can be obtained as follows

$$\mathbf{t}_{(\mathrm{ICA})} = \sqrt{n}\mathbf{k}_{new}\mathbf{A}_{(\mathrm{ICA})}\boldsymbol{\Lambda}^{-1} \tag{8}$$

where $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_a) \in \Re^{a \times a}$. $\mathbf{T}_{(\mathrm{ICA})} = [\mathbf{t}_1^{(\mathrm{ICA})} \;\; \mathbf{t}_2^{(\mathrm{ICA})} \ldots \mathbf{t}_a^{(\mathrm{ICA})}] \in \Re^{n \times a}$ is assumed to be generated as linear combinations of $d_{(\mathrm{ICA})}$ unknown independent components as follows

$$\mathbf{T}_{(\mathrm{ICA})} = \mathbf{S}\mathbf{A}^T_{(\mathrm{ICA})} \tag{9}$$

with $\mathbf{A}^{(\mathrm{ICA})} \in \Re^{d^{(\mathrm{ICA})} \times a}$ denoting the unknown mixing matrix and $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{d_{(\mathrm{ICA})}}] \in \Re^{n \times d_{(\mathrm{ICA})}}$ representing the independent component matrix. The purpose of KICA is to find a demixing matrix $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{d_{(\mathrm{ICA})}}] \in \Re^{a \times d^{(\mathrm{ICA})}}$ as follows

$$\mathbf{S} = \mathbf{T}_{(\mathrm{ICA})}\mathbf{W} \tag{10}$$

where the independent components $\mathbf{S}$ have the maximized statistical independency in terms of negentropy among each other [16]. Given a new sample vector $\mathbf{x}_{new}$, its corresponding independent component is expressed as

$$\mathbf{s}_{new}^{(\mathrm{ICA})} = \sqrt{n}\mathbf{k}_{new}\mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{W} \tag{11}$$

Further, the I^2 and SPE statistics can be defined for process monitoring as follows

$$I^2_{(\mathrm{ICA})} = \mathbf{s}_{new}^{(\mathrm{ICA})}\{\mathbf{s}_{new}^{(\mathrm{ICA})}\}^T \tag{12}$$

$$\mathrm{SPE}_{(\mathrm{ICA})} = \mathbf{k}_{new}\mathbf{A}\boldsymbol{\Lambda}^{-1}(\mathbf{I} - \mathbf{W}\mathbf{A}_{(\mathrm{ICA})}) \tag{13}$$

where the confidence limits for above two statistics can be estimated through kernel density estimation strategy [5].

2.3. Multiway kernel PLS

In MKICA method, independent components are calculated by using high-order statistics such as negentropy among different measurement variables. However, MKICA based process monitoring method may not be well suited because only the process measurement variables are incorporated in the MKICA model while the product quality variables are excluded. In order to take into account the nonlinear impact on output quality variables, MKPLS technique is developed. $\mathbf{X}$ can be mapped into high-dimensional feature space $\mathcal{F}$ through a nonlinear function $\boldsymbol{\phi} \in \Re^m$ as follows

$$\boldsymbol{\phi} : \mathbf{x}_j \in \Re^{v_m} \rightarrow \boldsymbol{\phi}(\mathbf{x}_j) \in \mathcal{F} \tag{14}$$

where $\mathbf{x}_j$ is the jth row vector of $\mathbf{X}$. The goal of KPLS is to construct the PLS model in kernel feature space as follows

$$\boldsymbol{\Phi}(\mathbf{X}) = \mathbf{T}\mathbf{P}^T + \mathbf{E} \tag{15}$$

$$\mathbf{Y} = \mathbf{T}\mathbf{Q}^T + \mathbf{F} \tag{16}$$

where $\boldsymbol{\Phi}(\mathbf{X}) = [\boldsymbol{\phi}(\mathbf{x}_1)^T, \boldsymbol{\phi}(\mathbf{x}_2)^T, \ldots, \boldsymbol{\phi}(\mathbf{x}_n)^T]^T \in \Re^{n \times m}$, $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_d] \in \Re^{n \times d}$ is the score matrix, $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_d] \in \Re^{m \times d}$ is the loading matrix of $\boldsymbol{\Phi}(\mathbf{X})$, $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_d] \in \Re^{v_q \times d}$ is the loading matrix of $\mathbf{Y}$, $\mathbf{E} \in \Re^{n \times m}$ is the residual matrix of

$\boldsymbol{\Phi}(\mathbf{X})$, and $\mathbf{F} \in \Re^{n \times v_q}$ is the residual matrix of $\mathbf{Y}$. The nonlinear iterative partial least squares (NIPALS) algorithm can be performed to determine the ith score vector $\mathbf{t}_i$ and $\mathbf{u}_i$ with the following optimization problem

$$\max \quad [cov(\mathbf{t}_i, \mathbf{u}_i)] = [cov(\boldsymbol{\Phi}(\mathbf{X}_i)\mathbf{w}_i, \mathbf{Y}_i\mathbf{q}_i)] \tag{17}$$

$$\text{s.t.} \quad ||\mathbf{t}_i|| = 1, ||\mathbf{u}_i|| = 1 \tag{18}$$

where $cov(\mathbf{t}_i, \mathbf{u}_i)$ represents the covariance matrix between the score vectors $\mathbf{t}_i \in \Re^n$ and $\mathbf{u}_i \in \Re^n$, and $\mathbf{w}_i \in \Re^m$ and $\mathbf{q}_i \in \Re^{v_m}$ denote weighting vectors. $\boldsymbol{\Phi}(\mathbf{X}_i)$ and $\mathbf{Y}_i$ are deflated matrices that can be calculated as follows

$$\boldsymbol{\Phi}(\mathbf{X}_{i+1}) = \boldsymbol{\Phi}(\mathbf{X}_i) - \mathbf{t}_i\mathbf{t}_i^T\boldsymbol{\Phi}(\mathbf{X}_i) \tag{19}$$

$$\mathbf{Y}_{i+1} = \mathbf{Y}_i - \mathbf{t}_i\mathbf{t}_i^T\mathbf{Y}_i \tag{20}$$

The calculation of the inner products in the high-dimensional feature space can be avoided by a kernel gram matrix $\mathbf{K}_i \in \Re^{n \times n}$ that is defined as follows

$$\mathbf{K}_i = \boldsymbol{\Phi}(\mathbf{X}_i)\boldsymbol{\Phi}(\mathbf{X}_i)^T \tag{21}$$

where the commonly used kernel function is the Gaussian kernel function. After the extraction of the score vectors, the matrix $\mathbf{K}_i$ and $\mathbf{Y}_i$ are deflated in each iteration. Given a new sample vector $\mathbf{x}_{new}$, the corresponding new score vector is computed as follows

$$\mathbf{t}_{new} = \mathbf{k}_{new}\mathbf{A}_{(\text{PLS})} \tag{22}$$

where $\mathbf{k}_{new} = \boldsymbol{\Phi}(\mathbf{x}_{new})\boldsymbol{\Phi}(\mathbf{X})^T \in \Re^n$ is a kernel vector for the new sample, $\mathbf{A}_{(\text{PLS})} = \mathbf{U}(\mathbf{T}^T\mathbf{K}\mathbf{U})^{-1}$ with $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_d] \in \Re^{n \times d}$ being a score matrix of $\mathbf{Y}$. The T^2 and SPE statistics can then be defined as follows

$$T^2_{(\text{PLS})} = \mathbf{t}_{new}\left\{\frac{1}{n-1}\mathbf{T}^T\mathbf{T}\right\}^{-1}\mathbf{t}_{new}^T \tag{23}$$

$$\text{SPE}_{(\text{PLS})} = 1 - 2\mathbf{k}_{new}\mathbf{A}_{(\text{PLS})}\mathbf{t}_{new}^T + \mathbf{t}_{new}\mathbf{A}_{(\text{PLS})}^T\mathbf{K}\mathbf{A}^{(\text{PLS})}\mathbf{t}_{new}^T \tag{24}$$

where the confidence limits of T^2 and SPE can be estimated from a χ^2 distribution approximately.

3. Quality relevant nonlinear batch process monitoring based on MKQNGLSP approach

Though MKPLS method can capture process nonlinearity from the measurement and quality variables, it relies on the second-order statistic of covariance only so that it may not effectively capture non-Gaussian process features. In this study, a novel multiway kernel based quality relevant non-Gaussian latent subspace projection method is developed to overcome the challenges of conventional MKPLS approach for nonlinear batch process monitoring. The basic idea of the presented method is to search for the latent directions in high-dimensional kernel feature spaces so that the statistical dependency in terms of multidimensional mutual information between measurement and quality variables is maximized. Compared to MKPLS, MKQNGLSP takes into account the higher-order statistic of mutual information and thus can effectively extract the non-Gaussian features from batch process data.

Since batch processes often have unequal durations across different batches, the synchronization of batch trajectories may be needed to solve the issue of batch-to-batch variations. In order to conduct batch trajectory alignment, the dynamic time warping (DTW) technique can be utilized [19]. Then the three-dimensional batch process data can be unfolded and scaled, as shown in Fig. 1. Let $\underline{\mathrm{X}} \in \Re^{b \times v_m \times s}$ and $\underline{\mathrm{Y}} \in \Re^{b \times v_q \times s}$ be the process measurement and quality data matrices with b batches, s sampling instants, v_m measurement variables, and v_q quality variables. Batch-wise unfolding is first performed to transform matrices $\underline{\mathrm{X}}$ and $\underline{\mathrm{Y}}$ into $\overline{\mathrm{X}} \in \Re^{b \times s v_m}$ and $\overline{\mathrm{Y}} \in \Re^{b \times s v_q}$. Then each column vectors of $\overline{\mathrm{X}}$ and $\overline{\mathrm{Y}}$ are normalized to zero-mean and unit-variance so that the mean trajectory of each batch can be eliminated. Further, the scaled matrices are converted into new matrices $\mathbf{X} \in \Re^{n \times v_m}$ and $\mathbf{Y} \in \Re^{n \times v_q}$ with $n = sb$ through variable-wise unfolding, where different sampling instants are stacked with various batches as row vectors.

After the above unfolded and scaled matrices are obtained, the next step is to find latent directions in high-dimensional kernel feature spaces so that the maximized mutual information between measurement and quality variables is obtained. In order to reduce the process nonlinearity, the measurement and quality variables are nonlinearly mapped into high-dimensional feature spaces. First, the measurement variables $\mathbf{X}$ can be mapped into high-dimensional feature space $\mathcal{F}$ through nonlinear mapping function $\boldsymbol{\phi} \in \Re^m$ as

$$\boldsymbol{\phi} : \mathbf{x}_j \in \Re^{v_m} \rightarrow \boldsymbol{\phi}(\mathbf{x}_j) \in \mathcal{F} \tag{25}$$

where $\mathbf{x}_j$ is the jth row vector of $\mathbf{X}$. The covariance matrix $\mathbf{C}$ in the kernel feature space can be expressed as

$$\mathbf{C} = \frac{1}{n}\sum_j^n \boldsymbol{\phi}(\mathbf{x}_j)^T\boldsymbol{\phi}(\mathbf{x}_j) \tag{26}$$

where $\boldsymbol{\phi}(\mathbf{x}_j)$ is assumed to be mean-centered. Then principal components in the kernel feature space can be computed by solving the following eigenvalue problem [5]

$$\mathbf{C}\mathbf{v}_i = \lambda_i\mathbf{v}_i \tag{27}$$

where $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ are the eigenvectors of $\mathbf{C} \in \Re^{m \times m}$ corresponding to the eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$. The eigenvalue λ_i satisfies

$$\frac{1}{n}\sum_j^n \boldsymbol{\phi}(\mathbf{x}_j)^T\{\boldsymbol{\phi}(\mathbf{x}_j)\mathbf{v}_i\} = \lambda_i\mathbf{v}_i \tag{28}$$

while the eigenvector $\mathbf{v}_i$ can be expressed as linear combination of $\boldsymbol{\phi}(\mathbf{x}_j)$ as follows

$$\mathbf{v}_i = \sum_j^n \alpha_{ij}\boldsymbol{\phi}(\mathbf{x}_j)^T \tag{29}$$

where α_{ij} is the linear coefficient. From Eqs. (28) and (29), $\boldsymbol{\alpha}_i = [\alpha_{i1}, \ldots, \alpha_{in}]^T \in \Re^n$ can be computed by solving the following eigenvalue decomposition

$$\mathrm{K}\alpha_i = \lambda_i n\boldsymbol{\alpha}_i \tag{30}$$

where $\mathbf{K}$ is kernel gram matrix defined as

$$[\mathbf{K}]_{ij} = \boldsymbol{\phi}(\mathbf{x}_i)\boldsymbol{\phi}^T(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) \tag{31}$$

and

$$\mathbf{K} = \boldsymbol{\Phi}(\mathbf{X})\boldsymbol{\Phi}(\mathbf{X})^T \tag{32}$$

where $K(\mathbf{x}_i, \mathbf{x}_j)$ is a kernel function and $\boldsymbol{\Phi}(\mathbf{X}) = [\boldsymbol{\phi}(\mathbf{x}_1)^T, \boldsymbol{\phi}(\mathbf{x}_2)^T, \ldots, \boldsymbol{\phi}(\mathbf{x}_n)^T]^T \in \Re^{n \times m}$. A commonly used kernel function is the Gaussian radial basis function defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{c}\right) \tag{33}$$

where c is the Gaussian kernel width and its value can be determined by cross-validation strategy. The use of kernel function can avoid the "dimensionality curse" due to the computations of all pairs of inner products of nonlinear mapping functions. In order

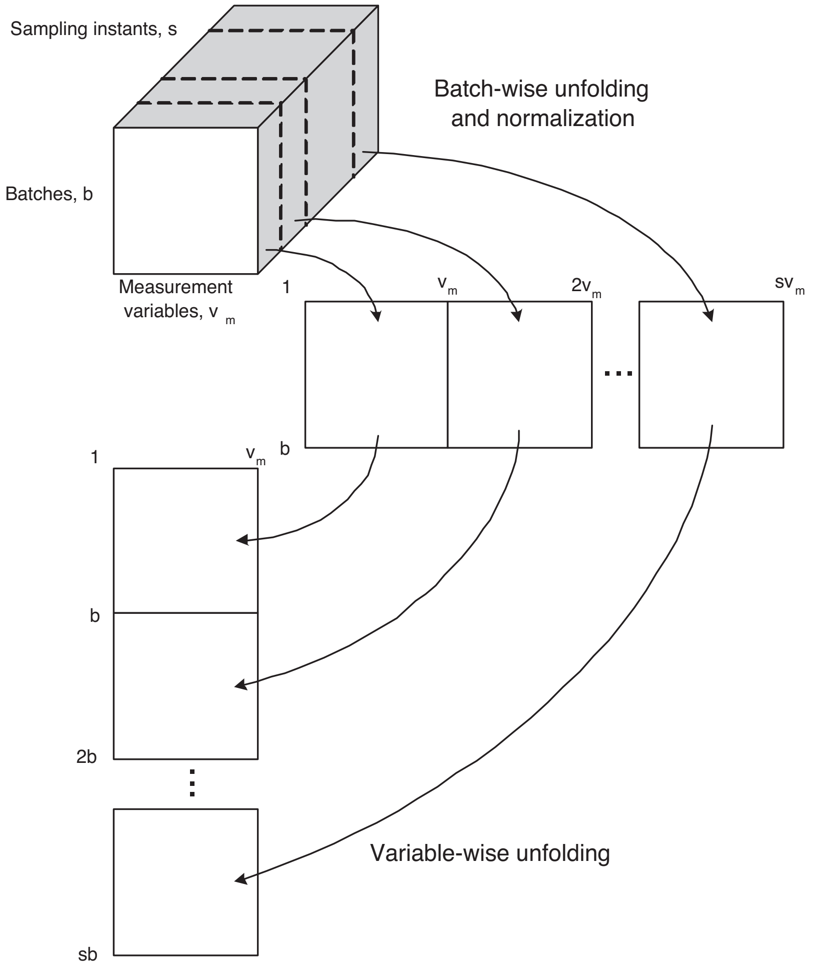


Fig. 1. Unfoling of three-dimensional batch process data into two-dimensional data matrix.

to satisfy the assumption of mean-centered $\boldsymbol{\phi}(\mathbf{x}_j)$, mean centering needs be conducted on the kernel gram matrix $\mathbf{K}$ as follows [5]

$$\overline{\mathrm{K}} = \mathbf{K} - \mathbf{1}_{nn}\mathbf{K} - \mathbf{K}\mathbf{1}_{nn} + \mathbf{1}_{nn}\mathbf{K}\mathbf{1}_{nn} \tag{34}$$

where

$$\mathbf{1}_{nn} = \frac{1}{n}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} \in \Re^{n\times n} \tag{35}$$

With the mean-centered kernel gram matrix $\overline{\mathrm{K}}$, Eq. (3) can be rewritten as

$$\overline{\mathrm{K}}\alpha_i = n\lambda_i\boldsymbol{\alpha}_i \tag{36}$$

After the eigenvalue decomposition is completed, the score matrix $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n] \in \Re^{n\times n}$ can be obtained as

$$\mathbf{Z}^T = \mathbf{A}^T\overline{\mathrm{K}} \tag{37}$$

where $\mathbf{A} = [\alpha_1, \ldots, \alpha_n] \in \Re^{n\times n}$. In this work, the number of principal components is determined according to the following criteria

$$\frac{\lambda_i}{\sum_{i=1}^{n}\lambda_i} \geq \epsilon \tag{38}$$

where ϵ represents a user-specified threshold and set to 0.95 in this work. Thus, A and Z can be extracted as

$$\mathrm{A} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_a] \in \Re^{n\times a} \tag{39}$$

$$\mathrm{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_a] \in \Re^{n\times a} \tag{40}$$

Similarly, principal components $\mathrm{V} \in \Re^{n\times e}$ for quality variables in the kernel feature space $\mathbf{Y}$ can be computed.

With the estimated kernel principal components for both measurement and quality variables in high-dimensional feature spaces, $\mathrm{Z} \in \Re^{n\times a}$ and $\mathrm{V} \in \Re^{n\times e}$ can be further projected onto lower-dimensional latent subspaces as follows

$$\mathrm{Z} = \mathrm{S}\mathrm{P}^T + \mathrm{E} \tag{41}$$

$$\mathrm{V} = \mathrm{S}\mathrm{Q}^T + \mathrm{F} \tag{42}$$

where $\mathrm{S} = [\mathrm{s}_1, \mathrm{s}_2, \ldots, \mathrm{s}_d] \in \Re^{n\times d}$ is the score matrix, $\mathrm{P} = [\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_d] \in \Re^{a\times d}$ is the loading matrix of Z, $\mathrm{Q} = [\mathrm{q}_1, \mathrm{q}_2, \ldots, \mathrm{q}_d] \in \Re^{e\times d}$ is the loading matrix of V, $\mathrm{E} \in \Re^{n\times a}$ is the residual matrix of Z, $\mathrm{F} \in \Re^{n\times e}$ is the residual matrix of V, and d is a number of latent directions. Then the objective is to find weighting vectors w and c from Z and V so that the mutual information between the score vectors corresponding to measurement and quality variables is maximized as follows

$$\max \quad I(\mathrm{s}_i; \mathrm{u}_i) = I(\mathrm{Z}_i\mathrm{w}_i; \mathrm{V}_i\mathrm{c}_i) \tag{43}$$

$$\text{s.}t. \quad \left|\left|\mathrm{w}_i\right|\right| = 1, \left|\left|\mathrm{c}_i\right|\right| = 1 \tag{44}$$

where $I(s_i, u_i)$ denotes the mutual information between the ith pair of score vectors s_i and u_i. The mutual information is defined as

$$I(s_i; u_i) = H(u_i) - H(u_i|s_i) = -\int_{u_i} f(u)\log f(u)ds + \int_{u_i}\int_{s_i} f(u,s)\log\frac{f(u|s)}{f(u,s)}duds \tag{45}$$

where $H(u_i)$ is the marginal entropy of u_i, $H(u_i|s_i)$ is the conditional entropy of u_i given s_i, and $f(\cdot)$ represents a marginal or conditional probability density function.

As Eq. (45) contains the complex integrals that are difficult to calculate analytically, a numerical optimization method termed as Nelder-Mead algorithm is adopted to solve the optimization problem given in Eq. (43) [29]. However, this algorithm cannot handle the constraints and thus w_i and c_i are normalized in the each iteration to guarantee the constraint conditions. Furthermore, the mutual information based optimization problem has strong non-linearity and multi-peak feature, which may easily lead the local optimum. To overcome this issue, the multi-start optimization strategy is utilized. First, $l_{\max}$ simplices each of which is a special polytope with $a+e+1$ vertices are formed through random initialization. It should be noted that the number of initial points $l_{\max}$ is a user-specified parameter and is set to 100 in this work. The integrated matrix of all weighting vectors is formulated as

$$\begin{bmatrix} w_i^{(1)(l)} & w_i^{(2)(l)} & \dots & w_i^{(v)(l)} & \dots & w_i^{(a+e+1)(l)} \\ c_i^{(1)(l)} & c_i^{(2)(l)} & \dots & c_i^{(v)(l)} & \dots & c_i^{(a+e+1)(l)} \end{bmatrix} \in \Re^{(a+e)\times(a+e+1)} \tag{46}$$

For all $l = 1, 2, \ldots, l_{\max}$, compute $w_i^{(l)}$ and $c_i^{(l)}$ such that the locally maximized mutual information can be obtained as many times as possible. First, $w_i^{(v)(l)}$ and $c_i^{(v)(l)}$ are normalized as

$$\begin{aligned} w_i^{(v)(l)} &= \frac{w_i^{(v)(l)}}{\left\|w_i^{(v)(l)}\right\|} \quad \forall v \\ c_i^{(v)(l)} &= \frac{c_i^{(v)(l)}}{\left\|c_i^{(v)(l)}\right\|} \quad \forall v \end{aligned} \tag{47}$$

On the basis of the computed mutual information of each pair of normalized weighting vectors, the optimal step sizes and can be estimated from Nelder–Mead algorithm and then the weighting vectors are updated as

$$\begin{aligned} w_i^{(v)(l)} &\leftarrow w_i^{(v)(l)} + \Delta w \\ c_i^{(v)(l)} &\leftarrow c_i^{(v)(l)} + \Delta c \end{aligned} \tag{48}$$

The above procedure is iterated until $w_i^{(v)(l)}$ and $c_i^{(v)(l)}$ are converged. Then lth weighting vectors $w_i^{(l)}$ and $c_i^{(l)}$ are set as

$$w_i^{(l)} = w_i^{(v)(l)}, \quad c_i^{(l)} = c_i^{(v)(l)} \tag{49}$$

After computing $w_i^{(l)}$ and $c_i^{(l)}$ for all $l = 1, 2, \ldots, l_{\max}$, the best w_i and c_i are chosen as follows

$$l_{opt} = \underset{l}{\operatorname{argmax}} \quad I(Z_i w_i^{(l)}; V_i c_i^{(l)}) \tag{50}$$

$$w_i = w_i^{l_{opt}} \tag{51}$$

$$c_i = c_i^{l_{opt}} \tag{52}$$

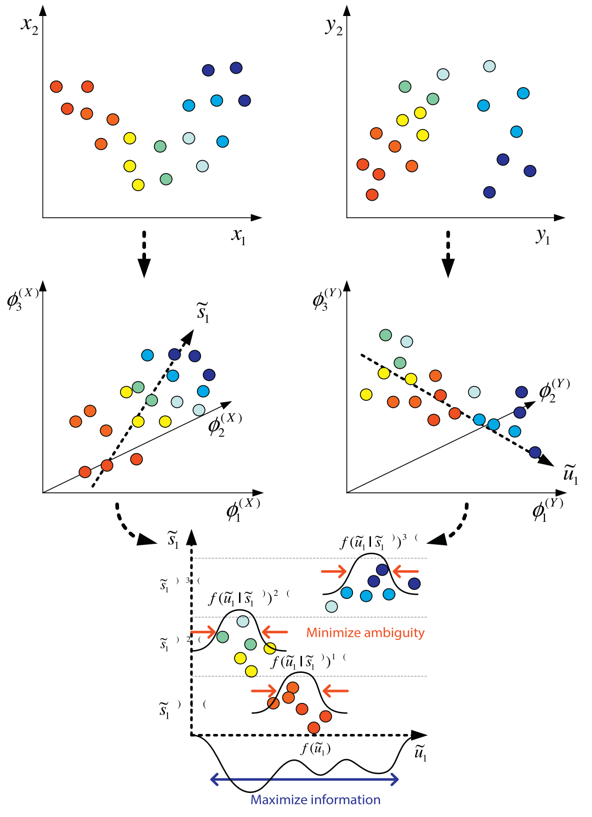


Fig. 2. Illustration of the proposed MKQNGLSP approach.

With the obtained ith weighting vectors w_i and c_i, the score vectors s_i and u_i can be computed as

$$s_i = Z_i w_i \tag{53}$$

$$u_i = V_i c_i \tag{54}$$

Further, the ith loading vectors p_i and q_i are estimated as

$$p_i = \frac{Z_i^T s_i}{s_i^T s_i} \tag{55}$$

$$q_i = \frac{V_i^T s_i}{s_i^T s_i} \tag{56}$$

Thus, the matrices Z_i and V_i are deflated as follows

$$Z_{i+1} = Z_i - s_i p_i^T \quad \text{and} \quad V_{i+1} = V_i - s_i q_i^T \tag{57}$$

Since s_i cannot be calculated from Z directly by utilizing w_i, the decomposition matrix $R = [r_1, r_2, \ldots, r_d]$ is then defined as

$$r_1 = w_1 \tag{58}$$

$$r_i = \prod_{j=1}^{i-1} (\mathbf{I}_m - w_j p_j^T) w_i \quad i \geq 2 \tag{59}$$

where $\mathbf{I}_m$ is $m \times m$ identity matrix. Therefore, the score matrix S can be computed from the kernel principal components Z as follows

$$S = ZR \tag{60}$$

After the MKQNGLSP model is built, it is important to select the number of leading latent variables for process monitoring. In this

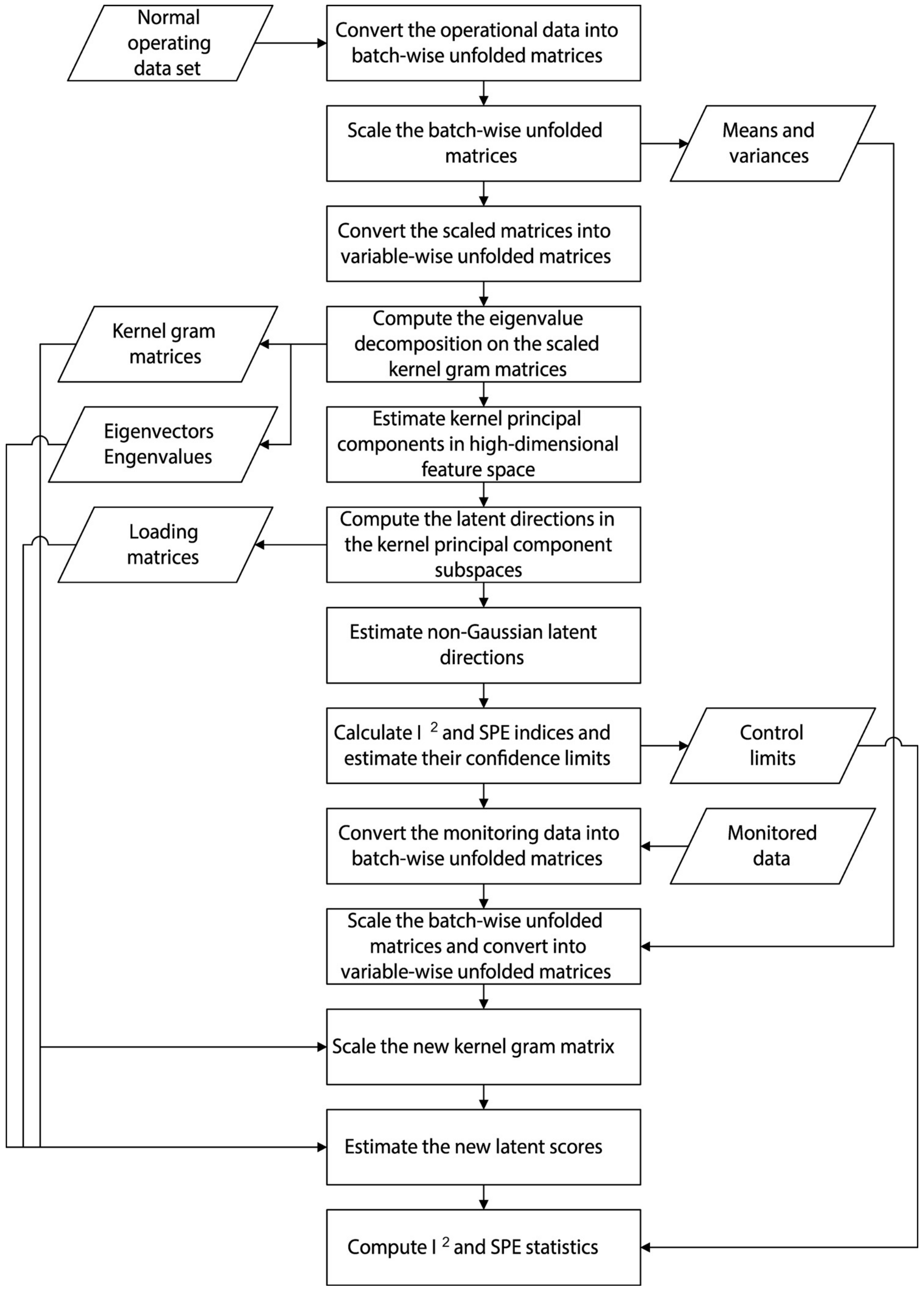

Normal operating data set
Convert the operational data into batch-wise unfolded matrices
Scale the batch-wise unfolded matrices
Means and variances
Convert the scaled matrices into variable-wise unfolded matrices
Kernel gram matrices
Compute the eigenvalue decomposition on the scaled kernel gram matrices
Eigenvectors Engenvalues
Estimate kernel principal components in high-dimensional feature space
Loading matrices
Compute the latent directions in the kernel principal component subspaces
Estimate non-Gaussian latent directions
Calculate I 2 and SPE indices and estimate their confidence limits
Control limits
Convert the monitoring data into batch-wise unfolded matrices
Monitored data
Scale the batch-wise unfolded matrices and convert into variable-wise unfolded matrices
Scale the new kernel gram matrix
Estimate the new latent scores
Compute I 2 and SPE statistics

Fig. 3. Schematic diagram of the proposed MKQNGLSP approach.

study, the column vectors of and S are sorted according to the normalized mutual information between the corresponding column vectors s_i and u_i, which is defined as follows

$$\bar{I}(s_i; u_i) = \frac{I(s_i; u_i)}{\sqrt{H(s_i)}\sqrt{H(u_i)}} \tag{61}$$

After rearranging the column vectors of S, the number of latent directions can then be determined. Too few latent directions may lead to inadequate non-Gaussian features captured in latent subspace for fault detection, while too many latent variables can cause redundant information and reduced sensitivity to process faults. Therefore, the best number of latent directions d is chosen to satisfy the following criteria based on the normalized mutual information

$$\frac{\sum_{i=1}^{d} \bar{I}(s_i; u_i)}{\sum_{i=1}^{a} \bar{I}(s_i; u_i)} \geq \tilde{\epsilon} \tag{62}$$

where $\tilde{\epsilon}$ is the user-specified threshold and set to 0.95 in this work. Thus, and R can be extracted as

$$P = [p_1, p_2, \ldots, p_d] \in \Re^{a \times d} \tag{63}$$

$$Q = [q_1, q_2, \ldots, q_d] \in \Re^{e \times d} \tag{64}$$

$$R = [r_1, r_2, \ldots, r_d] \in \Re^{a \times d} \tag{65}$$

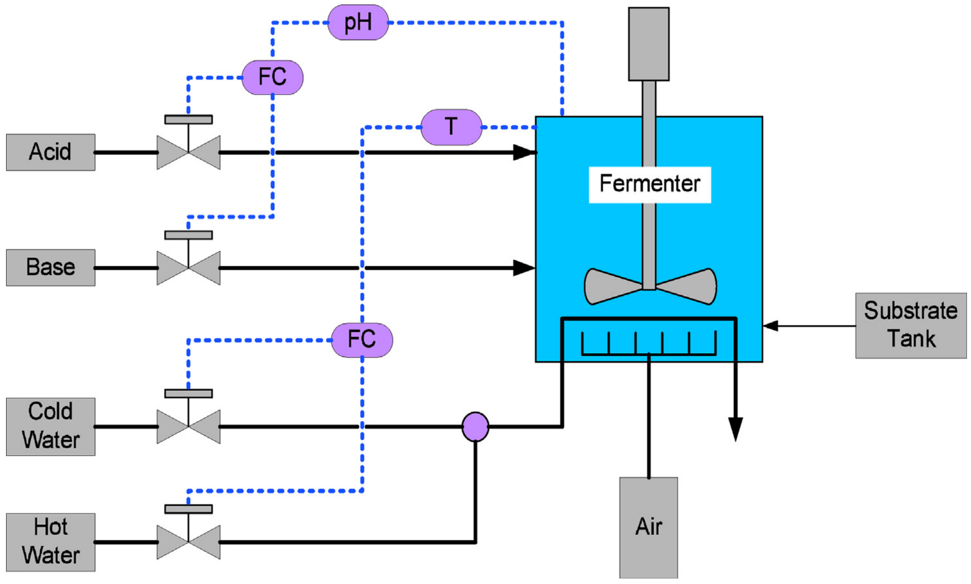


Fig. 4. Process flow diagram of the fed-batch penicillin fermentation process.

The score matrix S is further estimated as

$$S = \bar{K}AR \tag{66}$$

The searching strategy for the non-Gaussian latent directions within the nonlinear kernel feature space is illustrated in Fig. 2. It can be seen that both input and output data are first mapped into kernel feature spaces and then the latent directions are extracted. The aim of searching for latent variables is to maximize the amount of information of u_1 in terms of the marginal entropy $H(u_1)$ while minimizing the amount of ambiguity of u_1 given s_1 in terms of the conditional entropy $H(u_1|s_1)$. Such strategy is equivalent to maximize mutual information $I(s_1; u_1)$ between the score vectors u_1 and s_1 in the high-dimensional kernel feature space.

Table 1
Measurement and quality variables of the fed-batch penicillin fermentation process.

Variable no.	Variable description	Variable type
1	Dissolved aeration rate	Process variable
2	Agitator power	Process variable
3	Substrate feed temperature	Process variable
4	Substrate concentration	Quality variable
5	Dissolved oxygen concentration	Quality variable
6	Biomass concentration	Quality variable
7	Penicillin concentration	Quality variable
8	Culture volume	Process variable
9	Carbon dioxide concentration	Process variable
10	pH	Process variable
11	Fermenter temperature	Process variable
12	Generated heat	Process variable
13	Acid flow rate	Process variable
14	Base flow rate	Process variable
15	Cooling water flow rate	Process variable

Table 2
Three test cases of the fed-batch penicillin fermentation process.

Case no.	Case description
1	Substrate feed rate is increased by 2.5% from the 200th hour to the end of batch operation
2	Agitator power is increased by 3% from the 250th hour to the end of batch operation
3	Substrate feed rate is linearly increased with a slope of 0.002 from the 60th hour to the end of batch operation

Given a new sample vector $\mathbf{x}_{new}$, its corresponding score vector is computed as follows

$$s_{new} = \bar{k}_{new}AR \tag{67}$$

where $\bar{k}_{new} \in \Re^n$ denotes the normalized kernel vector for the new sample and is computed below

$$\bar{k}_{new} = \mathbf{k}_{new} - \mathbf{1}_{1n}\mathbf{K} - \mathbf{K}\mathbf{1}_{nn} + \mathbf{1}_{1n}\mathbf{K}\mathbf{1}_{nn} \tag{68}$$

with

$$\mathbf{k}_{new} = \boldsymbol{\phi}(\mathbf{x}_{new})\boldsymbol{\Phi}^T(\mathbf{X}) \tag{69}$$

$$\mathbf{1}_{1n} = \frac{1}{n}[1 \quad \cdots \quad 1] \in \Re^n \tag{70}$$

where $\boldsymbol{\phi}(\mathbf{x}_{new})$ can be obtained by mapping $\mathbf{x}_{new}$ through a nonlinear mapping function $\boldsymbol{\phi}$. With the MKQNGLSP model, I^2 and SPE statistics can be derived for batch process monitoring and fault detection. The I^2 statistic aims to detect the process abnormalities in the systematic part of the MKQNGLSP model while the SPE statistic is designed to capture the abnormality in the residual part of the model. In MKQNGLSP based monitoring method, the following I^2 and SPE indices are proposed:

$$I^2 = R^T A^T \bar{k}_{new}^T \bar{k}_{new} AR \tag{71}$$

and

$$\begin{aligned} SPE &= \left\|\boldsymbol{\phi}(\mathbf{x}_{new}) - \hat{\boldsymbol{\phi}}(\mathbf{x}_{new})\right\|^2 = 1 - 2\boldsymbol{\phi}(\mathbf{x}_{new})\hat{\boldsymbol{\phi}}^T(\mathbf{x}_{new}) \\ &\quad + \hat{\boldsymbol{\phi}}(\mathbf{x}_{new})\hat{\boldsymbol{\phi}}^T(\mathbf{x}_{new}) = 1 - 2\bar{k}_{new}A^T s_{new}^T + s_{new}P^T\bar{K}Ps_{new}^T \end{aligned} \tag{72}$$

After the above monitoring indices are defined, appropriate confidence limits need to be derived for isolating abnormal operating regions from normal batch operation. In this work, the confidence limits corresponding to I^2 and SPE indices are obtained through kernel density estimator [5]. Assume that a sequence of I^2 or SPE index

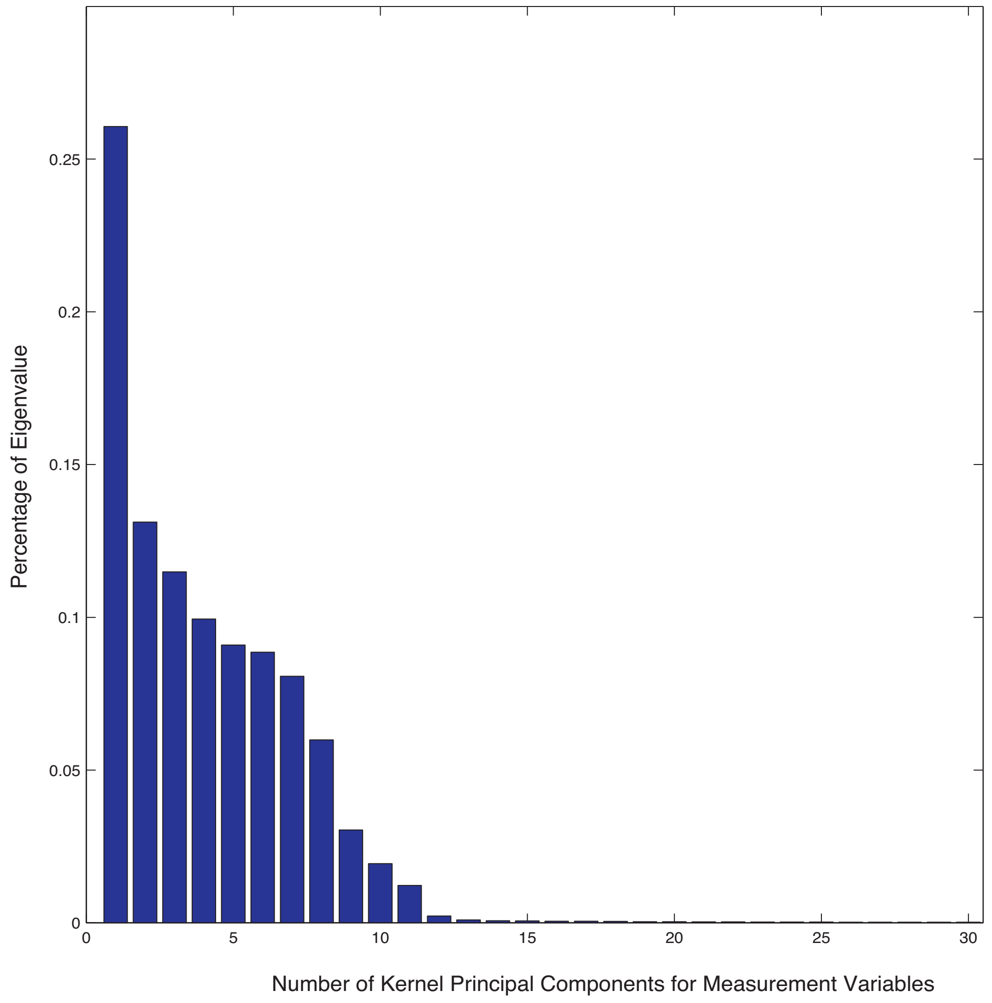


Fig. 5. Plot of the percentages of eigenvalues of kernel principal components of measurement variables.

values $(\beta_1, \beta_2, \ldots, \beta_n)$ are generated from an unknown probability density function $p(\beta)$ as follows

$$p(\beta) = \frac{1}{nD}\sum_{i=1}^{n} K\left\{\frac{\beta - \beta_i}{D}\right\} \tag{73}$$

where β denotes I^2 or SPE statistic under consideration, D is the kernel window width and K is a kernel function. The Gaussian kernel function based kernel density estimator can be expressed as

$$p(\beta) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}D}\exp\left\{-\frac{(\beta - \beta_i)^2}{2D^2}\right\} \tag{74}$$

With the estimated probability density function and the specified confidence level $(1-\gamma)\times 100\,\%$, the corresponding point with the cumulative probability $1-\gamma$ represents the confidence limit value.

The step-by-step procedure of the presented MKQNGLSP approach is listed below and the corresponding flowchart is shown in Fig. 3.

1) Collect training data from normal batch operation.
2) Convert both process measurement and quality data through batch-wise unfolding.
3) Normalize unfolded data matrices to zero-mean and unit-variance.
4) Convert the scaled data matrices through variable-wise unfolding.
5) Conduct eigenvalue decomposition on the scaled kernel gram matrices for both measurement and quality variables.
6) Estimate kernel principal components for both measurement and quality variables by extracting eigenvectors and eigenvalues.
7) Compute the multidimensional latent directions in the kernel-principal-components subspaces so that the two sets of latent variables have the maximized multidimensional mutual information between the measurement and quality variables.
8) Calculate I^2 and SPE indices and estimate the confidence limits of these statistics through kernel density estimation.
9) For new batch process data, conduct unfolding and normalization by by using same means and variances from the training data.
10) Compute a newly scaled kernel gram vector from the new batch data.
11) Estimate latent scores corresponding to measurement variables for the new batch data.
12) Calculate the values of I^2 and SPE statistics for the new batch data.

4. Application example

4.1. Fed-batch penicillin fermentation process

In this work, a fed-batch penicillin fermentation process is utilized to examine the performance of the proposed MKQNGLSP

Table 3
Comparison of fault detection rates (%) of three test cases in the fed-batch penicillin fermentation process.

Case no.	Fault detection rate (%)							
	MKPCA		MKICA		MKPLS		MKQNGLSP	
	T^2	*SPE*	I^2	SPE	T^2	*SPE*	I^2	SPE
1	47.13	54.86	84.04	24.19	4.74	17.71	91.77	91.27
2	91.69	27.91	28.90	1.99	68.77	65.12	97.67	86.71
3	82.23	80.62	93.98	73.57	58.59	68.58	96.92	96.92

Table 4
Comparison of false alarm rates (%) of three test cases in the fed-batch penicillin fermentation process.

Case no.	False alarm rate (%)							
	MKPCA		MKICA		MKPLS		MKQNGLSP	
	T^2	*SPE*	I^2	SPE	T^2	*SPE*	I^2	SPE
1	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00
2	0.00	0.00	0.00	0.00	0.60	0.40	3.01	3.01
3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

approach for nonlinear batch process monitoring [4]. The diagram of the penicillin fermentation process is shown in Fig. 4. The fermentation process starts with small amount of biomass and substrate that are added to the bioreactor from the beginning of the batch operation. After about 40 h, most of the initially added substrate is consumed and then the process is switched from batch to fed-batch operating mode. In the second stage, the substrate of glucose is being fed into fermenter continuously under open-loop condition. Meanwhile, in order to maintain the constant temperature and pH values (25°C and 5.0), two proportional-integral-derivative (PID) control loops are implemented in the fermenter for manipulating the acid/base and hot/cold water flow

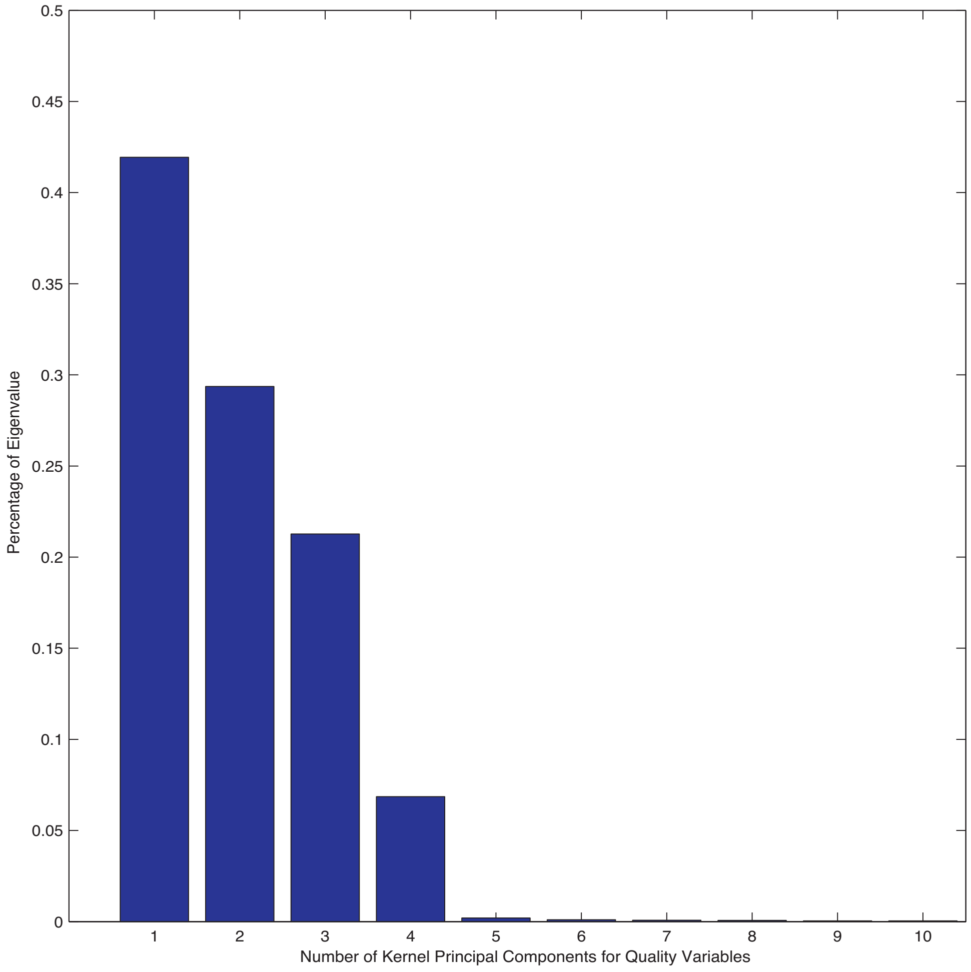


Fig. 6. Plot of the percentages of eigenvalues of kernel principal components of quality variables.

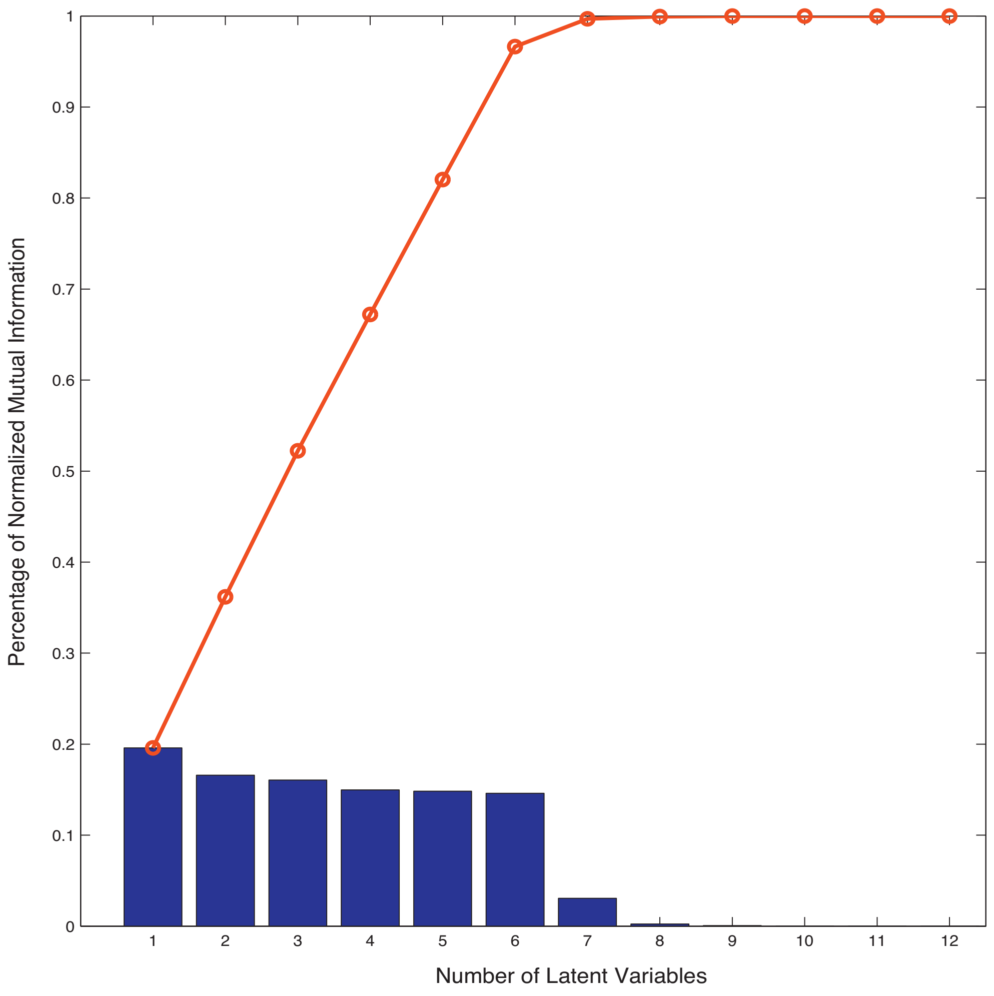


Fig. 7. Plot of the percentages and cumulative percentages of normalized mutual information of the extracted non-Gaussian latent variables.

ratios, respectively. The duration of each batch is 400 h, while the sampling time of both measurement and quality variables are set to 0.5 h. There are total 11 measurement variables as input variables $\underline{X}$ and 4 quality variables as output variables $\underline{Y}$. All the process measurement and quality variables are listed in Table 1. In our study, the MKQNGLSP model is built from the training data set consisting of 10 normal batches. Each batch has small variations in the process variables. In order to compare the effectiveness of the proposed MKQNGLSP approach with the MKPCA, MKICA and MKPLS methods, three test cases are designed with different types of faulty scenarios, as shown in Table 2. In the first test case, the fermentation process begins with normal operating conditions and then a 2.5% step increase on substrate feed rate occurs from the 200th hour until the end of batch operation. For the second case, the normal operation is followed by a fault of 3% step increase on agitator power, which begins at the 250th hour and remains until the end of batch duration. In the last test scenario, a drift error with the slope of 0.002 l/h takes place on substrate feed rate from the 60th hour until the end of batch operation.

4.2. Comparison of batch process monitoring result

After the KPCA model is obtained, the first 12 kernel principal components corresponding to measurement variables and 6 kernel principal components corresponding to quality variables are selected, as shown in Fig. 5. Then the MKQNGLSP model is built from 12 kernel PCs for measurement variables and 6 kernel PCs for quality variables. Typically, the selected number of kernel principal components is larger than that of original process variables, because KPCA extracts principal components from the high-dimensional kernel feature space. Further, the number of latent variables in the obtained MKQNGLSP model is determined by the normalized mutual information criterion. The percentage and the cumulative percentage of mutual information of different latent variables of MKQNGLSP model are shown in Fig. 7. It can be observed that the first 6 latent directions should be selected as their corresponding cumulative percentage of normalized mutual information is over 95%. For MKPCA, total 9 latent variables whose cumulative percentage of eigenvalues is over 95% are selected. As for MKICA, total 9 independent components are chosen because their cumulative percentage of L^2 norm of the demixing matrix is over 95% [24]. For MKPLS model, total 11 latent variables with the best monitoring performance of fault detection rate are selected (Fig. 6).

In the first test scenario, the step fault of substrate feed rate occurs from the 200th hour. The increase of substrate feed rate leads to the higher concentrations of biomass and penicillin than those under normal operation. The process monitoring results of the MKPCA, MKICA, MKPLS and proposed MKQNGLSP methods are shown in Fig. 8. Meanwhile, the fault detection and false alarm rates

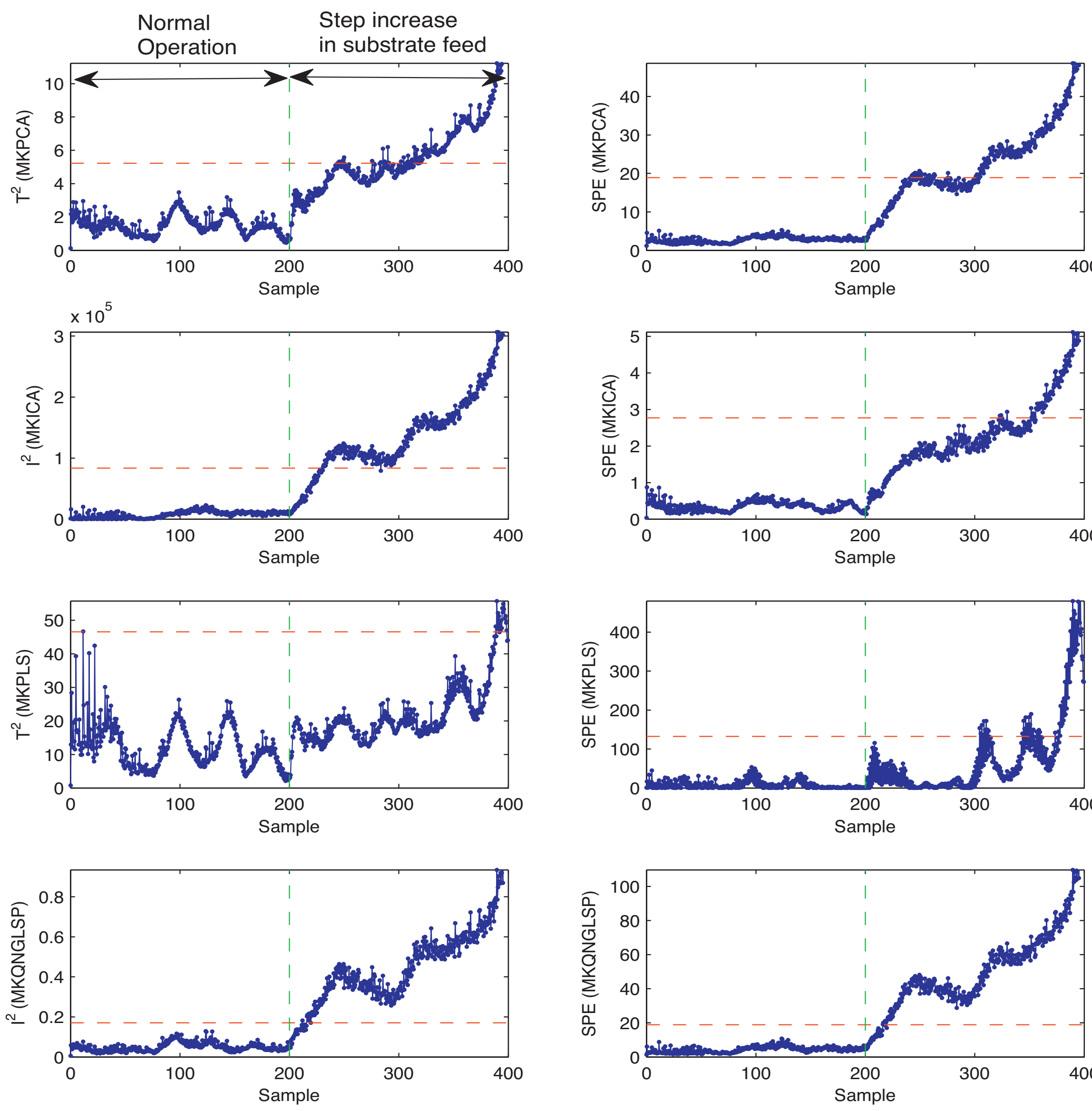


Fig. 8. Monitoring results of MKPCA (the first row), MKICA (the second row), MKPLS (the third row) and MKQNGLSP (the fourth row) methods in the first test case of the fed-batch penicillin fermentation process.

of different methods are listed in Tables 3 and 4, respectively. One can readily see that the MKQNGLSP based I^2 is able to accurately detect faulty operations with the high fault detection rate of 91.77% while the false alarm rate of 0.00%. In contrast, the MKPCA based T^2 index, the MKICA based I^2 index and the MKPLS based T^2 index result in lower fault detection rates of 47.13% , 84.04 % and 4.74%, respectively. Meanwhile, the MKQNGLSP based SPE index yields the high fault detection rate of 91.27% along with the false alarm rate of 0.00%. In comparison, the fault detection rates of MKPCA, MKICA and MKPLS based SPE index are 54.86% , 24.19 % and 17.71%, which are significantly lower than that of the MKQNGLSP based SPE index. These results indicate that the proposed MKQNGLSP method has strong capability to extract nonlinear and non-Gaussian process features from batch process data. Although the MKICA method takes into consideration nonlinear and non-Gaussian process features, it does not incorporate quality variables in the model and thus its performance is worse than that of MKQNGLSP method. Furthermore, even though the MKPLS method is able to deal with process nonlinearity and its model also includes both measurement and quality variables, it does not take into account the high-order statistics for capturing non-Gaussian process features so that its monitoring performance is not as satisfactory as that of the proposed MKQNGLSP approach.

In the second test case, the process operation includes 3% step increase of agitator power starting from the 250th hour. The process monitoring results of the MKPCA, MKICA, MKPLS and MKQNGLSP methods are compared in Fig. 9 as well as Tables 3 and 4. It can be observed that the MKQNGLSP based I^2 statistic can detect abnormal operating conditions with the high fault detection rate of 97.67% along with the low false alarm rate of 3.01%. Meanwhile, the MKPCA based T^2 index, the MKICA based I^2 index and the MKPLS based T^2 index can detect the faulty operation with lower fault detection rates of 91.69% , 28.90 % and 68.77%, respectively. On the other hand, the MKQNGLSP based SPE index is able to trigger the alarms of faulty operation with the fault detection rate of 86.71% and the false alarm rate of 3.01%. In comparison,

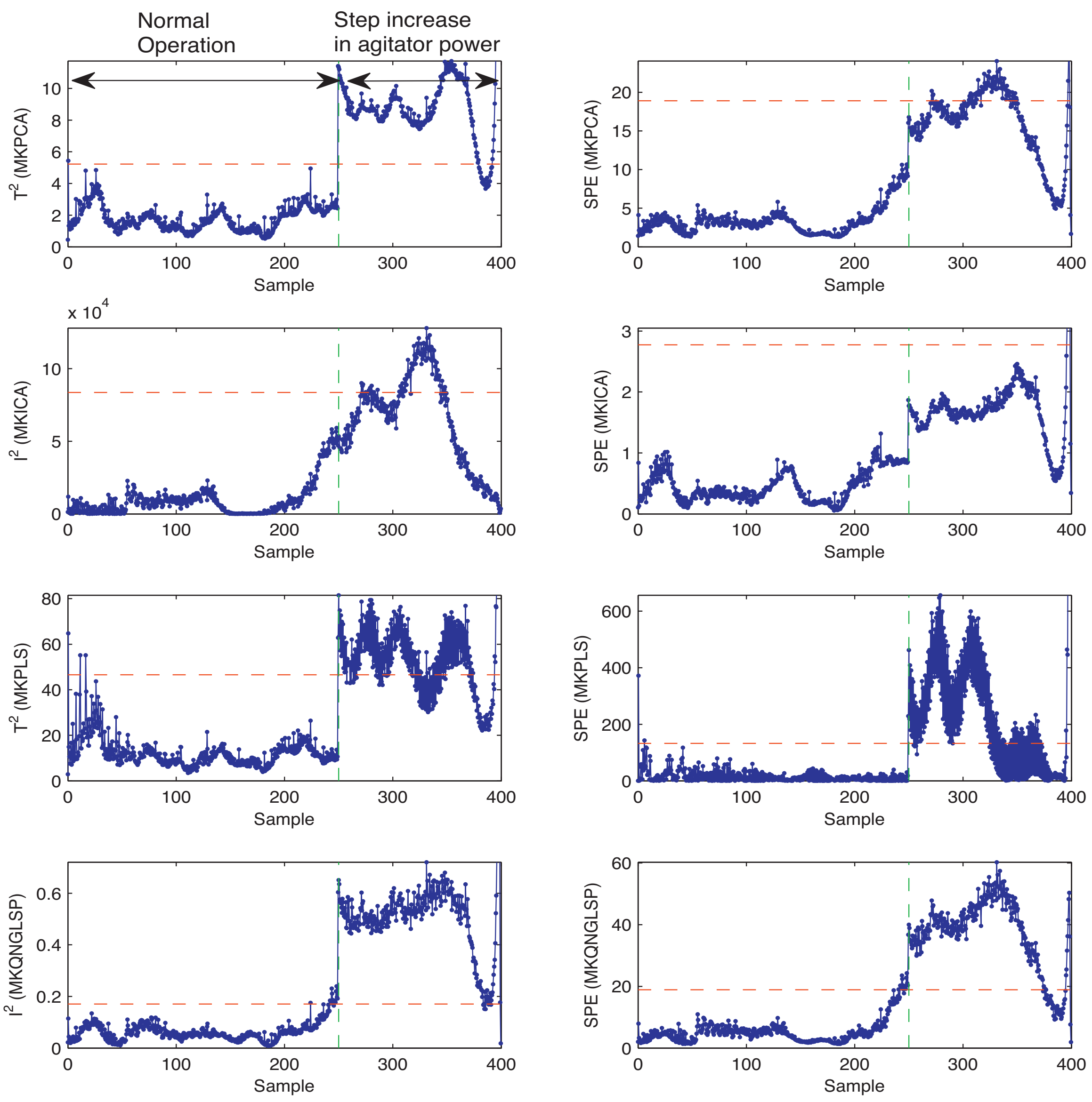


Fig. 9. Monitoring results of MKPCA (the first row), MKICA (the second row), MKPLS (the third row) and MKQNGLSP (the fourth row) methods in the second test case of the fed-batch penicillin fermentation process.

the MKPLS based SPE index can only detect 65.12% of faulty samples. For MKPCA and MKICA, their SPE indices result in much lower fault detection rates of 27.91% and 1.99%, respectively. The main reason why the MKPCA and MKICA show poor performance is that only the process measurement variables are involved in their models without any output quality variables. Moreover, the superiority of the proposed MKQNGLSP method over the MKPLS approach is due to the fact that the higher-order statistic of mutual information rather than the second-order statistic of covariance between the process measurement and product quality variables is maximized while searching for the latent directions. In this way, the non-Gaussian process features can be better extracted by the proposed MKQNGLSP method for quality relevant process monitoring.

For the third test case, a drift error with the slope of 0.002 l/h is added to the substrate feed rate starting from the 60th hour, which makes biomass and penicillin concentrations grow faster than the normal trajectories. The monitoring results of the MKPCA, MKICA, MKPLS and the proposed MKQNGLSP approaches are shown in Fig. 10, while the quantitative comparison is given in Tables 3 and 4. One can easily see that the MKPCA based T^2, MKICA based I^2 and MKPLS based T^2 indices lead to the fault detection rates of 82.23%, 93.98% and 58.59%, respectively, which are lower than that of the MKQNGLSP based T^2 index (96.92%). Likewise, the fault detection rates of the MKPCA, MKICA and MKPLS based SPE indices are 80.62%, 73.57% and 68.58%, respectively. In contrast, the MKQNGLSP based SPE index leads to the fault detection rate of 96.92%, which is significant higher than those of the other methods. These results further prove that the proposed MKQNGLSP approach can effectively extract the non-Gaussian process features in quality relevant latent subspaces and thus has substantially improved monitoring capability than the MKPCA, MKICA and MKPLS methods. All the above three test cases demonstrate that the proposed MKQNGLSP method is able to capture abnormal batch operation with higher fault sensitivity and detectability than the conventional MKPCA, MKICA and MKPLS approaches.

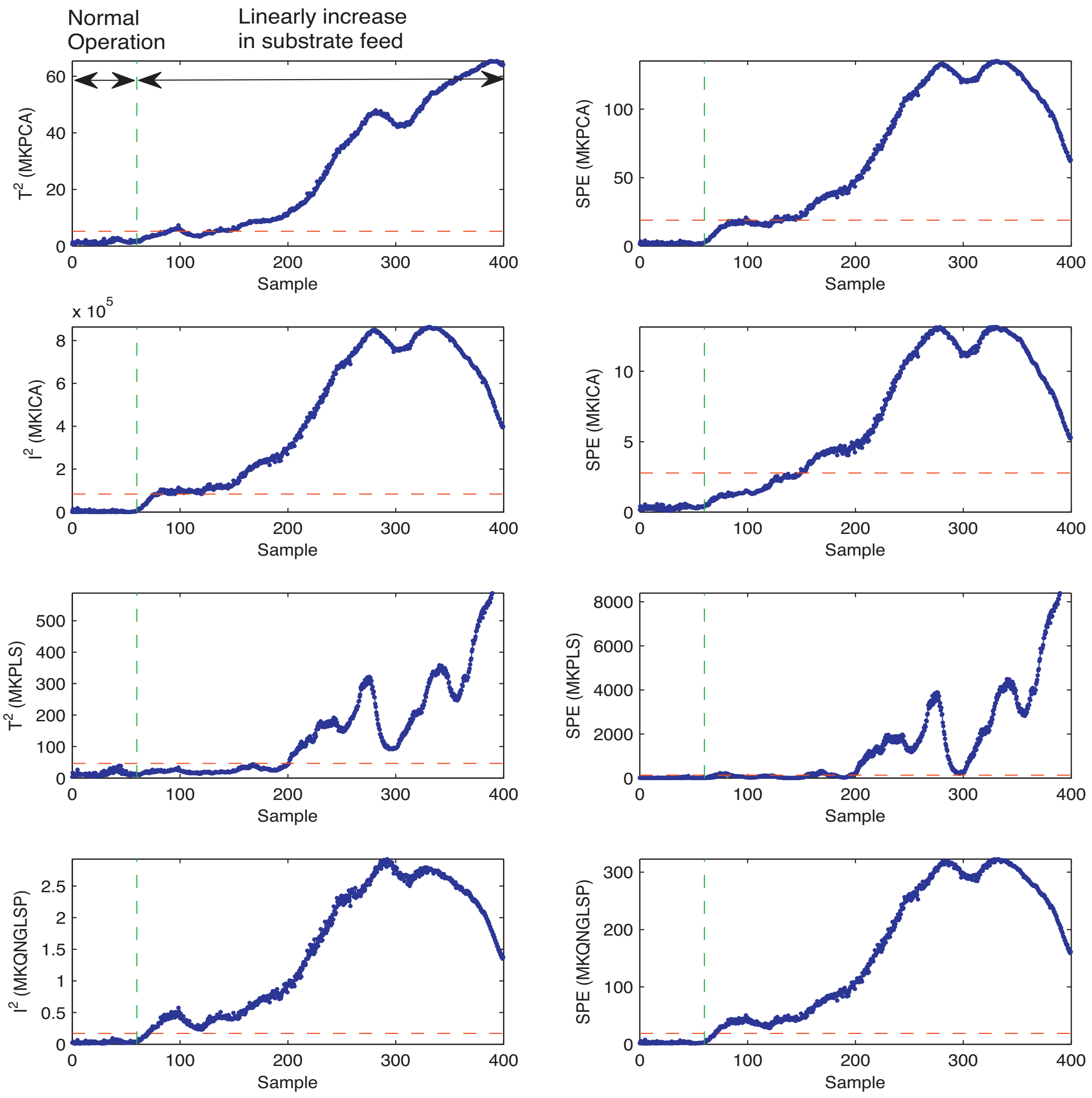


Fig. 10. Monitoring results of MKPCA (the first row), MKICA (the second row), MKPLS (the third row) and MKQNGLSP (the fourth row) methods in the third test case of the fed-batch penicillin fermentation process.

5. Conclusions

In this paper, a multiway quality relevant non-Gaussian latent subspace projection method is developed for nonlinear batch process monitoring. The presented approach can identify nonlinear dynamics and non-Gaussian relationships between measurement and quality variables. First, the kernel principal components are extracted from the unfolded and scaled measurement and quality data sets and thus the nonlinear process dynamics can be characterized in the high-dimensional kernel feature space. Secondly, the multidimensional latent directions in the kernel-principal-components subspaces corresponding to measurement and quality variables are searched concurrently by maximizing the higher-order statistic of mutual information instead of the second-order statistic of covariance. Hence, the non-Gaussian features relating measurement and quality variables can be well captured. Thirdly, a set of new monitoring indices are developed to capture abnormalities of batch process data within non-Gaussian latent subspace and residual subspace. The proposed MKQNGLSP method incorporates both measurement and quality variables, the latter of which are not included in the MKPCA and MKICA models. Furthermore, the MKQNGLSP model is characterized with the maximized mutual information instead of covariance and thus can effectively capture non-Gaussian process features that may not be well handled by the conventional MKPLS model.

The presented MKQNGLSP method is applied to a fed-batch Penicillin fermentation process, and the monitoring results demonstrate that the new I^2 and SPE indices outperform the MKPCA, MKICA and MKPLS based statistics. The MKQNGLSP approach provides an effective solution for nonlinear and non-Gaussian quality relevant batch process monitoring and fault detection. Future research will focus on extending MKQNGLSP method for fault isolation and diagnosis of nonlinear batch processes.

References

[1] H. Albazzaz, X.Z. Wang, Statistical process control charts for batch operations based on independent component analysis, Ind. Eng. Chem. Res. 43 (2004) 6731–6741.
[2] B.R. Bakshi, Multiscale PCA with application to multivariate statistical process monitoring, AIChE J. 44 (7) (1998) 1596–1610.

[3] G. Bastin, D. Dochain, On-line Estimation and Adaptive Control of Bioreactors, Elsevier, Amsterdam, The Netherlands, 1990.
[4] I. Birol, C. Ündey, G. Birol, E. Tatara, A. Ç inar, A web-based simulator for penicillin fermentation, Int. J. Eng. Simul. 2 (2001) 24–30.
[5] C.M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, New York, NY, USA, 2007.
[6] A. Chen, Z. Song, P. Li, Soft sensor modeling based on DICA-SVR, Lect. Notes Comput. Sci. 3644 (2005) 868–877.
[7] J. Chen, K. Liu, On-line batch process monitoring using dynamic PCA and dynamic PLS models, Chem. Eng. Sci. 57 (2002) 63–75.
[8] L.H. Chiang, E.L. Russell, R.D. Braatz, Fault Detection and Diagnosis in Industrial Systems, in: Advanced Textbooks in Control and Signal Processing, Springer-Verlag, London, Great Britain, 2001.
[9] L.H. Chiang, R. Leardi, R.J. Pell, M.B. Seasholtz, Industrial experiences with multivariate statistical analysis of batch process data, Chemom. Intell. Lab. Syst. 81 (2006) 109–119.
[10] S.W. Choi, J. Morris, I. Lee, Dynamic model-based batch process monitoring, Chem. Eng. Sci. 63 (3) (2008) 622–636.
[11] K. Desai, Y. Badhe, S.S. Tambe, B.D. Kulkarni, Soft-sensor development for fed-batch bioreactors using support vector regression, Biochem. Eng. J. 27 (3) (2006) 225–239.
[12] L. Di, Z. Xiong, X. Yang, Nonlinear process modeling and optimization based on multiway kernel partial least squares model, in: Proceedings of IEEE, 2008, pp. 1645–1651.
[13] F.J. Doyle, Nonlinear inferential control for process applications, J. Proc. Cont. 8 (1998) 339–353.
[14] J. Flores-Cerrillo, J.F. MacGregor, Multivariate monitoring of batch processes using batch-to-batch information, AIChE J. 50 (2004) 1219–1228.
[15] Z. Ge, Z. Song, Process monitoring based on independent component analysis–principal component analysis (ICA–PCA) and similarity factors, Ind. Eng. Chem. Res. 46 (2007) 2054–2063.
[16] A. Hyv??rinen, J. Karhunen, E. Oja, Independent Component Analysis, John Wiley and Sons, New York, NY, USA, 2001.
[17] H. Kaneko, M. Arakawa, K. Funatsu, Development of a new soft sensor method using independent component analysis and partial least squares, AIChE J. 55 (2009) 87–98.
[18] M. Kano, S. Tanaka, S. Hasebe, I. Hashimoto, H. Ohno, Monitoring independent components for fault detection, AIChE J. 49 (4) (2003) 969–976.
[19] A. Kassidas, J.F. MacGregor, P.A. Taylor, Synchronization of batch trajectories using dynamic time warping, AIChE J. 44 (1998) 864–875.
[20] W. Ku, R.H. Storer, C. Georgakis, Disturbance detection and isolation by dynamic principal component analysis, Chemom. Intell. Lab. Syst. 30 (1995) 179.
[21] J.M. Lee, C.K. Yoo, I.B. Lee, Fault detection of batch processes using multiway kernel principal component analysis, Comput. Chem. Eng. 28 (2004) 1837–1847.
[22] J.M. Lee, C.K. Yoo, I.B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chem. Eng. Sci. 59 (2004) 223–234.
[23] J.M. Lee, C.K. Yoo, I.B. Lee, Statistical monitoring of dynamic processes based on dynamic independent component analysis, Chem. Eng. Sci. 59 (2004) 2995–3006.
[24] J.M. Lee, C.K. Yoo, I.B. Lee, Statistical process monitoring with independent component analysis, J. Proc. Cont. 14 (2004) 467–485.
[25] J.-M. Lee, C. Yoo, S.W. Choi, P.A. Vanrolleghem, I.-B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chem. Eng. Sci. 59 (2004) 223–234.
[26] B. Lennox, G.A. Montague, H.G. Hiden, G. Kornfeld, P.R. Goulding, Process monitoring of an industrial fed-batch fermentation, Biotechnol. Bioeng. 74 (2001) 125–135.
[27] B. Lennox, H.G. Hiden, G.A. Montague, G. Kornfeld, P.R. Goulding, Process monitoring of an industrial fed-batch fermentation, Biotechnol. Bioeng. 74 (2001) 125–135.
[28] X. Liu, U. Kruger, T. Littler, L. Xie, S. Wang, Moving window kernel PCA for adaptive monitoring of nonlinear processes, Chemom. Intell. Lab. Syst. 96 (2009) 132–143.
[29] A. Nelder, R. Mead, A simplex method for function minimization, Comput. J. 7 (4) (1965) 308–313.
[30] P. Nomikos, J.F. MacGregor, Multivariate SPC charts for monitoring batch processes, Technometrics 37 (1) (1995) 41–59.
[31] M.J. Piovoso, K.A. Hoo, Multivariate statistics for process control, IEEE Cont. Syst. Mag. 22 (2002) 8–9.
[32] M.J. Piovoso, K.A. Kosanovich, Applications of multivariate statistical methods to process monitoring and controller design, Int. J. Control 59 (1994) 743–765.
[33] M.N. Pons, A. Rajab, J.M. Flaus, J.M. Engasser, A. Cheruy, Comparison of estimation methods for biotechnological processes, Chem. Eng. Sci. 8 (1988) 1909–1914.
[34] S.J. Qin, Statistical process monitoring: basics and beyond, J. Chemometrics 17 (2003) 480–502.
[35] A. Raich, A. Ç inar, Statistical process monitoring and disturbance diagnosis in multivariable continuous processes, AIChE J. 42 (1996) 995–1009.
[36] M.M. Rashid, J. Yu, Nonlinear and non-Gaussian dynamic batch process monitoring using a new multiway kernel independent component analysis and multidimensional mutual information based dissimilarity approach, Ind. Eng. Chem. Res. 51 (2012) 10910–10920.
[37] X. Tian, X. Zhang, X. Deng, S. Chen, Multiway kernel independent component analysis based on feature samples for batch process monitoring, J. Neurocomputing 72 (2009) 1584–1596.
[38] C. Ündey, S. Ertunç, A. Ç inar, Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis, Ind. Eng. Chem. Res. 42 (2003) 4645–4658.
[39] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Kavuri, A review of process fault detection and diagnosis: Part I: Quantitative model-based methods, Comput. Chem. Eng. 27 (2003) 293–311.
[40] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Kavuri, A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies, Comput. Chem. Eng. 27 (2003) 313–326.
[41] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Kavuri, A review of process fault detection and diagnosis: Part III: Qualitative models and search strategies, Comput. Chem. Eng. 27 (2003) 327–346.
[42] B.M. Wise, N.B. Gallagher, The process chemometrics approach to process monitoring and fault detection, J. Proc. Cont. 6 (1996) 329–348.
[43] Y. Yao, F. Gao, A survey on multistage/multiphase statistical modeling methods for batch processes, Annu. Rev. Control 33 (2009) 172–183.
[44] J. Yu, Nonlinear bioprocess monitoring based multiway kernel localized Fisher discriminant analysis, Ind. Eng. Chem. Res. 50 (2011) 3390–3402.
[45] J. Yu, A nonlinear kernel Gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes, Chem. Eng. Sci. 68 (2012) 506–519.
[46] J. Yu, Online quality prediction of nonlinear and non-Gaussian chemical processes with shifting dynamics using finite mixture model based Gaussian process regression approach, Chem. Eng. Sci. 82 (2012) 22–30.
[47] J. Yu, A particle filter driven dynamic Gaussian mixture model approach for complex process monitoring and fault diagnosis, J. Proc. Cont. 22 (2012) 778–788.
[48] J. Yu, S.J. Qin, Multiway Gaussian Mixture Model Based Multiphase Batch Process Monitoring, Ind. Eng. Chem. Res. 48 (18) (2009) 8585–8594.
[49] J. Yu, S.J. Qin, Variance component analysis based fault diagnosis of multi-layer overlay lithography processes, IIE Trans. 41 (2009) 764–775.
[50] H. Zhang, B. Lennox, Integrated condition monitoring and control of fed-batch fermentation processes, J. Proc. Cont. 14 (2004) 41–50.
[51] X. Zhang, W. Yan, H. Shao, Nonlinear multivariate quality estimation and prediction based on kernel partial least squares, Ind. Eng. Chem. Res. 47 (2008) 1120–1131.
[52] Y. Zhang, Fault detection and diagnosis of nonlinear processes using improved Kernel independent component analysis (KICA) and support vector machine (SVM), Ind. Eng. Chem. Res. 47 (2008) 6961–6971.
[53] Y. Zhang, S.J. Qin, Fault detection of nonlinear processes using multiway kernel independent component analysis, Ind. Eng. Chem. Res. 46 (23) (2007) 7780–7787.
[54] S.J. Zhao, J. Zhang, Y.M. Xu, Performance monitoring of processes with multiple operating modes through multiple PLS models, J. Proc. Cont. 16 (2006) 763–772.
[55] Z. Zhu, Z. Song, A. Palazoglu, Process pattern construction and multi-mode monitoring, J. Proc. Cont. 22 (2012) 247–262.

Chapter 4

Identification of probabilistic graphical network model for root-cause diagnosis in industrial processes

Computers and Chemical Engineering 71 (2014) 171–209

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Identification of probabilistic graphical network model for root-cause diagnosis in industrial processes

Junichi Mori, Vladimir Mahalec[*], Jie Yu

Department of Chemical Engineering, McMaster University, Hamilton, Ontario, Canada L8S 4L7

A R T I C L E I N F O

Article history:
Received 5 April 2014
Received in revised form 16 July 2014
Accepted 22 July 2014
Available online 1 August 2014

Keywords:
Probabilistic graphical model
Cause–effect relationship
Structure learning
Incidence matrix
Markov chain Monte Carlo simulation
Root-cause diagnosis

A B S T R A C T

Identification of faults in process systems can be based purely on measurement (e.g. PCA), or can exploit knowledge of process model structure to construct a causal network. This work introduces a method to identify most likely causal network in cases when process model is not known. An incidence matrix, showing location of measurements in the plant network structure, and historical process data are used to identify the optimal causal network structure by means of maximizing Bayesian scores for alternative causal networks. Causal subnetworks, corresponding to subgraphs of the process network, are identified by finding the most probable graph based on highest posterior probability of graph features computed via Markov Chain Monte Carlo simulation. Novel Bayesian contribution indices within the probabilistic graphical network are proposed to identify the potential root-cause variables. Application to Tennessee Eastman Chemical plant demonstrates that the presented method is significantly more accurate than the current methods.

1. Introduction

Process monitoring and fault diagnosis are gaining significant attention in order to improve product quality, yields, energy efficiency, plant safety, and eco-sustainability (Piovoso and Hoo, 2002; Venkatasurbramanian et al., 2003). Rapid development of measurement, automation, advanced computing, and database technologies has enabled many companies to record a huge number of process variables in industrial plant historians. Methods that extract useful information from such a large number of the historical process data provide valuable insight into plant operation.

The approaches to process monitoring, fault detection and fault diagnosis analysis fall into two categories, which are: (i) model based and (ii) data driven techniques (Gertler, 1988). Model-based process monitoring methods may be applicable only if accurate mechanistic models of processes can be developed (Pons et al., 1988; Doyle, 1998). However, these first-principle models require in-depth knowledge about processes and a significant effort to build precise mechanistic models for large-scale complex industrial plants. By contrast, data-driven monitoring techniques have become increasingly attractive because they do not require in-depth fundamental knowledge and mechanistic models but instead depend on the historical process data only.

Multivariate statistical process monitoring (MSPM) techniques have been developed to extract useful information from a large number of highly correlated process variables (Nomikos and MacGregor, 1995). Two widely used methods in the MSPM field are principal component analysis (PCA) and partial least squares or projection to latent structure (PLS). These methods can build the data-driven models within the low-dimensional subspace that retains most of the variance or covariance structure (Mejdell and Skogestad, 1991; Kresta et al., 1994; Kosanovich et al., 1996; Zhang and Lennox, 2004; Zamprogna et al., 2005; Lin et al., 2007). Then the statistics such as T^2 and SPE are proposed for extracting the critical features of process data for fault detection and diagnosis. Furthermore, in order to handle non-Gaussian processes, independent component analysis (ICA) has been applied to project multivariate process data into latent subspace of statistically independent components (IC) (Yan et al., 2004; Chen et al., 2005; Desai et al., 2006; Kaneko et al., 2009; Yu, 2012b). ICs are assumed to be non-Gaussian and mutually independent based on high-order statistics and they retain the non-Gaussian process features that cannot be extracted by traditional PCA/PLS methods. Similar to PCA/ICA based monitoring methods, the ICA based statistics like I^2 and SPE are developed for detecting faulty operation (Lee et al., 2004). More recently, the PLS method has been extended to the non-Gaussian decomposition version called quality relevant non-Gaussian latent

* Corresponding author. Tel.: +1 905 525 9140x26386.
E-mail address: mahalec@mcmaster.ca (V. Mahalec).

http://dx.doi.org/10.1016/j.compchemeng.2014.07.022
0098-1354/© 2014 Elsevier Ltd. All rights reserved.

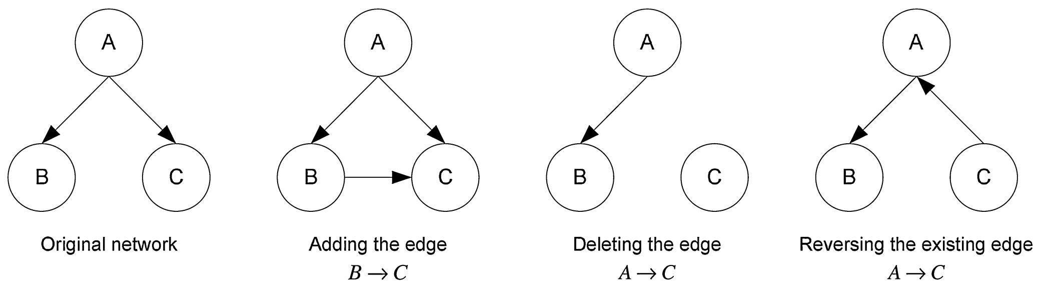


Fig. 1. Neighbouring graphs.

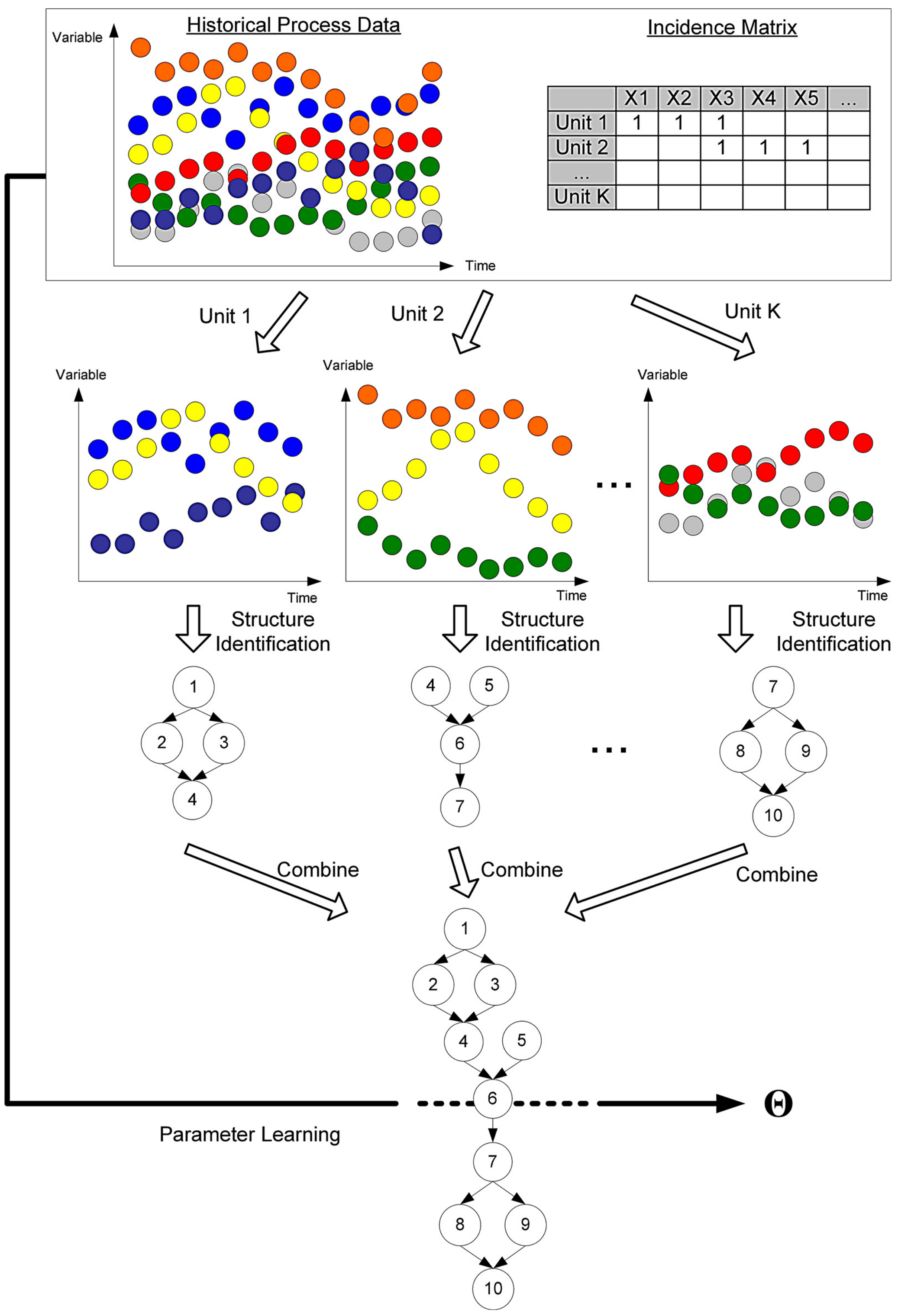
subspace projection (QNGLSP) (Mori and Yu, 2014a, 2013) and its non-linear decomposition version called multi-way kernel quality relevant non-Gaussian latent subspace projection (MKQNGLSP) (Mori and Yu, 2014b). Once a fault is detected, all the above methods are capable of generating contribution plots to identify the major fault effect variables without prior process knowledge. Nevertheless, variable contribution methods are unable to identify the root causes of faulty operations among the fault effect variables without in-depth knowledge about processes because the abnormal operational events can propagate throughout the process due to the intricate variable interactions, process dynamics, closed-loop control systems, etc.

In order to identify the cause–effect relationship, signed directed graph (SDG) approach for incipient fault diagnosis has been proposed. SDG method has the possibility to identify the cause–effect relationship and the direction of the fault effect (Maurya et al., 2004). Moreover, qualitative trend analysis (QTA) that incorporates data driven approach has been proposed in order to reduce the number of spurious alarms (Maurya et al., 2007). However, the above method identifies candidate faults that the prior fault database contains and thus it can be challenging to diagnose abnormal events that the prior fault database does not include. In addition, these methods cannot trace the fault propagation pathways and especially the root-cause variables. SDG is also used in order to automatically interpret the PCA-based contributions (Vedam and Venkatasubramanian, 1999). If model equations are available, this method is capable of detecting multiple root causes. More recently, Granger causality methods are proposed for revealing the root causes of plant oscillations (Yuan and Qin, 2014). Granger causality is based on linear prediction of time series and can extract the cause effect relationship in terms of prediction accuracy in autoregressive (AR) model. This method is very useful to identify the cause–effect relationship among process variables and capture the root cause of the plant oscillations that degrades the prediction performance, but it cannot explain whether the calculated Granger causality comes from abnormal operations or not. Therefore, it may not be effective in identifying the root-cause variables that lead to the process upset. Alternatively, causal maps that are graphs with directed arcs between nodes have been employed to identify the cause and effect relationships among process variables using the Kullback–Leibner distance (Chiang and Braatz, 2003). However, the fault propagation is derived from in-depth process knowledge and thus it is challenging to apply this method to fault diagnosis when the propagation pathways are difficult to identify due to the lack of process knowledge. Moreover, the cross-correlation function (CCF) is estimated from two time series and its results are arranged in a causal matrix (Bauer and Thornhill, 2008). The transfer entropy is also utilized to identify the direction of disturbance propagation (Bauer et al., 2007). These two approaches do not require any process knowledge, but the cross-correlation function and transfer entropy can consider relationship between two random variables only and hence it may not model more intricate dependencies containing over three variables. To make matters worse, these methods require a large number of samples to identify the precise propagation pathways and hence the small number of samples may cause the poor indicator of causal relationships.

Alternatively, pattern recognition algorithms play an important role in process monitoring and analysis. For example, support vector machines (SVM), artificial neural networks (ANN) and Gaussian mixture model (GMM) methods have been developed for chemical process monitoring (Sorsa and Koivo, 1993; Yélamos et al., 2009; Yu and Qin, 2008; Yu, 2012a). However, they may not be able to isolate accurately the root-cause variables, particularly when the fault propagation pathways include complex variable interactions across different process units.

Instead of using purely probabilistic models, probabilistic graphical models are highly advantageous for analyzing the cause–effect relationship. The probabilistic graphical model consists of a graphical structure and a probabilistic description of the relationships among random variables under system uncertainty. Bayesian network is one of the major class of graphical models and has been applied to various fields including medical diagnostics, gene modeling, cancer classification and reliability analysis (Gevaert et al., 2006; Boudali and Dugan, 2005; Friedman et al., 2000; Nikovski, 2000). Bayesian network based process diagnosis techniques have been developed in the past decade. For instance, Bayesian networks have been employed in order to identify the root cause of process variations and give a probabilistic confidence level of the diagnosis (Weidl et al., 2005; Dey and Stori, 2005). Bayesian networks have also been applied for control loop performance diagnosis and have made it possible to synthesize various existing monitoring methods (Huang, 2008). Furthermore, Bayesian networks have been adopted for identifying the cause–effect relationship among process patterns, process information and possible root causes (Alaeddini and Dogan, 2011). Nevertheless, all of the above Bayesian network based process monitoring techniques can deal with discrete variables only, and if there are continuous variables, these variables should be discretized. In addition, potential root causes need to be specified and added to hidden nodes in advance. The biggest problem with application of Bayesian network based methods is that they require the in-depth process knowledge to design the network structure for well-performed process diagnosis.

In order to apply Bayesian networks to process monitoring without descrtizing any process variables and specifying the potential root causes in advance, more recently, Bayesian network based networked process monitoring and diagnosis approach are proposed to detect root-cause variables in dynamic processes without any specifications of fault types (Yu and Rashid, 2013). This method makes use of likelihood of each node to identify fault propagation pathways as well as root-cause variables. Similar to the other existing methods, process knowledge is needed for the design of the network structure. In addition, it can be time-consuming to build precise graphical model for complex processes, and it is also challenging to check the accuracy of the inferred structure. If the



	X1	X2	X3	X4	X5	...
Unit 1	1	1	1			
Unit 2			1	1	1	
...						
Unit K						

Fig. 2. Illustrative diagram of learning Bayesian network framework.

cause–effect relationships are difficult to identify due to the lack of process knowledge or too intricate processes, data-driven techniques are useful to design the network structure. The basic idea of network structure learning is to find the graph so that the score function representing likelihood is maximized. Since this computational task is a combinatorial optimization problem, which is known as *NP*-hard (Chickering, 1996), it is challenging to compute the precise network structure for industrial plants where there are a large number of process variables and historical data set. The standard methodology to overcome this issue is to carry out a heuristic search. Some powerful searching spaces such as equivalence classes (Chickeringn, 2002), skeletons (Steck, 2000), and orderings (Teyssier and Koller, 2005) have been proposed, but their combinatorial problems remain NP-hard.

Motivated by the above consideration, we propose a novel data-driven structure learning algorithm that is able to construct the probabilistic graphical model for process monitoring, fault diagnosis, and process analysis without in-depth process knowledge.

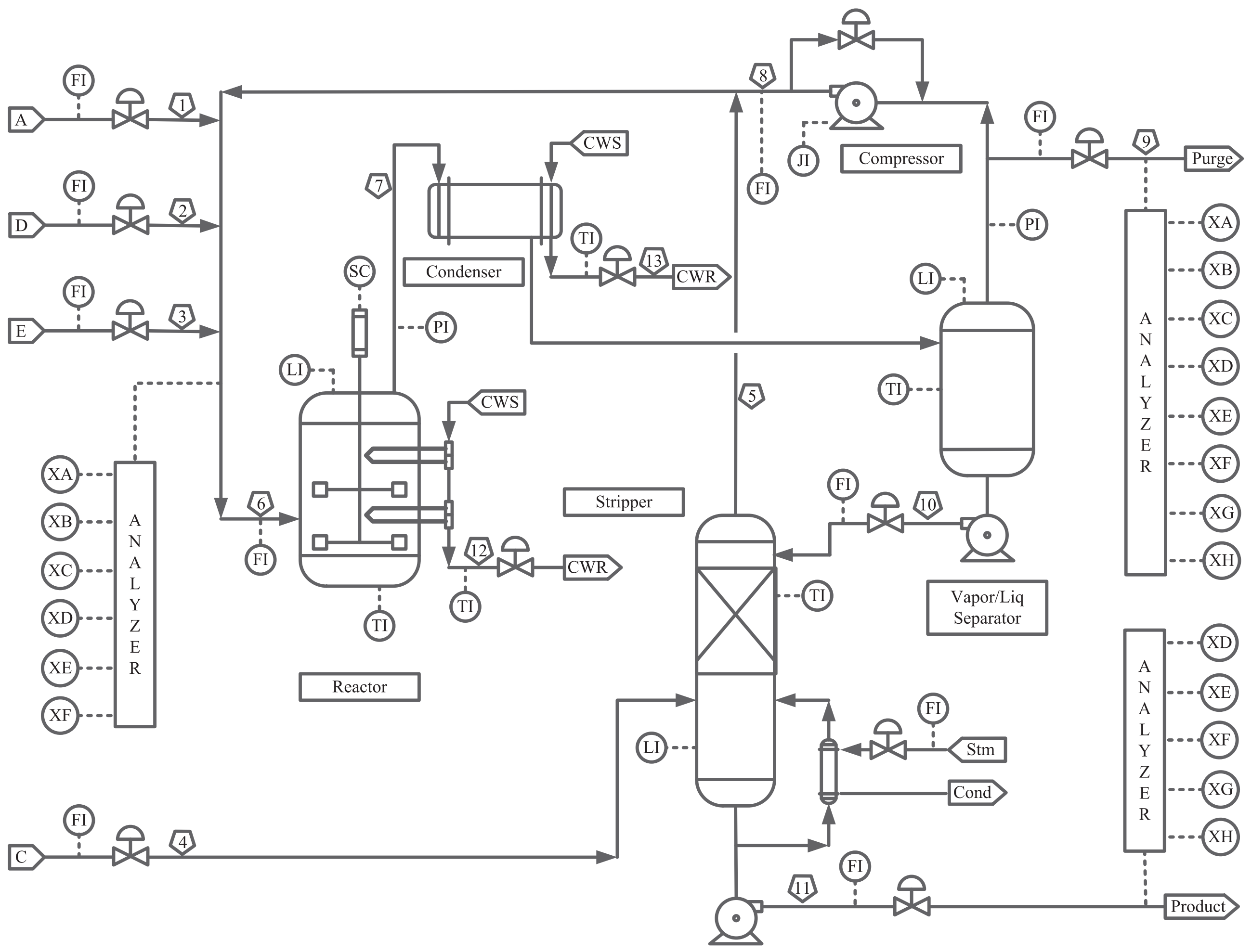


Fig. 3. Process flow diagram of the Tennessee Eastman Chemical process.

Table 1
Monitored variables of Tennessee Eastman Chemical process.

Variable no.	Symbol	Variable description
1	F_1	A feed
2	F_2	D feed
3	F_3	E feed
4	F_4	Total feed
5	F_5	Recycle flow
6	F_6	Reactor feed rate
7	P_7	Reactor pressure
8	L_8	Reactor level
9	T_9	Reactor temperature
10	F_{10}	Purge rate
11	T_{11}	Separator temperature
12	L_{12}	Separator level
13	P_{13}	Separator pressure
14	F_{14}	Separator underflow
15	L_{15}	Stripper level
16	P_{16}	Stripper pressure
17	F_{17}	Stripper underflow
18	T_{18}	Stripper temperature
19	F_{19}	Stripper steam flow
20	J_{20}	Compressor work
21	T_{21}	Reactor coolant temperature
22	T_{22}	Condenser coolant temperature

Table 2
Predefined faults of the Tennessee Eastman Chemical process.

Fault ID.	Fault description
IDV(1)	Step in A/C feed ratio, B composition constant
IDV(2)	Step in B composition, A/C ratio constant
IDV(3)	Step in D feed temperature (stream 2)
IDV(4)	Step in reactor cooling water inlet temperature
IDV(5)	Step in condenser cooling water inlet temperature
IDV(6)	A feed loss (step change in stream 1)
IDV(7)	C header pressure loss (step change in stream 4)
IDV(8)	Random variation in A + C feed composition (stream 4)
IDV(9)	Random variation in D feed temperature (stream 2)
IDV(10)	Random variation in C feed temperature (stream 4)
IDV(11)	Random variation in reactor cooling water inlet temperature
IDV(12)	Random variation in condenser cooling water inlet temperature
IDV(13)	Slow drift in reaction kinetics
IDV(14)	Sticking reactor cooling water valve
IDV(15)	Sticking condenser cooling water valve

The summary of the proposed structure learning algorithm is listed below:

1. Divide the set of measured variables into smaller subsets in accordance with modularity of the process flow diagram;
2. Compute the most reasonable sub-networks for all subsets;
3. Combine all determined sub-networks into one entire network;
4. Compute the network parameters including the conditional probability density functions of all nodes.

Table 3
Incidence matrix of the Tennessee Eastman Chemical process.

Unit	F_1	F_2	F_3	F_5	F_6	P_7	L_8	T_9	T_{21}	T_{22}	T_{11}	L_{12}	P_{13}	F_{14}	F_4	L_{15}	P_{16}	F_{17}	T_{18}	F_{19}	F_{10}	J_{20}
Mixer	1	1	1	1	1																	
Reactor	1	1	1	1	1	1	1	1	1													
Condenser								1		1												
Separator						1	1	1		1	1	1	1	1								
Stripper											1	1	1	1	1	1	1	1	1	1		
Compressor				1							1	1	1								1	1

Table 4
Posterior probability of the graph edges in the mixer and reactor.

		Child nodes								
		F_1	F_2	F_3	F_5	F_6	P_7	L_8	T_9	T_{21}
Parent nodes	F_1	0	0	0	0	1	1	1	0.106	1
	F_2	0	0	0	0	1	0.024	0.998	0.18	0.086
	F_3	0	0	0	0	1	0.02	0.056	0.051	1
	F_5	0	0	0	0	1	1	1	1	1
	F_6	0	0	0	0	0	1	0.93	1	1
	P_7	0	0	0	0	0	0	0	0	0
	L_8	0	0	0	0	0	1	0	0.002	0.026
	T_9	0	0	0	0	0	1	0.018	0	0.586
	T_{21}	0	0	0	0	0	1	0.974	0.414	0

Table 5
Root-cause (text in bold represents correct identification of root-cause variables).

Test case number	PCA		ICA		Transfer entropy		Proposed
	T^2 Rank of true root-cause variable	SPE Rank of true root-cause variable	I^2 Rank of true root-cause variable	SPE Rank of true root-cause variable	Potential root-cause variables include true one? (Y/N)	Number of potential root-cause variables (If Y)	Identify root-cause variable? (Y/N)
IDV(3)	16th	2nd	–	4th	N	–	**Y**
IDV(4)	2nd	10th	2nd	3rd	N	–	**Y**
IDV(5)	3rd	**1st**	2nd	**1st**	**Y**	7	**Y**
IDV(6)	**1st**	10th	**1st**	**1st**	N	–	**Y**
IDV(7)	**1st**	2nd	2nd	3rd	**Y**	6	N
IDV(9)	2nd	3rd	5th	16th	N	–	**Y**
IDV(10)	**1st**	18th	**1st**	18th	**Y**	6	N
IDV(11)	20th	14th	20th	14th	N	–	N
IDV(12)	3rd	**1st**	2nd	3rd	N	–	**Y**
IDV(13)	3rd	2nd	6th	3rd	**Y**	4	**Y**
IDV(14)	**1st**	8th	**1st**	**1st**	N	–	**Y**
IDV(15)	7th	**1st**	2nd	3rd	N	–	**Y**

After the probabilistic graph is obtained, the likelihood based monitoring indices are proposed for detecting faulty operations. With the captured process abnormality, the Bayesian contribution indices are developed to capture the potential root-cause variables. Finally, the posterior probability of fault propagation pathways is computed so as to identify the most probable fault propagation pathway.

The organization of this article is as follows. Section 2 defines the probabilistic graphical model used in this work and reviews the existing structure learning algorithm. Section 3 describes the proposed structure identification algorithm for probabilistic graphical network. The proposed probabilistic graphical network based process monitoring approach for fault detection and root-cause diagnosis is proposed in Section 4. The presented method is applied to the Tennessee Eastman Chemical process in Section 5. Finally, the conclusions are drawn in Section 6.

2. Probabilistic graphical model

2.1. Model definition

Probabilistic graphical models represent complex causal relationships among a set of random variables and their conditional dependencies. A popular graphical model is a Bayesian network which is essentially a directed acyclic graph (DAG) consisting of hidden and observed nodes, each of which is connected to the various nodes. Nodes with arrows pointing toward them are termed as child nodes while the ones with departing arrows are parent nodes. Meanwhile, the nodes without any parent nodes are called root nodes and the nodes without any child nodes are called leaf nodes.

The structure of BN can be designed from prior knowledge and then the qualitative causal reasoning is conducted within the network models. Further, the conditional probability distributions with model parameters $\theta_i \in \Theta$ can be estimated from historical data to infer the quantitative causal relationships among variables.

Given all the multivariate measurements $\mathbf{x} = [x_1, x_2, \ldots, x_I] \in \Re^I$ with x_i being the ith node, the joint distribution is expressed as

$$p(x) = \prod_{i=1}^{I} p(x_i|\mathrm{pa}(x_i)) \qquad (1)$$

where $\mathrm{pa}(x_i)$ are the parent nodes of x_i.

Once the network structure is determined, model parameters Θ can be identified by maximizing the likelihood. In the case when some nodes are unobservable, expectation–maximization

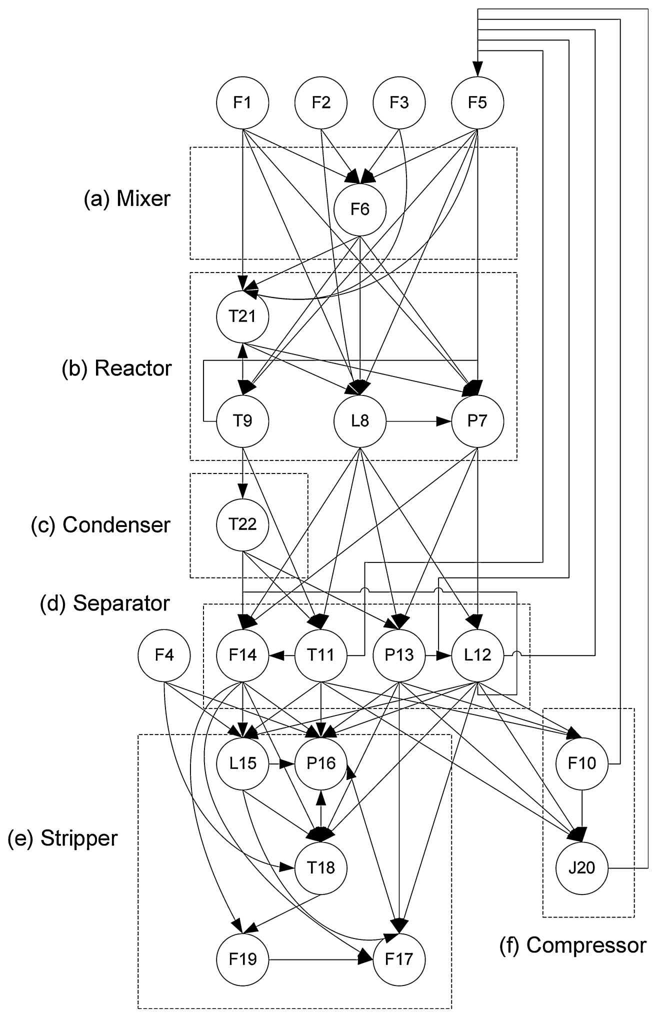


Fig. 4. Most probable network structure of the Tennessee Eastman Chemical process.

(EM) algorithm can be used. The details on BN model learning procedure can be found in literature (Bishop, 2006).

It should be pointed out that in this paper each node of the probabilistic graphical model represents a monitored process variable and the data are assumed to be complete or fully observed. Therefore, no inference is needed, which means the proposed probabilistic graphical model does not have to be an acyclic graph. In addition, the linear Gaussian model is employed to parameterize variables. We denote the monitored variables as $\mathbf{x} = [x_1, x_2, \ldots, x_I] \in \Re^I$. A single continuous random variables x_i has a Gaussian distribution and the mean of the distribution is computed as a linear combination of the state of its parent nodes $\mathrm{pa}(x_i)$ of ith node as follows:

$$p(x_i|\mathrm{pa}(x_i)) = P(x_i|\mathrm{pa}(x_i); \Theta_i) = \mathcal{N}(x_i| \sum_{j\in \mathrm{pa}(x_i)} w_{i,j}x_j + b_i, \sigma_i^2) \tag{2}$$

where $w_{i,j}$ and b_i are parameters controlling the mean, σ_i represents the standard deviation of the conditional distribution for x_i, and $\Theta_i = \{w_{i,j}, b_i, \sigma_i\}$ is the model parameters of ith node. Each variable x_i can be computed as

$$x_i = \sum_{j\in \mathrm{pa}(x_i)} w_{i,j}x_j + b_j + \sigma_i\epsilon_i \tag{3}$$

where ϵ_i is a Gaussian random variable with a zero mean and unit variance. By taking the expectation of Eq. (3), the following system of linear equation is obtained:

$$E[x_i] = \sum_{j\in \mathrm{pa}(x_i)} w_{i,j}\mathrm{E}[x_j] + b_j \tag{4}$$

where $E[\bullet]$ represents the mathematical expectation operator. First, when probabilistic network structure is already known, the parameter $w_{i,j}$ can be computed by solving the linear equation Eq. (4). In the same way, from Eqs. (3) and (4) the parameters σ_i can be obtained by solving the following equation:

$$\mathrm{cov}[x_i, x_j] = \sum_{k\in \mathrm{pa}(x_j)} w_{jk}\mathrm{cov}[x_i, x_k] + I_{ij}\sigma_j^2 \tag{5}$$

where I_{ij} is the i, j element of the identity matrix and $\mathrm{cov}[x_i x_j]$ is the covariance between x_i and x_j. In this way, all model parameters $\boldsymbol{\Theta} = \{\Theta_1, \ldots, \Theta_I\}$ can be computed (Bishop, 2006).

2.2. Structure learning

First and the most important task for networked process monitoring is to infer the precise network structure. The most straight forward way is to construct the network structure from process knowledge. However, by-hand structure design requires in-depth process knowledge because identification of cause–effect relationships is needed to characterize the complex physical, chemical and biological phenomena in systems. On the other hand, data-driven based techniques such as score function based structure learning is useful when there is lack of process knowledge or considered processes are too complicated to analyze.

The goal of structure learning is to find a network structure $\mathcal{G}$ that is a good estimator for the data $\mathbf{x}$. The most common approach is *score-based structure learning*, which defines the structure learning problem as an optimization problem (Koller and Friedman, 2009). A score function score($\mathcal{G}:\mathcal{D}$) that measures how well the network model fits the observed data $\mathcal{D}$ is defined. The computational task is to solve the combinational optimization problem to finding the network so that the highest score can be obtained.

Several kinds of scoring functions have been developed. Most of them have the strong property where the score function is decomposable as follows:

$$\mathrm{score}(\mathcal{G}) = \sum_{i=1}^{I} \mathrm{score}(x_i, \mathrm{pa}_{\mathcal{G}}(x_i)) \tag{6}$$

where $\mathrm{pa}_{\mathcal{G}}(x_i)$ are the parent nodes of x_i given graph $\mathcal{G}$. For instance, the BIC score (for Bayesian information criterion) can be defined as

$$\mathrm{score}_{\mathrm{BIC}}(\mathcal{G}:\mathcal{D}) = \ell(\hat{\theta}_{\mathcal{G}}:\mathcal{D}) - \frac{logM}{2}\mathrm{Dim}[\mathcal{G}] \tag{7}$$

where $\hat{\theta}_{\mathcal{G}}$ represents the maximum likelihood parameters for a graph $\mathcal{G}$, $\ell(\hat{\theta}_{\mathcal{G}}:\mathcal{D})$ denotes the logarithm of the likelihood function, M is the number of samples, and $\mathrm{Dim}[\mathcal{G}]$ is the model dimension. Owing to the term of model dimension, the BIC score includes a tradeoff between the likelihood and model complexity. The BIC scores decompose as follows:

$$\mathrm{score}_{\mathrm{BIC}}(\mathcal{G}:\mathcal{D}) = M\sum_{i=1}^{N} \mathbf{I}(x_i; \mathrm{pa}_{\mathcal{G}}(x_i)) - M\sum_{i=1}^{N} H(x_i) - \frac{logM}{2}\mathrm{Dim}[\mathcal{G}] \tag{8}$$

where $\mathbf{I}(x_i; \mathrm{pa}_{\mathcal{G}}(x_i))$ is the mutual information between x_i and $\mathrm{pa}_{\mathcal{G}}(x_i)$, and $H(x_i)$ is the marginal entropy of x_i. Since the linear Gaussian model is employed in this work, the mutual information

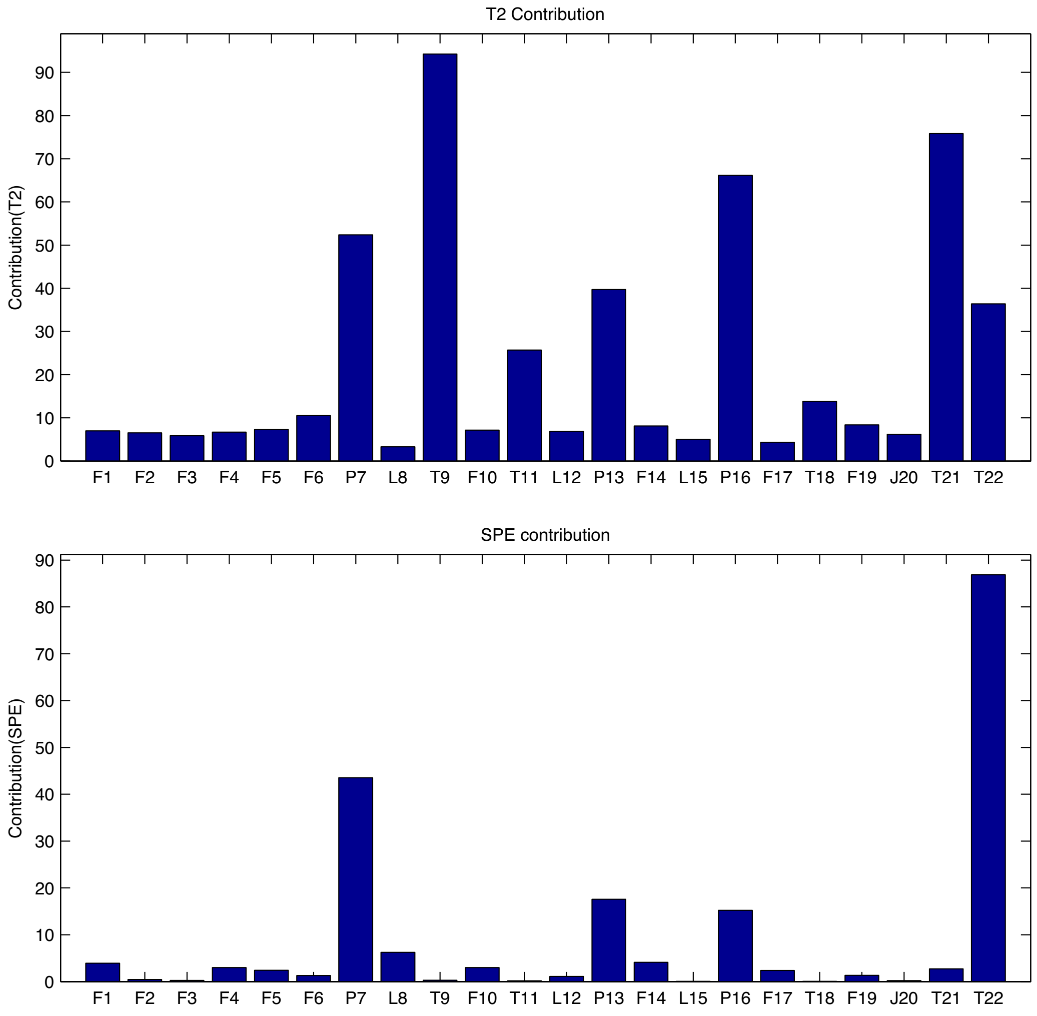


Fig. 5. Variable contribution plots of PCA based process monitoring in the case of IDV(4).

between two multidimensional Gaussian variables **X** and **Y**, and the entropy of **X** can be computed by:

$$\mathbf{I}(\mathbf{X};\mathbf{Y}) = \frac{1}{2}\log\left(\frac{\det(\Sigma_{\mathbf{XX}})\det(\Sigma_{\mathbf{YY}})}{\det(\Sigma)}\right) \tag{9}$$

and

$$\mathbf{H}(\mathbf{X}) = \ln\left(\sigma\sqrt{(2\pi e)}\right) \tag{10}$$

with

$$\Sigma = \begin{pmatrix} \Sigma_{\mathbf{XX}} & \Sigma_{\mathbf{XY}} \\ \Sigma_{\mathbf{YX}} & \Sigma_{\mathbf{YY}} \end{pmatrix} \tag{11}$$

where σ is a standard deviation of **X**, e is Napier's Number, det(•) represents the determinant of a matrix and $\Sigma_{\mathbf{XY}}$ is the covariance matrix between **X** and **Y**.

Score decomposability is important property because if the decomposability is satisfied, a local change in the structure does not change the score of other parts of the structure that remains same. It also should be noted that the mutual information term grows linearly in M while the model complexity term grows logarithmically, and thus the larger M leads to the obtained graph which better represents data (Koller and Friedman, 2009).

With the score function, the optimal graph $\mathcal{G}^*$ can be computed as follows:

$$\mathcal{G}^* = \underset{\mathcal{G}}{\operatorname{argmax}}\,\mathrm{score}_{\mathrm{BIC}}(\mathcal{G}:\mathcal{D}). \tag{12}$$

It should be noted that this combinational optimization problem is known to be *NP*-hard (Chickering, 1996). Therefore, it is challenging to obtain the optimal graph for industrial plants which often include a large number of process variables.

3. Structure identification method

3.1. Incidence matrix based decomposition

In order to overcome the issue of computational cost as well as explicitly incorporate relationships between the location of measurements and the plant structure, the incidence matrix is utilized to divide a large set of variables into smaller subsets. The incidence matrix has one row for each of the units in the process under consideration and one column for each of the measured variables. The incidence matrix is a $K \times I$ matrix $[b_{ki}]$ where K and I represent the number of units and measured variables, respectively. Each entry $[b_{ki}]$ is assigned 1 or 0 in accordance with the rule where $b_{ki} = 1$ if the measured variable x_i is monitored either in the unit k or in the outlet stream of the unit k, and $b_{ki} = 0$ otherwise. Next, the entry is set to be $b_{ki} = 1$ when the variable i monitored in the adjacent upper unit $k-1$ have an influence on the unit k under consideration. If the

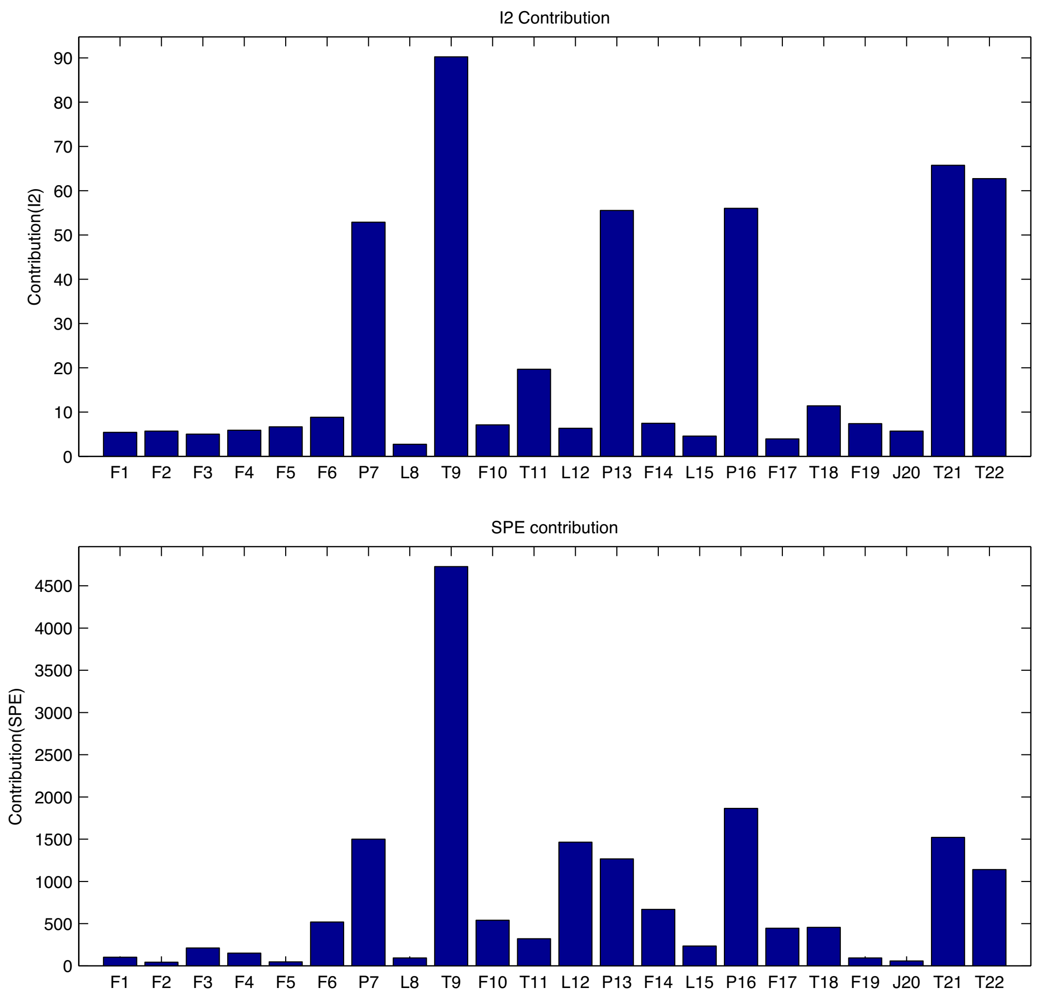


Fig. 6. Variable contribution plots of ICA based process monitoring in the case of IDV(4).

variable i monitored in the upper unit $k-t$, $t \geq 2$ and the variables monitored in the unit k are not conditionally independent given variables monitored in units $k-1, k-2, \ldots, k-t+1$, it is useful to set the entry to be $b_{ki} = 1$ to avoid losing important information. It is true that this procedure requires some process knowledge, but it can be readily implemented from very basic understanding of process diagram.

With the generated incidence matrix, a large set of training data is divided into smaller subsets $\mathcal{D}^k \in \Re^K$. Each subset corresponds to a row of the incidence matrix and contains the monitored variables whose entries are 1 in that row. Subsets of training data should satisfy the following condition:

$$\mathcal{D}^1 \cup \ldots \cup \mathcal{D}^k \cup \cdots \cup \mathcal{D}^K = \mathcal{D}. \tag{13}$$

After defining the subsets from the incidence matrix, the following combinational optimization problem is solved for each subset $\mathcal{D}^k$.

$$\mathcal{G}^{*k} = \underset{\mathcal{G}}{\text{argmax}}\,\text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}^k) \tag{14}$$

where $\mathcal{G}^{*k}$ is an optimal graph for subset $\mathcal{D}^k$. Therefore, the total number of computed subgraphs is equal to the number of units under consideration. If the user likes to incorporate the process knowledge into the network structure, any variables are allowed to be set on the root nodes or the leaf nodes in subgraphs. A set of

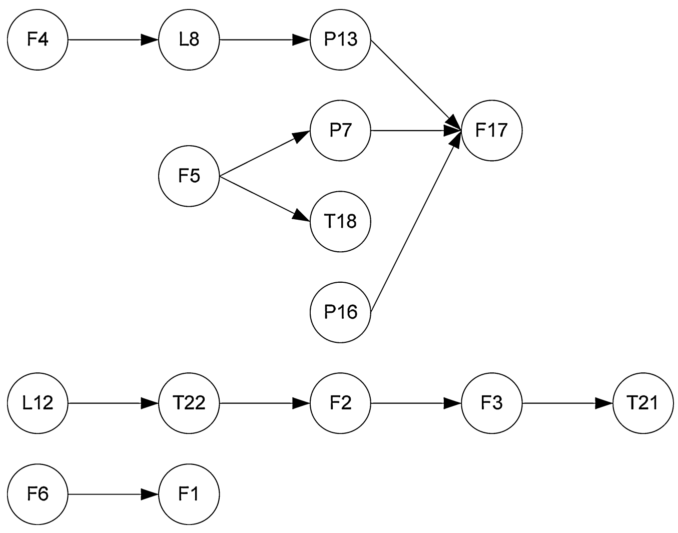
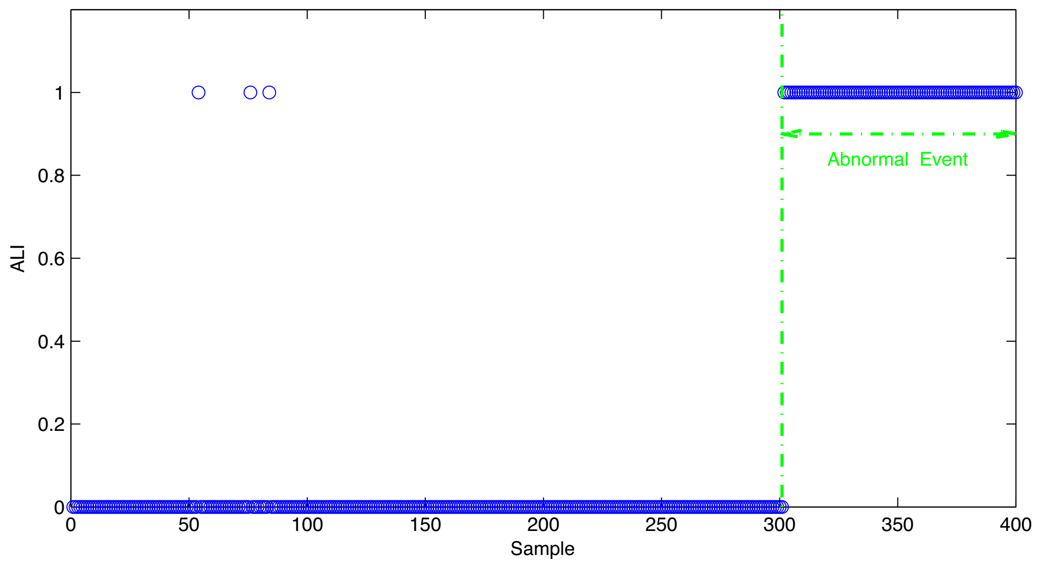


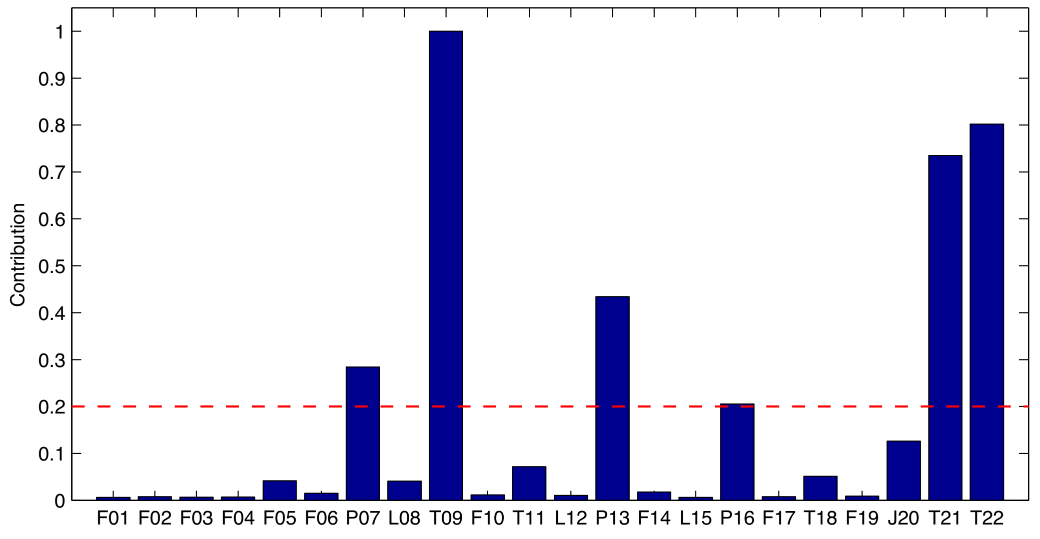
Fig. 7. Causal map of transfer entropy method in the case of IDV(4).

these fixed root nodes and leaf nodes are denoted as $\mathcal{R}^k \in \mathcal{D}^k$ and $\mathcal{L}^k \in \mathcal{D}^k$ respectively ($\mathcal{R}^k \cap \mathcal{L}^k = \emptyset$).

Moreover, in order to restrict the solution space further, some predetermined variable ordering $\prec$ over x is introduced. If the ordering covers order relationships over all variables as $x_1 \prec x_2, \ldots,$



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 8. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(4).

x_l and the maximum number of parents for each node set to be at most d, the number of possible parent sets for x_i is at most

$$\binom{i-1}{d}$$

rather than 2^{l-1} (Koller and Friedman, 2009). However, in general these complete orderings are very difficult to set and they require in-depth process knowledge to identify. Instead of setting complete orderings, incomplete orderings are proposed here. Unlike the complete ordering, the incomplete ordering does not cover order relationships over all variables but restrict the ordering over subsets of variables. The followings are the example of the incomplete ordering.

$$\{x_1, x_2, x_3\} \prec \{x_4, x_5, x_6, x_7\} \quad (15)$$

This incomplete ordering suggests that a computed network should be consistent with the order relationships where x_1, x_2 and x_3 precede x_4, x_5, x_6 and x_7, but any constraints about orders within each subset are not imposed. Similar to the fixed root nodes and leaf nodes, the incomplete orderings are optional to set for the structure learning, but they can further reduce the computational cost because they make the combinational search space much smaller. Fortunately, some incomplete orderings are almost automatically determined from the process flow diagram. For instance, the variables monitored in the upper units must precede the ones monitored in the lower units.

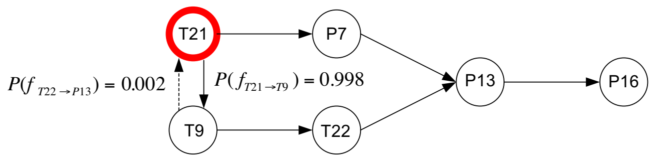


Fig. 9. Identified fault propagation pathway in the case of IDV(4).

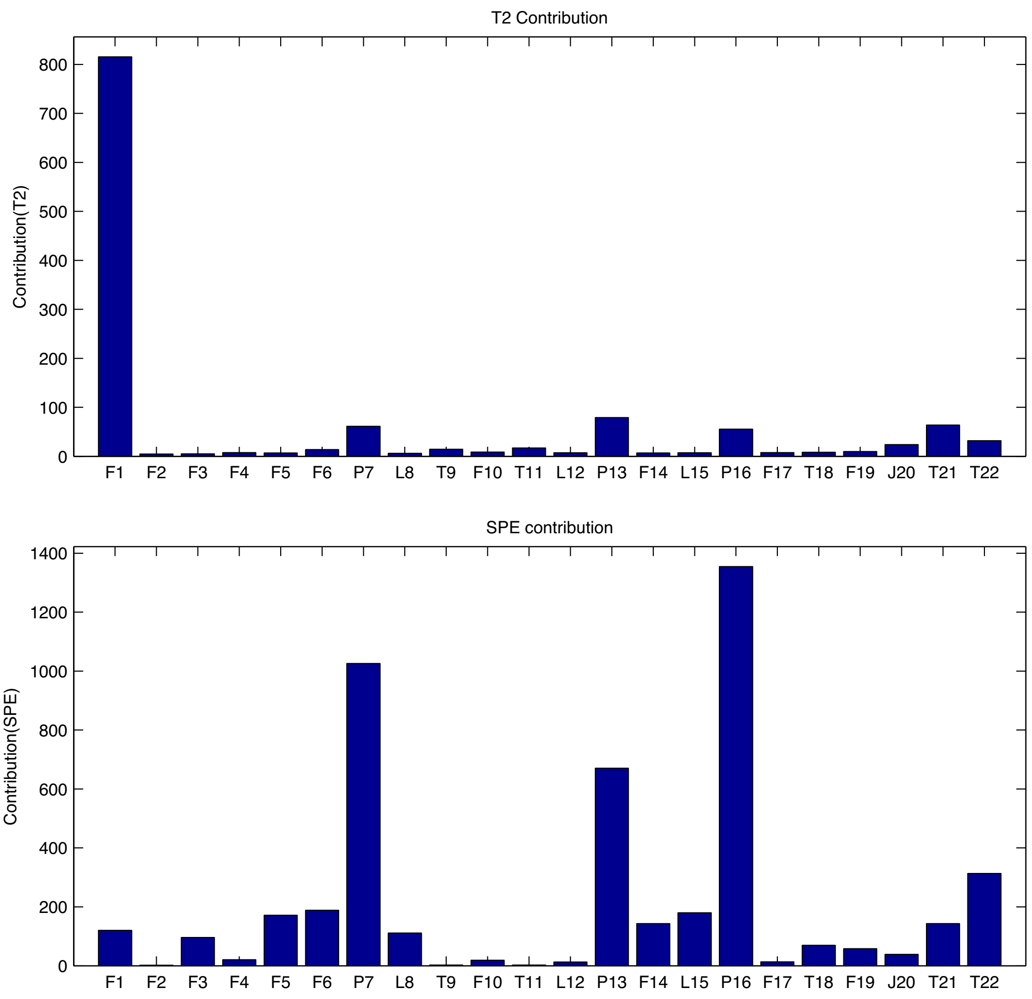


Fig. 10. Variable contribution plots of PCA based process monitoring in the case of IDV(6).

3.2. Optimization algorithm

For optimization of subnetworks, the search space is restricted to the neighbouring graphs that can be obtained by either adding one edge, deleting one edge, or reversing the existing edge as shown in Fig. 1 (Koller and Friedman, 2009). The tabu search algorithm is adopted to decide whether the neighbouring graphs are accepted (Glover, 1986). Tabu search employs a neighbour search algorithm to move from one solution candidate to an improved solution in the neighbourhood unless the termination criterion is satisfied. In order to avoid local minima as much as possible, this algorithm allows one solution candidate to move to a worse solution unless that tabu list includes that solution. Tabu list is a set of solutions that have been visited over the last *t* iterations, where *t* is termed as tabu size and should be specified in advance. In this work, the tabu size is set to be 100. It should be noted that the neighbouring graphs should satisfy the constraints including the specified incomplete orderings. Therefore, the illegal graphs that violate these constraints are removed from neighbouring solutions during searching.

In addition, we employ both the first admissible move strategy and the best admissible move strategy to update the current solution. In the first admissible move strategy, a current graph is updated immediately after better one is found, while in the best admissible move strategy, a graph is updated to the best graph after searching all possible neighbouring solutions. The first admissible move strategy can find passable solution very quickly but may move slowly around the local minima. Meanwhile, the best admissible move strategy can search deeply for the local minima but due to the large size of search space, it takes a quite a lot of time to move to passable solutions around the local minima. Therefore, the first admissible move strategy is employed until the passable solution is found, and then the strategy is changed to the best admissible move strategy in order to find the optimal solution unless the termination criteria is satisfied. This enables discovery of the single high-scoring subgraphs on all subsets.

3.3. Graph identification

It is very common in structure learning that some networks have similar scores as the optimal one. If we are interested in the density estimation only and do not pay attention to the graph structure in itself, we can ignore the networks that have similar scores. However, our main purpose is to identify the cause–effect relationships such as root-cause variables and propagation pathways that describe process upsets, and hence we are very interested in the network structure rather than the density estimation. Therefore, we should not ignore these high-scoring graphs. The proposed optimization algorithm can find the single high-scoring subgraph for all subsets of training data. Instead of identifying the one unique highest-scoring graph, the posterior probabilities of corresponding graph features are computed for all possible graphs and then the most probable network structure is identified. Graph feature in subset $\mathcal{D}^k$ is represented by the function $f^k_{i\to j}(\mathcal{G}^k)$ that returns 1 if

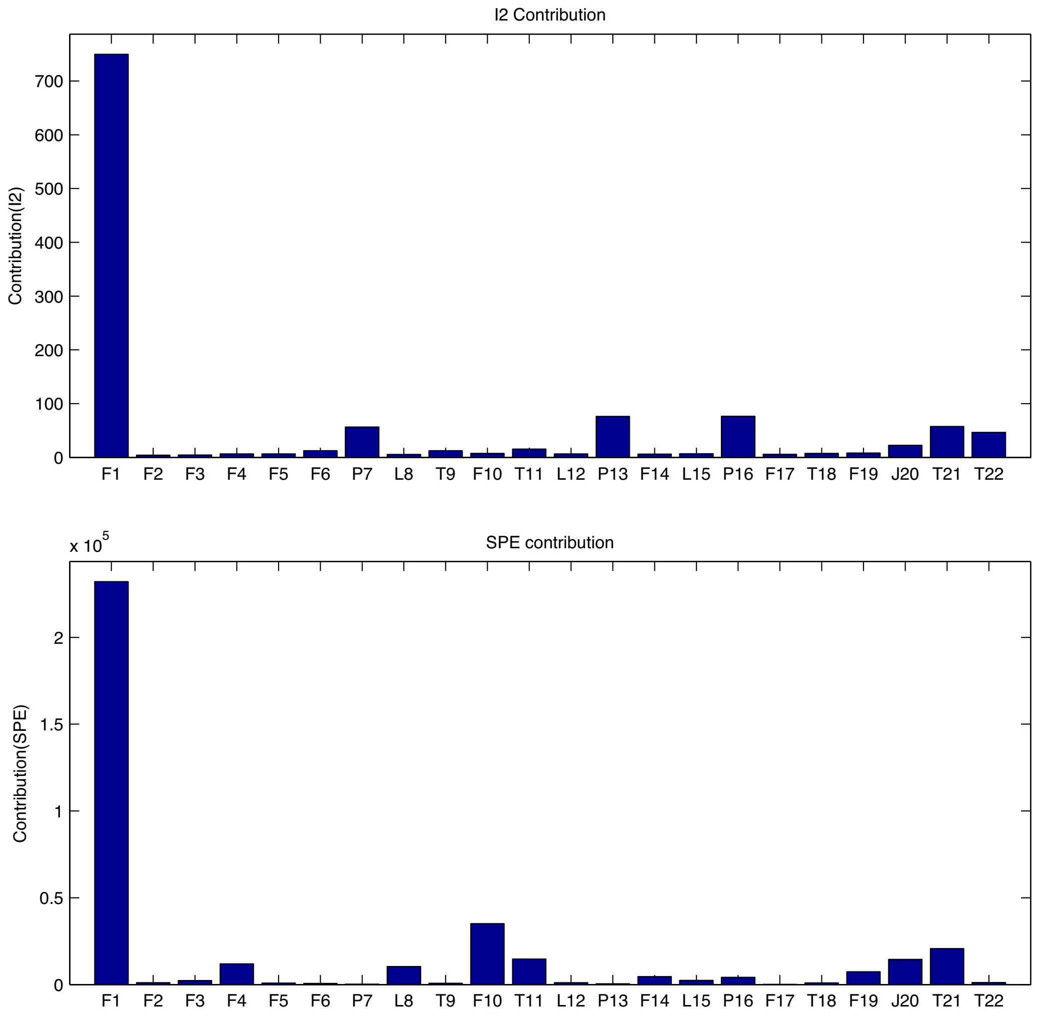


Fig. 11. Variable contribution plots of ICA based process monitoring in the case of IDV(6).

ith node is a parent of jth node and otherwise 0. Our goal is to compute the posterior probability of features $f^k_{i\to j}(\mathcal{G}^k)$ over all possible graphs $\mathcal{G}^k$ given the training data $\mathcal{D}^k$ as follows:

$$P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k) = \sum_{\mathcal{G}^k} f^k_{i\to j}(\mathcal{G}^k)P(\mathcal{G}^k|\mathcal{D}^k). \tag{16}$$

It should be noted that Bayesian score of a graph $\mathcal{G}^k$ is equal to its posterior probability $P(\mathcal{G}|\mathcal{D}^k)$. Since the number of potential graphs increases exponentially in the number of nodes, we cannot directly compute the posterior probability $P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k)$. Instead of using the all possible graphs, graph samples are used for computing the posterior probability. If we can generate the graph samples $\mathcal{G}^k_1, \ldots, \mathcal{G}^k_N$ that follow the true posterior distribution of $\mathcal{G}^k$ given $\mathcal{D}^k$, Eq. (16) can be approximated as follows:

$$P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k) \approx \frac{1}{N}\sum_{n=1}^{N} f^k_{i\to j}(\mathcal{G}^k_n). \tag{17}$$

Since we do not know the true posterior distribution but can compute the posterior probability $P(\mathcal{G}^k|\mathcal{D}^k)$, Markov chain Monte Carlo (MCMC) simulation can be utilized to obtain a sequence of graph samples $\mathcal{G}^k_1, \ldots, \mathcal{G}^k_N$. In order to construct the Markov chain, a standard Metropolis algorithm is used (Metropolis et al., 1953). Similar to the search space in the structure optimization, three operations of adding edge, deleting edge and reversing edge are considered to transform one structure to another. Then, we assume that a proposal distribution $\mathcal{T}^Q$ over these operations follow uniform distribution. Finally, we generate the graph samples in accordance with the Markov chain over the space of structure by accepting the graph sample $\mathcal{G}'^k$ transformed from the current graph $\mathcal{G}^k$ with the following probability (Friedman and Koller, 2003)

$$\min\left[1, \frac{P(\mathcal{G}'^k, \mathcal{D}^k)\mathcal{T}^Q(\mathcal{G}'^k \to \mathcal{G}^k)}{P(\mathcal{G}^k, \mathcal{D}^k)\mathcal{T}^Q(\mathcal{G}^k \to \mathcal{G}'^k)}\right]. \tag{18}$$

Markov chain is initialized from the structure G^* found by structure learning. Once the graph samples that follow the Markov chain over the space of structure are generated, the posterior probability of graph features $P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k)$ is computed for all potential edges.

With the posterior probability of graph features, the following statistical rules are introduced to identify the final probabilistic graphical model. First, among all potential edges, the features satisfying the following criteria are selected as the edge in the final model.

$$P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k) \geq \alpha \tag{19}$$

where α is the confidence limit. Next, bidirectional edges between ith and jth nodes are selected in the final graph when the corresponding posterior probabilities satisfies the following criteria:

$$P(f^k_{i\to j}(\mathcal{G}^k)|\mathcal{D}^k) + P(f^k_{j\to i}(\mathcal{G}^k)|\mathcal{D}^k) \geq \alpha \tag{20}$$

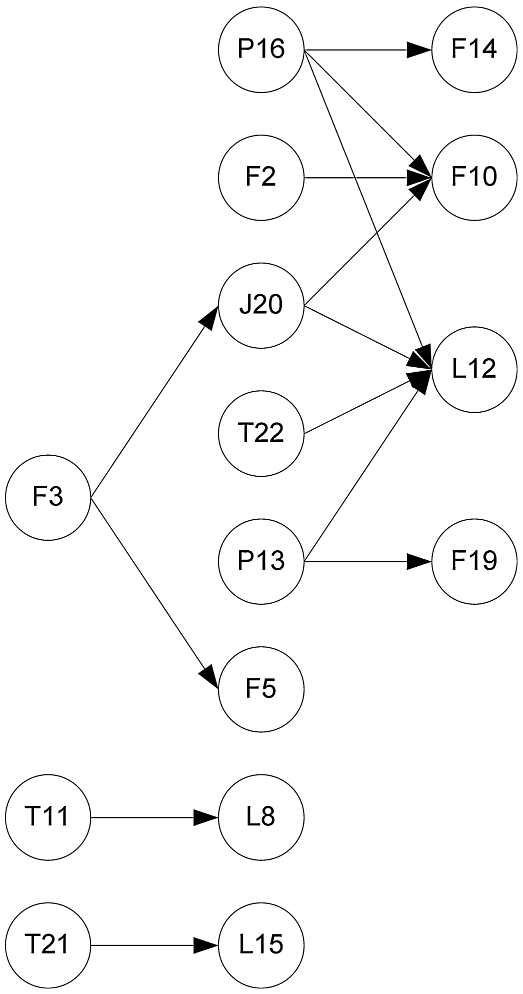


Fig. 12. Causal map of transfer entropy method in the case of IDV(6).

and

$$P(f_{i\to j}^k(\mathcal{G}^k)|\mathcal{D}^k) \geq \beta \tag{21}$$

$$P(f_{j\to i}^k(\mathcal{G}^k)|\mathcal{D}^k) \geq \beta \tag{22}$$

with the confidence limit β. In this work, α is set to be 0.9, which is a common value of confidence limits. The confidence limit β is set to be 0.1 since bidirectional edges that have less than 10% of existence probability are assumed to be noise in this paper. After obtaining the all subgraphs $\mathcal{G}^k$, they are combined into one entire graph in accordance with the incidence matrix.

4. Fault detection and root-cause identification

4.1. Fault detection

Since probabilistic graphical model has the ability to characterize the stochastic processes using conditional probability, it is possible to determine the operational status of processes, identify the fault propagation pathways and diagnose the root causes of abnormal event. After the model parameters are computed from historical data, the log-likelihood function for the new observation of the network nodes $\mathbf{x}^{\text{new}} = [x_1^{\text{new}}, x_2^{\text{new}}, \ldots, x_I^{\text{new}}]$ can be defined as follows:

$$\ln p(x_i^{\text{new}}) = \ln p(x_i^{\text{new}}|\text{pa}(x_i^{\text{new}})). \tag{23}$$

This equation represents the probability of each observable network nodes under normal operational conditions. In other words, the smaller the value of the log-likelihoods, the more probable is that abnormal event has occurred. Index γ is defined to identify the abnormal operation on each node.

$$\gamma(i) = \begin{cases} 1 & \text{if } \zeta(x_i^{\text{new}}) > c_i \\ 0 & \text{else} \end{cases} \tag{24}$$

where $\zeta(x_i^{\text{new}}) = -\ln p(x_i^{\text{new}})$ is a minus of the log-likelihood of x_i^{new} and c_i is the corresponding confidence limit that can be estimated from kernel density estimation (KDE) algorithm (Bishop, 2006). Given T samples $(\zeta(x_i^1), \zeta(x_i^2), \ldots, \zeta(x_i^T))$ from unknown probability distribution g_i, g_i can be estimated by Kernel density estimator as follows:

$$g_i(\zeta) = \frac{1}{TW}\sum_{t=1}^{T} k\left\{\frac{\zeta - \zeta(x_i^t)}{W}\right\} \tag{25}$$

where ζ denotes the random variable under consideration, W represents the kernel window width and k is a kernel function. The Gaussian kernel function is commonly used as follows

$$g_i(\zeta) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\sqrt{2\pi}W}\exp\left\{-\frac{(\zeta - \zeta(x_i^t))^2}{2W^2}\right\} \tag{26}$$

After a probability density function is estimated, the corresponding point with cumulative density function value at $1-\eta$ is the confidence limit under the confidence level of $\eta \times 100\,\%$. The new example x^{new} is identified as faulty operation data when the following *abnormal likelihood index* (ALI) is one.

$$\text{ALI} = \begin{cases} 1 & \text{if } \sum_{i}^{I}\gamma(i) \geq 1 \\ 0 & \text{else} \end{cases} \tag{27}$$

4.2. Root-cause diagnosis

With the detected abnormal operation, it is desirable to identify fault propagation pathways and diagnose root-cause variables in order to prevent or fix the fault operations. In this study, a Bayesian probability index (BPI) is developed from the conditional probability function $p(x_i^{\text{new}}(t)|\text{pa}(x_i^{\text{new}}(t)))$ of the ith node as follows:

$$\gamma_i(t) = \int_{x_i^1(t)}^{x_i^2(t)} p(x_i^{\text{new}}|\text{pa}(x_i^{\text{new}}(t)))dx \tag{28}$$

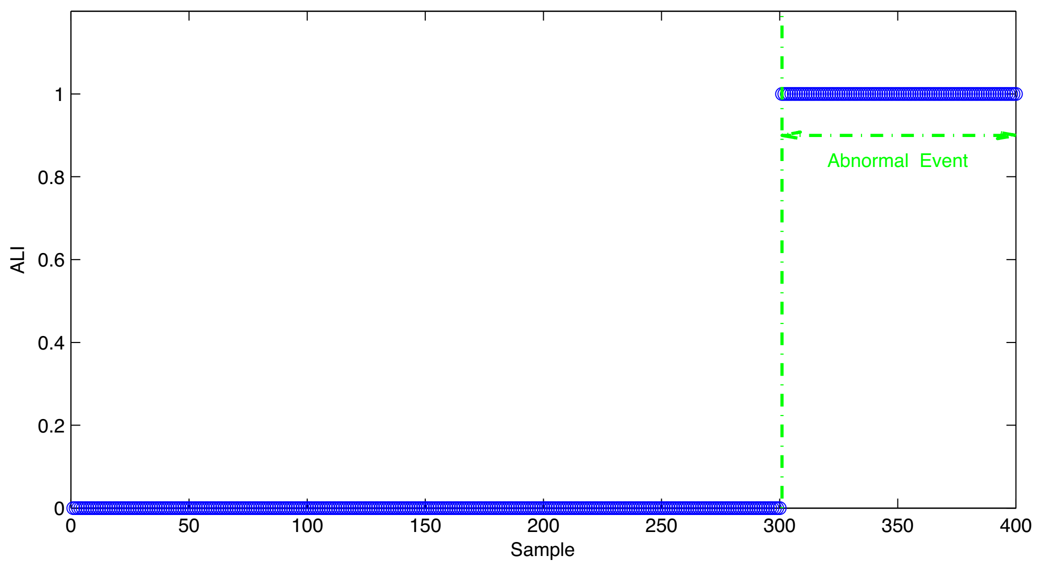
where

$$x_i^1(t) = \begin{cases} x_i^{\text{new}}(t) & (x_i^{\text{new}}(t) < \mu_i(t)) \\ 2\mu_i - x_i^{\text{new}}(t) & (x_i^{\text{new}}(t) \geq \mu_i(t)) \end{cases} \tag{29}$$

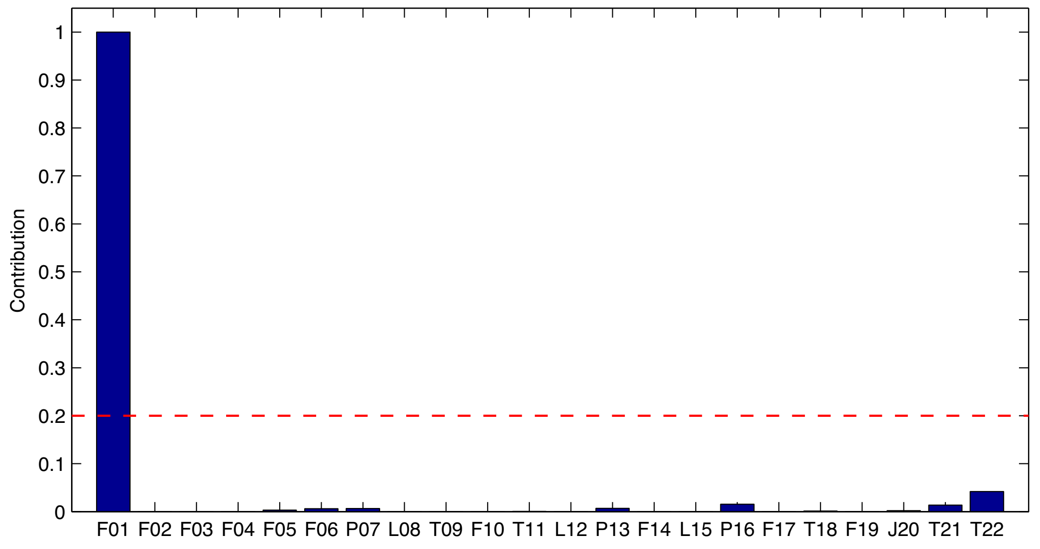
$$x_i^2(t) = \begin{cases} 2\mu_i - x_i^{\text{new}}(t) & (x_i^{\text{new}}(t) < \mu_i(t)) \\ x_i^{\text{new}}(t) & (x_i^{\text{new}}(t) \geq \mu_i(t)) \end{cases} \tag{30}$$

with $\mu_i(t) = \sum_{j\in\text{pa}(x_i^{\text{new}}(t))} w_{i,j}x_j^{\text{new}}(t) + b_j$. BPI is a type of cumulative distribution function and can be used to quantify the effect of each monitored variable on the abnormal event by its parent nodes in a probabilistic manner. With the BPI values, the following Bayesian contribution index is defined to identify the root-cause variables:

$$\Gamma_i = \frac{1}{|\mathbf{T}|}\sum_{t\in\mathbf{T}}\gamma_i(t) \tag{31}$$



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 13. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(6).

where $\mathbf{T} = [t_0 - t', t_0 - t' + 1, \ldots, t_0 + t' - 1, t_0 + t']$ is a sampling period with the time t_0 when the abnormal event is first detected and t' should be specified by users. Γ_i is the mean of BPI over $\mathbf{T}$ and named Bayesian contribution index (BCI) and it represents the likelihood of each process variable with significantly abnormal behaviour and can be utilized as an indicator identifying the root causes due to process faults. Then, a subset of potential root-cause variables, whose BCI values are greater than the confidence limit ϵ, is extracted as follows:

$$\mathbf{Z} = \{\eta : \Gamma_\eta \geq \epsilon\}. \tag{32}$$

When only one variable is included in $\mathbf{Z}$, it can be isolated as the root-cause variable leading to process upset. On the other hand, in the case when $|\mathbf{Z}| \geq 2$, the short paths that connect potential root-cause variables in $\mathbf{Z}$ so that all of them are connected are selected as the fault propagation pathways and then the root nodes of the identified paths are isolated as the root-cause variable due to abnormal operations. However, this approach gives us the question of how we identify the root-cause variables when more than 2 variables become the root-cause variables in the identified pathways. Our approach is to select the pathways that have the largest posterior probability given the faulty data $\mathcal{D}^{\text{new},k} = \left\{\mathbf{x}(t) : t \in \mathbf{T}\right\}$ among them. Similar to the approach for identification of the network structure, the posterior probability of graph features $f_{i\to j}$ given the faulty data $\mathcal{D}^{\text{new},k}$ as follows:

$$P(f^k_{i\to j}(\mathcal{G}^{\text{new},k})|\mathcal{D}^{\text{new},k}) \approx \frac{1}{N}\sum_{n=1}^{N} f^k_{i\to j}(\mathcal{G}^{\text{new},k}_n) \tag{33}$$

where graph sample $\mathcal{G}'^{\text{new},k}$ can be generated from Markov chain where the new sample is accepted as the following probability:

$$\min\left[1, \frac{P(\mathcal{G}'^{\text{new},k}, \mathcal{D}^{\text{new},k})\mathcal{T}^Q(\mathcal{G}'^{\text{new},k} \to \mathcal{G}^{\text{new},k})}{P(\mathcal{G}^{\text{new},k}, \mathcal{D}^{\text{new},k})\mathcal{T}^Q(\mathcal{G}^{\text{new},k} \to \mathcal{G}'^{\text{new},k})}\right]. \tag{34}$$

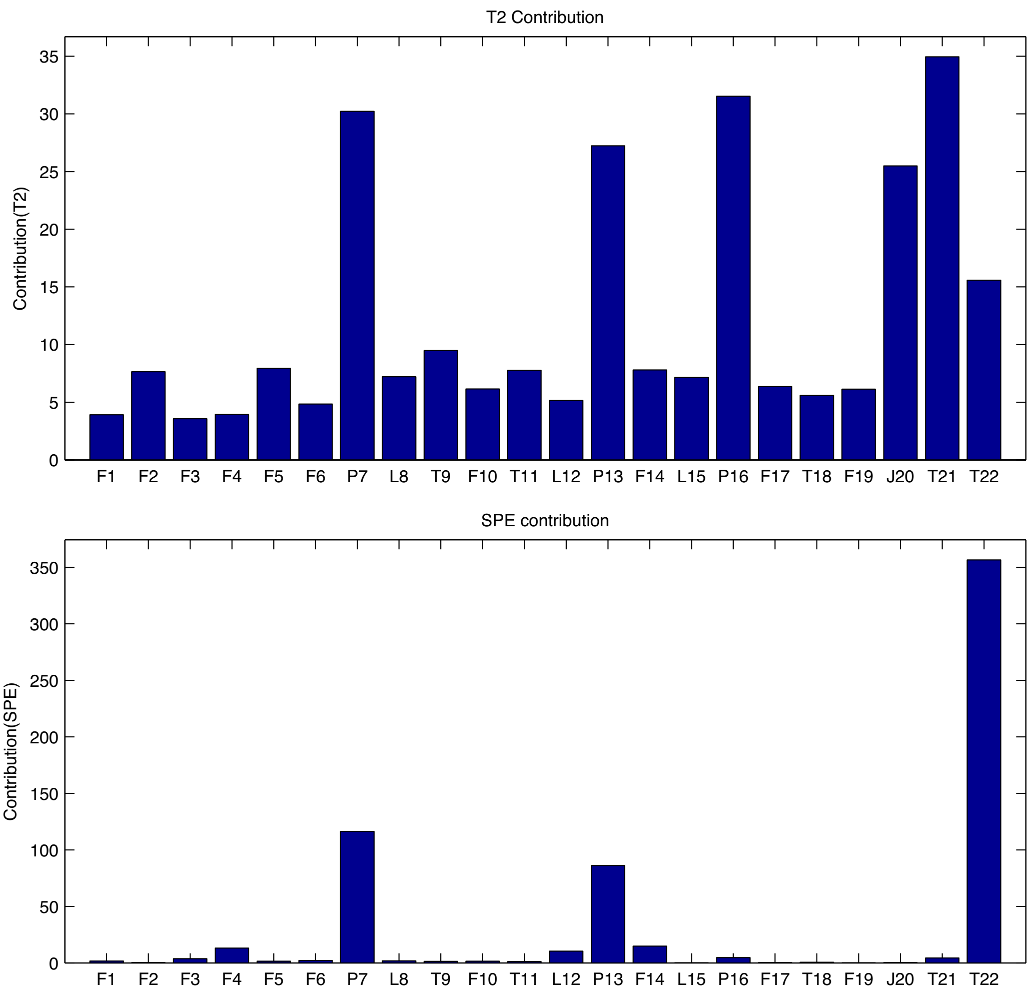


Fig. 14. Variable contribution plots of PCA based process monitoring in the case of IDV(13).

A proposal distribution $\mathcal{T}^Q$ over these operations assumes to be uniform distribution and Markov chain is initialized from the structure $\mathcal{G}^{*k}$ found by structure learning. Then the edge that has the highest probability among competing edges is identified as the most probable one below:

$$j^* = \underset{j \in \mathbf{Z} \cap pa_{\mathcal{G}^{*k}}(x_i)}{\mathrm{argmax}} P(f^k_{i \to j}(\mathcal{G}^{\mathrm{new},k}) | \mathcal{D}^{\mathrm{new},k}) \quad (35)$$

The proposed structure learning framework for probabilistic graphical network is illustrated in Fig. 2. Step-by-step procedure of the presented networked process diagnosis methodology is listed below:

(1) Create the (equipment, measured variables) incidence matrix from process flow diagram;
(2) Divide the set of measured process variables into smaller subsets in accordance with the incidence matrix;
(3) Set root nodes $\mathcal{R}^k$ and leaf nodes $\mathcal{L}^k$ for all unit $k = (1, 2, \ldots, K)$ by the user if necessary;
(4) Set incomplete ordering by the user if possible;
(5) Solve the optimization problem (14) from the historical process data and obtain an optimal graphs $\mathcal{G}^{*k}$ for all k;
(6) Generate graph samples from the MCMC simulation and compute the posterior probability over the structure;
(7) Determine the final sub-network structure based on the posterior probability over the structure;
(8) Combine all subgraphs into one entire graph;
(9) Learn the parameters of linear Gaussian model from the historical process data by solving Eqs. (4) and (5);
(10) Compute the log-likelihood and determine the confidence limits using kernel density estimation;
(11) Calculate the values of ALI for new process data and determine abnormal operation when ALI(i) = 1;
(12) Calculate the BCI values of the detected faulty samples for all variable nodes in the network and produce the corresponding contribution plots;
(13) Identify the fault propagation pathways and root-cause variable using the BCI values and the posterior probability of edge over the potential graphs given the faulty operation data.

5. Application example

5.1. Tennessee Eastman Chemical process

In this study, the Tennessee Eastman Chemical process is utilized to examine the performance of the proposed networked process monitoring method. The diagram of the Tennessee Eastman Chemical process is shown in Fig. 3. This process has six major process units which are a feed mixer, a exothermic 2-phase reactor, a product condenser, a vapor-liquid separator, a recycle compressor and product stripper (Downs and Vogel, 1993). This process produces two liquid products of G and H, along with a byproduct of F from four gaseous reactants A, C, D and E. An inert B is fed into the reactor which formed G and H. The process has total 22

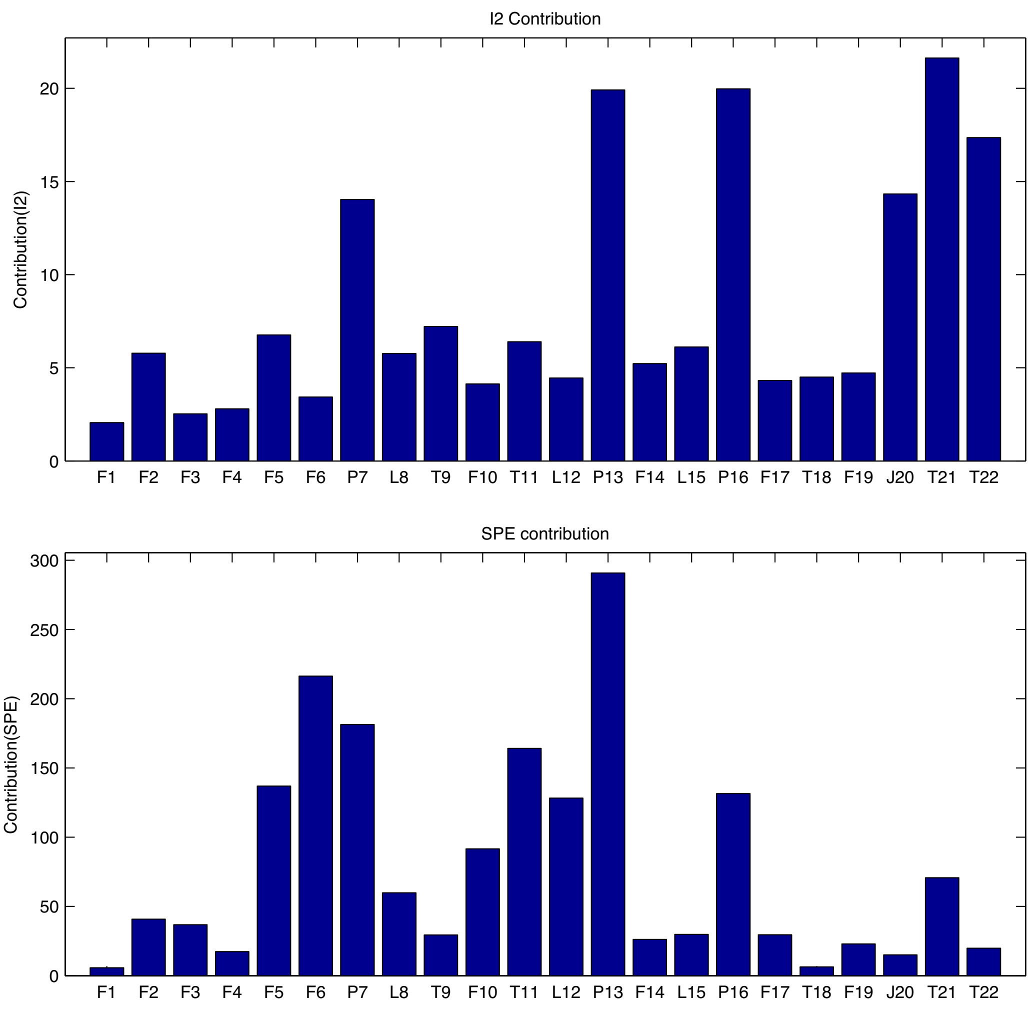


Fig. 15. Variable contribution plots of ICA based process monitoring in the case of IDV(13).

continuously measured variables, 16 sampled measured variables and 12 manipulated variables. The process includes a plant-wide decentralized control implementation with different feedback control loops.

For process monitoring purpose, 22 continuously measured variables among the total 41 variables are selected, as shown in Table 1. The variable symbols shown in the table are used later in this section. The sampling time of process measurement variables is set to 15 min. The training data set consisting of 1000 samples generated under normal operations is used to learn both network structure $\mathcal{G}$ and network parameters $\boldsymbol{\Theta}$.

Furthermore, 12 out of 15 predefined test cases are used to compare the monitoring performance. All the predefined test scenarios are shown in Table 2 (Downs and Vogel, 1993). The eliminated test cases of IDV(1), IDV(2) and IDV(8) are considered non-applicable, because all of three faulty operations are derived from the variation of the feed composition ratio and no variable related to component ratio is monitored, which means that it is difficult to select the most significant root-cause variable among the measured ones. In addition all of these 12 cases include both normal and faulty operations. In all the cases, the process begins with normal operating condition from the first through the 300th samples. Then the faulty operation occurs from the 301st through 400th samples.

As a comparison, the PCA and ICA based process monitoring methods are used to examine the effectiveness of the proposed algorithm. In both methods, the variable that has the largest value of contribution index is considered to be root-cause one. For the sake of fair comparison, the same diagnosis period $|\mathbf{T}| = 2[h]$ is used in the PCA, ICA and proposed methods. As for the causal map method (Chiang and Braatz, 2003), this method requires in-depth process knowledge to design the causal maps and we assume that the expert process knowledge is unavailable here, and hence this approach is not used for the comparison in this paper. Instead, purely data-driven causal matrix based process diagnosis method (Bauer et al., 2007) is compared with the proposed method. This method employs the transfer entropies to design the causal map in order to find the direction of disturbance propagation pathways. The diagnosis period $\mathbf{T}$ is set to be same value as the proposed method. For the existing Bayesian network based process diagnosis methods, unlike the proposed method, all the methods (Weidl et al., 2005; Dey and Stori, 2005; Huang, 2008; Alaeddini and Dogan, 2011) require the in-depth process knowledge to design the network structure, and hence they are excluded from the comparison in this paper.

5.2. Network structure identification

The first task in the proposed approach is to generate the incidence matrix from the process flow diagram and very superficial knowledge about processes. As shown in Fig. 3, the process includes 6 major units of the mixer, reactor, condenser, separator, stripper, and compressor, which correspond to the rows of the incidence matrix. Columns of the incidence matrix correspond to the total 22 measurement variables and thus the incidence matrix is a 6×22

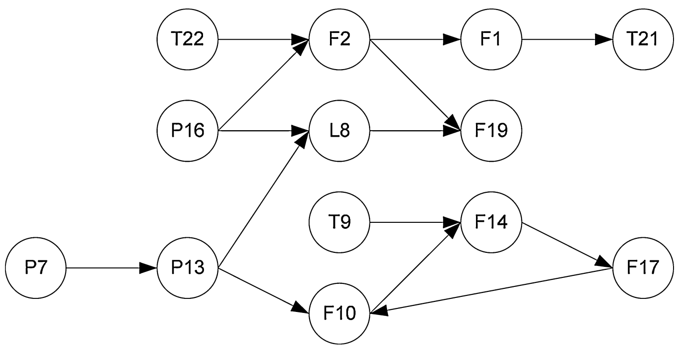


Fig. 16. Causal map of transfer entropy method in the case of IDV(13).

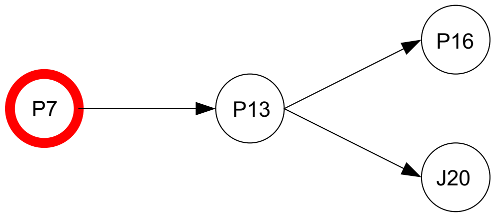
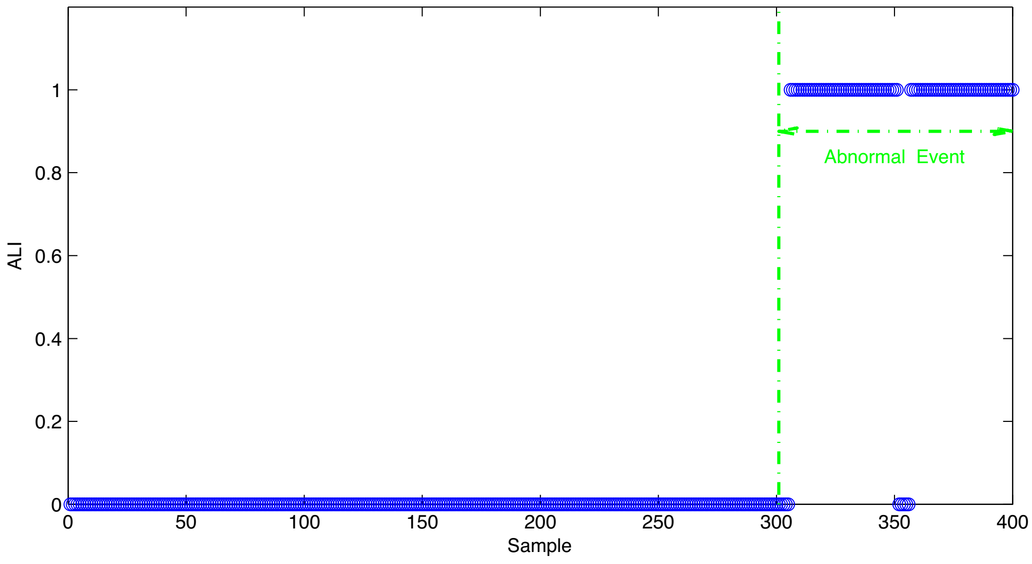
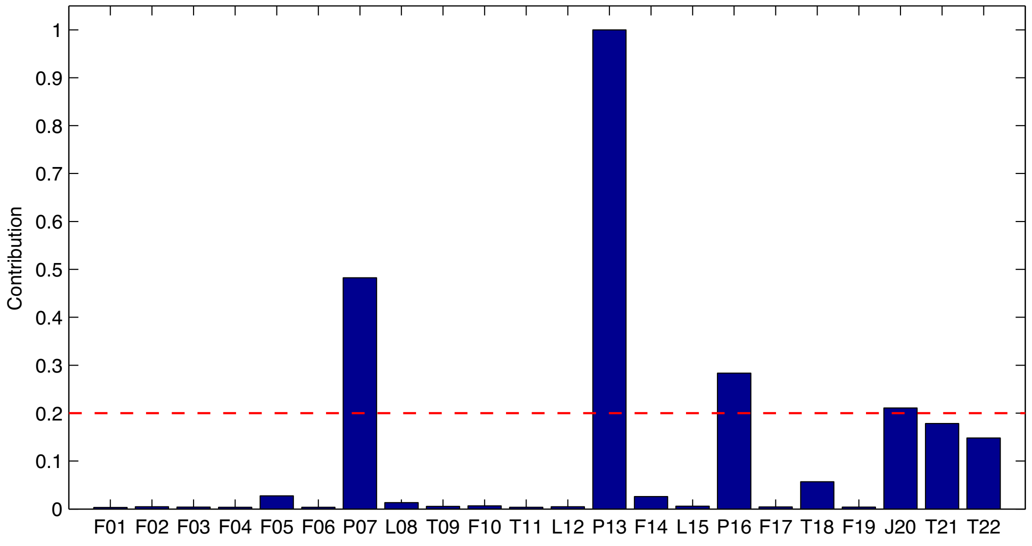


Fig. 18. Identified fault propagation pathway in the case of IDV(13).

matrix. F_1, F_2, F_3, F_5 and F_6 are measured in the mixer, P_7, L_8, T_9 and T_{21} are measured in the reactor, T_{22} is monitored in condenser, T_{11}, L_{12}, P_{13} and F_{14} are monitored in the separator, L_{15}, P_{16}, F_{17}, T_{18} and F_{19} are measured in the stripper, and J_{20} is monitored in the compressor. Hence, the corresponding entries are 1 in the incidence matrix. In addition, one can readily notice that variables monitored in the mixer have an influence on the adjacent reactor, so the corresponding entries in the row of reactor are 1. Similarly, some entries of the other rows are assigned 1 accordingly and finally the incidence matrix shown in Table 3 can be obtained.



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 17. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(13).

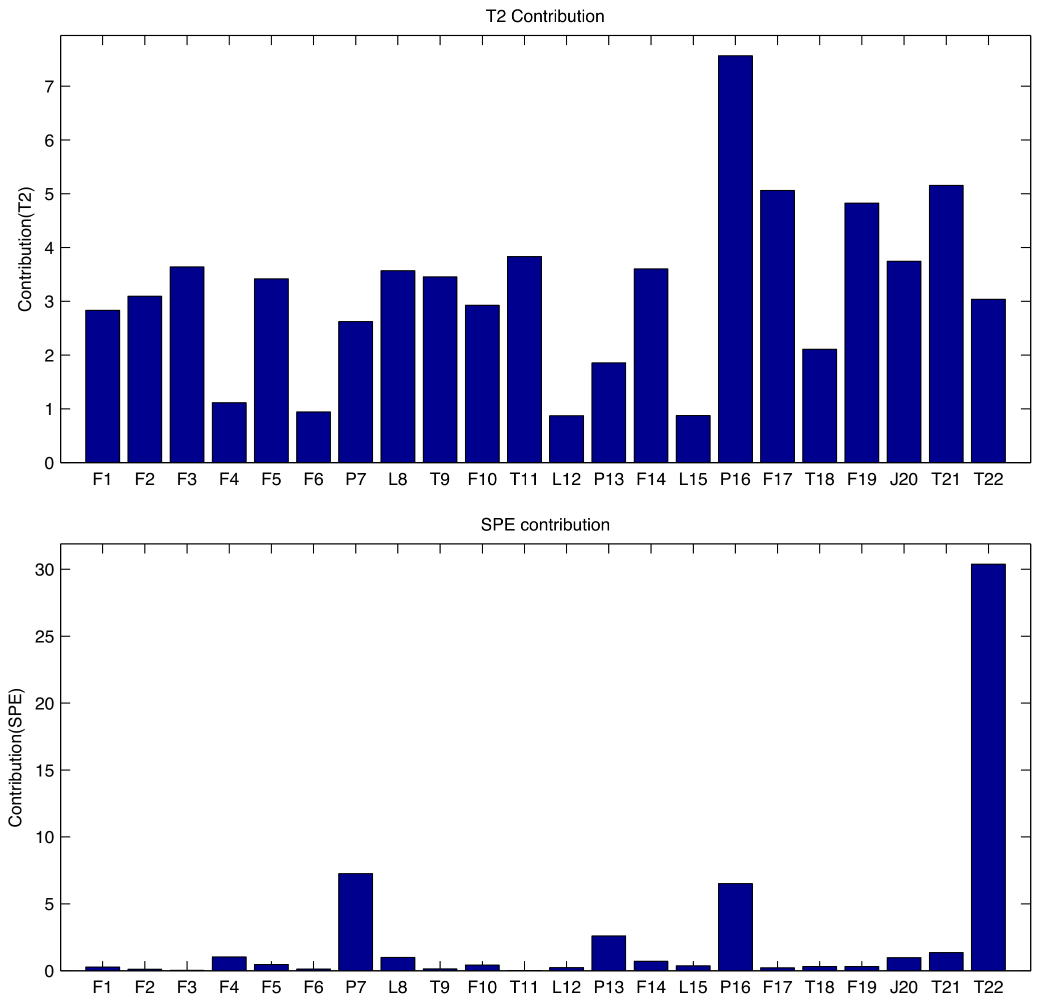


Fig. 19. Variable contribution plots of PCA based process monitoring in the case of IDV(3).

With the constructed incidence matrix, a set of all measured variables is broken into smaller subsets as follows:

$$\mathcal{D}^1 = \{F_1, F_2, F_3, F_5, F_6\}$$
$$\mathcal{D}^2 = \{F_1, F_2, F_3, F_5, F_6, P_7, L_8, T_9, T_{21}\}$$
$$\mathcal{D}^3 = \{T_9, T_{22}\}$$
$$\mathcal{D}^4 = \{P_7, L_8, T_9, T_{22}, T_{11}, L_{12}, P_{13}, F_{14}\}$$
$$\mathcal{D}^5 = \{T_{11}, L_{12}, P_{13}, F_{14}, F_4, L_{15}, P_{16}, F_{17}, T_{18}, F_{19}\}$$
$$\mathcal{D}^6 = \{F_5, T_{11}, L_{12}, P_{13}, F_{10}, J_{20}\}.$$

The next step is to place any variables on root nodes or leaf nodes as appropriate. In this example, it is clear from process diagram as shown in Fig. 3 that A feed rate (F_1), D feed (F_2), E feed (F_3), and recycle flow (F_5) should be placed on the root nodes in the mixer, and total feed (F_4) should be placed on the root node in the stripper as follows:

$$\mathcal{R}^1 = \{F_1, F_2, F_3, F_5\}$$
$$\mathcal{R}^5 = \{F_4\}.$$

In addition, the following incomplete orderings are also readily identified from process flow diagram.

$$\{F_1, F_2, F_3, F_5\} \prec F_6$$
$$F_6 \prec \{P_7, L_8, T_9, T_{21}\}$$
$$T_9 \prec T_{22}$$
$$\{P_7, L_8, T_9, T_{22}\} \prec \{T_{11}, L_{12}, P_{13}, F_{14}\}$$
$$\{T_{11}, L_{12}, P_{13}, F_{14}, F_4\} \prec \{L_{15}, P_{16}, F_{17}, T_{18}, F_{19}\}$$
$$\{T_{11}, L_{12}, P_{13}\} \prec F_{10} \prec J_{20} \prec F_5.$$

After discovering the single high-scoring subgraphs, the posterior probability of the graph features over graph structures given the training data is computed for each subset. Table 4 shows the posterior probability of the graph features over graph structure of the reactor. In this table, for instance, the probability of the edge from T_{21} to L_8 is 0.974 while that of the reverse edge from L_8 to T_{21} is 0.026. Since the confidence limit α and β is set to be 0.9 and 0.1 respectively, the final subgraph of the reactor is obtained as shown in Fig. 4(b). In this way, all subgraphs are identified and then they are combined into the entire graph as shown in Fig. 4.

5.3. Comparison of process diagnosis results

Once network structure is obtained, the network parameters denoting the conditional probability density function of all nodes

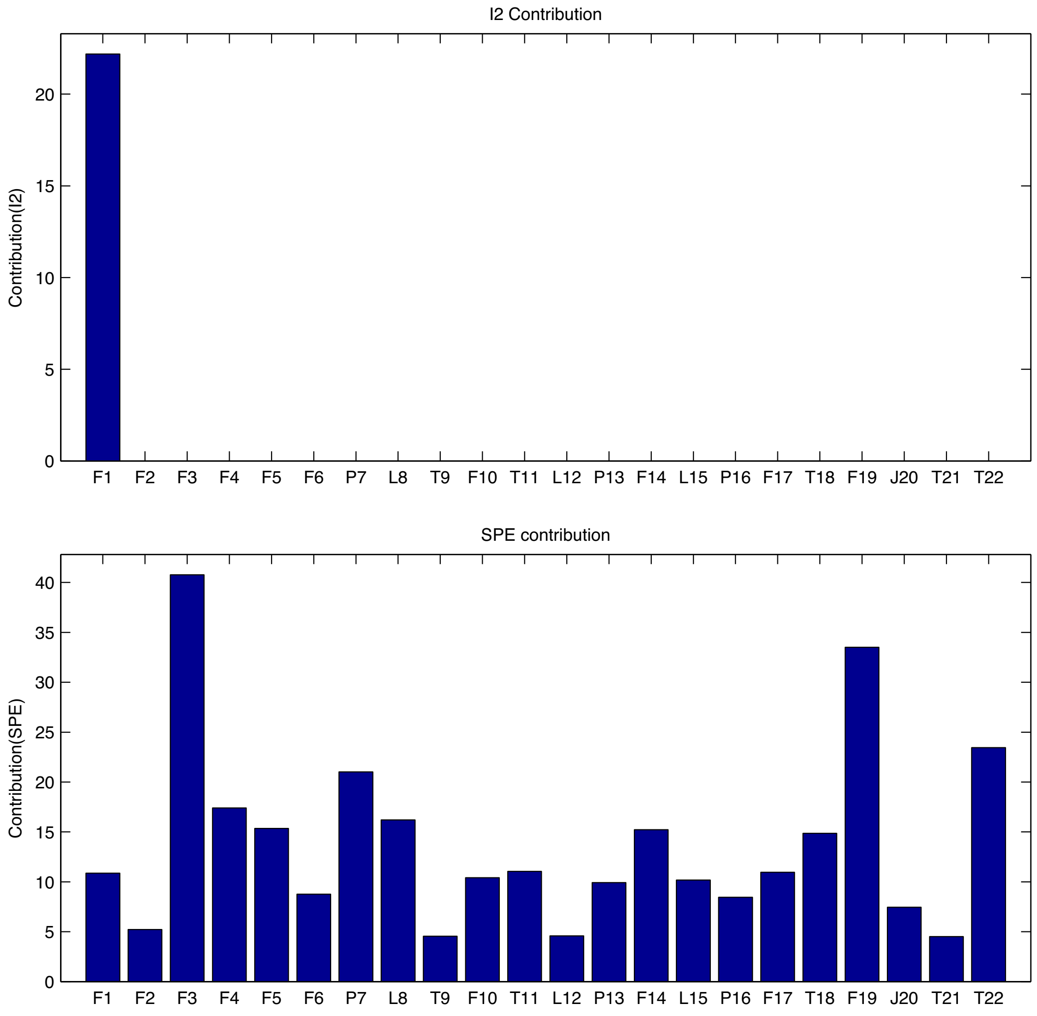


Fig. 20. Variable contribution plots of ICA based process monitoring in the case of IDV(3).

are computed from the training data. With the network model constructed, the abnormal likelihood index values for new process data are calculated for detecting the faulty operation. Once the abnormal event is detected, the fault propagation pathway is searched and the root-cause variable for the event is identified. As mentioned in the previous section, 12 out of 15 predefined test scenarios, IDV(3), IDV(4), IDV(5), IDV(6), IDV(7), IDV(9), IDV(10), IDV(11), IDV(12), IDV(13), IDV(14) and IDV(15) are used to examine the performance of the proposed method. In addition, the confidence limit ϵ is set to be 0.2, which means that the variables whose BCI value is greater than 0.2 are considered candidates for the root-cause variables. This value is determined from our experience. As for the transfer entropy method, the prediction horizon is set to be 4, which is recommended in the literature (Bauer et al., 2007). Furthermore, the literature recommends that the significant level of causality measures be greater than six sigma, but that value is too severe to accept any causality measures and thus we have set it to three sigma instead of six sigma in order to get reasonable causal maps in the transfer entropy method.

We will present discussion of three test scenarios. The remaining test scenarios are discussed in the supplementary material which is available in Appendix A.

In the case of IDV(4), the abnormal operating event is the step change of reactor cooling water inlet temperature from the 301st to the 400th samples. The ALI trend plot and the BCI values are shown in Fig. 8. One can readily see that the proposed method is

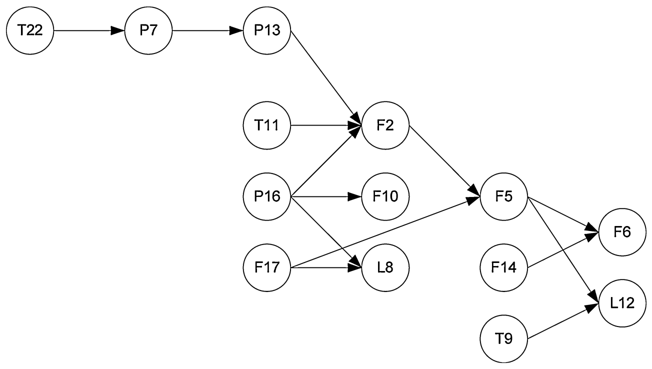
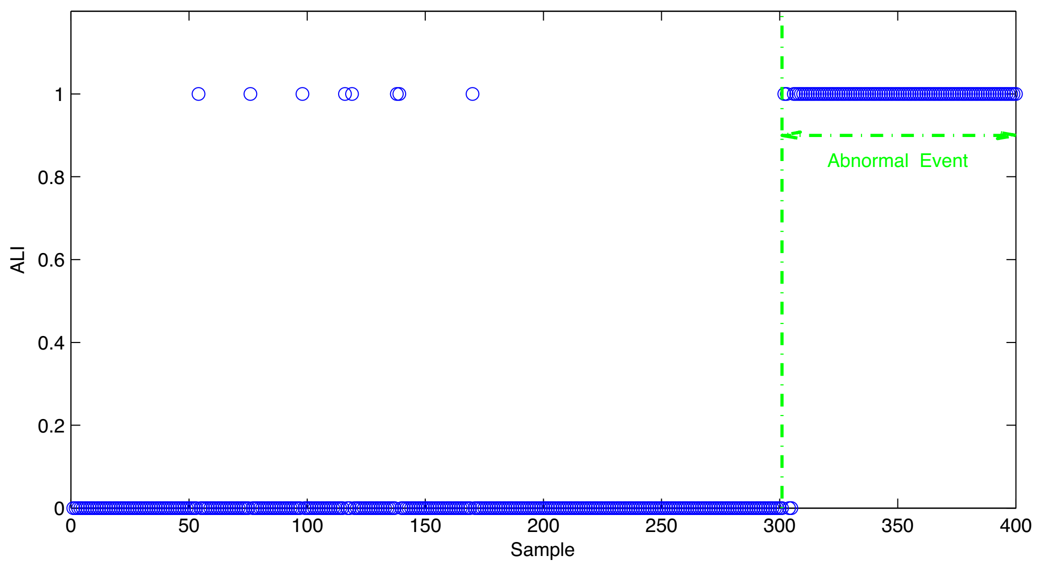
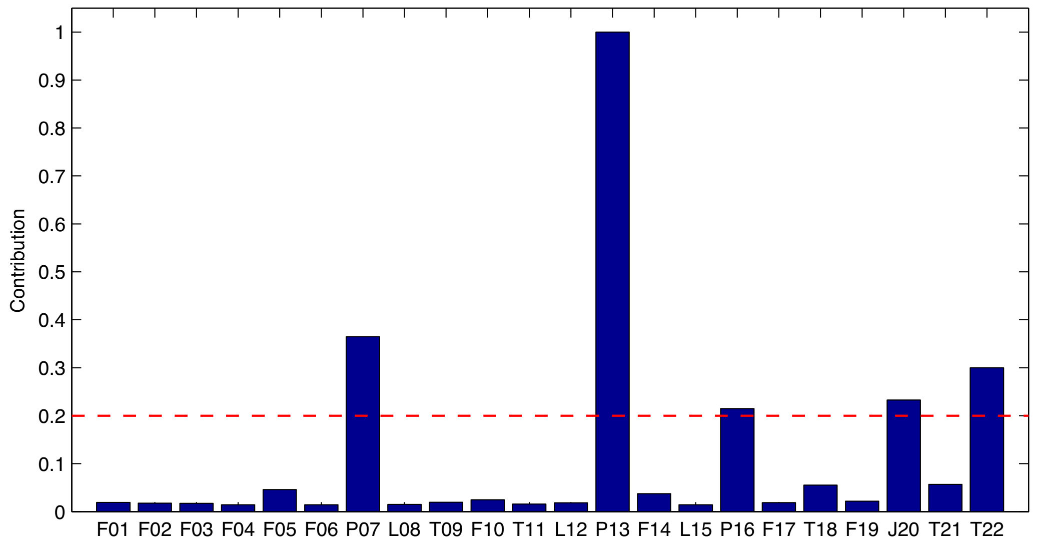


Fig. 21. Causal map of transfer entropy method in case 3.

able to accurately detect faulty operations. In addition, it can be seen that reactor pressure (P_7), reactor temperature (T_9), separator pressure (P_{13}), stripper pressure (P_{16}), reactor coolant water temperature (T_{21}), and condenser coolant temperature (T_{22}) are greater than the confidence level of 0.2. The identified fault propagation pathways are shown in Fig. 9. Since the bidirectional edge between T_{21} and T_9 is included in the pathways, the posterior probability of each directional edge is computed as $P(f_{T_{21}\rightarrow T_9}|\mathcal{D}^{\text{new}}) = 0.998$ and $P(f_{T_9\rightarrow T_{21}}|\mathcal{D}^{\text{new}}) = 0.002$. Therefore, it is concluded that reactor coolant water temperature (T_{21}) is located in the root node of the



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 22. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(3).

fault propagation pathways and accurately identified as the root cause of the process fault because this process upset is induced by the step change of the reactor coolant water temperature. Meanwhile, the PCA and ICA-based contribution plots are shown in Figs. 5 and 6. In the PCA-based T^2 contribution plot and the ICA-based I^2 one, it is true that both the contribution values of T_{21} are large, but they are not large enough compared with the other high-contribution variables and hence it is still challenging to identify the actual root-cause variables of T_{21}. As for both the ICA-based and the PCA-based SPE contribution plots, the contribution values of T_{21} are so small that it may be impossible to extract the root-cause variable based on these contribution plots. Furthermore, the causal map of the transfer entropy method is described in Fig. 7. The causal map indicates that F_4, F_5, L_{12} and F_6 are the root nodes of the identified network and they are the candidates of the root-cause variable that causes the process upset. However, the actual root-cause variable of T_{21} is not included among them, so the transfer entropy method cannot capture the root-cause variable in this test scenario. Meanwhile, the proposed method can well-isolate

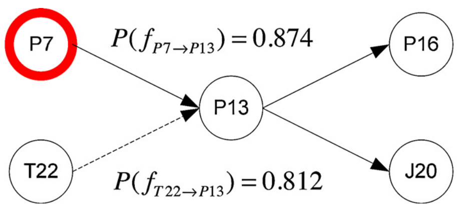


Fig. 23. Identified fault propagation pathway in the case of IDV(3).

the plausible fault propagation pathways and the true root-cause variable.

In the case of IDV(6), the process operation includes the A feed loss starting from the 301st samples. The process monitoring results of the ALI trend and BCI values are shown in Fig. 13. It is obvious that the ALI value remains 0 for the first 300 samples and then becomes 1 once the process fault of A feed loss takes place. After the faulty operation is captured, the BPI values can be generated. The BPI values of A feed (F_1) is far larger than that of any other

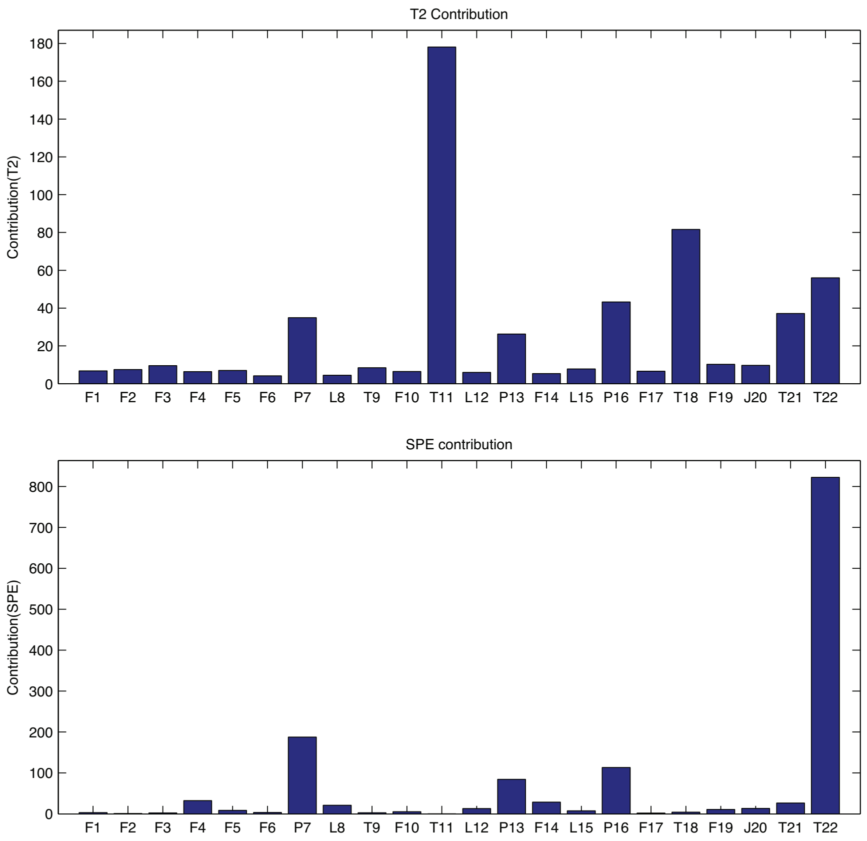


Fig. 24. Variable contribution plots of PCA based process monitoring in the case of IDV(5).

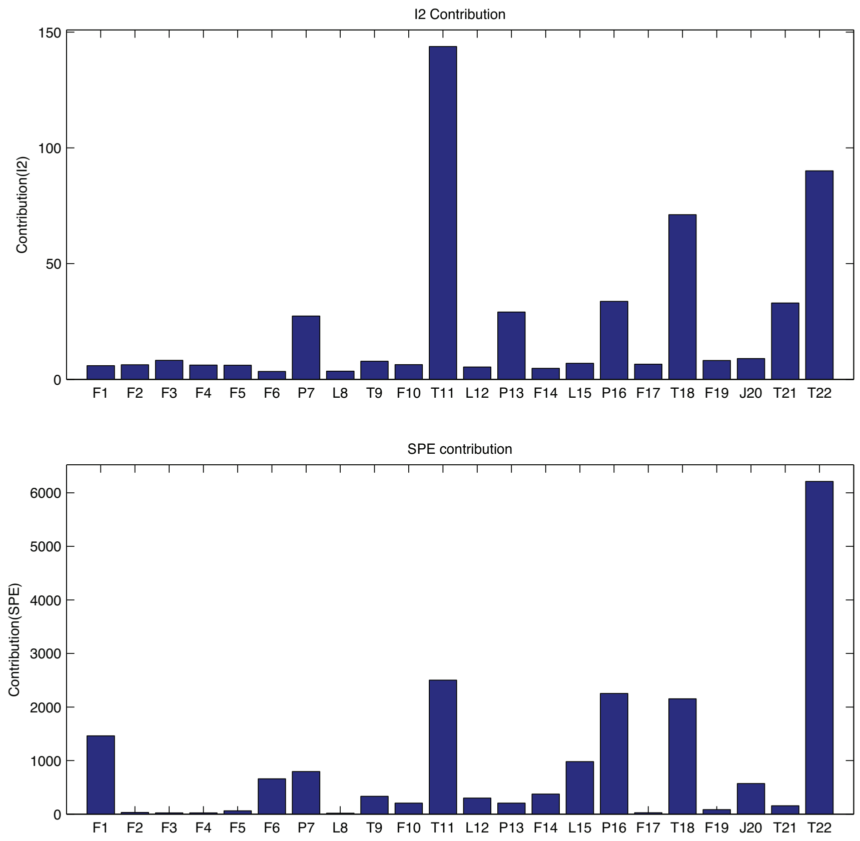


Fig. 25. Variable contribution plots of ICA based process monitoring in the case of IDV(5).

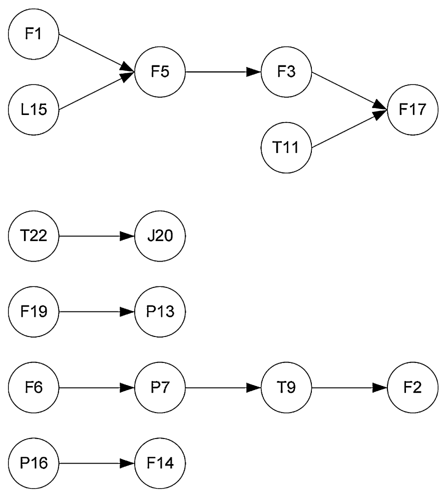


Fig. 26. Causal map of transfer entropy method in the case of IDV(5).

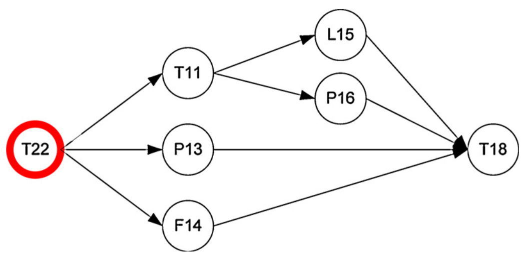
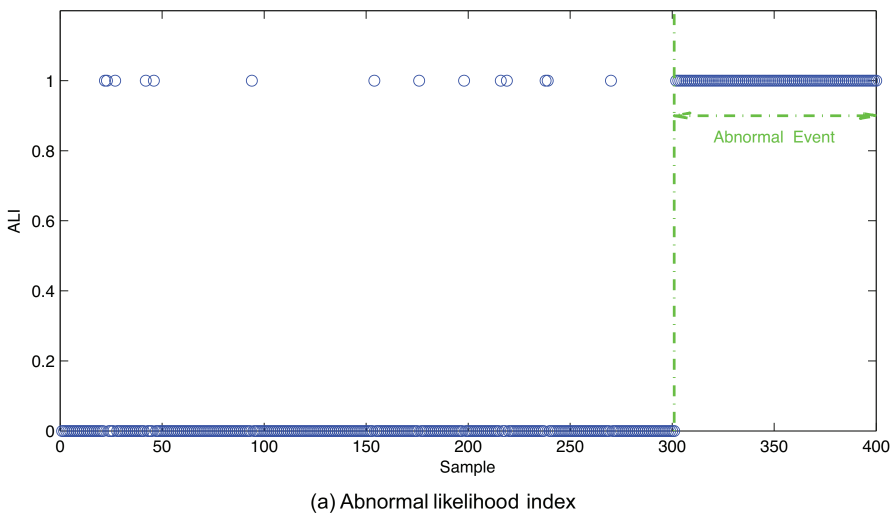


Fig. 28. Identified fault propagation pathway in the case of IDV(5).

variable and this means that A feed rate (F_1) is identified as the root-cause variable, which is consistent with the actual root cause. As a comparison, the PCA and ICA-based contribution plots shown in Figs. 10 and 11 indicate that the contribution values except for PCA-based SPE are so larger than that of other variables that they can be accurately identified as the root-cause variable. As for PCA-based SPE, the contribution values do not reflect the true fault causes. On the other hand, the causality map is computed by the transfer entropy method in order to capture the root-cause variable and the fault propagation pathways. As this map shows, the set of potential root-cause variables of F_3, T_{11} and T_{21} exclude the actual root-cause variable of F_1. In contrast, the proposed networked process monitoring can capture the actual fault propagation pathway due to its well described cause–effect relationship (Fig. 4).

In the test case of IDV(13), the trend plot of the ALI and BCI-based contribution plot are shown in Fig. 17. It is readily observed that the ALI values remain zero and then jump to the one once the slow drift of reaction kinetics takes place. Further, it can be seen



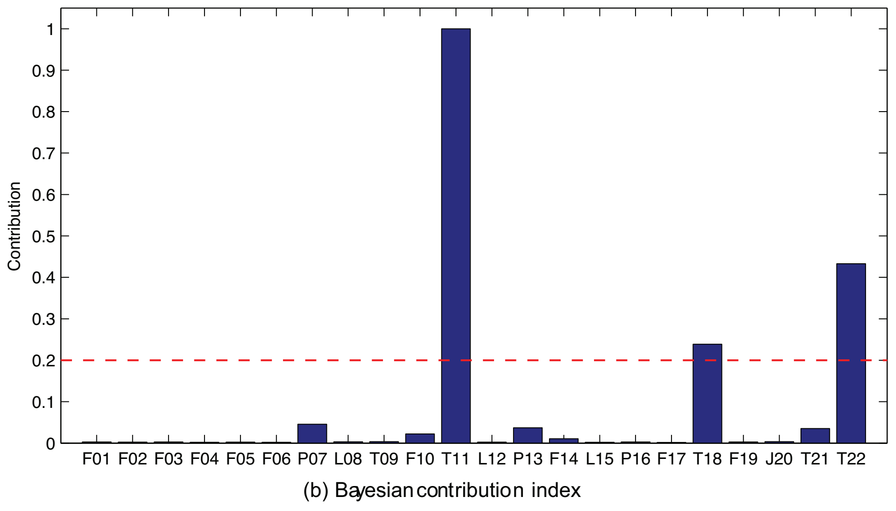


Fig. 27. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(5).

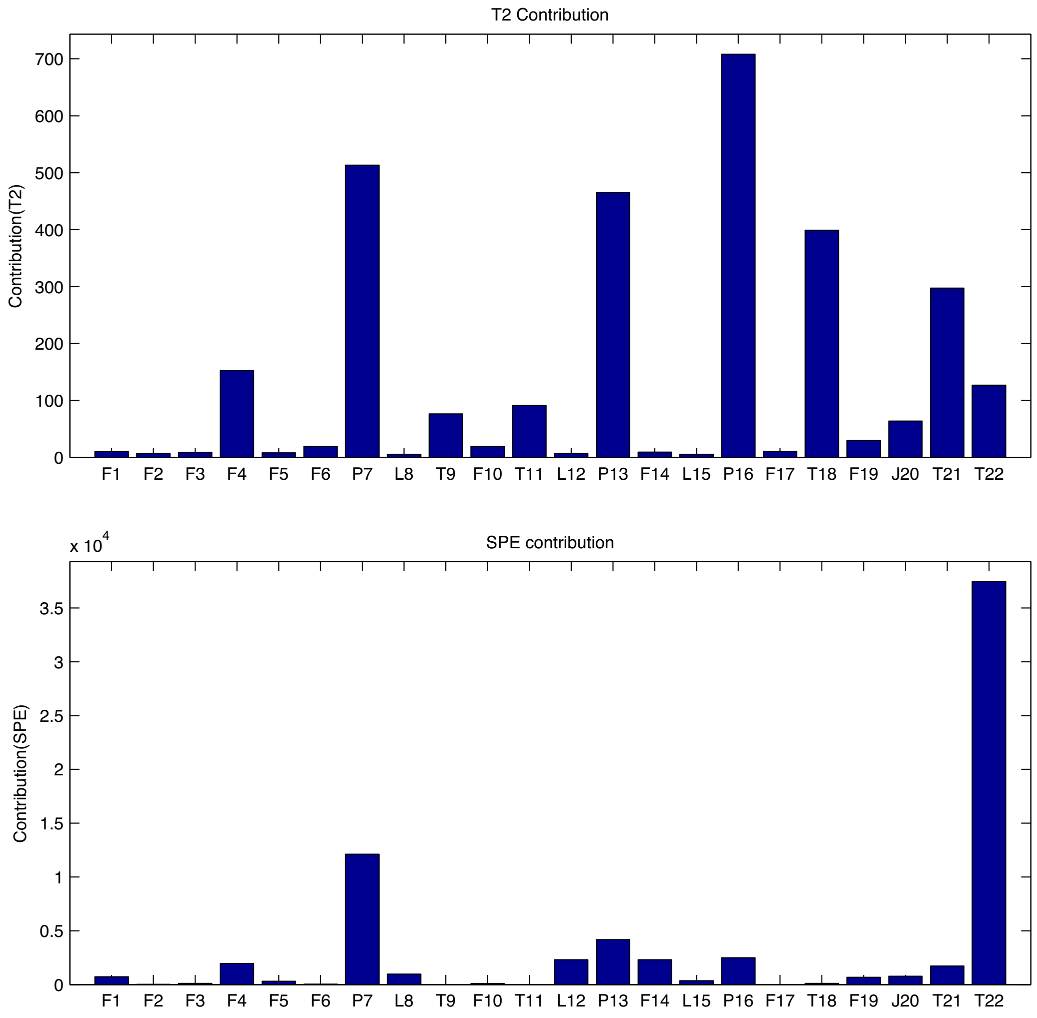


Fig. 29. Variable contribution plots of PCA based process monitoring in the case of IDV(7).

that four leading variables with higher BCI values than the confidence level are reactor temperature (P_7), separator pressure (P_{13}), stripper pressure (P_{16}) and compressor work (J_{20}). Since there are 4 potential root-cause variables, the fault propagation pathways are analyzed based on the constructed network structure. The computed pathways are described in Fig. 18. It is obvious that reactor pressure (P_7) is the root node of the identified pathways. Meanwhile, the variation of reaction kinetics is directly connected with the reactor pressure, and thus the identified root-cause variable of P_7 is accurately placed on the root cause of the process upset. As a comparison, the PCA and ICA-based variable contribution plots are shown in Figs. 14 and 15. One can easily see that the leading variables that have large contribution values include reactor pressure P_7 in all contribution plots, but these plots are unable to correctly point the faulty variable. Furthermore, the transfer entropy method is used in order to compute the causal map as shown in Fig. 16. It can be observed that the root nodes of the network are T_{22}, P_{16}, and P_7, which include the actual root-cause variable of P_7, but cannot isolate it among the three potential root-cause variables. In contrast, the proposed probabilistic networked process monitoring approach can further identify the root-cause variable.

Table 5 shows the root-cause identification results of the proposed probabilistic graphical network based process monitoring, PCA and ICA-based contribution plot methods and the transfer entropy approach. It should be pointed out that the PCA and ICA-based contribution plots can accurately identify the most significant variable that ranks 1st as the root-cause variable in 3 or 4 test cases. T^2 contribution, for instance, can well-isolate the root-cause variable in four test cases of IDV(6), IDV(7), IDV(10) and IDV(14). As for the transfer entropy method, in 4 out of 12 test cases the sets of the potential root-cause variables include the true root-cause ones, but the number of the potential root-cause variables is between 4 and 6. This means that the transfer entropy approach cannot well-isolate the actual root-cause variable. By contrast, the proposed graphical network process monitoring approach can identify the true root-cause variable in 9 out of 12 test scenarios, which is the highest identification rate among all methods. Such comparison shows that the proposed approach has the unique capability to locate the true root-cause variable due to the complicated network structure based on both the historical data and the structural information such as the incidence matrix.

6. Conclusion

In this study, a novel probabilistic graphical model based networked process monitoring is proposed for the identification of the root-cause variable. The main contribution of this research is a methodology to construct the most probable probabilistic graphical model from historical process data as well as the process measurement incidence matrix, and our approach requires very little knowledge about the process. First, the incidence matrix is

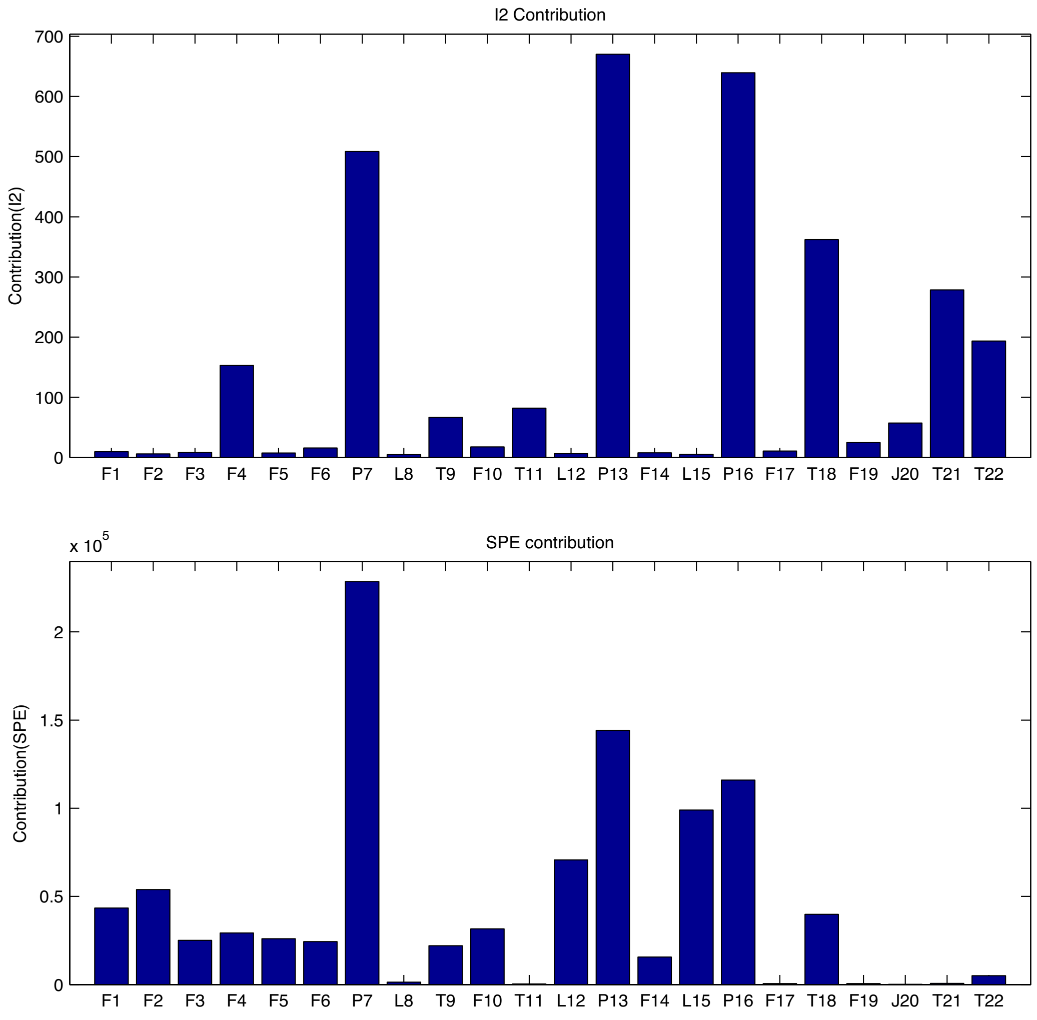


Fig. 30. Variable contribution plots of ICA based process monitoring in the case of IDV(7).

created from the process flow diagram. Then a set of monitored variables is broken into smaller subsets in accordance with the incidence matrix. Then subgraph corresponding to each subset is optimized from the historical process data in terms of Bayesian score that can take into account the trade-off between the likelihood and the model complexity. In general, several graphs have similar scores as the optimal network, which means that it is impossible to identify the true network structure from the historical data. Therefore, instead of identifying the true network structure, the most probable one is computed using graph samples generated from the Markov chain Monte Carlo simulation. After combining all the identified subgraphs into the one entire network, network parameters including conditional probability density functions of all different nodes are computed from the historical process data. Moreover, an abnormality likelihood index is developed to capture the abnormal operating condition. Once the fault is detected, Bayesian contribution indices are developed to capture the potential root-cause variables. In addition, for the further identification of root-cause variable in the case when there are more than two potential root-cause variables, the posterior probability of the edge is computed to isolate the true root-cause variable among them. The Tennessee Eastman Chemical process has been used to examine the effectiveness of the proposed method. The results demonstrate that the proposed method can accurately detect faults, identify fault propagation pathways and capture root-cause variable of process faults. Since the proposed method may not work well for time-varying system, future work will focus on employing dynamic Bayesian networks in order to take into account the dynamic nature of process data. In addition, in order to detect multiple root-cause variables, the current algorithm to identify fault propagation pathways will be modified in future work.

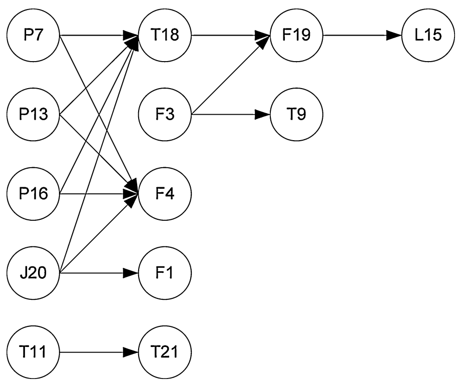
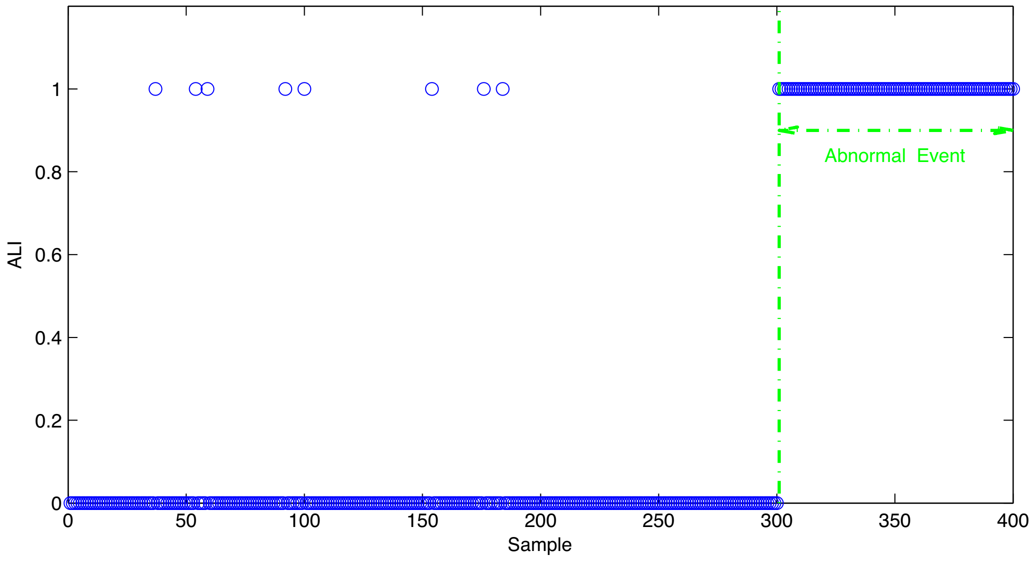
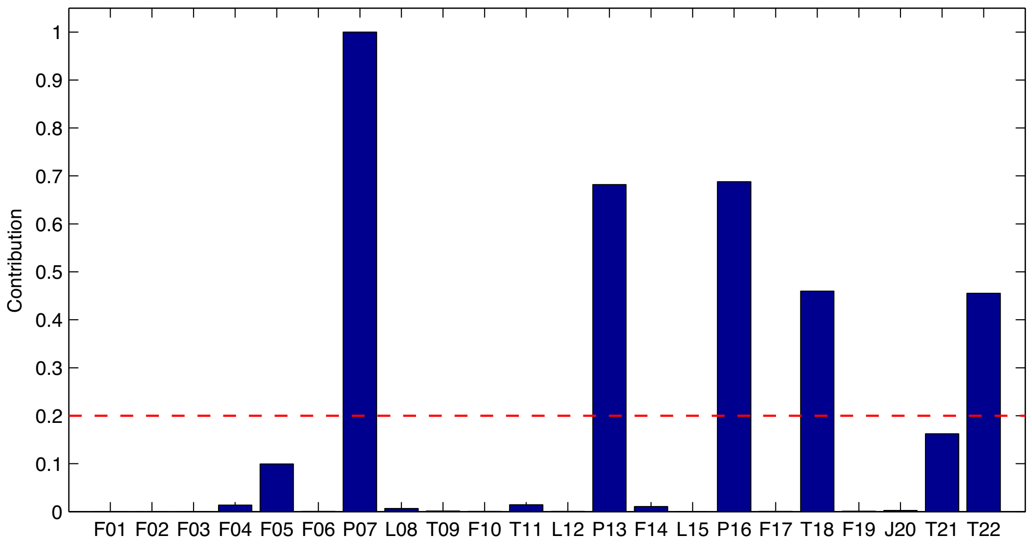


Fig. 31. Causal map of transfer entropy method in the case of IDV(7).



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 32. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(7).

Appendix A. Remaining test scenarios

For the case of IDV(3), the trend plot of the abnormality likelihood index and the Bayesian probability index based contribution plot are shown in Fig. 22. It can be seen that most of the ALI values are 0 during the normal operation and then become 1 with very minimal delay once the process fault of a step change in D feed temperature occurs at 301st samples. Please note that the feed temperature is not measured. Hence, the fault detection algorithm cannot identify it as the root cause, but it should be able to identify a measured variable which is directly impacted by the change in feed temperature. After the fault is detected, it is obvious that the five leading variables with the larger contribution values than 0.2 are reactor pressure (P_7), separator pressure (P_{13}), stripper pressure (P_{16}), compressor work (J_{20}), and condenser coolant temperature (T_{22}). After finding the short paths in Fig. 23 so that these all variables are connected, one can readily find that P_7 and T_{22} are the potential root-cause variables that lead process upset. In order to identify the root-cause variable between them, the posterior probability of each graph edge is computed and they turned out to be $P(f_{P_7 \to P_{13}}|\mathcal{D}^{\mathrm{new}}) = 0.874$ and $P(f_{T_{22} \to P_{13}}|\mathcal{D}^{\mathrm{new}}) = 0.812$. Therefore, reactor pressure (P_7) is more likely be associated with the root cause of this process upset than P_{13}. One should note that the variations in D feed temperature directly affect the pressure in the reactor because the stream of D feed is connected to the reactor. Hence, the identified root cause is accurately consistent with the actual root-cause variable. In contrast, the PCA and ICA based variable contribution plots are shown in Figs. 19 and 20. These contribution plots show the major affected process variables, but cannot identify the root-cause variable among them. As a

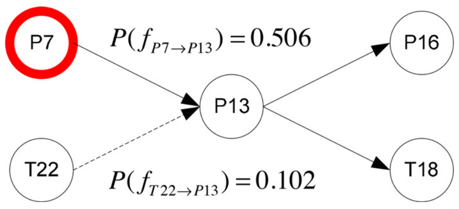


Fig. 33. Identified fault propagation pathway in the case of IDV(7).

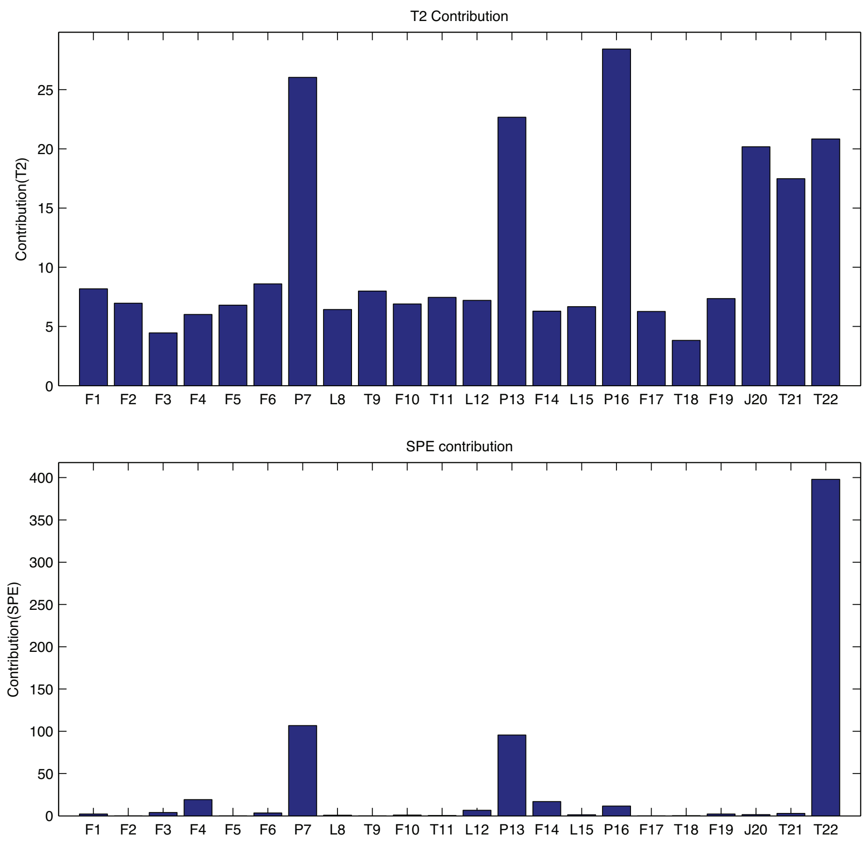


Fig. 34. Variable contribution plots of PCA based process monitoring in the case of IDV(9).

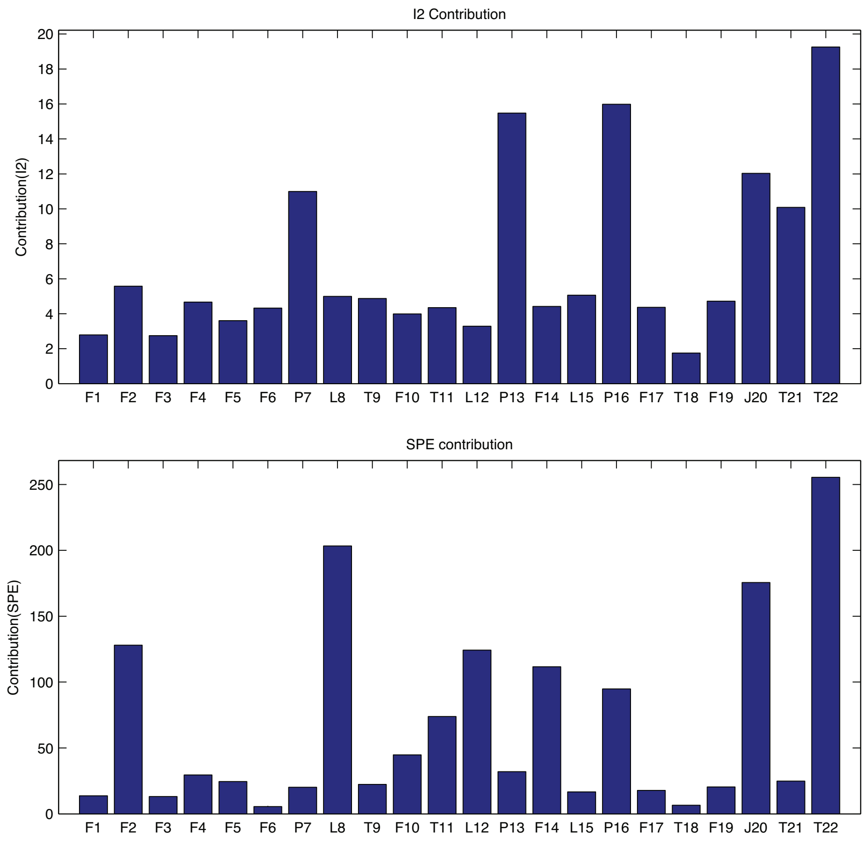


Fig. 35. Variable contribution plots of ICA based process monitoring in the case of IDV(9).

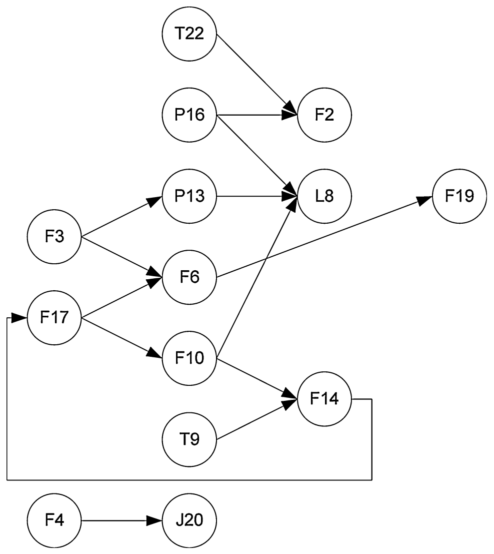


Fig. 36. Causal map of transfer entropy method in the case of IDV(9).

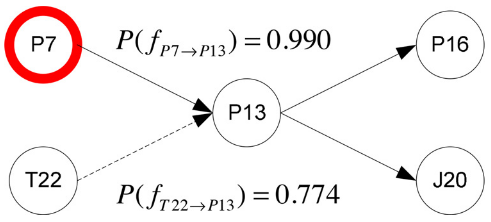


Fig. 38. Identified fault propagation pathway in the case of IDV(9).

comparison, the causal map of the transfer entropy approach is shown in Fig. 21. The root nodes of causal map are T_{22}, T_{11}, P_{16} and F_{17} that are the potential root causes, but the true root-cause variables cannot be identified among them without in-dept process knowledge. Conversely, the proposed networked process monitoring taken into account more complicated cause–effect relationship and further identify the major propagation pathway and root-cause variable.

For the case of IDV(5), the step change of condenser coolant water inlet temperature takes place after the 300 samples of normal operation. As the trend plot of the abnormality likelihood index shows in Fig. 27, they detect abnormal operating events immediately after the process fault occurs at 301st samples. Then, Bayesian contribution index values are calculated to identify the variables responsible for this abnormal event, as shown in Fig. 27. It is obvious that the three leading variables with the largest contribution values are separator temperature (T_{11}) stripper temperature (T_{18}) and condenser coolant temperature (T_{22}). Because the process abnormality occurs in separator coolant temperature, separator

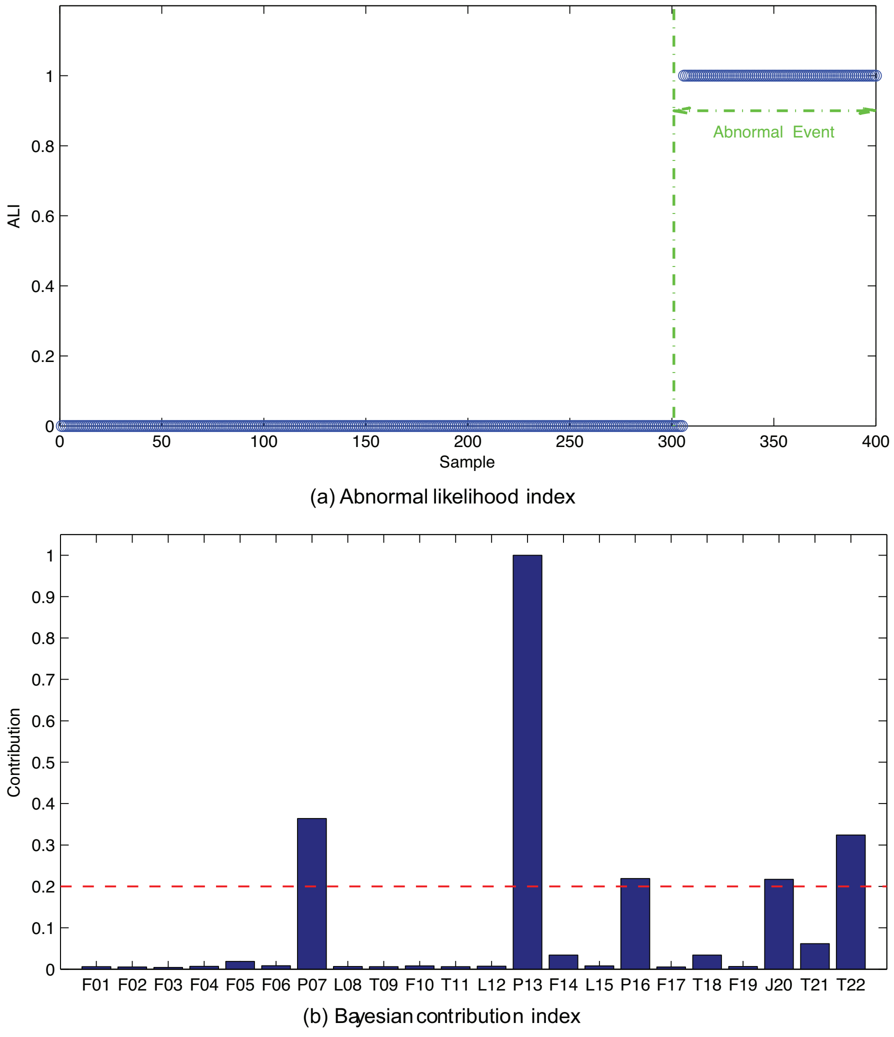


Fig. 37. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(9).

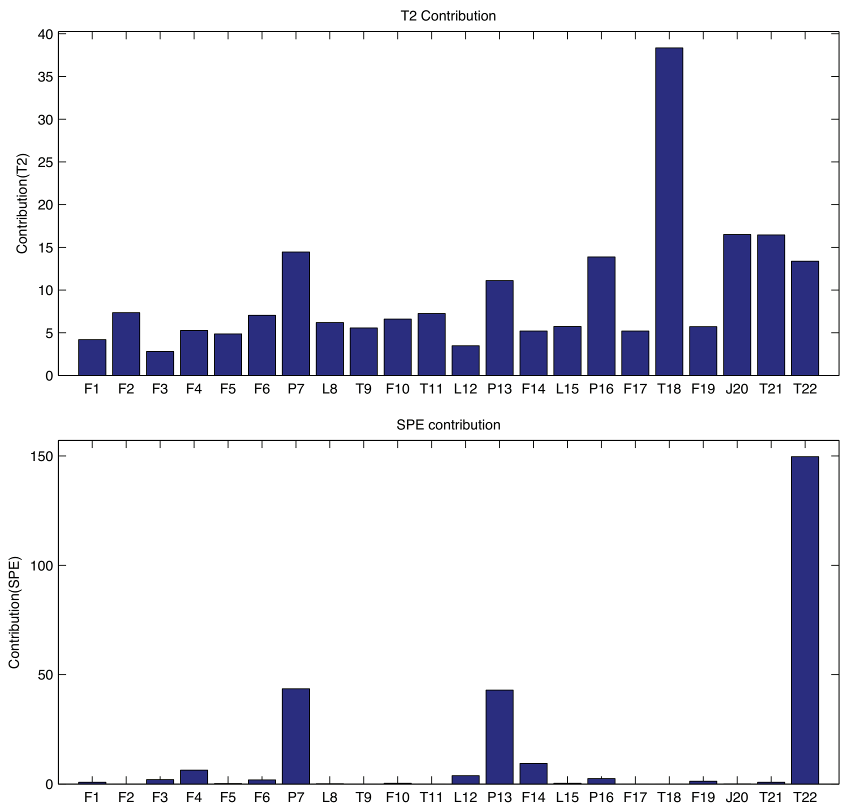


Fig. 39. Variable contribution plots of PCA based process monitoring in the case of IDV(10).

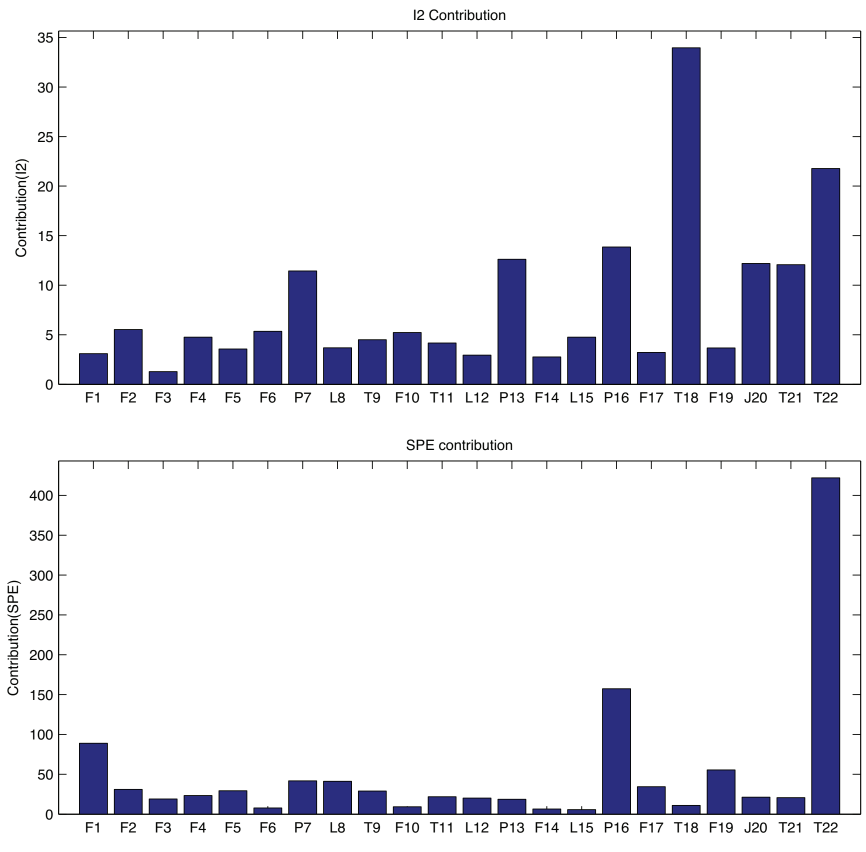


Fig. 40. Variable contribution plots of ICA based process monitoring in the case of IDV(10).

temperature (T_{11}) and stripper temperature (T_{18}) are consequently affected due to the variable interaction. The identified fault propagation pathways are shown in Fig. 28. The pathways include T_{11}, P_{13}, F_{14}, L_{15}, P_{16}, and T_{18}, and the root nodes of T_{22} is accurately identified as the root-cause variable leading to the process variations. On the other hand, as shown in Figs. 24 and 25, the values on T_{22} of both PCA-based and ICA-based SPE contribution are significantly larger than that of any other variable and thus it can be said that T_{22} are accurately identified as the root-cause variable. Meanwhile, in the T^2 and I^2 contribution plots, many variables including T_{22} have the large values of contribution, so it is challenging to extract the true root-cause variable without in-depth knowledge about processes. As a comparison, causal map is computed by the transfer entropy is shown in Fig. 26. According to the causal map, F_1, L_{15}, T_{22}, F_{19}, F_6, and P_{16} are identified as the potential root-cause variables since they are placed on the root nodes of the identified pathways. Although T_{22} is included among these variables, we cannot identify the root-cause variable without in-depth process knowledge. Conversely, since the proposed

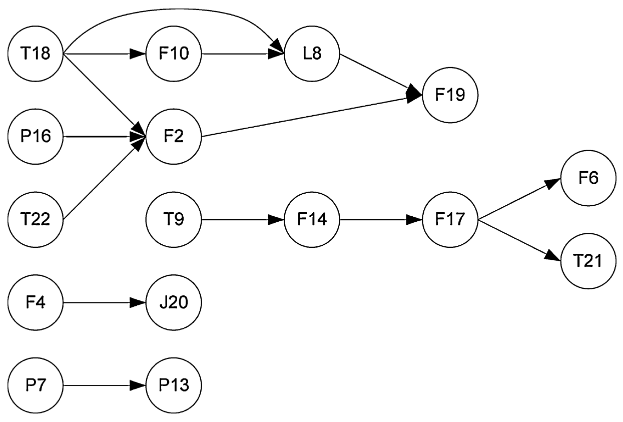
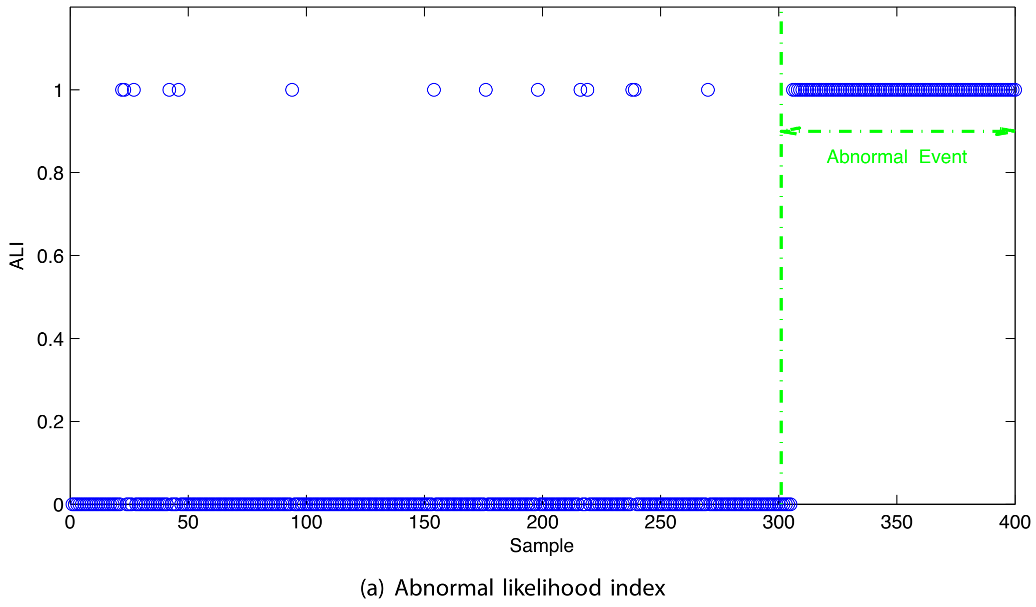


Fig. 41. Causal map of transfer entropy method in the case of IDV(10).



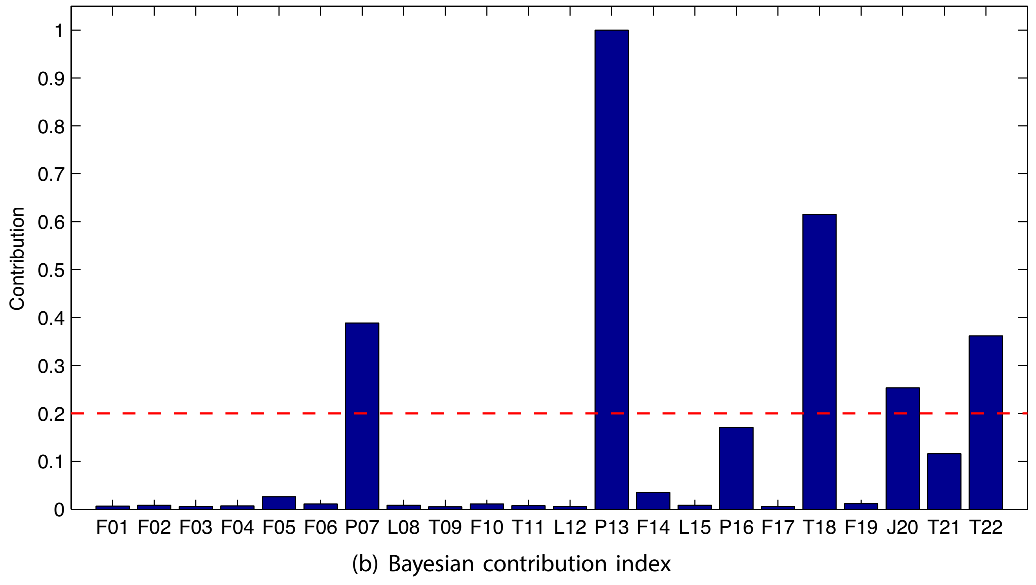


Fig. 42. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(10).

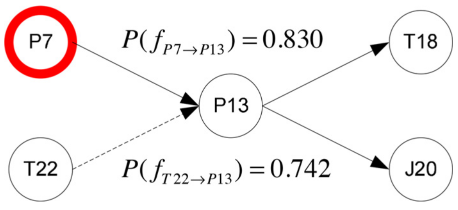


Fig. 43. Identified fault propagation pathway in the case of IDV(10).

networked process monitoring takes into account more complicated cause–effect relationship and structural information based on the incidence matrix, it can further identify the major propagation pathways and root-cause variable (see Figs. 27 and 28).

For the case of IDV(7), C header pressure loss occurs from the 301st to 400th samples after normal operation. The process monitoring results including the ALI trend and BCI plot are shown in Fig. 32. ALI values accurately capture abnormal operating events during C header pressure loss takes place. Then, Bayesian contribution index values are calculated to identify the potential root-cause variables. The significant variables in terms of BCI values are P_7, P_{13}, P_{16}, T_{18} and T_{22}. They include the actual root-cause variable of stripper pressure P_{16} that should be directly affected by the variation of the C header pressure because the stream of C feed is connected to the stripper. In order to identify the root-cause variable among them, the fault propagation pathways are identified as shown in Fig. 33. Although the actual root-cause variable is P_7, the proposed method mistakenly identified reactor pressure of P_7 as the root-cause variable. By contrast, the value on P_{16} in PCA-based T^2 contribution plot is large enough to be identified as the root-cause variable that is consistent with the actual one. Meanwhile other contribution values such as PCA-based SPE, ICA-based T^2 and SPE do not conclude that P_{16} is the root-cause variable (see Figs. 29 and 30). In comparison, the transfer entropy based causal map is shown in Fig. 31 and its root nodes are P_7, P_{13}, P_{16}, J_{20} and T_{11}, one of which is identified as the root-cause variable leading process variations. Although these variables include the actual root-cause one P_7, it is not easy to identify it among these 5 potential variables without in-depth process knowledge. Therefore, in this test scenario, only PCA-based T^2 contribution plot can accurately identify the root-cause variable of faulty operation.

For the test case of IDV(9), the trend plot of the ALI and the BPI values are shown in Fig. 37. It can be observed that all ALI values are 0 during the normal operation and then become 1 with very minimal delay once the process fault of a random variation in D feed temperature happens at 301st samples. Furthermore, it can be seen that in the BCI plot reactor pressure (P_7), separator pressure (P_{13}), stripper pressure (P_{16}), compressor work (J_{20}), and condenser coolant temperature (T_{22}) are greater than the confidence limit of 0.2. Next step is to identify the root-cause variable among them. According to the identified fault propagation pathways shown in Fig. 38, there are two potential root-cause variables of P_7 and T_{22}. Therefore, the posterior probability of each edge given the faulty data is computed and compared. As a result it is found that the probability of the edge from P_7 to P_{13} is larger than the other and then it is concluded that the root-cause variable is accurately identified as the reactor pressure P_7, which leads the actual root-cause variables because the D feed stream is connected to the reactor and the variation of D feed temperature should directly affected the pressure in the reactor. Meanwhile, Figs. 34 and 35 show the contribution plots of the PCA-based and ICA-based contribution plots. Although the actual root cause of P_7 is included among the variables having high-contribution values except in the ICA-based SPE contribution plot, their contribution values are not the highest

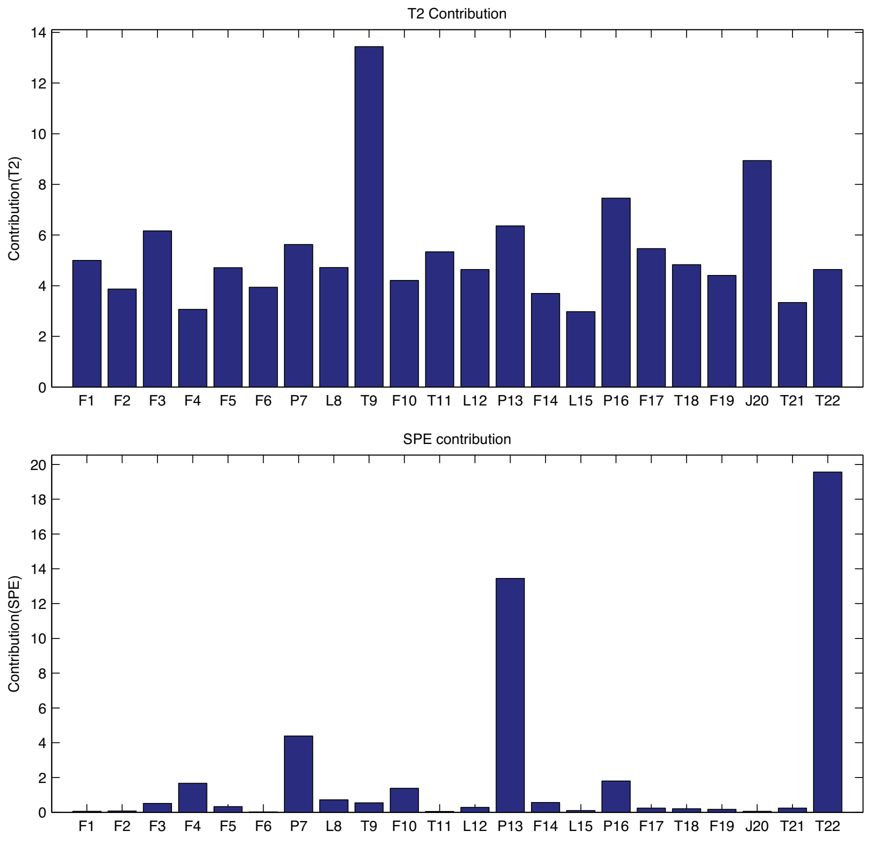


Fig. 44. Variable contribution plots of PCA based process monitoring in the case of IDV(11).

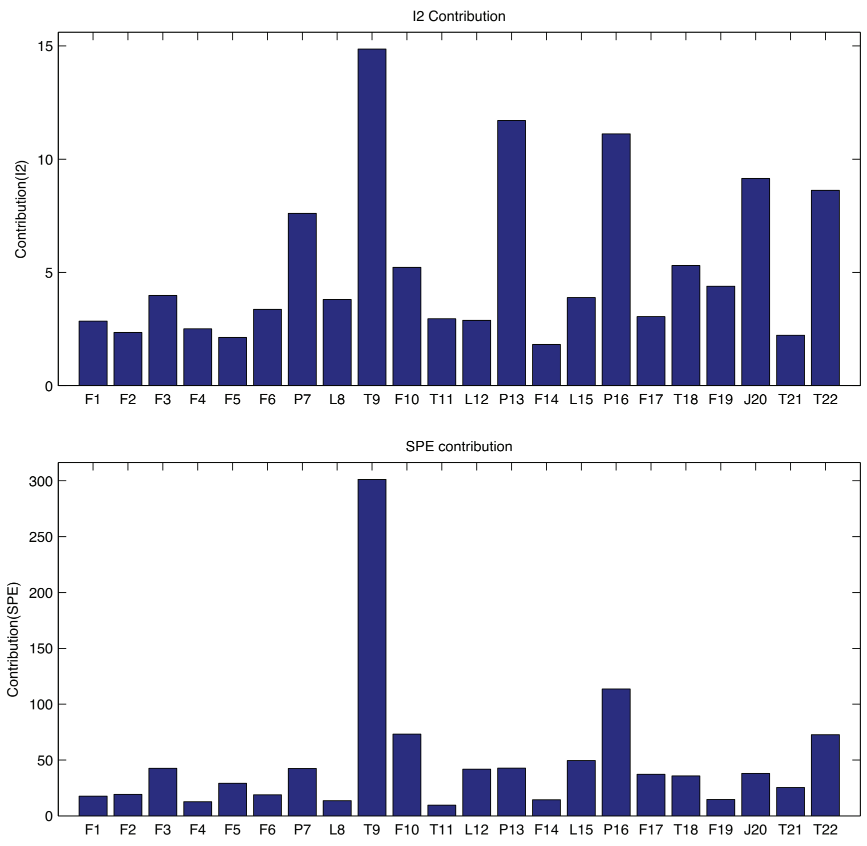


Fig. 45. Variable contribution plots of ICA based process monitoring in the case of IDV(11).

of all variables and thus we cannot identify the root-cause variable without in-depth process knowledge. Furthermore, the causal map generated by the transfer entropy method suggests that the number of potential root-cause variables are 6 as shown in Fig. 36, but the true root-cause variable of P_7 is excluded. By contrast, the proposed method can capture the plausible fault propagation pathways and identify the actual root-cause variable.

For the case of IDV(10), the fault detection results of the ALI are shown in Fig. 42. It is obvious that the alert is triggered with very minimal delay after process fault of random variation in C feed temperature occurs. Once faulty event is detected, BCI values of all the variables are further compared, as shown in Fig. 42. In addition, networked process diagnosis results are shown in Fig. 43. Three leading variables with the largest contribution values are reactor pressure (P_7), separator pressure (P_{13}), stripper temperature (T_{18}), compressor work (J_{20}) and separator coolant temperature (T_{22}). It is true that these potential root-cause variables include the true root-cause one T_{18} that leads to the process upset, but the identified fault propagation suggests that the reactor pressure P_7 is the measured variable which is associated with the root cause in Fig. 43 and it can be said that the identified one conflict with the actual root cause. On the other hand, the PCA-based and ICA-based contribution plots are shown in Figs. 39 and 40. Both T^2 and I^2 contribution plots can well-isolate the root-cause variable of T_{18} that has far larger contribution value than the other variable does. As for PCA-based and ICA-based SPE contribution plots do not work well for identifying the root-cause variable. Meanwhile, the transfer entropy method computes the causal map in order to identify the root-cause variable and the fault propagation pathways as shown in Fig. 41. Although one can readily see that total 6 potential root nodes include the actual root-cause variable of T_{18}, it is still challenging to extract the true one from these potential variables. In this test scenario, we found that only T^2 and I^2 based contribution plots can accurately identify the root-cause variable causing the faulty operation.

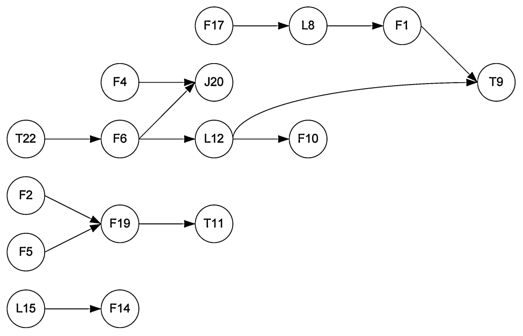
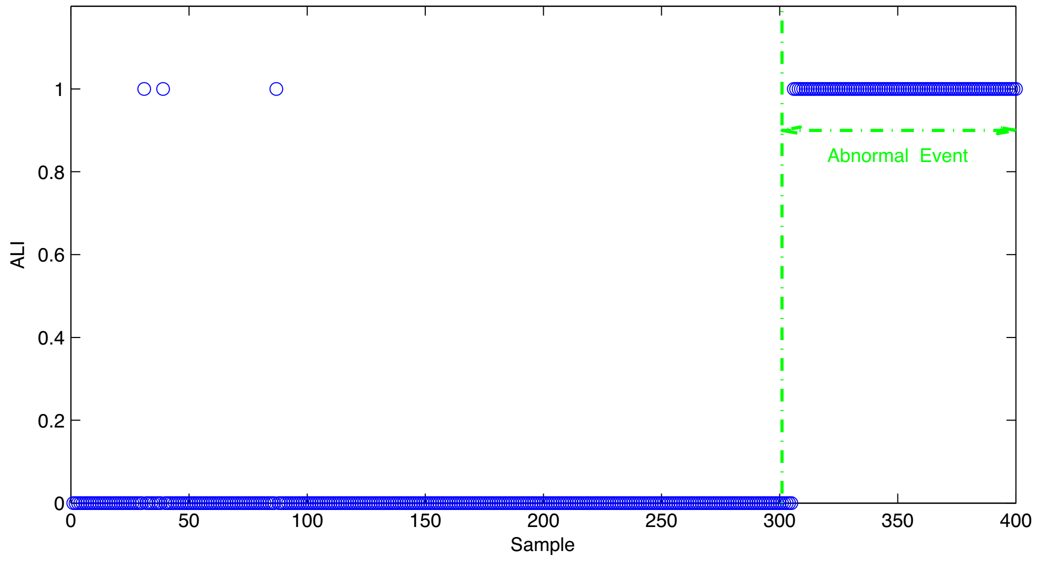
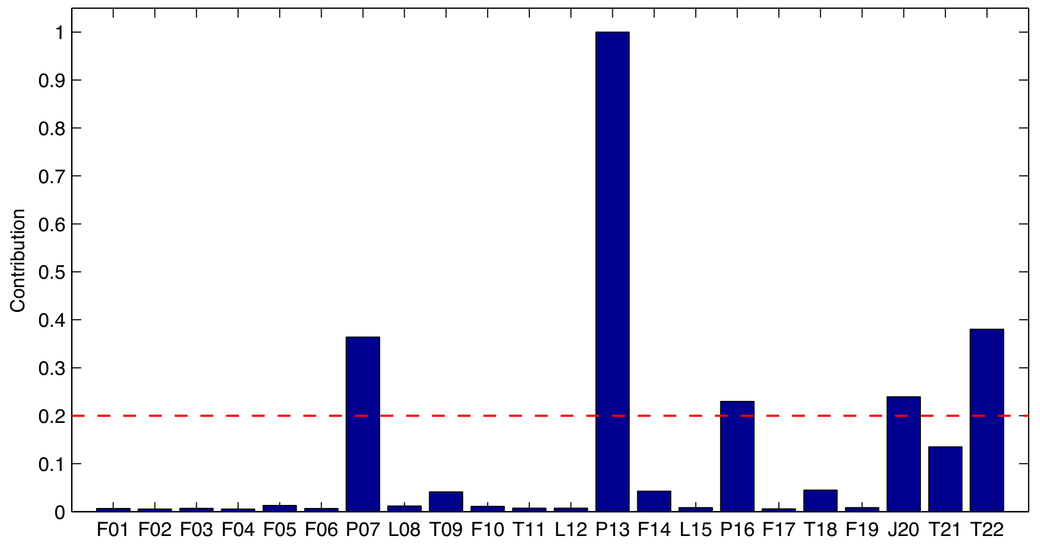


Fig. 46. Causal map of transfer entropy method in the case of IDV(11).

In the case of IDV(11), the random variation of the reactor cooling water inlet temperature occurs from 301st to 400th samples. The ALI trend and BCI values are shown in Fig. 47. These figures indicate that the ALI values can detect the faulty operation with the high detection rate but the set of variables that have high BCI values excludes the actual root-cause variable of T_{11}. Similarly, in the PCA and ICA-based contribution plots shown in Figs. 44 and 45, T_{11} is not taken into account significant variable that causes the process upset and thus these results also conflict with the actual root cause. As a comparison, the causal map computed by the transfer entropy method is shown in Fig. 46. Similar to the other methods, the root nodes of the identified causal map do not include the reactor cooling water of T_{11} that is the actual root-cause variable leading to the process variation. In summary, all the methods including the proposed one cannot identify the root-cause variable of the reactor cooling water temperature T_{11} in this test scenario.



(a) Abnormal likelihood index



(b) Bayesian contribution index

Fig. 47. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(11).

For the test case of IDV(12), the trend plot of the ALI and the BCI values are shown in Fig. 52. One can readily see that the proposed ALI is able to accurately detect faulty operations after the random variation of the condenser cooling water inlet temperature takes place. Once the faulty operation is captured, the Bayesian contribution index values are computed to identify the root-cause variable for the process upset. It is obvious that the three variables with the largest contribution values are separator temperature (T_{11}), stripper temperature (T_{18}), and condenser coolant temperature (T_{22}). Since there are three potential root-cause variables, the fault propagation pathways are identified as shown in Fig. 53. The fault propagation pathways show that process variation takes place in separator coolant temperature (T_{22}) and then T_{11}, F_{14}, L_{15}, P_{16} and T_{18} are consequently affected due to the variable interaction. The identified fault propagation pathway is well consistent with the actual faulty scenario and the separator coolant temperature (T_{22}) is accurately placed as the root-cause variable that leads to process upset. On the other hand, the PCA and ICA-based contribution plots that are shown in Figs. 49 and 50 can identify the fault affected variables including true root-cause variables of separator coolant temperature (T_{22}). The SPE contribution value of T_{22} in the PCA-based monitoring method is significantly large and hence it can be said that T_{22} is accurately identified as the root-cause variable. Meanwhile, the other contribution plot cannot well-isolate the root-cause variable among high-contribution-value variables. By contrast, the transfer entropy method is used to compute the causal map and identify the root-cause variable and the fault propagation pathways. The identified causal map is shown in Fig. 51. It can be seen that the identified fault propagation pathway

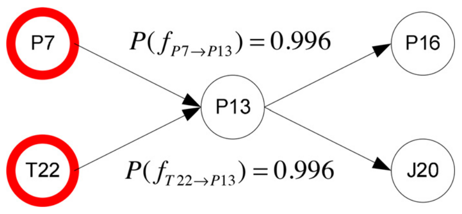


Fig. 48. Identified fault propagation pathway in the case of IDV(11).

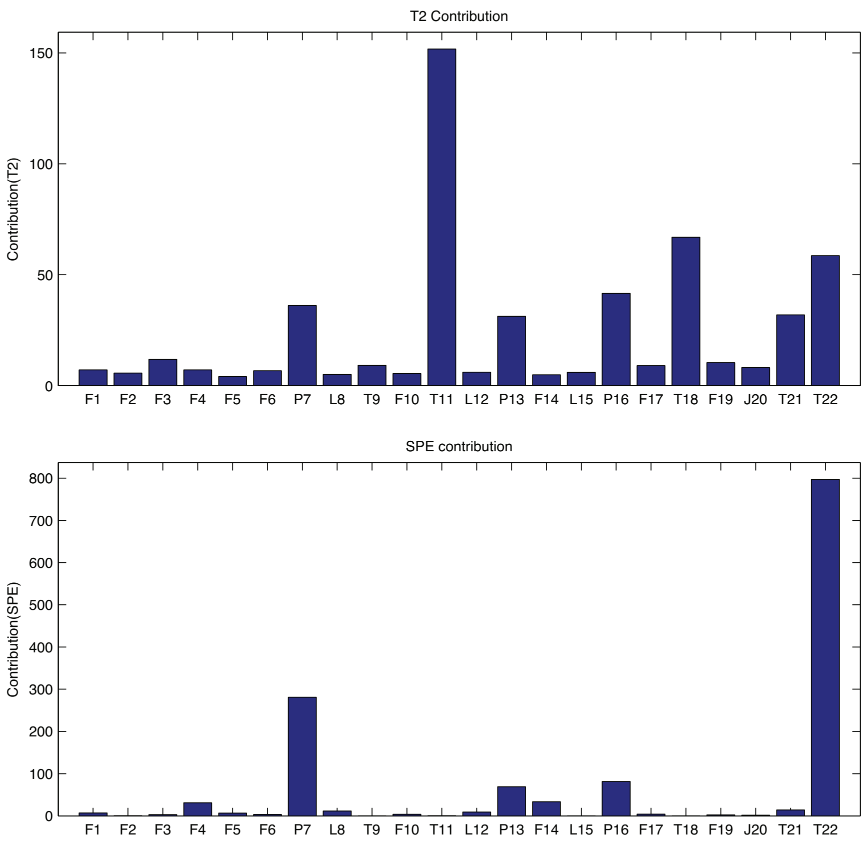


Fig. 49. Variable contribution plots of PCA based process monitoring in the case of IDV(12).

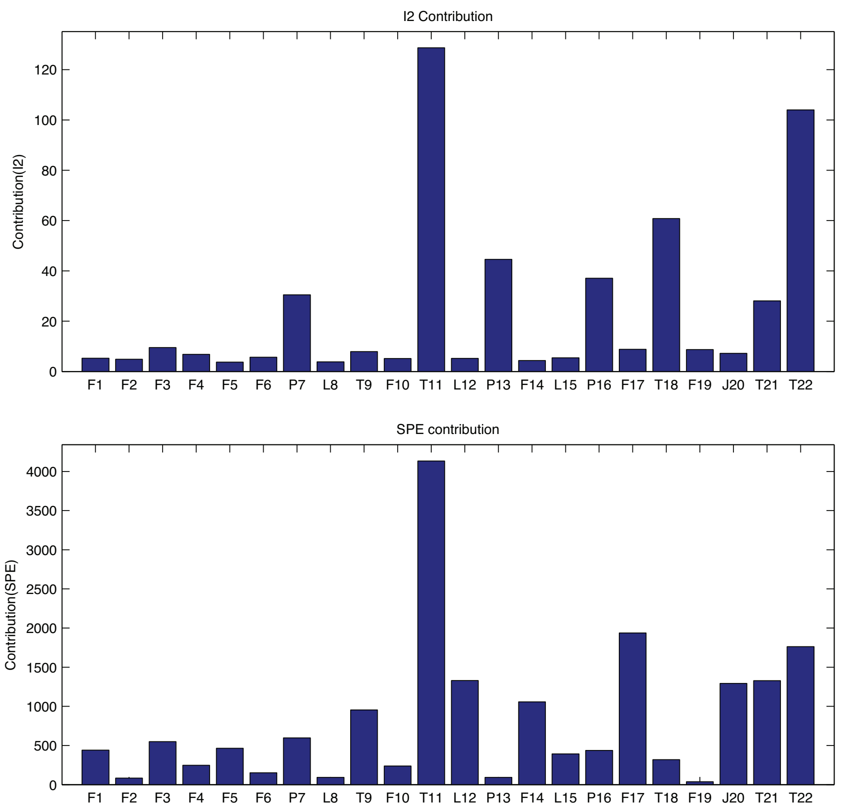


Fig. 50. Variable contribution plots of ICA based process monitoring in the case of IDV(12).

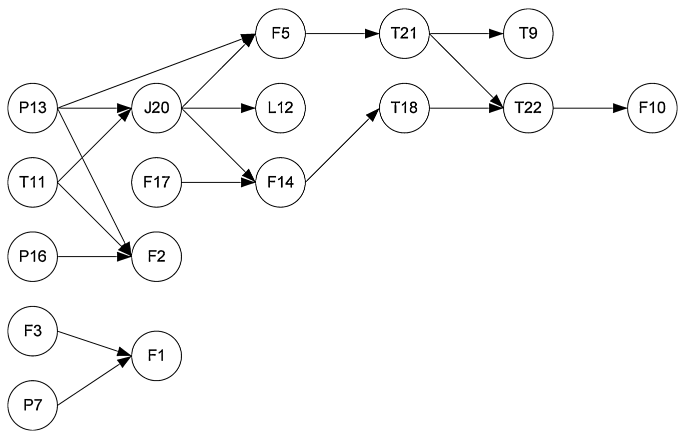


Fig. 51. Causal map of transfer entropy method in the case of IDV(12).

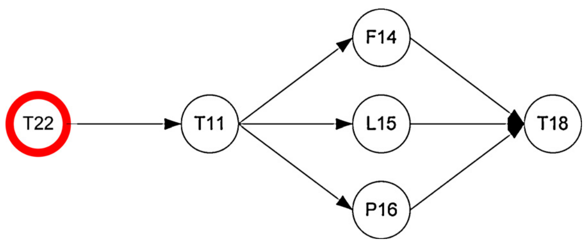


Fig. 53. Identified fault propagation pathway in the case of IDV(12).

excludes the actual root cause of separator coolant temperature (T_{22}). On the other hand, the fault propagation pathways and root-cause variable identified by the proposed networked process monitoring approach coincides well the actual fault propagation pathways. These above all comparisons demonstrate that the proposed networked process monitoring approach has the unique capability to capture the major fault propagation pathways and identify the root-cause variable due to the fact that the proposed

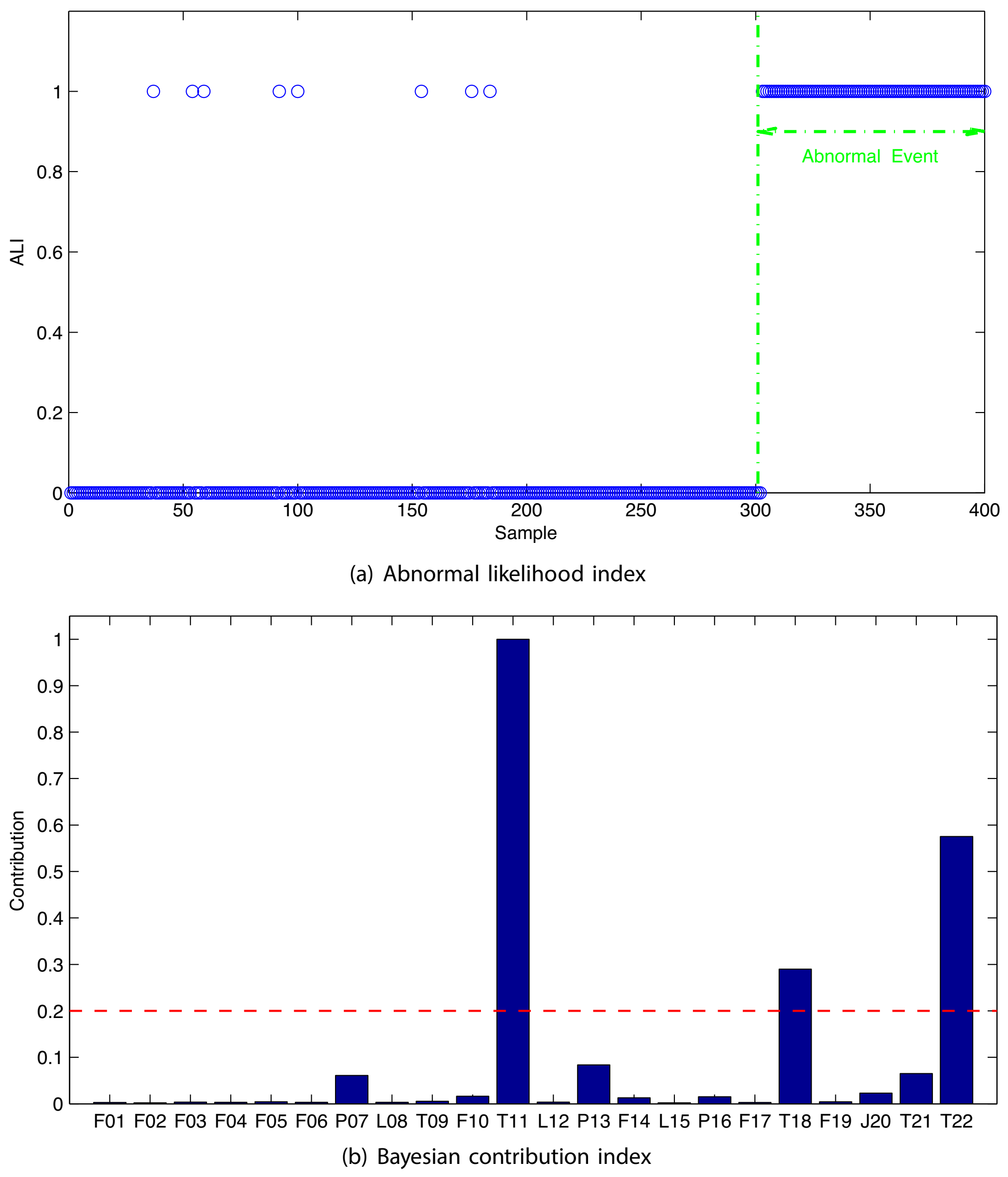


Fig. 52. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(12).

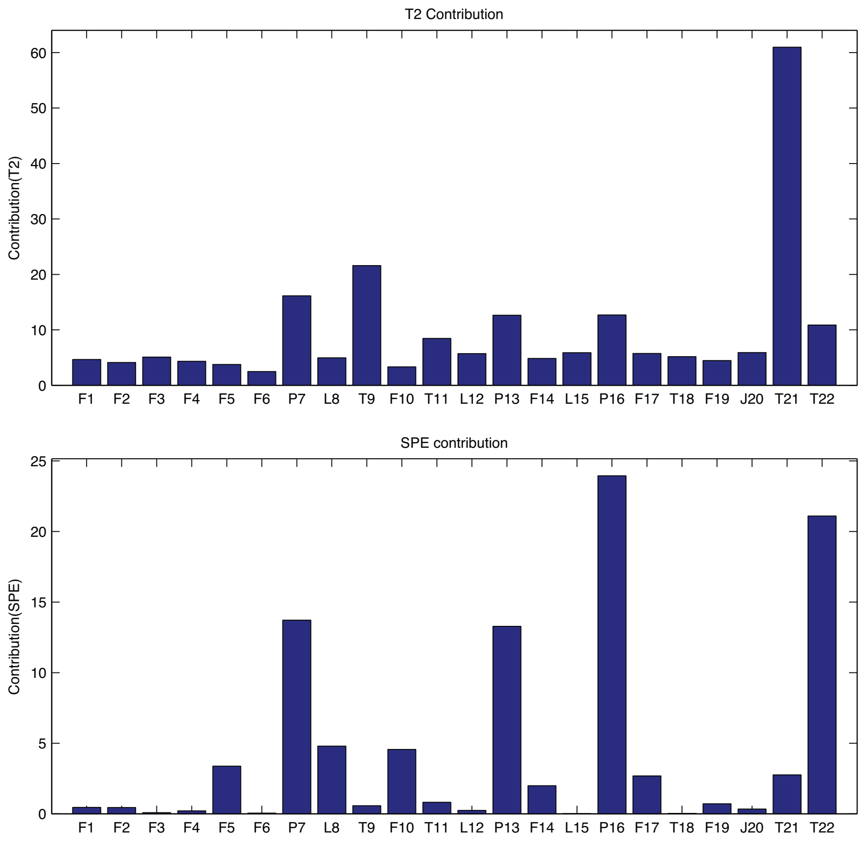


Fig. 54. Variable contribution plots of PCA based process monitoring in the case of IDV(14).

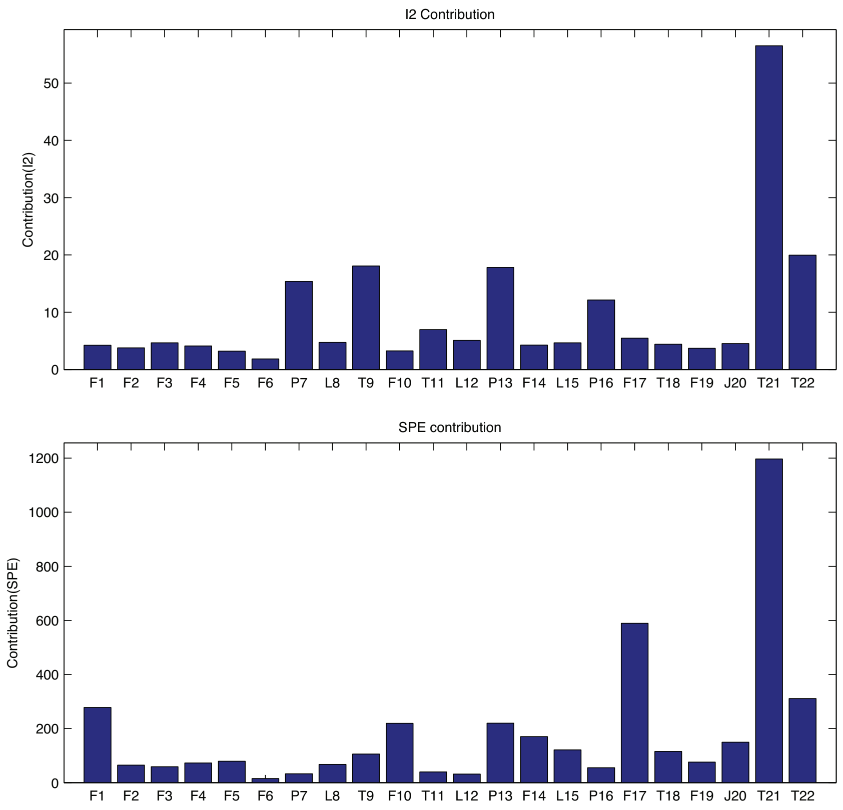


Fig. 55. Variable contribution plots of ICA based process monitoring in the case of IDV(14).

networked model takes into account more detailed relationship and is capable of describing the complicated phenomena.

For the test case of IDV(14), the plant operation includes normal condition along with sticking of the reactor cooling water valve. Fig. 57 describes the ALI-based fault detection result and BCI-based variable contribution plot. It is obvious that the alarm is accurately triggered with a short delay. The variable with higher BCI value than the confidence level is reactor cooling water temperature (T_{21}) only and thus T_{21} is turned out to be the identified root-cause variable. Since the sticking of the reactor cooling water valve should has a direct influence on T_{21}, it can be said that the proposed method can accurately isolate the true root-cause variable. As a comparison, Figs. 54 and 55 show the PCA and ICA-based contribution plot. It is obvious that except in PCA-based SPE contribution plot the contribution values of T_{21} are so large that T_{21} can be correctly identified as the root-cause variable. Meanwhile, the transfer entropy method is also carried out to identify the root-cause variable. The causal map computed by the transfer entropy method is shown in Fig. 56. One can easily observe that the true root-cause variable of T_{21} is not

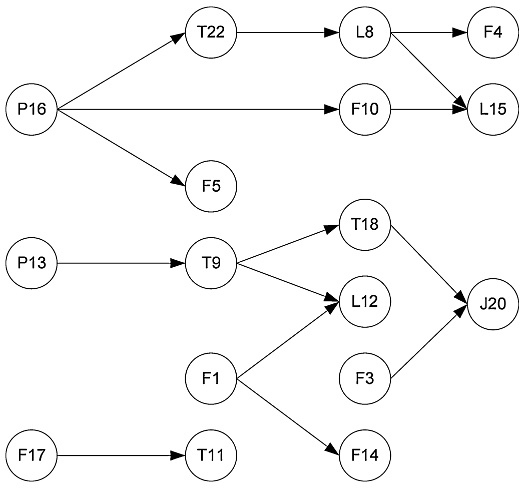


Fig. 56. Causal map of transfer entropy method in the case of IDV(14).

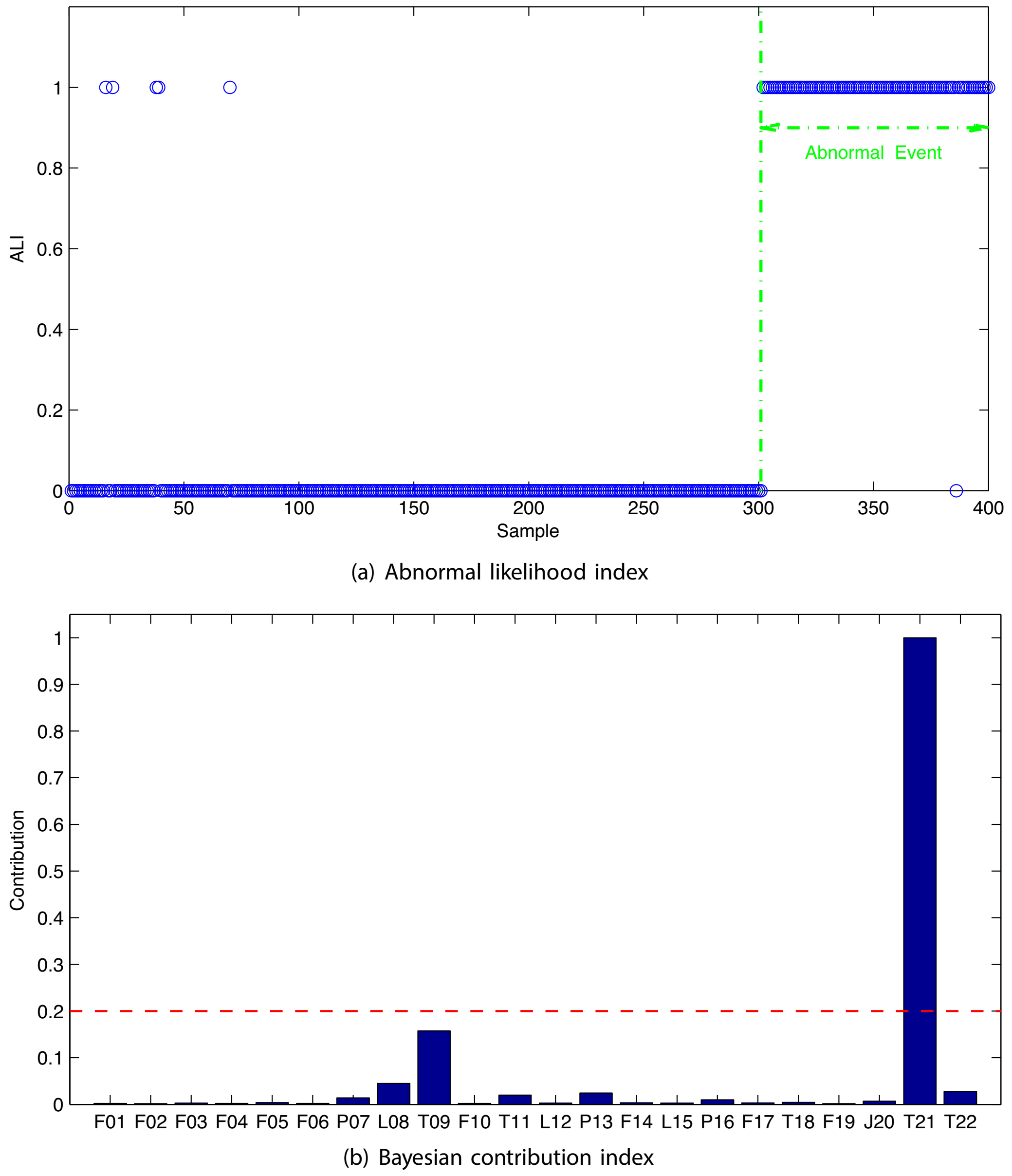


Fig. 57. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(14).

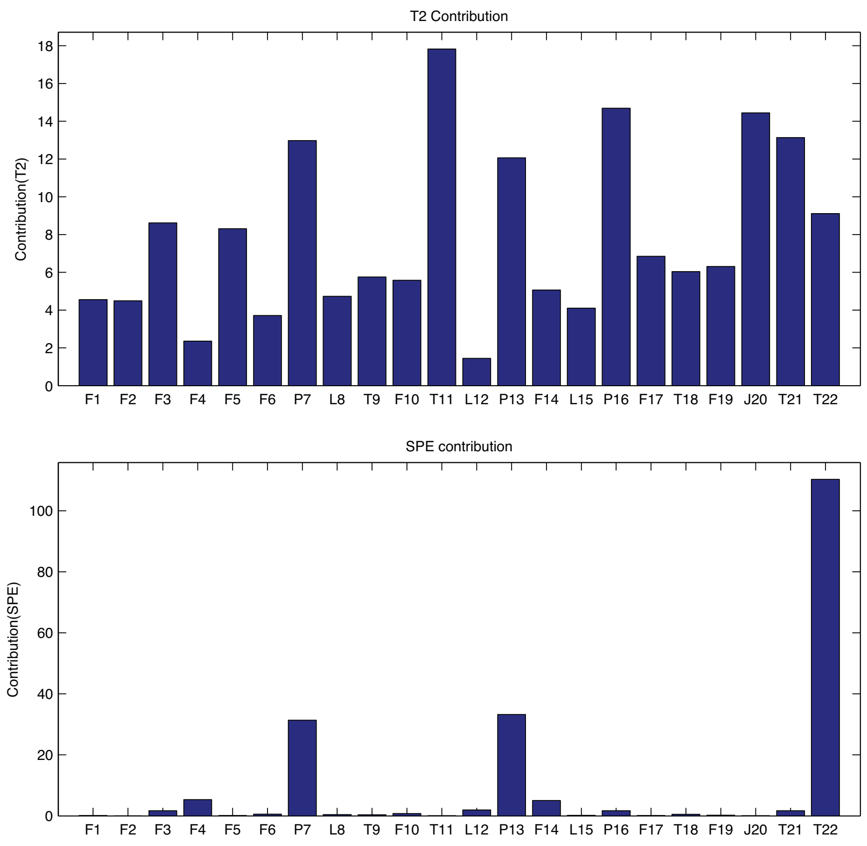


Fig. 58. Variable contribution plots of PCA based process monitoring in the case of IDV(15).

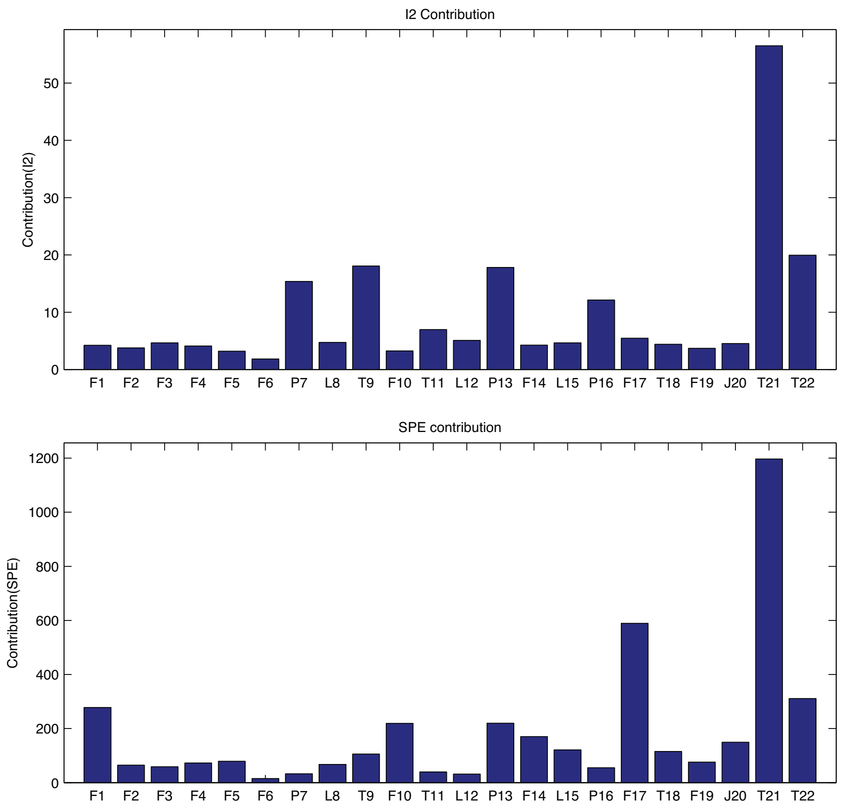


Fig. 59. Variable contribution plots of ICA based process monitoring in the case of IDV(15).

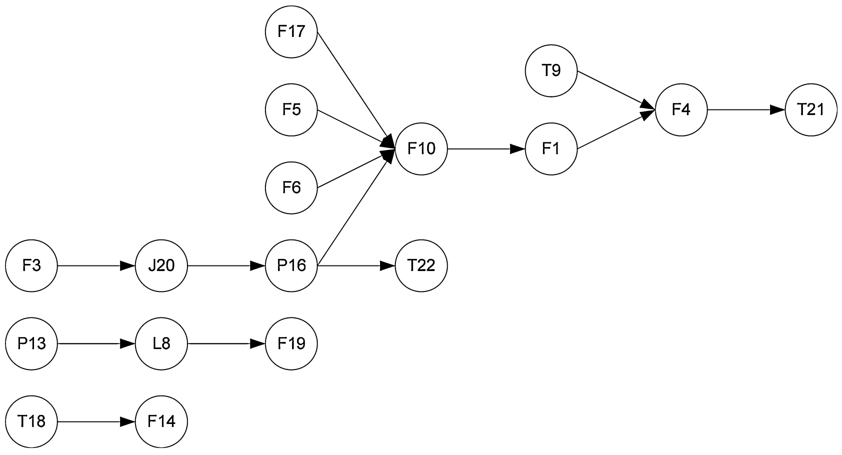
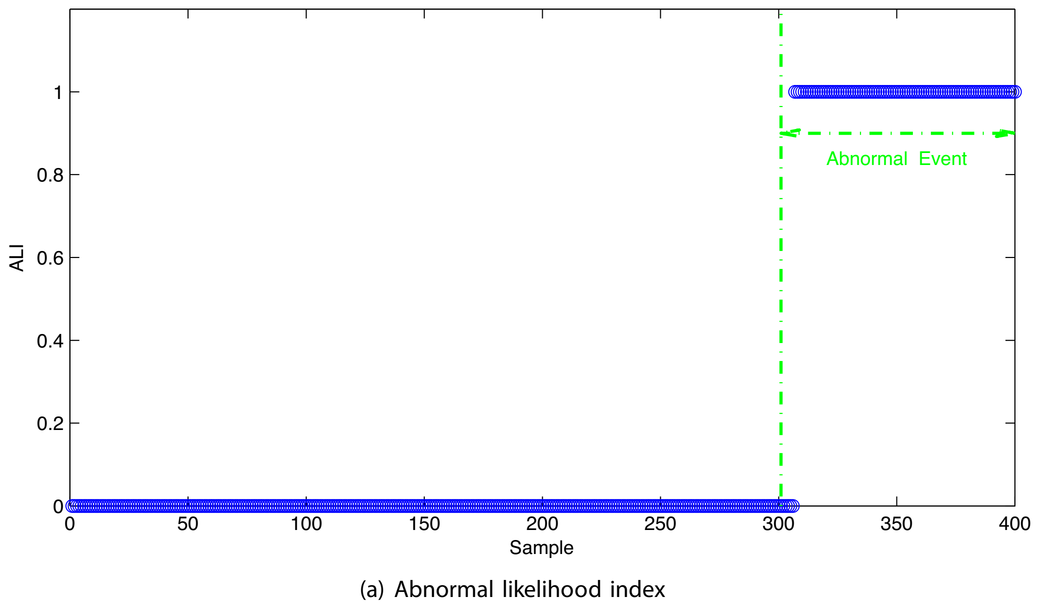


Fig. 60. Causal map of transfer entropy method in the case of IDV(15).



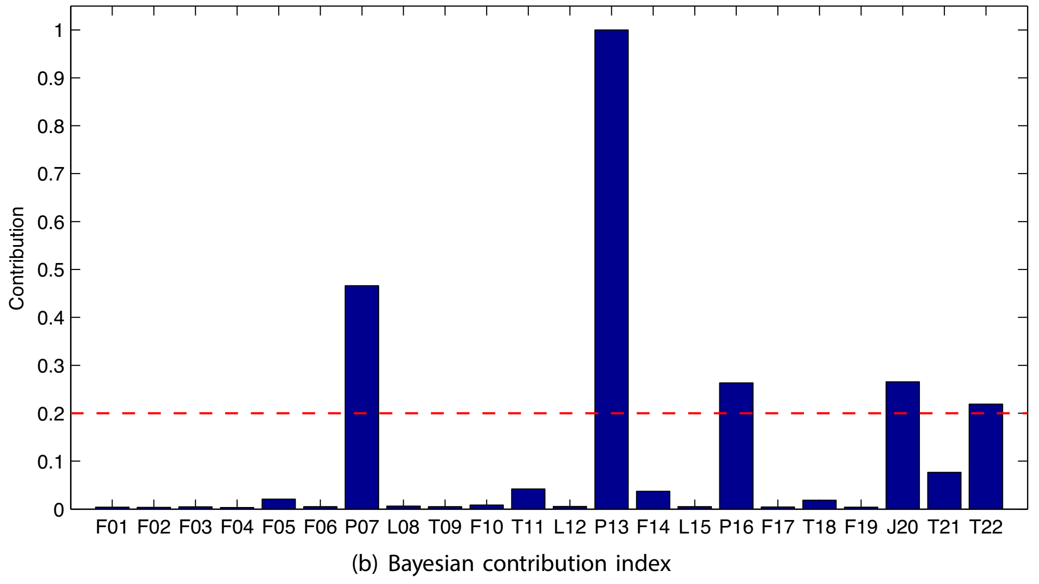


Fig. 61. Trend plot of abnormal likelihood index (ALI) and Bayesian contribution index (BCI) plot in the case of IDV(15).

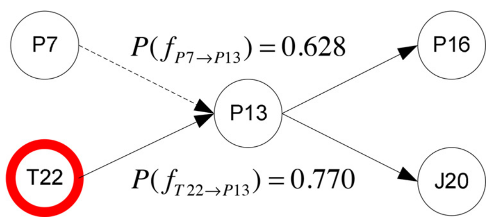


Fig. 62. Identified fault propagation pathway in the case of IDV(15).

included in the identified causal map. Such comparison shows that the proposed probabilistic graphical model approach and the PCA and ICA-based contribution plot can accurately capture the root-cause variable in this test case, while the transfer entropy method does not well-work.

For the last test case of IDV(15), sticking of condenser cooling water valve occurs from 301st to 400th samples. The ALI trend plot and BCI values are shown in Fig. 61. It can be seen that the proposed ALI can correctly trigger the alarm with very minimal delay. Moreover, there is no false alarm triggered on the normal samples. Further, it is obvious that five leading variables with higher BCI values than the confidence level of 0.2 are reactor temperature (P_7), separator pressure (P_{13}), stripper pressure (P_{16}), compressor work (J_{20}) and condenser cooling water temperature (T_{22}). In order to isolate the root-cause variable among these 4 potential root-cause variables, the fault propagation pathways are computed from the constructed network structure as shown in Fig. 62. One can readily observe that reactor pressure (P_7) and reactor cooling water temperature (T_{22}) are the root nodes of the identified pathways. So as to identify the root-cause variable between them, the posterior probability of each edge is computed as $P(f_{P_7\to P_{13}}|\mathcal{D}^{\text{new}}) = 0.628$ and $P(f_{T_{22}\to P_{13}}|\mathcal{D}^{\text{new}}) = 0.770$ using the MCMC simulation technique. Finally, reactor cooling water temperature (T_{22}) is accurately place on the root-cause variable that leads the process upset. As a comparison, the PCA and ICA-based variable contribution plot are shown in Figs. 58 and 59. While the ICA-based contribution plots identify T_{21} as the most significant variable, the PCA-based ones cannot well-isolate T_{21} among the variables with high-contribution values. Meanwhile, the causal map based on the transfer entropy approach is computed as shown in Fig. 60. It can be seen that the true root-cause variable T_{21} is mistakenly located in the leaf nodes instead of the root nodes.

References

Alaeddini A, Dogan I. Using Bayesian networks for root cause analysis in statistical process control. Expert Syst Appl 2011;38:11230–43.
Bauer M, Thornhill NF. A practical method for identifying the propagation path of plant-wide disturbances. J Process Control 2008;18:707–19.
Bauer M, Cox JW, Caveness MH, Downs JJ, Downs F. Finding the direction of disturbance propagation in a chemical process using transfer entropy. IEEE Trans Control Syst Technol 2007;15:12–21.
Bishop CM. Pattern recognition and machine learning. New York, NY: Springer-Verlag; 2006.
Boudali H, Dugan JB. A discrete-time Bayesian network reliability modeling and analysis framework. Reliab Eng Syst Saf 2005;87:337–49.
Chen A, Song Z, Li P. Soft sensor modeling based on DICA-SVR. Lect Notes Comput Sci 2005;3644:868–77.
Chiang LH, Braatz RD. Process monitoring using causal map and multivariate statistics fault detection and identification. Chemom Intell Lab Syst 2003;65:159–78.
Chickering DM. Learning Bayesian networks is NP-complete. New York, NY: Springer-Verlag; 1996.
Chickeringn DM. Learning equivalence classes of Bayesian-network structures. J Mach Learn Res 2002;2:445–98.
Desai K, Badhe Y, Tambe SS, Kulkarni BD. Soft-sensor development for fed-batch bioreactors using support vector regression. Biochem Eng J 2006;27(3):225–39.
Dey S, Stori JA. A Bayesian network approach to root cause diagnosis of process variations. Int J Mach Tools Manuf 2005;45:75–91.
Downs JJ, Vogel EF. Plant-wide industrial process control problem. Comput Chem Eng 1993;17:245–55.
Doyle FJ. Nonlinear inferential control for process applications. J Process Control 1998;8:339–53.
Friedman N, Koller D. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. Mach Learn 2003;50:95–125.
Friedman N, Linial M, Nachman I, Peer D. Using Bayesian networks to analyze expression data. J Comput Biol 2000;7:601–20.
Gertler JJ. Survey of model-based failure detection and isolation in complex plants. IEEE Control Syst Mag 1988;8:3–11.
Gevaert O, Smet FD, Timmerman D, Moreau Y, Moor BD. Predicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks. Bioinformatics 2006;22:184–90.
Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res 1986;13:533–49.
Huang B. Bayesian methods for control loop monitoring and diagnosis. J Process Control 2008;18:829–38.
Kaneko H, Arakawa M, Funatsu K. Development of a new soft sensor method using independent component analysis and partial least squares. AIChE J 2009;55:87–98.
Koller D, Friedman N. Probabilistic graphical models: principles and techniques. Cambridge, MA: MIT Press; 2009.
Kosanovich KA, Dahl KS, Piovoso MJ. Improved process understanding using multiway principal component analysis. Ind Eng Chem Res 1996;35:138–46.
Kresta JV, Marlin TE, MacGregor JF. Development of inferential process models using PLS. Comput Chem Eng 1994;18:597–611.
Lee JM, Yoo CK, Lee IB. Statistical process monitoring with independent component analysis. J Process Control 2004;14:467–85.
Lin B, Recke B, Knudsen JKH, Jorgensen SB. A systematic approach for soft sensor development. Comput Chem Eng 2007;31:419–25.
Maurya MR, Rengaswamy R, Venkatasubramanian V. Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets. Eng Appl Artif Intell 2004;17:501–18.
Maurya MR, Rengaswamy R, Venkatasubramanian V. A signed directed graph and qualitative trend analysis-based framework for incipient fault diagnosis. Chem Eng Res Des 2007;85:1407–22.
Mejdell T, Skogestad S. Estimation of distillation compositions from multiple temperature measurements using partial least-squares regression. Ind Eng Chem Res 1991;30:2543–55.
Metropolis N, Rosenbluth M, Rosenbluth A. Equation of state calculation by fast computing machines. J Chem Phys 1953;21:1087–92.
Mori J, Yu J. Maximized mutual information based non-Gaussian latent subspace projection. In: Proceedings of 52nd IEEE conference on decision and control; 2013. p. 4361–6.
Mori J, Yu J. A quality relevant non-Gaussian latent subspace projection method for chemical process monitoring and fault detection. AIChE J 2014a;60:485–99.
Mori J, Yu J. Quality relevant nonlinear batch process performance monitoring using a kernel based multiway non-Gaussian latent subspace projection approach. J Process Control 2014b;24:57–71.
Nikovski D. Constructing Bayesian networks for medical diagnosis from incomplete and partially correct statistics. Trans Knowl Data Eng 2000;12:509–16.
Nomikos P, MacGregor JF. Multivariate SPC charts for monitoring batch processes. Technometrics 1995;37(1):41–59.
Piovoso MJ, Hoo KA. Multivariate statistics for process control. IEEE Control Syst Mag 2002;22:8–9.
Pons MN, Rajab A, Flaus JM, Engasser JM, Cheruy A. Comparison of estimation methods for biotechnological processes. Chem Eng Sci 1988;8:1909–14.
Sorsa T, Koivo HN. Application of artificial neural networks in process fault diagnosis. Automatica 1993;29:843–9.
Steck H. On the use of skeletons when learning in Bayesian networks. In: Proceedings of UAI-2000; 2000. p. 558–65.
Teyssier M, Koller D. Ordering-based search: a simple and effective algorithm for learning Bayesian networks. In: Proceedings of the twenty-first conference on uncertainty in artificial intelligence; 2005. p. 584–90.
Vedam H, Venkatasubramanian V. PCA-SDG based process monitoring and fault diagnosis. Control Eng Pract 1999;7:903–17.
Venkatasurbramanian V, Rengaswamy R, Kavuri SN, Yin K. A review of process fault detection and diagnosis: Part III. Process history based methods. Comput Chem Eng 2003;27:327–46.
Weidl G, Madsen AL, Israelson S. Applications of object-oriented Bayesian networks for condition monitoring, root cause analysis and decision support on operation of complex continuous processes. Comput Chem Eng 2005;29:1996–2009.
Yan W, Shao H, Wang X. Soft sensing modeling based on support vector machine and Bayesian model selection. Comput Chem Eng 2004;28:1489–98.
Yélamos I, Escudero G, Graells M, Puigjaner L. Performance assessment of a novel fault diagnosis system based on support vector machines. Comput Chem Eng 2009;33:244–55.
Yu J. A Bayesian inference based two-stage support vector regression framework for soft sensor development in batch bioprocesses. Comput Chem Eng 2012a;41:134–44.
Yu J. A nonlinear kernel Gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes. Chem Eng Sci 2012b;68:506–19.
Yu J, Qin J. Multimode process monitoring with Bayesian inference-based finite Gaussian mixture models. AIChE J 2008;54:1811–29.

Yu J, Rashid MM. A novel dynamic Bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis. AIChE J 2013;59:2348–65.
Yuan T, Qin J. Root cause diagnosis of plant-wide oscillations using Granger causality. J Process Control 2014;24:450–9.
Zamprogna E, Barolo M, Seborg DE. Optimal selection of soft sensor inputs for batch distillation columns using principal component analysis. J Process Control 2005;15:39–52.
Zhang H, Lennox B. Integrated condition monitoring and control of fed-batch fermentation processes. J Process Control 2004;14:41–50.

Chapter 5

Inference in hybrid Bayesian networks with large discrete and continuous domains

Expert Systems With Applications 49 (2016) 1–19

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Inference in hybrid Bayesian networks with large discrete and continuous domains

Junichi Mori[a], Vladimir Mahalec[b,*]

[a] Nippon Steel & Sumitomo Metal Corporation, Kimitsu Works, Department of Computer System, Kimitsu City, Chiba Prefecture 299-1141, Japan
[b] McMaster University, Department of Chemical Engineering, Hamilton, Ontario L8S 4L7, Canada

ARTICLE INFO

Keywords:
Large domain of discrete and continuous variables
Decision tree
Hybrid Bayesian network
Belief propagation algorithm
Context-specific independence

ABSTRACT

Inference in Bayesian networks with large domain of discrete variables requires significant computational effort. In order to reduce the computational effort, current approaches often assume that discrete variables have some bounded number of values or are represented at an appropriate size of clusters. In this paper, we introduce decision-tree structured conditional probability representations that can efficiently handle a large domain of discrete and continuous variables. These representations can partition the large number of values into some reasonable number of clusters and lead to more robust parameter estimation. Very rapid computation and ability to treat both discrete and continuous variables are accomplished via modified belief propagation algorithm. Being able to compute various types of reasoning from a single Bayesian network eliminates development and maintenance issues associated with the use of distinct models for different types of reasoning. Application to real-world steel production process data is presented.

1. Introduction

This work has been motivated by the need to optimize production of steel plates manufacturing. Steel plates manufacturing is a complex, multi-stage process. A manufacturing plant often produces several thousands of different steel plate SKUs. Each stage of the manufacturing is not a fully deterministic process. Sometimes there may be defects at some stage, which are then corrected by modifying the manufacturing process. In other instances, a customer may order a new steel plate, something that has not been manufactured yet. Due to these circumstances, the exact times required to produce any specific SKU is not known. Production planning and scheduling models require that we estimate the production times for each grade of steel plates. Such estimate can be made from a Bayesian network representing the manufacturing plant and probabilities of processing a steel plate at each stage of the manufacturing. Due to the complexity of the steel manufacturing plant, it is not possible to use a first-principle model of the plant to construct such Bayesian network. In this paper we use Bayesian statistics to construct the most likely Bayesian network for such complex manufacturing process. Having constructed Bayesian network, we then proceed to estimate most likely production times for each grade of steel plates. The challenging nature of the problem is magnified by the large number of different grades of steel plates. This paper presents new inference algorithm in Bayesian network with large domain discrete variables to enable us to:

1. Estimate the probability distributions of production time from historical data with large domain discrete variables and continuous variables.
2. Deal with unobservable (unavailable) variables such that we have a single model and avoid multiple models that meet with specific problems.

We present a new method for constructing most likely structure of the Bayesian network representing a complex manufacturing process, e.g. manufacturing of steel plates. Such networks contain a large number of discrete variables and also contain continuous variables parent nodes which have discrete variables children nodes. An application to a steel plate manufacturing process, which produces a large number of distinct steel plates and also has uncertain production times at different manufacturing steps, demonstrates that the proposed method can successfully compute inferences for very large hybrid Bayesian networks.

In order to learn the tree structured CPTs, scores such as Bayesian scores are often used as objective functions. However, greedy hill climbing approach cannot be used since it can easily get stuck in a local minimum at the early stage. Some kinds of approach to avoid local minimum as much as possible are proposed, but they are computationally expensive. Therefore, we employ decision trees algorithm

* Corresponding author. Tel.: +1 905 525 9140 ext.26386.
E-mail addresses: mori.nn4.junichi@jp.nssmc.com (J. Mori), mahalec@mcmaster.ca (V. Mahalec).

http://dx.doi.org/10.1016/j.eswa.2015.11.019
0957-4174/© 2015 Elsevier Ltd. All rights reserved.

(Breiman, Friedman, Olshen, & Stone, 1984) based on classification trees in order to learn the tree structured CPTs. Classification trees predict the dependent variables following decisions in the tree from the root node down to the leaf node. Since the classification trees group the values to capture important distinctions of continuous or discrete variables, this method can be used to construct the context-specific CPTs in the hybrid Bayesian networks. The classification tree classifies discrete variables into a small number of subsets so that the values of continuous or discrete child nodes can be distinguished well. If Bayesian networks include continuous parent nodes with discrete child nodes, the corresponding continuous variables can be discretized as finely as needed, because the domain size of discretized variable does not increase the number of parameters in intermediate factors due to decision-tree structured CPTs. Since the classification algorithms are typically greedy ones, the computational cost is relatively small. Consequently, the intermediate factors can be described compactly using a simple parametric representation called the canonical form.

We also introduce the decision-tree structured CPT based inference algorithm in Bayesian network, which employs belief propagation algorithm to deal with hybrid networks with large domain discrete variables. In order for multiplying and marginalizing factors during belief propagation, novel types of operations to dynamically construct CPTs are introduced. In order to carry out other types of inferences, such as causal, diagnostic, intercausal and mixed reasoning, we employ the loopy belief propagation in the decision-tree structured CPT based Bayesian networks.

The organization of the article is as follows. Review of the related prior work is presented in Section 2. Section 3 describes construction of decision-tree structured CPTs for hybrid Bayesian networks with large domain discrete variables. Section 4 describes the detailed inference algorithm including operations of product and marginalization of factors. The presented method is applied to the steel production processes data in Section 5. Finally, the conclusions are presented in Section 6.

2. Review of prior related works

In this section we review prior related works and discuss the limitations with respect the size of the problem and complexity of computation.

Let us first review prior related works and discuss the capabilities of the proposed methods with respect the size of the problem and complexity of computation.

Probabilistic graphical models are popular for representing conditional independencies among random variables under system uncertainty. Such models are comprised of nodes representing random variables and the links between the nodes which express probabilistic relationships among the corresponding random variables. Two major classes of graphical models are *Bayesian networks* and *Markov random fields*. Bayesian networks are also called *directed graphical models* since the links of the graphs represent direct dependence among the variables and are described by arrows between links. Markov random fields are also called *undirected graphical models* since they provide a simple definition of independence among random variables and do not have a particular directionality indicated by arrows (Bishop, 2006; Pearl, 1988). Both graphical models are popular in the machine learning community and have been applied to various fields including medical diagnostics, speech recognition, gene modeling, cancer classification, target tracking, sensor validation, and reliability analysis.

In particular, Bayesian network has been widely used for systems including many uncertainties. For instance, scenario analysis under changing conditions is implemented by means of Bayesian inference techniques, since Bayesian network performs well in uncertainty environment (Buyukozkan, Kayakutlu, & Karakadlar, 2015; Cai, Sun, Si, & Yannou, 2011). Bayesian network is also applied to predicting the risk of software development or maintenance projects, because it is suitable for representing the knowledge of experts under uncertainty of conditions (Melo & Sanchez, 2008; Perkusich & Soares, 2014). As these applications indicate, Bayesian network is a powerful tool for knowledge representation and reasoning under uncertainties since it can visually represents the probabilistic relationships among measured and unmeasured variables.

Each node in a Bayesian network is associated with conditional probability distributions (CPD). The most common representations of CPDs are conditional probability tables (CPTs), which specify marginal probability distributions for each combination of values of its discrete parent nodes. The number of parameters required to represent CPTs grows exponentially both with the number of discrete variables and with the cardinality of discrete variables. In order to reduce the number of parameters, context-specific independence representations have been proposed (Boutilier, Friedman, Goldszmidt, & Koller, 1996). Furthermore, an efficient inference algorithm that exploits context-specified independence has also been proposed (Poole & Zhang, 2003). As for identification of parameters of context-specific independence, learning methods such as tree-structured CPTs (Friedman & Goldszmidt, 1996) and graph-structured CPTs (Chickering, Heckerman, & Meek, 1997) have been developed. However, since learning structured CPTs is NP-hard problems, all of these methods assume that all discrete variables have a bounded number of values or that they are already grouped at an appropriate level of domain size. In the real world problem, discrete variables often have large domains and the task of grouping discrete values requires expert knowledge that enables us to identify a reasonable set of groups that well distinguish the values of discrete variables. In order to group the discrete values in a Bayesian network learning, $attribute - value\ hierarchies$ (AVHs) which capture meaningful groupings of values in a particular domain are integrated with the tree-structured CPTs (DesJardins & Rathod, 2008). However, if large domain discrete variables do not contain hierarchal structures, AVHs cannot capture the useful abstracts of values in that domain. In addition, this model cannot handle the continuous variables without discretizing them. The authors, DesJardins and Rathod (2008) also do not apply AVH-derived CPTs to inference in Bayesian networks.

Sharma and Poole (2003) have proposed an inference method for Bayesian Networks containing CPTs which are represented as decision trees. The inference algorithm is based on variable elimination (VE) algorithm; the authors introduced operations for decision trees computations, namely multiplying factors and summing out variable from a factor. However, because the computational complexity of the exact inference algorithm such as VE grows exponentially with the size of the network, this method may not be appropriate for Bayesian networks in real world. In addition, computational cost of reconstructing decision trees as required to compute multiplication and marginalization of factors makes this method too expensive to apply to large decision trees. An alternative approach is to employ algebraic decision diagrams (ADDs) for the purpose of inference in Bayesian network with large domain discrete variables. For instance, ADDs have been used to represent factors and their multiplying and summing-out operations have been proposed (Chavira & Darwiche, 2007). In addition, structured message passing has been proposed to utilize powerful approximate inference algorithms such as cluster-graph Belief propagation (Gogate & Domingos, 2013). In the worst case, ADDs have the same space complexity as CPTs. To make matters worse, the factor operations of multiplication and summing-out are polynomial in time. It should be noted that all the above methods have not considered an application of decision-tree structured CPTs to hybrid Bayesian networks, where both discrete and continuous variables appear simultaneously.

In hybrid Bayesian networks, the most commonly used model that allows exact inference is the conditional linear Gaussian (CLG) model (Lauritzen, 1992; Lauritzen & Jensen, 2001). The corresponding graphical model does not allow discrete variables to have continuous parents. To overcome this limitation, the CPDs of these nodes are typically modeled as softmax function. Since there is no exact inference algorithm for such model, the inference is typically carried out by means of approximate inference such as Monte Carlo method (Koller, Lerner, & Angelov, 1999). One problem with the Monte Carlo method is that the convergence can be quite slow especially when we deal with the hybrid networks with large domain of discrete variables. An alternative is to discretize all continuous variables in a network and treat them as if they are discrete (Kozlov & Koller, 1997). However, since the number of variables in intermediate factors can be quite large, it is typically impossible to discretize the continuous variables as finely as required to obtain reasonable solutions. This discretization leads to a trade-off between accuracy of the approximation and the cost of computation.

An alternative approach is the mixture of truncated exponential (MTE) model to represent the hybrid Bayesian networks (Moral, Rumi, & Salmern, 2001). MTE models approximate arbitrary probability distribution functions (PDFs) using exponential terms and allow implementation of inference in hybrid Bayesian networks. The main advantage of this method is that standard propagation algorithm can be used. Parameter estimation and propagation algorithms for MTE models have been extensively studied (Cobb, Rumi, & Salmeron, 2007; Rumi & Salmeron, 2007). This method is also limited to relatively modest size models, since the number of regression coefficients in exponential functions linearly grows in the domain size of discrete variables. Hence, MTE model may not work well in the large Bayesian networks that contain large domain of discrete variables. Extended probability trees for probabilistic graphical models are also proposed (Cano, Gómez-Olmedo, Moral, & Pérez-Ariza, 2014). These trees allow the representation of multiplicative and additive factorization. However, similar idea as MTE is employed to operate with discrete distributions and thus this algorithm is also limited to modest size models.

3. Decision-tree structured CPT for large domain of discrete and continuous variables

A hybrid Bayesian network represents a probability distribution over random variables in which each node is either discrete or continuous. $\mathcal{X} = \Gamma \cup \Delta$ represents sets of random variables ,where Γ denotes the continuous variables and Δ represents the discrete variables. We denote random discrete variables by upper case letters from the beginning of the alphabet (e.g. A, B, A_1) while continuous variables are represented by upper case letters near the end (e.g. X, Y, X_1). Their actual values are represented by the lower case letters (e.g. x). We denote sets of variables by bold upper case letters (e.g. $\mathbf{X}$) and the assignments of those sets by the bold lower case letters (e.g. $\mathbf{x}$). Val(X) is used to represent the set of values that a random variable X can take.

The most widely used representation of hybrid Bayesian networks is the conditional linear Gaussian (CLG) model. Let $X \subseteq \Gamma$ be a continuous node, $A \subseteq \mathrm{pa}(X) \cap \Delta$ be its discrete parent nodes and $Y_1, \ldots, Y_k \subseteq \mathrm{pa}(X) \cap \Gamma$ be its continuous parent nodes where pa(X) denotes a state of parent nodes of X. X has a Gaussian distribution and the mean of the distribution is computed as a linear combination of the state of its parent nodes pa(X) for every instantiation $a \in \mathrm{Val}(A)$ as follows:

$$p(X|\mathrm{pa}(X)) = P(X|\mathrm{pa}(X);\, \Theta_a) = \mathcal{N}\left(X \,\middle|\, \sum_j^k w_{a,j} y_j + b_a, \sigma_a^2\right) \quad (1)$$

where $w_{a,j}$ and b_a are parameters controlling the mean, σ_a represents the standard deviation of the conditional distribution for X, and

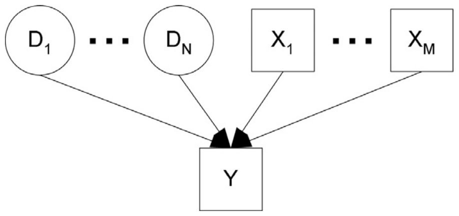


Fig. 1. Bayesian network that contains continuous child node.

$\Theta_a = \{w_{a,j}, b_a, \sigma_a\}$ is the set of model parameters of instantiation a. Since this representation is not Gaussian but a conditional distribution, canonical form is used as a more general representation. In the canonical table representation, a factor $C(\mathbf{X};\, K_a, h_a, g_a)$ which is a function over a set of variables is defined as (Lauritzen & Wermuth, 1989):

$$C(\mathbf{X};\, K_a, h_a, g_a) = \exp\left(-\frac{1}{2}\mathbf{X}^T K_a \mathbf{X} + \mathbf{h}_a^T \mathbf{X} + g_a\right) \quad (2)$$

where K_a, $\mathbf{h}_a$ and g_a represent parameters of ath instantiation in canonical table representations. The canonical table representation can express both the canonical form used in continuous networks and the table factors used in discrete networks. When $A = \emptyset$, only a single canonical form ($a = 1$) is obtained. Meanwhile, when $X = \emptyset$, parameters of K_a and $\mathbf{h}_a$ are vacuous and only a canonical form $\exp(g_a)$ remains for each instantiation a.

Unfortunately, the canonical table representation cannot represent the dependence of a discrete child node with continuous parent nodes. The simplest approach to resolve this issue is to discretize the variables of continuous parent nodes. As discussed in the introduction section, there is a trade-off between the accuracy of the approximation and the domain size of discretized variables. However, the decision-tree structured CPDs proposed in this paper can handle unbounded number of values in discrete variables and we can ignore this issue.

In order to compute the parameters of canonical table, one first needs to define the instantiations of the set of discrete variables. The simplest way is to use conditional probability tables. However, the number of parameters required to describe a Bayesian network increases exponentially with the domain size. In the case when a discrete node A has discrete parent nodes pa(A), the number of conditional probabilities is $|A| \times |\mathrm{pa}(A)|$ and it is impossible to specify all the parameters of the large domain of discrete variables. This is the case particularly when we discretize very finely the continuous parent nodes in order to get an accurate approximation, since the domain size of the discretized variables becomes large. To exploit conditional independence that holds only in certain contexts in the Bayesian networks, several works about structured CPTs have been done to capture this independence (Boutilier et al., 1996; Poole & Zhang, 2003). However, since learning structured CPTs is an NP-hard problem, all of these methods assume either that all discrete variables have a bounded number of values or that they are already grouped at an appropriate level of domain size.

To overcome these limitations of the existing methods, we employ decision tree algorithms such as classification trees and regression trees for learning structured CPTs. Classification trees give the prediction of the dependent variables by following decisions in the tree from the root node down to the leaf node. The classification trees can predict the discrete and continuous variables and the trees are identified by choosing a split based on the some statistical measures such as Gini impurity for the classification trees and mean square error for the regression trees.

Let us first consider a factor that contains both continuous nodes $\mathbf{X}$ and discrete nodes $\mathbf{D}$ with continuous child node Y as shown in Fig. 1.

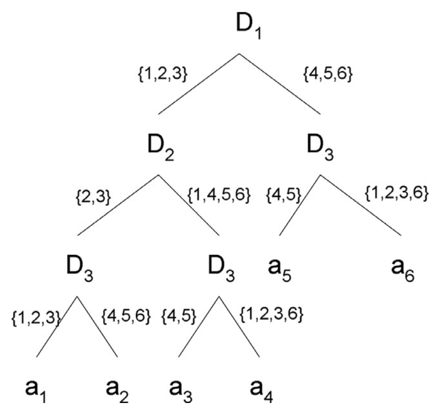


Fig. 2. Decision tree of a factor that contains continuous child node.

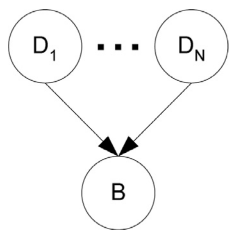


Fig. 3. Bayesian network that contains discrete child node.

Continuous variables are shown as rectangles while discrete variables are ovals. The classification tree f classifies the discrete variable set $\mathbf{D}$ into a small number of subsets $A = \{a_1, a_2, \ldots, a_L\}$ as $a = f(\mathbf{d})$ where L is the number of the leaves in the decision tree f. An example of the constructed decision tree is given in Fig. 2.

For instance, the decision tree indicates that if $D_1 \in \{1, 2, 3\}$, $D_2 \in \{2, 3\}$ and $D_3 \in \{1, 2, 3\}$, then the corresponding records are assigned the instantiation a_1. Since the domain size of each discrete variable is 6, the number of parameters required to describe the CPT of discrete variables $\mathbf{D}$ is $|\mathrm{Val}(D_1)| \times |\mathrm{Val}(D_2)| \times |\mathrm{Val}(D_3)| = 216$. On the other hand, the decision tree representation requires only $|\mathrm{Val}(A)| = 6$ parameters to describe the behavior of Y, instead of 216 in the CPT representation. Therefore, the decision-tree representation requires smaller training data to learn the conditional probability distribution and leads to much more robust estimation than the traditional CPT representation.

After giving instantiation $a_i \in A$ to all records in the training data, parameters of the canonical table representation can be computed as follows:

$$\begin{aligned} K_{a_i} &= \Sigma_i^{-1} \\ \mathbf{h}_{a_i} &= \Sigma_i^{-1}\mu_i \\ g_{a_i} &= -\frac{1}{2}\mu_i^T \Sigma_i^{-1}\mu_i - \log\left((2\pi)^{n/2}|\Sigma_i|^{1/2}\right) \end{aligned} \tag{3}$$

with

$$\Sigma_i = \mathrm{cov}[\mathbf{Z}_i] \tag{4}$$

$$\mu_i = \mathrm{mean}[\mathbf{Z}_i] \tag{5}$$

where $\mathbf{Z} = \mathbf{X} \cup Y$, $\mathbf{Z}_i = \{z : f(\mathbf{d}) = a_i\}$ and $\mathrm{cov}[\mathbf{Z}_i]$ and $\mathrm{mean}[\mathbf{Z}_i]$ are the covariance matrix and mean vector of $\mathbf{Z}_i$, respectively. Tree structured CPTs induce refined independence that holds only in certain contexts, which cannot be achieved by the table representation. In the example shown in Fig. 2, if the evidence $D_1 = 4$,5 or 6 is given, we immediately know that the continuous variables $\mathbf{Z} = \mathbf{X} \cup Y$ follow the distribution either $C(\mathbf{Z}; K_{a_5}, \mathbf{h}_{a_5}, g_{a_5})$ or $C(\mathbf{Z}; K_{a_6}, \mathbf{h}_{a_6}, g_{a_6})$, regardless of the value of D_2. Thus, we can conclude that D_2 and Y are contextually independent given the evidence $D_1 = \{4, 5, 6\}$ denoted $(D_2 \perp_c Y | D_1 = \{4, 5, 6\})$, which is described as follows:

$$P(D_2|Y, D_1 = \{4, 5, 6\}) = P(D_2|D_1 = \{4, 5, 6\}) \ . \tag{6}$$

Next, let us consider the case when a factor contains discrete nodes $\mathbf{D}$ with discrete child node B as shown in Fig. 3.

The classification tree classifies discrete variables $\mathbf{D}$ into a small number of subsets $A = \{a_1, a_2, \ldots, a_L\}$ as $a = f(\mathbf{d})$ with L being the number of leaves so that the values of a discrete child node B can be categorized well. The decision tree can categorize well the discrete parent variables $\mathbf{D}$, but a discrete child variable B (which is used

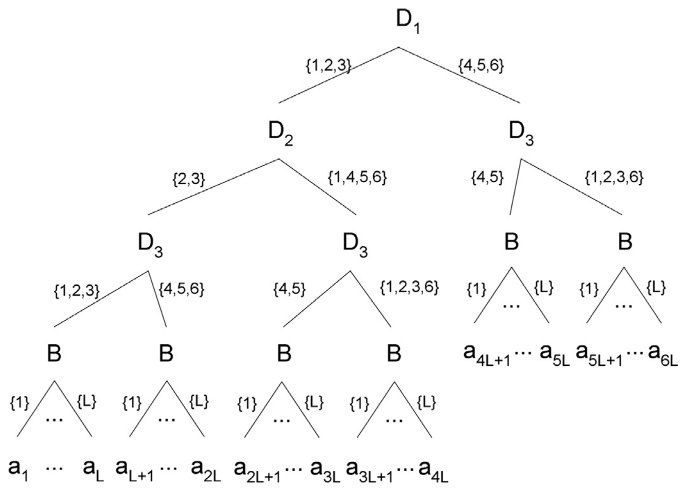


Fig. 4. Decision tree of a factor that contains discrete child node.

as a dependent variable in the classification tree algorithm) is not included in the nodes of the decision tree. Therefore, the obtained decision tree classifies only the discrete parent variables $\mathbf{D}$ only and excludes a discrete child variable B from its branching nodes. In order to use all discrete variables (including a discrete child variable) as the brunching nodes, the discrete child variable is placed at all leaf nodes to continue categorizing discrete variables by using their values. Therefore, all values of a discrete child variable $b_1 \ldots b_L \in B$ are used as categorical splits and thus the number of branches at each leaf node in the original tree is same as the domain size of $\mathrm{Val}(B) = L$. As a result, the total number of instantiations assigned in the final decision tree is $\mathrm{Val}(B)$ times as many as the number of leaves in the original decision tree. Fig. 4 shows the example of a decision tree where a factor contains a discrete child node.

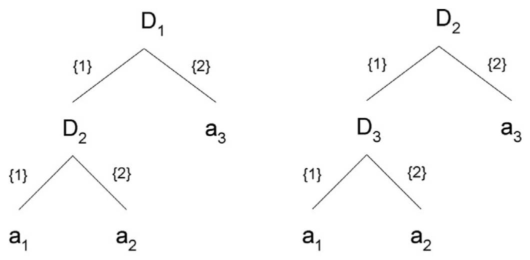
The decision tree indicates that if $D_1 \in \{1, 2, 3\}$, $D_2 \in \{2, 3\}$, $D_3 \in \{1, 2, 3\}$ and $B \in \{1\}$, then the corresponding records are assigned the instantiation a_1. The number of instantiations are $\mathrm{Val}(B) \times 6 = 6L$ rather than $|\mathrm{Val}(D_1)| \times |\mathrm{Val}(D_2)| \times |\mathrm{Val}(D_3)| \times |\mathrm{Val}(B)| = 216L$ which is the number of parameters required to describe the traditional CPT of discrete variables $\mathbf{D} \cup B$.

With assigned instantiations $a_i \in A$, parameters g_{a_i} of the canonical table representation can be computed below:

$$g_{a_i} = \log \frac{1}{N_{a_i}} \sum_{z} \mathbf{1}(f(\mathbf{d}) = a_i) \tag{7}$$

where $\mathbf{1}$ is an indicator function, N_{a_i} is the number of samples belonging to the instantiation a_i and the other parameters K_{a_i} and $\mathbf{h}_{a_i}$ are vacuous. Similar to the previous case, tree structured CPTs hold context specific independence. In the example shown in Fig. 4, if the evidence $D_1 = 4$, 5 or 6 is given, we compute each probability of $\mathrm{Val}(B)$, regardless of the value of D_2. Thus, it can be said that D_2 and B are contextually independent given the evidence $D_1 = \{4, 5, 6\}$ denoted $(D_2 \perp_c B | D_1 = \{4, 5, 6\})$, which is described as follows:

$$P(D_2|B, D_1 = \{4, 5, 6\}) = P(D_2|D_1 = \{4, 5, 6\}). \tag{8}$$



(a) Tree for Factor A (b) Tree for Factor B

Fig. 5. Example of two decision trees.

Since this implementation uses a binary tree algorithm to create decision tree, the complexity is $O(\log N)$ with N being the number of samples.

4. Inference

The computational task of inference in Bayesian network is to answer the *conditional probability query*, $P(\mathcal{X}|\mathbf{E}=\mathbf{e})$ where the evidence $\mathbf{E} \subseteq \mathcal{X}$ is a subset of random variables in the Bayesian network and $\mathbf{e} \in \mathrm{Val}(\mathbf{E})$ is the set of values given to these variables. For causal reasoning, the sum-product algorithm is carried out from top node factors to bottom node factors. Meanwhile, other types of inference such as diagnostic, intercausal and mixed reasoning do not have monotonic logic based algorithm and the basic algorithm for exact inference for these types of reasoning in graphical models is *variable elimination* (VE). VE can be performed in a clique tree and each clique in the clique tree is the factor in VE. The computational cost of the exact inference such as clique tree algorithm exponentially grows with the width of the network and thus the exact inference algorithms are infeasible for large width networks. On the other hand, approximate inference algorithms such as loopy belief propagation (BP) algorithm are tractable for many real-world graphical models. All the above inference techniques require computation of the multiplying factors and marginalizing variables in factors. In this work, since the factors are represented by using decision-tree structured CPTs as proposed in the previous section, the operations of multiplication and marginalization are described in the first parts of this section.

4.1. Multiplying factors and marginalizing over variables in factors

Initial potentials of factors are equal to the decision-tree structured CPTs while potentials of final and intermediate factors including messages between clusters are computed by the operations of multiplying and marginalizing factors. These operations require us to divide the large domain discrete variables in accordance with the decision-tree structure. For this purpose, in this section we describe three types of two main operations: multiplying factors and marginalizing over variables in factors.

4.1.1. Full conditional probability table based operations

The simplest approach is to convert the tree structured CPTs into conditional probability tables which can be analyzed by using standard multiplying and marginalizing techniques. We shall call this CPT *full conditional probability table* (full CPT). Let us consider the simple example of multiplying two factors described in Fig. 5. Domain size of each discrete variable is 2 ($\mathrm{Val}(D_1) = \mathrm{Val}(D_2) = \mathrm{Val}(D_3) = 2$) and the number of instantiations of each tree is 3 (a_1, a_2, a_3).

First, a full CPT is designed in accordance with the decision-tree structured CPTs as shown in Table 1 and using these full CPTs, a new factor C of multiplying two factors is determined as shown in Table 3.

For instance, let us consider the case where $D_1 = 1, D_2 = 2, D_3 = 1$. According to the full CPTs, the first row in the full CPT for factor A shown in Table 1 and the second row in the full CPT for factor B shown

Table 1
Full CPT for factor A.

D_1	D_2	Probability distribution
1	1	$C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1})$
2	1	$C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3})$
1	2	$C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2})$
2	2	$C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3})$

Table 2
Full CPT for factor B.

D_2	D_3	Probability distribution
1	1	$C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1	$C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
1	2	$C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	2	$C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$

Table 3
Full CPT for factor C.

D_1	D_2	D_3	Probability distribution
1	1	1	$C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1	1	$C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	2	1	$C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
2	2	1	$C(\mathbf{X}; K^C_{a_4}, \mathbf{h}^C_{a_4}, g^C_{a_4}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
1	1	2	$C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	1	2	$C(\mathbf{X}; K^C_{a_6}, \mathbf{h}^C_{a_6}, g^C_{a_6}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
1	2	2	$C(\mathbf{X}; K^C_{a_7}, \mathbf{h}^C_{a_7}, g^C_{a_7}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
2	2	2	$C(\mathbf{X}; K^C_{a_8}, \mathbf{h}^C_{a_8}, g^C_{a_8}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$

in Table 2 are used for computing a new factor $C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3})$. The canonical parameters of this new factor are computed as follows:

$$
\begin{aligned}
K^C_{a_3} &= K^A_{a_2} + K^B_{a_3} \\
\mathbf{h}^C_{a_3} &= \mathbf{h}^A_{a_2} + \mathbf{h}^B_{a_3} \\
g^C_{a_3} &= g^A_{a_2} + g^B_{a_3}.
\end{aligned}
\tag{9}
$$

Since the scopes of two factors are different, we need to add zero entries to the K matrices and $\mathbf{h}$ vectors of both factors so that their scopes are consistent.

Next, let us consider summing out variables in factors. If we need to marginalize factors that contain continuous variables by summing out discrete variables, the resulting distribution is not a Gaussian but a mixture of Gaussian distributions. Therefore, mixtures of Gaussians model can be used to represent the factors and Monte Carlo integration are carried out to compute the marginalization (Yuan & Druzdzel, 2006). However, since the number of Gaussian components grow every time after summing out the discrete variables, we typically approximate the mixture of Gaussian distributions by collapsing them into a single Gaussian distribution. Similarly to multiplying the factors, first we convert the decision-tree structured CPTs into full conditional probability tables. In practice since marginalization operations are needed only after multiplying operations, full conditional probability tables are already obtained when marginalization is necessary. Let us consider summing out variable D_3 in the full CPT given in Table 3.

The marginal distribution over D_1 and D_2 assigns probability distributions to specific events such as $P(D_1 = 1, D_2 = 1)$, $P(D_1 = 2, D_2 = 1)$, $P(D_1 = 1, D_2 = 2)$ and $P(D_1 = 2, D_2 = 2)$. For instance, the marginal probability distribution function $P(D_1 = 1, D_2 = 1)$ can be computed as follows:

$$P(D_1 = 1, D_2 = 1) = C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) + C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}). \quad (10)$$

As mentioned before, however, this distribution is not a Gaussian and thus it cannot be represented in a canonical form. Instead, we employ the weak discrete marginalization where the summing operation uses the collapsing operation (Koller & Friedman, 2009). The collapsing operation approximates a mixture of k Gaussian distribution $\mathcal{N}(\mu_i, \Sigma_i)$ by a single Gaussian distribution $\mathcal{N}(\mu_j, \Sigma_j)$. Its mean vector $\boldsymbol{\mu}_j$ and covariance matrix Σ_j for new instantiation j are defined as follows:

$$\mu_j = \sum_{i=1}^{k} p(i|j)\mu_i \quad (11)$$

$$\Sigma_j = \sum_{i=1}^{k} p(i|j)\Sigma_i + \sum_{i=1}^{k} p(i|j)(\mu_i - \mu)(\mu_i - \mu)^T \quad (12)$$

where $p(i|j)$ is the conditional probability of the original instantiation i given the new instantiation j and it satisfies $\sum_{i=1}^{k} p(i|j) = 1$. Therefore, the parameters $(K_{\text{new}}, \mathbf{h}_{\text{new}}, g_{\text{new}})$ of the canonical form $P(D_1 = 1, D_2 = 1)$ after summing out variable D_3 can be computed as follows:

$$K_{\text{new}} = \Sigma^{-1}_{\text{new}} \quad (13)$$

$$\mathbf{h}_{\text{new}} = \Sigma^{-1}_{\text{new}}\mu_{\text{new}} \quad (14)$$

where

$$\begin{aligned}\mu_{\text{new}} &= p(D_3 = 1|D_1 = 1, D_2 = 1)K^{C-1}_{a_1}\mathbf{h}^C_{a_1}\\ &\quad + p(D_3 = 2|D_1 = 1, D_2 = 1)K^{C-1}_{a_5}\mathbf{h}^C_{a_5}\end{aligned} \quad (15)$$

$$\begin{aligned}\Sigma_{\text{new}} &= p(D_3 = 1|D_1 = 1, D_2 = 1)K^{C-1}_{a_1}\\ &\quad + p(D_3 = 2|D_1 = 1, D_2 = 1)K^{C-1}_{a_5}\\ &\quad + p(D_3 = 1|D_1 = 1, D_2 = 1)(K^{C-1}_{a_1}\mathbf{h}^C_{a_1} - \mu_{\text{new}})\\ &\quad \times (K^{C-1}_{a_1}\mathbf{h}^C_{a_1} - \mu_{\text{new}})^T + p(D_3 = 2|D_1 = 1, D_2 = 1)\\ &\quad \times (K^{C-1}_{a_5}\mathbf{h}^C_{a_5} - \mu_{\text{new}})(K^{C-1}_{a_5}\mathbf{h}^C_{a_5} - \mu_{\text{new}})^T.\end{aligned} \quad (16)$$

After summing out discrete variables, the operation of continuous marginalization is carried out if we need to integrate the continuous variables $\mathbf{Y}$. We assume that the canonical table consists of a set of canonical forms $C(\mathbf{X}, \mathbf{Y}; K^C_{a_i}, \mathbf{h}^C_{a_i}, g^C_{a_i})$ indexed by a_i after summing out discrete variables where

$$K_{a_i} = \begin{bmatrix} K_{\mathbf{XX}} & K_{\mathbf{XY}} \\ K_{\mathbf{YX}} & K_{\mathbf{YY}} \end{bmatrix} \quad (17)$$

$$\mathbf{h}_{a_i} = \begin{pmatrix} \mathbf{h}_{\mathbf{X}} \\ \mathbf{h}_{\mathbf{Y}} \end{pmatrix}. \quad (18)$$

If $K_{\mathbf{YY}}$ is positive definite, then the integral over the variables $\mathbf{Y}$ becomes a canonical form $C(\mathbf{X}; K'^C_{a_i}, \mathbf{h}'^C_{a_i}, g'^C_{a_i})$ computed as (Lauritzen, 1992):

$$\begin{aligned}K'^C_{a_i} &= K_{\mathbf{XX}} - K_{\mathbf{XY}}K^{-1}_{\mathbf{YY}}K_{\mathbf{YX}}\\ \mathbf{h}'^C_{a_i} &= \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}}K^{-1}_{\mathbf{YY}}\mathbf{h}_{\mathbf{Y}}\\ g'^C_{a_i} &= g + \frac{1}{2}\left(\log|2\pi K^{-1}_{\mathbf{YY}}| + \mathbf{h}^T_{\mathbf{Y}}K^{-1}_{\mathbf{YY}}\mathbf{h}_{\mathbf{Y}}\right).\end{aligned} \quad (19)$$

Above methodology enables accurate computation of multiplication and marginalization of factors using full CPTs. This method is easy to carry out and computationally it is very fast. However, to describe the full CPT of a factor, the number of required rows grows exponentially both with the size of domain and with the number of discrete variables that a factor contains. For instance, let the size of the domain be D and the number of child nodes be C, then the number of rows of full CPT is D^C, which is too large for a large domain of discrete variables. Therefore, this approach is not efficient in terms of memory and it cannot work for large hybrid Bayesian networks where factors contain a large number of discrete variables or a huge domain of discrete variables. In addition, the complexity of multiplying is $O(D^{2C})$.

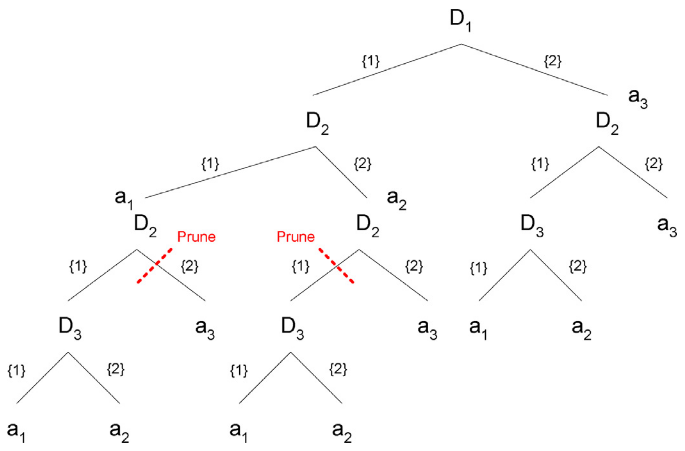


Fig. 6. Merging tree T_A and tree T_B.

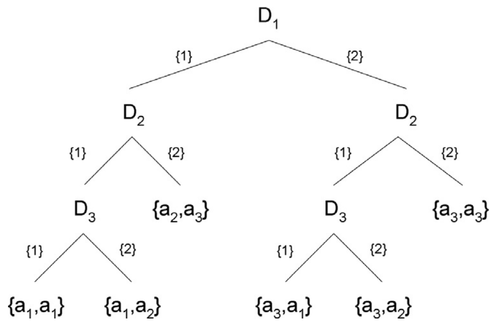


Fig. 7. Pruning merged tree.

4.1.2. Tree structured CPT based operations

The full CPT based multiplying and marginalizing operations may be impossible to carry out for a large scale real-world Bayesian networks due to the finite size of computer memory. Hence, we propose another approach by making use of the decision-tree structure. In this approach, we dynamically reconstruct the decision trees every time after multiplying or marginalizing factors and thus full CPTs are not needed. As an example, let us consider the product of factors using the example from the previous section, Fig. 5. Two trees, named as T_A and T_B, are combined to form a single tree so that all the distinctions that are made both in tree T_A and in tree T_B are made. The scopes of these trees are represented by $\mathbf{C}_A = \text{scope}(T_A)$ and $\mathbf{C}_B = \text{scope}(T_B)$. Pval($j$, $\mathbf{D}$, T) is used to represent the set of possible combination of values that a set of discrete variables $\mathbf{D}$ can have at the node j in the tree T. First, the leaves of the tree T_A are replaced by the tree T_B as shown in Fig. 6.

One can readily confirm that this merged tree can make all the distinction that both trees T_A and T_B make, but it also contains redundant sub-trees that are incompatible with its ancestors in the decision tree. For example, after branching at $D_1 = 1$ and $D_2 = 1$ the right child node specifies $D_2 = 2$ which conflicts with its ancestors. Therefore, we prune the right child node and also remove its corresponding node. This leads to the simplified tree as shown in Fig. 7.

After merging two trees, we can compute the canonical parameters of the new factor. For example, the canonical parameters given

Table 4
Algorithm 1. Merging Trees.

```
   Procedure Merge-Trees (
   Tree $T_A$, $T_B$ // Decision trees
   )
1  $\mathbf{S} = \mathbf{C}_A \cap \mathbf{C}_B$ // Sepset of discrete variables among two factors
2  $T_{\text{new}} = T_A$
3  Let $a_1, \ldots, a_M$ be indices of leaf nodes in $T_{\text{new}}$
4  for $i = 1, ldots, M$
5    $T'_B = T_B$
6    if $S \neq \emptyset$
7      $List = \emptyset$ // List of next visiting nodes in $T_B$
8      Add the top node of $T_B$ to $List$
9      while $List \neq \emptyset$
10       if node $List(1)$ has child nodes
11         Let $l$ and $r$ be the left and right child nodes of the node $List(1)$
12         if $\text{Pval}(a_i, \mathbf{S}, T_A) \cap \text{Pval}(\text{List}(l), \mathbf{S}, T_B) = \emptyset$
             and $\text{Pval}(a_i, \mathbf{S}, T_A) \cap \text{Pval}(\text{List}(r), \mathbf{S}, T_B) \neq \emptyset$
13           Replace current node $List(1)$ with node $r$ and remove node $l$
14           Add node $r$ to the end of $List$
15         elseif $\text{Pval}(a_i, \mathbf{S}, T_A) \cap \text{Pval}(\text{List}(l), \mathbf{S}, T_B) \neq \emptyset$
             and $\text{Pval}(a_i, \mathbf{S}, T_A) \cap \text{Pval}(\text{List}(r), \mathbf{S}, T_B) = \emptyset$
16           Replace current node $List(1)$ with node $l$ and remove node $r$
17           Add node $l$ to the end of $List$
18         else
19           Add $l$ and $r$ to the end of $List$
20       Delete $List(1)$
21   Replace the nodes $a_i$ of tree $T_{\text{new}}$ with the new structure $T'_B$
22 return $T_{\text{new}}$
```

$\{D_1, D_2, D_3\} = \{1, 1, 1\}$ are computed as follows:

$$K^C = K^A_{a_1} + K^B_{a_1}$$
$$\mathbf{h}^C = \mathbf{h}^A_{a_1} + \mathbf{h}^B_{a_1}$$
$$g^C = g^A_{a_1} + g^B_{a_1}. \quad (20)$$

The algorithm for merging two trees is presented in Table 4.

Next, let us consider summing out variables in decision trees. As an example, we consider summing out discrete variables D_1 and D_2 in the decision tree described in Fig. 2. We search for summed out variables D_1 and D_2 from the root node down to the leaf node. In this example, it can be immediately found that the root node is the summed out variable D_1. Therefore, we divide the tree into two sub-trees at the node D_1, then two sub-trees are merged by using *Merge-Trees* algorithm as shown in Fig. 8.

After that, we continue searching the summed out variables from the root node down to the leaf node in the obtained new tree. We find that the summed out variable D_2 is placed on the root node of the tree and then similar to the previous step, we divide the tree at the node D_2 into two sub-trees and combine them into one tree using *Merge-Trees* algorithm as shown in Fig. 9. Since there is no summed out variable in the current tree, marginalization step of discrete variables is achieved.

With the obtained new tree, weak marginalization is carried out for summing out discrete variables. For instance, the mean vector and covariance matrix of a canonical form $P(D_3 = 1)$ after summing out variables D_1 and D_2 can be computed as follows:

$$\mu_{\text{new}} = w_1 K^{-1}_{a_1}\mathbf{h}_{a_1} + w_4 K^{-1}_{a_4}\mathbf{h}_{a_4} + 2w_6 K^{-1}_{a_6}\mathbf{h}_{a_6} \quad (21)$$

$$\begin{aligned}\Sigma_{\text{new}} = {} & w_1 K^{-1}_{a_1} + w_4 K^{-1}_{a_4} + 2w_6 K^{-1}_{a_6} \\ & + w_1 (K^{-1}_{a_1}\mathbf{h}_{a_1} - \mu_{\text{new}})(K^{-1}_{a_1}\mathbf{h}^C_{a_1} - \mu_{\text{new}})^T \\ & + w_4 (K^{-1}_{a_4}\mathbf{h}_{a_4} - \mu_{\text{new}})(K^{-1}_{a_4}\mathbf{h}^C_{a_4} - \mu_{\text{new}})^T \\ & + 2w_6 (K^{-1}_{a_6}\mathbf{h}_{a_6} - \mu_{\text{new}})(K^{-1}_{a_6}\mathbf{h}^C_{a_6} - \mu_{\text{new}})^T \end{aligned} \quad (22)$$

with

$$w_1 = p(D_3 = 1 | D_1 = \{1, 2, 3\}, D_2 = \{2, 3\})$$
$$w_4 = p(D_3 = 1 | D_1 = \{1, 2, 3\}, D_2 = \{1, 4, 5, 6\})$$
$$w_6 = p(D_3 = 1 | D_1 = \{4, 5, 6\}). \quad (23)$$

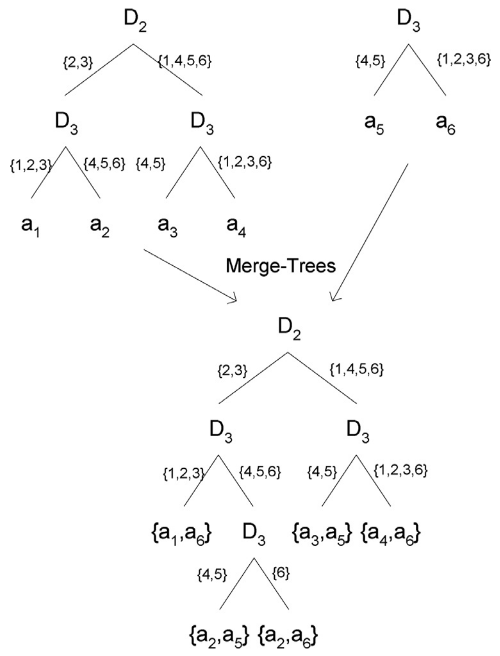


Fig. 8. Summing out D_1.

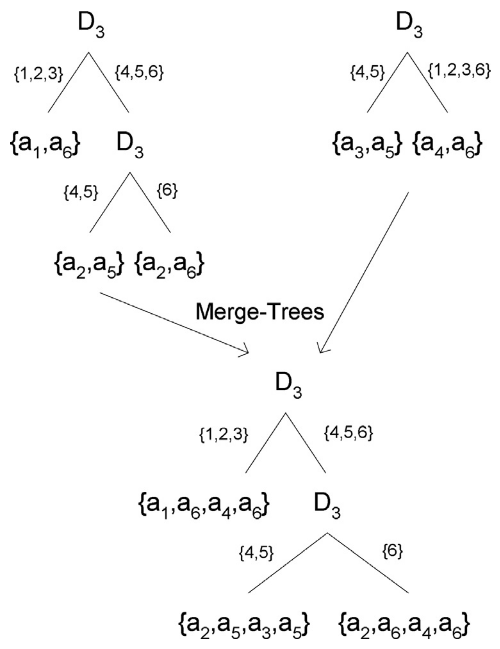


Fig. 9. Summing out D_2.

After summing out the discrete variables, continuous variables are integrated out by using Eq. (19). The detailed algorithm for merging two trees is shown in Table 5.

Above procedure enables us to multiply factors and marginalize over both discrete and continuous variables through dynamically reconstructing decision trees. This method can retain the tree structure since the number of operations and the memory size required to multiply and marginalize factors are much smaller compared to the full CPT based operations. It should be pointed out that the computational cost of marginalization increases exponentially with the number of discrete nodes that we need to sum out. At worst case, the number of summed out nodes are increasing with $O(N^2)$ during marginalization if we cannot prune the redundant nodes. Therefore, depending on the situation, the algorithm may not converge in an

Table 5
Algorithm 2. Summing out Tree.

```
   Procedure Sum-Out-Tree (
   Tree $T$ // Decision tree
   A set of discrete variables $\mathbf{V}$ // Summed out variables
   )
1  $T_{\text{new}} = T$
2  while scope($T_{\text{new}}$)$\cap \mathbf{V} \neq \emptyset$
3    Let $i$ be the top node of scope($T_{\text{new}}$)$\cap \mathbf{V}$
4    Make sub-trees $T_A$ and $T_B$ whose root nodes are the child nodes of node $i$
5    $T_{\text{new}}$ = Merge-Trees ($T_A$, $T_B$)
6  return $T_{\text{new}}$
```

Table 6
Compact CPT for factor *A*.

D_1	D_2	Probability distribution
1	1	$C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1})$
1	2	$C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_3}, g^A_{a_2})$
2	1,2	$C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_2}, g^A_{a_3})$

Table 7
Compact CPT for factor *B*.

D_2	D_3	Probability distribution
1	1	$C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	2	$C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	1,2	$C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$

Table 8
Combined compact CPT of factors *A* and *B*.

Factor *A*		Factor *B*		Probability
D_1	D_2	D_2	D_3	distribution
		1	1	$C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	1	1	2	$C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
		2	1,2	Inconsistent
		1	1	Inconsistent
1	2	1	2	Inconsistent
		2	1,2	$C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
		1	1	$C(\mathbf{X}; K^C_{a_4}, \mathbf{h}^C_{a_4}, g^C_{a_4}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1,2	1	2	$C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
		2	1,2	$C(\mathbf{X}; K^C_{a_6}, \mathbf{h}^C_{a_6}, g^C_{a_6}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$

acceptable length of time. Nevertheless, compared to the previous approaches, the proposed method allows us to deal with larger domain of discrete variables.

4.1.3. Compact conditional probability table based operations

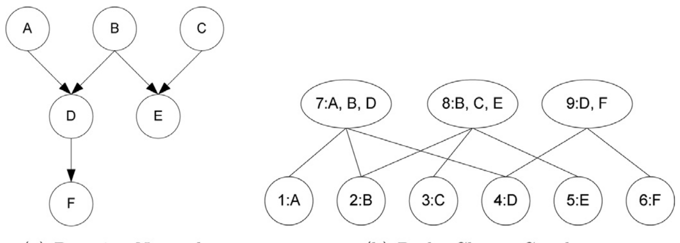
Last type of the operations for factor product and marginalization is based on the set of possible combination of values that a set of discrete variables $\mathbf{D}$ can take at each leaf node. Let us consider again example in Fig. 5 used in the previous section. First we convert the tree structured CPTs into conditional probability tables where each row corresponds to each leaf node and each entry represents a probability distribution as shown in Tables 6 and 7.

This conditional probability table will be called *compact conditional probability table* (compact CPT). Next, we combine two compact CPTs into a single table in order to be able to make all distinctions that original two tables make. Therefore, the compact CPT for factor *B* is combined with each row of the compact CPT for factor *A* as shown in Table 8.

After combining two tables, we eliminate inconsistent rows. The canonical table representation of a new factor *C* of multiplying two factors is computed as shown in Table 9.

Table 9
Compact CPT for factor *C*.

D_1	D_2	D_3	Probability distribution
1	1	1	$C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	1	2	$C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
1	2	1,2	$C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
2	1	1	$C(\mathbf{X}; K^C_{a_4}, \mathbf{h}^C_{a_4}, g^C_{a_4}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1	2	$C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	2	1,2	$C(\mathbf{X}; K^C_{a_6}, \mathbf{h}^C_{a_6}, g^C_{a_6}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$



(a) Bayesian Network (b) Bethe Cluster Graph

Fig. 10. Example of Bethe cluster graph.

Next, let us consider summing out discrete variables D_1 in the compact CPT described in Table 9. We sum up probability distribution to make all distinction except for D_1 that original table makes. For instance, since the first and forth rows of the table are assigned same values of $D_2 = 1$ and $D_3 = 1$, the corresponding two canonical forms are summed up by carrying out weak marginalization that we discussed in the previous section. After summing out discrete variables, we integrate out continuous variables from Eq. (19).

By using compact CPTs, we can compute product and marginalization of factors. Compared to the full CPT based operations, the compact CPTs do not require large memory to carry out these operations. In contrast, the summing out operations need to create groups of rows in the CPTs so that all rows in the each group have the same values for all discrete variables except for summed out variables and thus the computational cost of grouping grows with $O(N \log N)$. In comparison to the tree structured CPT based operations, the computational cost of the compact CPT based operations does not increase exponentially. One possible drawback is that the compact CPT based operations discard tree structures once they are multiplied or marginalized and thus the decision tree structures are no longer available.

4.2. Belief propagation algorithm

In this section, we apply the *belief propagation* (BP) algorithm to handle a large hybrid Bayesian network containing large domain of discrete variables. The four major reasoning patterns in inference are *causal reasoning*, *diagnostic reasoning*, *intercausal reasoning* and *mixed reasoning*. The causal reasoning is to predict the downstream effects of factors from top nodes to bottom nodes while the diagnostic reasoning is to predict the upstream causes of factors from bottom nodes to top nodes. The intercausal reasoning is the inference where different causes of the same effect can interact. The mixed inference is the combination of two or more of the above reasoning.

4.2.1. Causal reasoning

The purpose of causal reasoning is to predict the conditional probability of downstream effects of factors given the evidence of their ancestor variables. Conditional probability can be calculated by a local message passing algorithm known as sum-product algorithm. Since our Bayesian network contains both discrete and continuous variables, a final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ of factor $\mathbf{C}_i = \mathbf{A}_i \cup \mathbf{X}_i$ is computed using

Table 10
Algorithm 3. Hybrid loopy belief propagation algorithm.

```
    Procedure Hybrid-LBP (
    $\Phi$ // Set of factors
    e // Evidence
    Cgraph // Bethe cluster graph
    )
1   Set $\mathcal{E}$ to be set of edges in Cgraph
2   Initialize all factors so that all of them are consistent with the evidence e
3   Initialize all messages as $\{K_{a_i}, \mathbf{h}_{a_i}, g_{a_i}\} = \{1, 0, 0\}$ for each instantiation $a_i$
4   while true
5     Select $(i, j) \in \mathcal{E}$
6     if scope$(\psi_i)$ = scope$(e)$
7       continue
8     Update message $\mu_{i\to j}$ from Eq. (26)
9     if all messages are converged
10      break
11  Compute final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ from (27)
12  return $\tilde{P}_\Phi(\mathbf{C}_i)$
```

Table 11
Prediction accuracies of BN, ANN and SVM for the simple system.

Known variables	BN	ANN	SVM
$[X_1, X_2, X_3, Y_1, Y_2]$ (fully observed)	0.970	0.958	0.958
$[X_1, X_2, X_3]$ (partially observed)	0.838	0.734	0.734

Table 12
System variables of steel production process.

Variable No.	Variable description	Domain size
1	Customer demand A	12
2	Customer demand B	2
3	Customer demand C	20
4	Customer demand D	20
5	Customer demand E	20
6	Customer demand F	10 (discritized)
7	Customer demand G	10 (discritized)
8	Customer demand H	10 (discritized)
9	Customer demand I	10 (discritized)
10	Operating condition A	5
11	Operating condition B	2
12	Operating condition C	2
13	Operating condition D	2
14	Operating condition E	2
15	Operating condition F	2
16	Operating condition G	2
17	Production load A	2
18	Production load B	2
19	Production load C	2
20	Production load D	2
21	Production load E	2
22	Production load F	2
23	Production load G	2
24	Production load H	2
25	Production load I	2
26	Process time	Continuous

sum and integral operators as follows:

$$\tilde{P}_\Phi(\mathbf{C}_i) = \int \sum_{\mathbf{A}_i - A_i} \psi_i \prod_{k\in \mathrm{pa}(i)} P_\Phi(\mathbf{C}_k) dx_1 dx_2 \ldots dx_M \tag{24}$$

where ψ_i is the initial potential of $\mathbf{C}_i$, $P_\Phi(\mathbf{C}_k)$ is the normalized final potential of $\mathbf{C}_k$, $x_m \in \{\mathbf{X}_i - X_i\}$ are integrated variables and X_i or A_i is the child node of the factor $\mathbf{C}_i$ (If the child node variable is continuous, $A_i = \emptyset$, otherwise $X_i = \emptyset$).

Causal inference is carried out as follows. First, we create a factor for each variable, so the number of factors is equal to the number of nodes in Bayesian networks. A factor corresponding to the top root node variable is a singleton factor that contains the corresponding top root node variable only. Since singleton factors are given evidence due to causal reasoning, cardinalities of their initial and final potentials are set to be 1. Note that a factor corresponding to the variable that is not placed on the top root node is a non-singleton factor that contains the corresponding variable and its all parent node variables. Therefore, the initial potentials for non-singleton factors are the conditional probability distributions of its corresponding variable given its parent node variables. After creating factors, the sum-product algorithm is carried out to compute the final potentials of all factors. We look for the non-singleton factor for which all parent nodes are assigned final potentials and compute a final potential for the non-singleton factor by using Eq. (24). Since the computed final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ is not normalized, the normalized final potential $P_\Phi(\mathbf{C}_i)$ is computed every time after computing final potentials. We continue computing the final potentials using the sum-product algorithm until the final potentials of all factors are computed. Because computation of the final potentials is carried out from top node factors to bottom node factors, the probability propagation directions are the same as arcs in Bayesian networks.

4.2.2. Diagnostic and intercausal reasoning

Unlike the causal reasoning, conditional probability for diagnostic, intercausal and mixed reasoning cannot be computed by monotonic algorithms. However, the sum-product algorithm can also be utilized for these inferences. Since the computational cost of exact inference algorithm such as VE exponentially increases in the width of the network, the loopy belief propagation (LBP) algorithm is employed in this work. LBP schemes use a cluster graph rather than a clique tree that is used for exact inference (Murphy, Weiss, & Jordan, 1999). Since the constraints defining the clique tree are indispensable for exact inference, the answer of message passing scheme is not correct in LBP.

LBP is carried out as follows: (i) create the singleton factors for all nodes and intermediate factors for all conditional probability distributions. Therefore, unlike in the causal reasoning, the number of factors is larger than the number of nodes in Bayesian networks. Given the cluster graph, we assign each factor ϕ_k to a cluster $\mathbf{C}_{\alpha(k)}$ so that scope$(\phi_k) \subseteq \mathbf{C}_{\alpha(k)}$. (ii) Then, the initial potentials ψ_i can be computed as follows:

$$\psi_i = \prod_{k:\alpha(k)=i} \phi_k \tag{25}$$

(iii) Initialize all messages as $\{K, \mathbf{h}, g\} = \{1, 0, 0\}$. A message from cluster $\mathbf{C}_i = \mathbf{A}_i \cup \mathbf{X}_i$ to another factor $\mathbf{C}_j = \mathbf{A}_j \cup \mathbf{X}_j$ is computed using sum and integral operators as follows:

$$\delta_{i\to j} = \int \sum_{\mathbf{A}_i - \mathbf{S}_{i,j}} \psi_i \prod_{k\in \mathrm{Nb}_i - \{j\}} \delta_{k\to i} dx_1 dx_2 \ldots dx_M \tag{26}$$

where $x_m \in \{\mathbf{X}_i - \mathbf{S}'_{i,j}\}$ are integrated variables, $\mathbf{S}_{i,j} = \mathbf{A}_i \cap \mathbf{A}_j$ is the subset of discrete variables, $\mathbf{S}'_{i,j} = \mathbf{X}_i \cap \mathbf{X}_j$ is the subset of continuous variables, and Nb_i is the set of indices of factors that are neighbors of $\mathbf{C}_i$. (iv) The messages are updated by using Eq. (26) in accordance with some message passing schedule until the canonical parameters of all messages are converged. Once the factor receives all messages, final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ can be computed by multiplying them with its initial potential as follows:

$$\tilde{P}_\Phi(\mathbf{C}_i) = \psi_i \prod_{k\in \mathrm{Nb}_i} \delta_{k\to i}. \tag{27}$$

In order to carry out LBP, we need to create a cluster graph that satisfies the family preservation property. For this purpose, the *Bethe cluster* graph which is a bipartite graph and holds the family preservation property can be used. The first layer of the Bethe cluster graph consists of large clusters $\mathbf{C}_k$ = scope(ϕ_k) while the second layer consists of a singleton cluster X_i for each random variable. Then edges are placed between a large cluster $\mathbf{C}_k$ and a singleton cluster X_i if $X_i \in \mathbf{C}_k$. Let us consider the Bethe cluster graph represented as shown in Fig. 10. This Bethe cluster graph has 6 singleton clusters $\{A\}$, $\{B\}$, $\{C\}$,

{D}, {E}, {F} and 3 large clusters {A, B, D}, {B, C, E}, {D, E}. Because of {B} $\in$ {A, B, D} and {B} $\in$ {B, C, E}, we place the edges between cluster 2 and 7 and between cluster 2 and 8.

4.2.3. Belief propagation with evidence

Factors need to be modified in accordance with the evidence. Let us consider initializing the factor ψ_i given the discrete evidence $e \in E \cap \Delta$. We delete all canonical parameters K, $\mathbf{h}$, g in the factor ψ_i that are not consistent with the evidence e.

Next we consider initializing the factor ψ_i given the continuous evidence $e \in E \cap \Gamma$. If the factor does not contain discrete variables, a canonical form of the factor ψ_i is reduced to a context representing the evidence e. In Eq. (17), we set $\mathbf{Y} = y$ with y being the value of the evidence e, then the new canonical form given continuous evidence is described as follows (Lauritzen, 1992):

$$K' = K_{\mathbf{XX}}$$
$$\mathbf{h}' = \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}}y$$
$$g' = g + \mathbf{h}_{\mathbf{Y}}^T y - \frac{1}{2} y^T \mathbf{K}_{\mathbf{YY}} y. \tag{28}$$

If the factor contains discrete variables and does not have a continuous variable except for scope(E), the canonical parameter g_{a_i} is updated for each instantiation a_i as follows:

$$g_{a_i} = -\frac{1}{2} y K_{a_i} y + \mathbf{h}_{a_i}^T y + g_{a_i}. \tag{29}$$

Since the new factor contains no continuous variable, the canonical parameters K_{a_i}, h_{a_i} become vacuous. If the factor contains both discrete and continuous variables except for scope(E), the parameters of the canonical form are computed for each instantiation a_i using Eq. (28). After modifying all factors so that all of them are consistent with the evidence, the reasoning algorithms mentioned above can be carried out.

We should note that, even if the compact CPT based operations are employed for factor product and marginalization, the computational task for multiplying incoming messages exponentially increases with the number of incoming messages. This is the case when diagnostic, intercausal or mixed inference is carried out since the LBP is an iterative algorithm which requires a large number of calculation of factor multiplication and marginalization. In order to reduce the computational cost during LBP, we make use of the property of Bethe cluster graph. In the Bethe cluster graph, the scope of messages is always one variable since there is no edge among any large clusters. If we give the evidence to some variable $X_i = e$, the message departed from the corresponding singleton cluster $\mathbf{C}_i$ never changes during iterations. Therefore, we can fix the messages $\delta_{i \to k}$ as e and thus we can skip the computation described by Eq. (26). Table 10 shows the loopy belief propagation algorithm given evidence.

5. Application example

5.1. Comparison with SVM and ANN

A simulated example is used to evaluate the validity and performance of the Bayesian networks under the assumption where single model can be used even though we have to deal with any unobservable variables. In this example, we use ANN and SVM for comparison with the presented method. The input and output data are generated from the simple network system where Y_1 is a parent of X_1 and X_2, Y_2 is a parent of X_3 and Z is a parent of Y_1 and Y_2. All variables are linearly

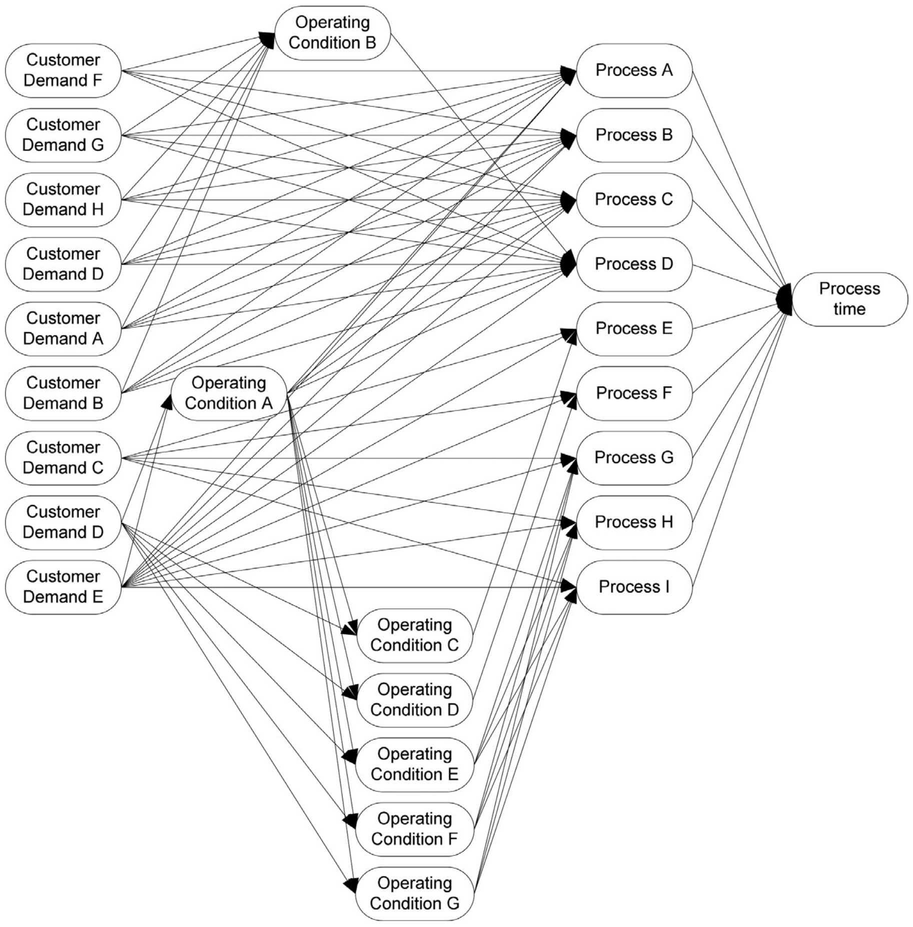


Fig. 11. Bayesian network for steel production process.

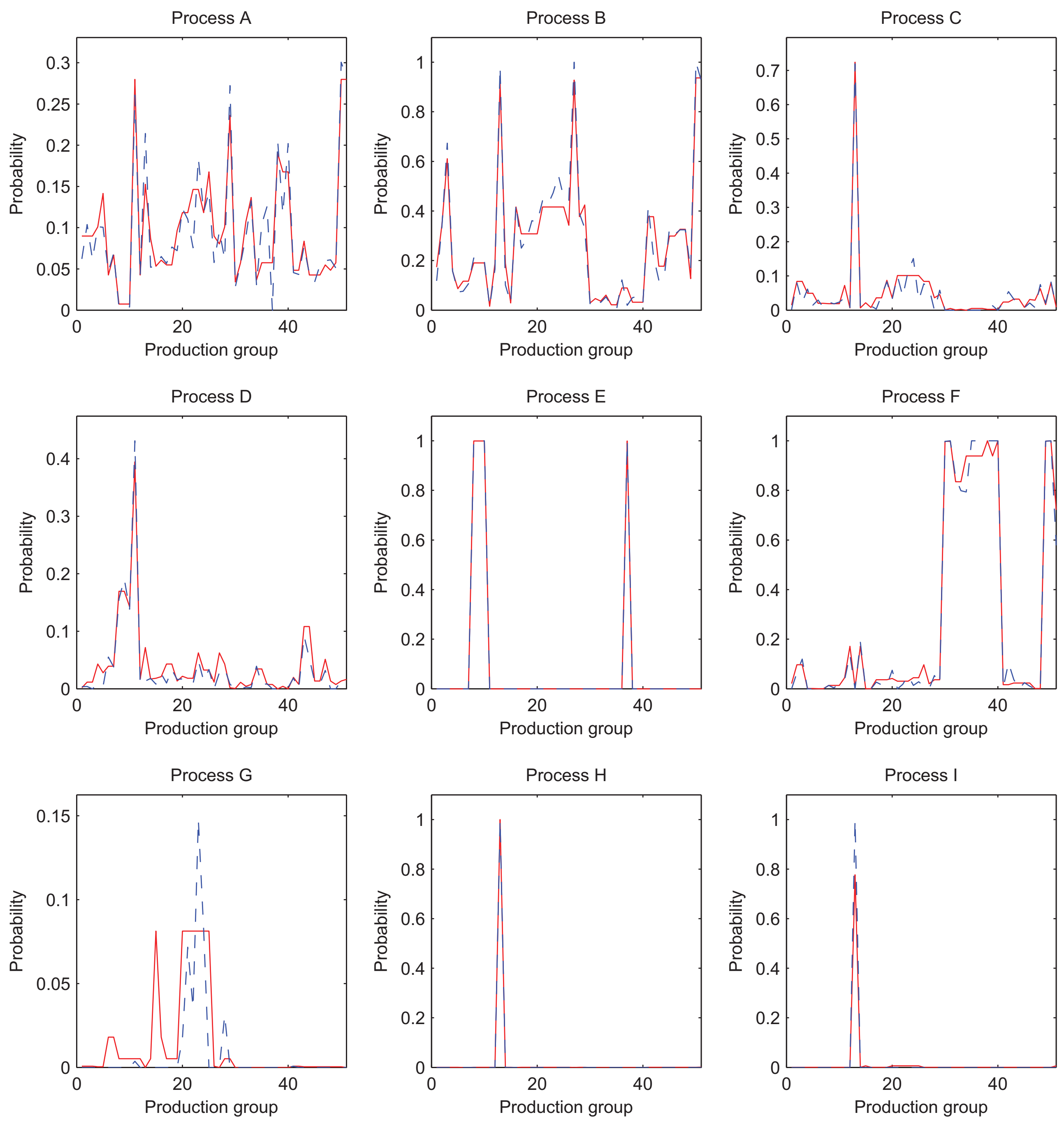


Fig. 12. Probability of the predicted production loads given customer demands and operating conditions in case 1.

dependent when they are connected with the arc. In this example [X_1, X_2, X_3, Y_1, Y_2] are input variables while Z is an output variable, all of which are binarized and thus the domain size of each variable is two.

The radial basis kernel function is used for SVM and its parameters are determined automatically by grid search algorithm. As for ANN, the number of hidden variables is set to be 4, which is determined from cross-validation.

Table 11 shows the computational results. When all variables without output variables can be observed, the accuracies for test data of BN, ANN and SVM are greater than 0.9. Therefore we can say that all methods can accurately predict the output variables. Meanwhile in the case when only input data are partially known, the accuracies of both ANN and SVM are 0.734 and 0.734, respectively. On the other hand, BN predicts the output variable with the high accuracy of 0.848 even when partial input data can be observed. These computational results demonstrate that BN is superior to SVM and ANN in terms of accuracy when input data are partially obtained.

5.2. Steel production process

In this study, the real-world steel production data are utilized to examine the performance of the proposed hybrid inference method. All case studies have been computed on a DELL Optiplex 990 (Intel(R) Core(TM) i-7-2600 CPU, 3.40 GHz and 8.0 GB RAM). Steel production is managed via two major tasks, production planning and production scheduling. At the production planning stage, the sales department receives orders from the customers and considers customer demand, plant profit, production capacity, inventory and shipment dates. For

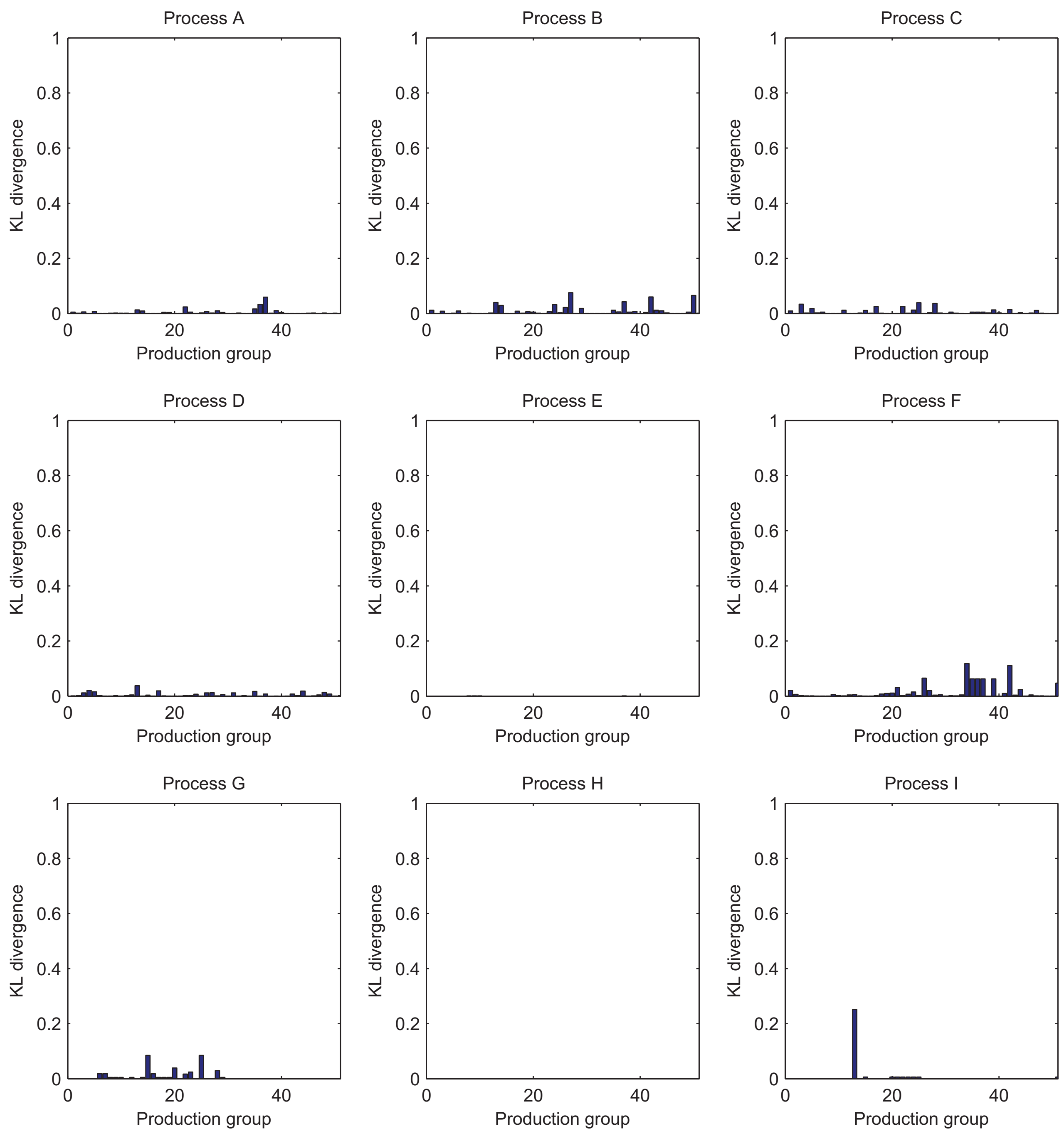


Fig. 13. KL divergence between the predicted and true probability distributions of production loads given customer demands and operating conditions in case 1.

this purpose, the production planning task requires a model that can predict production loads and processing times required to meet the customer demands. Once orders are finalized, the manufacturing department makes a production schedule that satisfies customer deadlines and production capacity constraints. Similar to the production planning task, this task also requires a model that can predict the production load and process time. At the production scheduling stage, additional information about operating conditions is also known.

Due to complexity of the manufacturing operations, potential for product defects which need to be rectified, and also the reality that exact production times for new grades of steel plates (which have not been produced before and need to be produced to meet a new customer order), the exact production times are not known. We apply the methodology described in the previous sections to construct the most likely structure of the Bayesian network representing the steel plates manufacturing process. This network is then used to estimate most likely production time for each grade of steel plates. Once the production times are known (estimated), one can proceed to plan and schedule the production.

In this application example, we create the Bayesian network that represents the relationship among system variables including customer demands, operating conditions, production loads and a process time. With the constructed Bayesian network, production loads and a process time are estimated so that they are consistent with the evidence such as customer demands or operating conditions.

To construct the Bayesian network, 21 discrete variables and 5 continuous variables are selected as shown in Table 12.

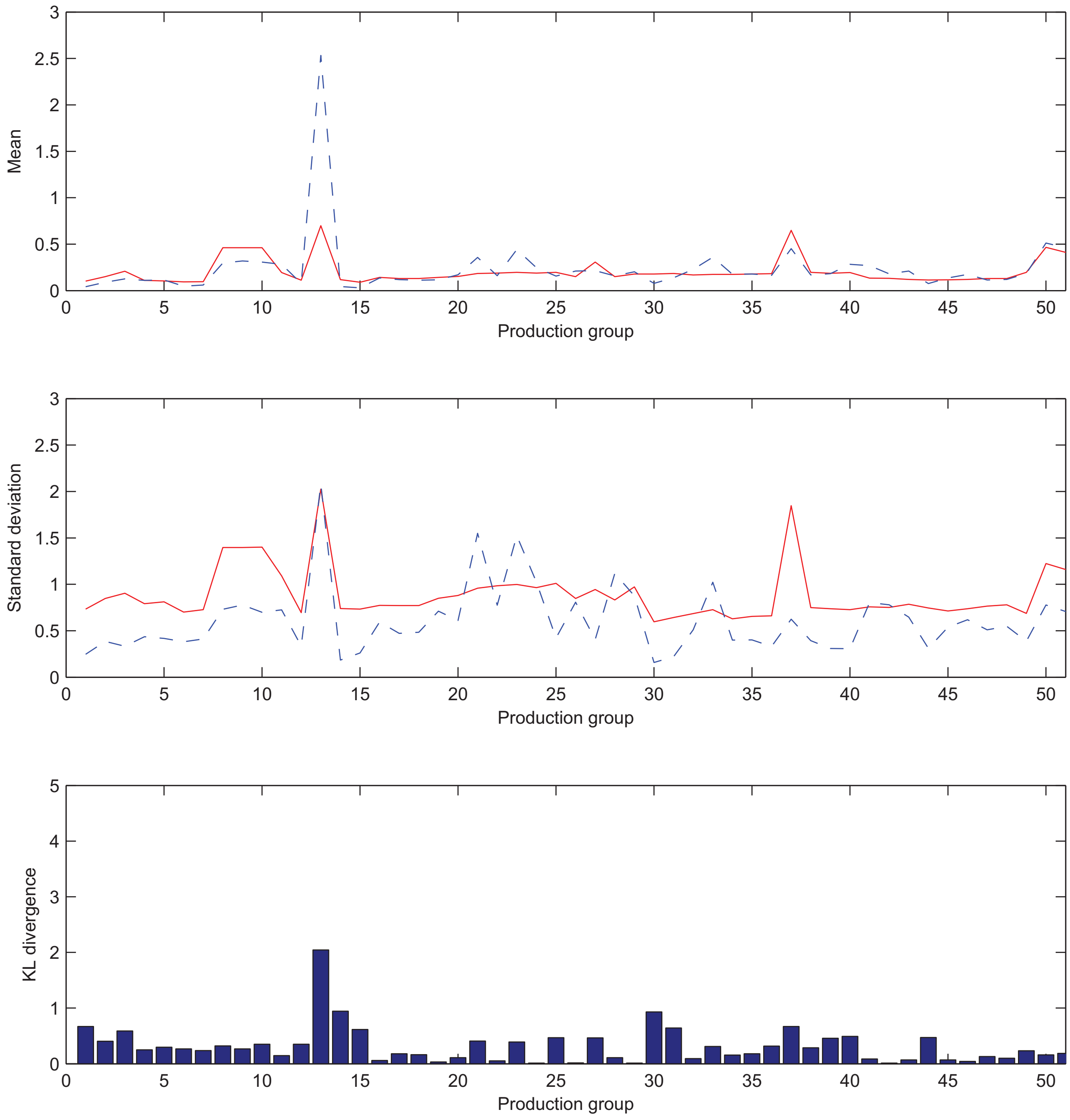


Fig. 14. Mean, standard deviation and KL divergence between the predicted and true probability distributions of process time given customer demands and operating conditions in case 1.

The variables related to the customer demands include continuous variables such as size of plates and strength and they are converted into discrete variables because each node of these variables has a discrete child node and the canonical tables cannot represent factors containing continuous nodes with a discrete child node. The variables related to the operating condition are all discrete variables such as heat and primer conditions. The variable corresponding to each production load is an integer variable; if the corresponding production process is not needed, it is 1, otherwise it is 2.

5.3. Inference results

First, we design the Bayesian network structure that represents relationship among system variables from historical process data shown in Fig. 11. The method to learn network structure from historical dataset is described in Appendix A.

With the designed network structure, the decision-tree structured CPTs are computed from historical process data. The traditional simple CPT method cannot be used in this application example for the following reason. Let us consider the non-singleton factor that contains variables of customer demand A, B, E, F, G, H, operating condition A, and process A. Since the product of cardinalities of all variables is $\times 12 \times 2 \times 20 \times 10 \times 10 \times 10 \times 10 \times 5 \times 2 = 48$ million, it would require a huge amount of training data to compute the appropriate parameters of all elements in the table. Among the proposed three types of multiplying and marginalizing operations, we employ the compact CPTs based operations. That is because the full CPT based operations would require a huge amount of memory and the

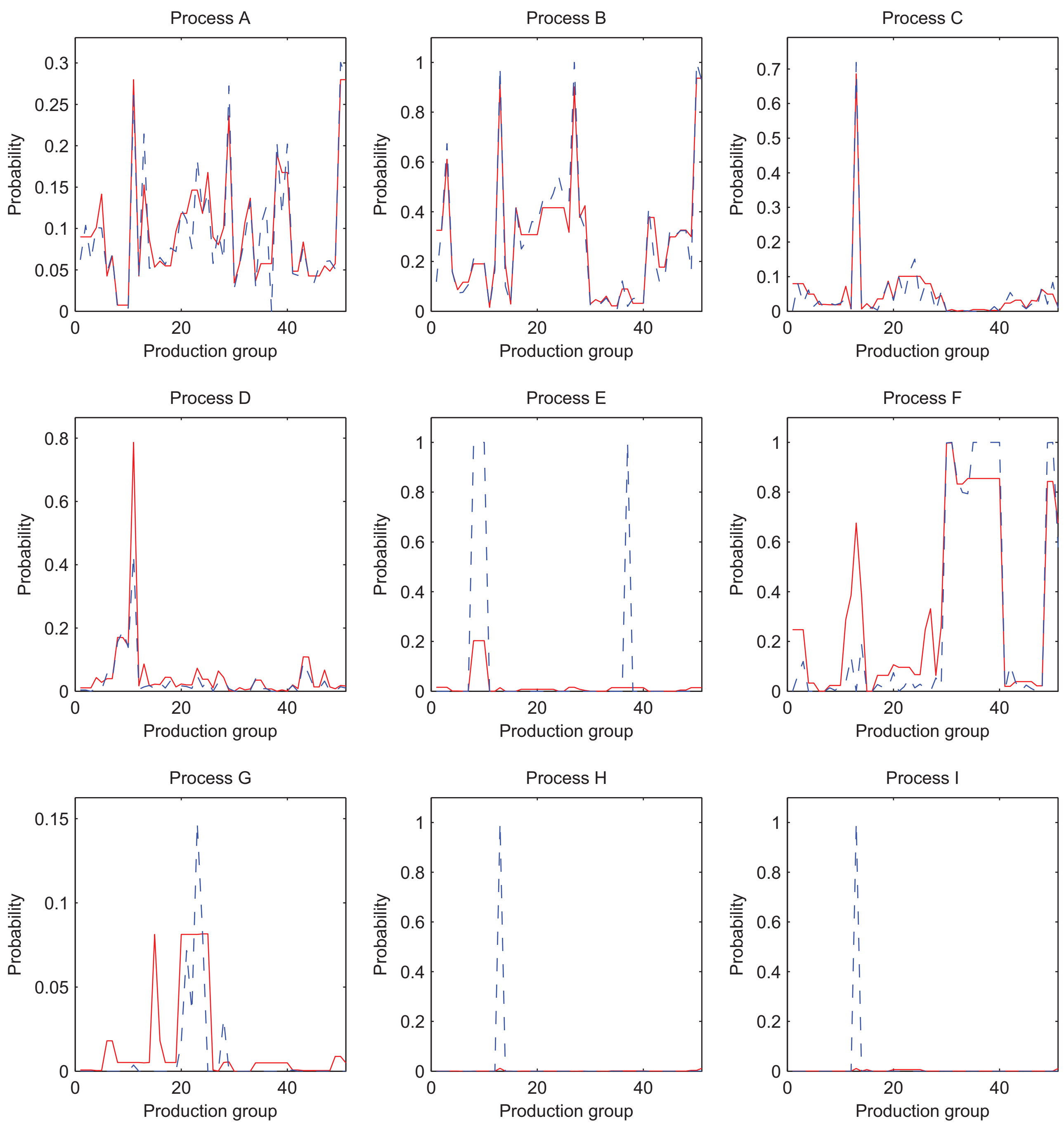


Fig. 15. Probability of the predicted production loads given customer demands in case 2. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

tree-structured CPT based operations needs a large execution time for marginalization due to a size of decision tree. It should be pointed out that since we use the compact CPTs, the decision-tree structure of final potentials cannot be obtained.

5.3.1. Case 1: variables of both customer demands and operating conditions are known and the variables of both production loads and process time are unknown

In this case, we need to estimate the production loads and process time given all evidence except for them. Such situation typically arises at the production scheduling stage. This types of inference is causal reasoning. The training data set consisting of 100, 000 steel plates is used to learn the decision-tree structured CPTs. The test data set of other 100, 000 plates is used to evaluate prediction accuracy. We compute the true probability distributions for comparison on each production group, which contains plates that have same customer demands and operating conditions. We select the production groups that have more than 200 plates for this purpose, while the remaining production groups are discarded because these production groups do not have a sufficient number of plates to compute the actual probability distributions. Figs. 12 and 13 show the prediction results of the production loads.

It can be seen that there is little difference between predicted and actual probability distributions in all production loads. Compared to case 2, the prediction accuracy of production loads for process D, process E, process F, process G, process H and process I is significantly

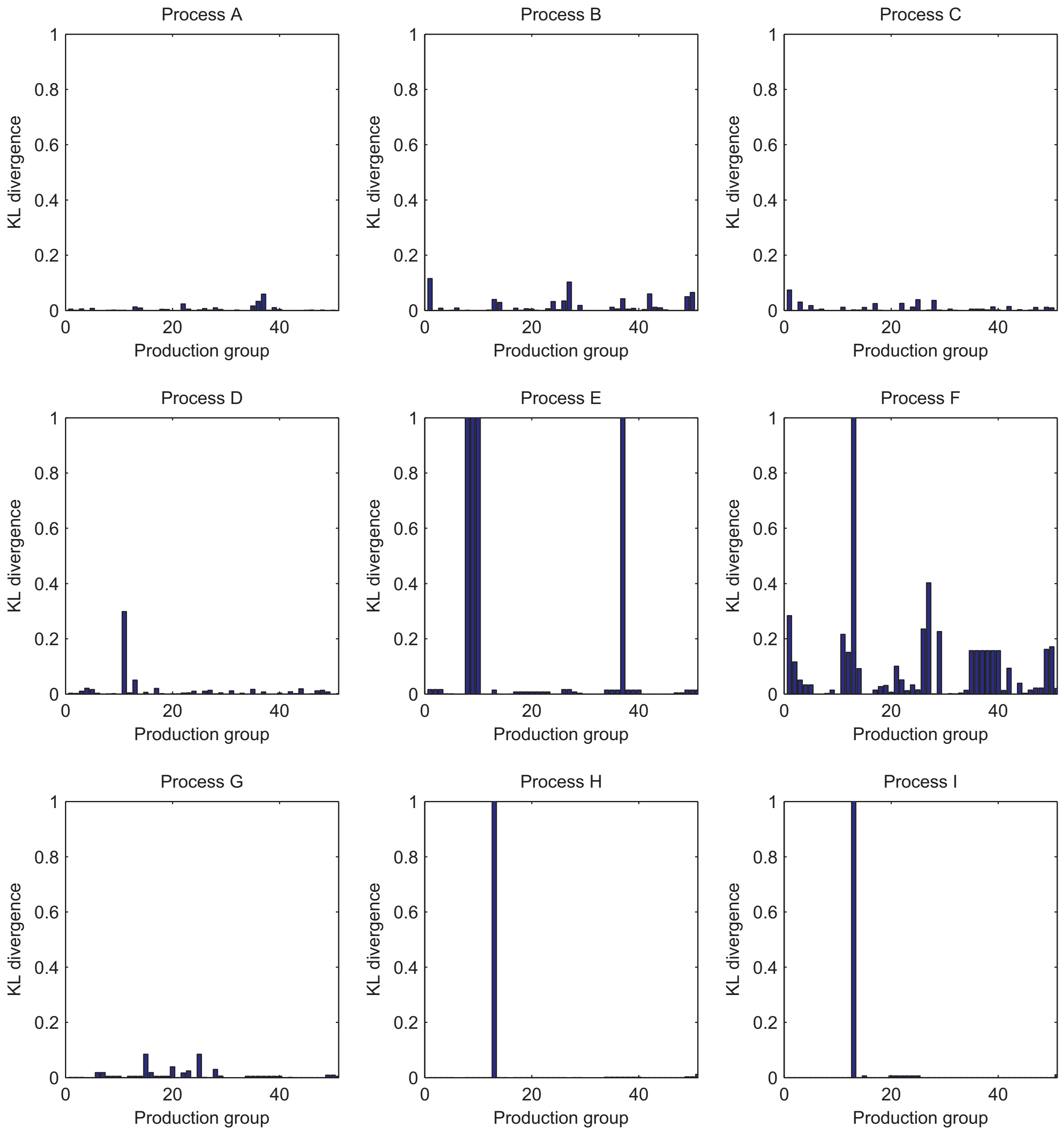


Fig. 16. KL divergence between the predicted and true probability distributions of production loads given customer demands in case 2. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

better. This is mainly due to the fact that we know the values of the operating conditions which have a significant influence on these production loads. Fig. 14 shows the means and variances of the inferred and actual process time and KL divergences between them.

One can see that KL divergences are smaller than the second test case. For further improvement, it may be desirable to add other variables such as the amount of in-process inventory, the number of workers and the failure rate of each machine, which will have an impact on the process time. However, these kinds of discussion are outside the scope of our research. The average execution time for the inference in all production group is 9.60[s], which is almost same as the second test scenario. This example demonstrates that, even though our Bayesian network includes a large domain of discrete variables, the causal reasoning can be carried out by using the proposed decision-tree structured conditional probability table representations.

5.3.2. *Case 2: variables representing the customer demands are known and the other variables are unknown*

This situation arises typically when persons in the sales department receive orders from customers and want to know the production loads and the process times for these orders. Since the purpose of this inference is to predict the effects of production loads and the process time given the causes of the customer demands, this type of inference is causal reasoning. Compared to case 1, there are more uncertainties here, since the production loads are not known and one can

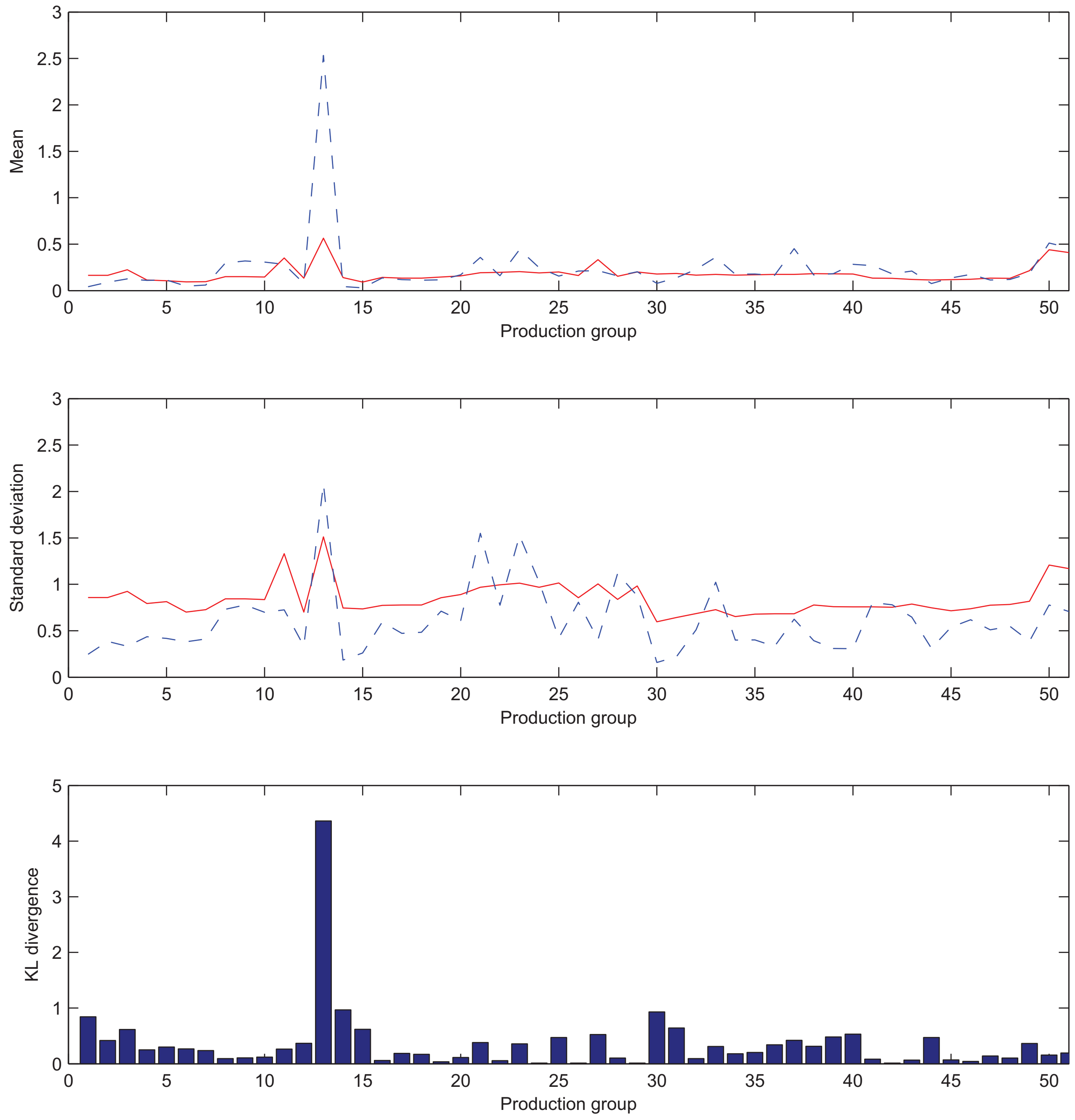


Fig. 17. Mean, standard deviation and KL divergence between the predicted and true probability distributions of process time given customer demands in case 2.

expect reduced certainty of predictions. We use the same decision-tree structured CPTs that was learned in case 1. The probabilities of the values of production loads being 2 and their KL divergences between the predicted and actual probability distributions are shown in Figs. 15 and 16.

The red solid lines represent the predicted values while the blue dash lines denote the actual values. One can readily observe that the probability distributions of process A, process B, process C and process D are very close to actual ones. However, the results of the other production loads such as process E, process F, process G, process H and process I show the bad prediction performance. This is most likely due to the fact that these production loads are significantly affected by the operating conditions, which are assumed to be unknown in this test case. In addition, the prediction results for the process time are shown in Fig. 17.

It can be seen that the inferred probability distributions are largely different from the true ones. That is because the process time is computed from the final potentials of production loads and the probability distributions of some production loads are not accurately predicted. The average execution time for the inference in all production group is 9.96[s].

Results of test cases 1 and 2 indicate that we can carry out the most plausible reasoning from the values of known variables using a single Bayesian network model. This is very useful particularly in instances when available variables are different at different steps. Having a single Bayesian network model avoids the need to have

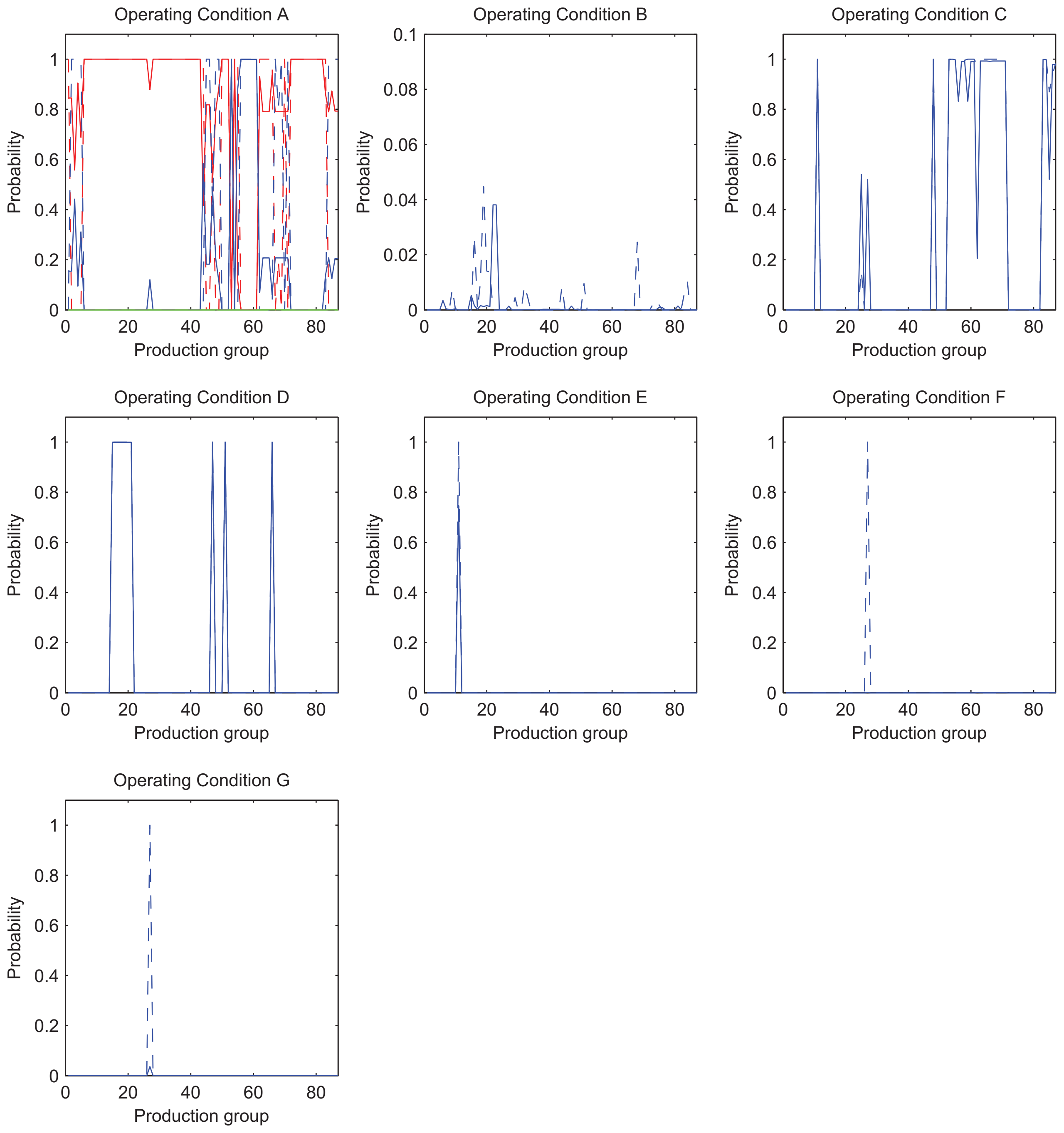


Fig. 18. Probability of operating conditions in case 3. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

multiple specific models tailored for specific purposes, which causes model maintenance issues and lack of model consistency.

5.3.3. Case 3: customer demands and all production loads are known while the operating conditions are unknown

This kind of is required when we want to decide which operating conditions to use in production so that both the customer demands and production capacity can be satisfied. Since this type of the inference is mixed reasoning, the proposed hybrid LBP algorithm is employed. Similarly to the previous two cases, we evaluate the prediction accuracy for each production that contains steel plates that have the same customer demands and production loads. We have selected the production groups containing more than 100 plates to compute the true probability distributions. The computed probability distributions of production loads are shown in Fig. 18. Different colors of the lines are used to represent different values of each variable while the solid lines denote the predicted probabilities and the dashed lines represent the actual probabilities. In addition, the KL divergences between the predicted and actual probability distributions are shown in Fig. 19. It can be seen that most KL divergences are very small and the proposed inference algorithm can accurately predict the probability distributions of unknown variables. Therefore, we can say that the proposed inference method can carry out the mixed reasoning even though this Bayesian network has a large domain of discrete variables. The average execution time for the inference in all production groups is 1.91[s].

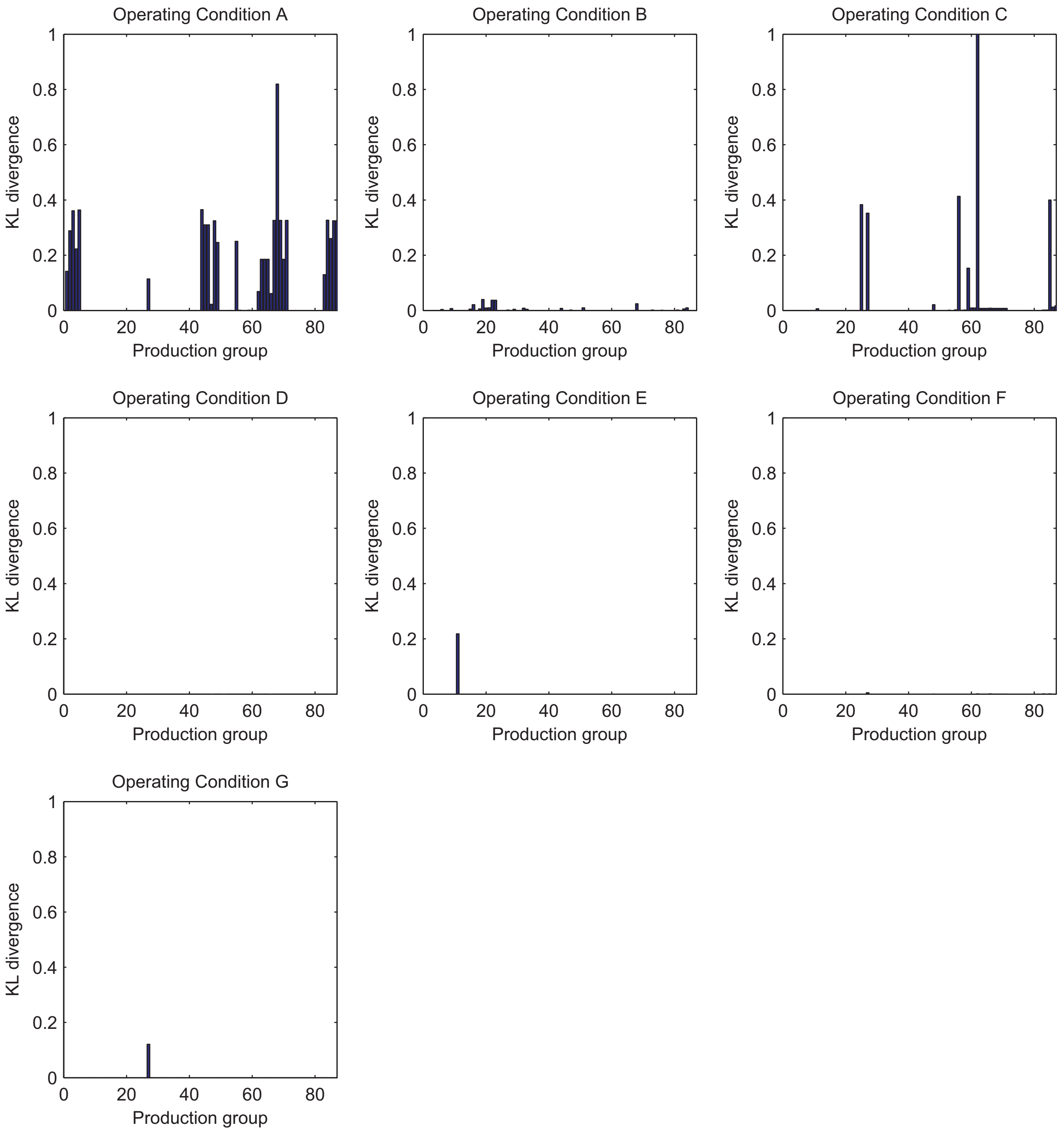


Fig. 19. KL divergence of operating conditions in case 3.

6. Conclusion

Complex multi-stage manufacturing processes, such as steel plates manufacturing, have many manufacturing steps which are not fully deterministic. Production planning and scheduling for such systems requires that probabilities of specific outcomes at each production step be accounted for. Business decisions, e.g. committing to deliver a specific order, scheduling production, or deciding on operating conditions, require different types of reasoning to conclude the most likely outcomes. Instead of using different models for different types of reasoning, we have used the same inference technique for all decision making. The method employs decision-tree structured conditional probability tables (CPTs) based hybrid inference technique for the hybrid Bayesian network containing large domain discrete variables. The main contribution of this research is a new methodology to construct the context-specific CPTs represented by decision trees through classification tree algorithm. We have introduced three new types of operators for computing multiplication and marginalization of factors represented by decision-tree structure based CPTs. These novel operations enable us to carry out inference by using belief propagation algorithms. For causal reasoning, the direction of probability propagation is downstream from the top node variables to the bottom nodes variables. Since other types of reasoning cannot be carried out using such monotonic logic based algorithm, we use more complicated algorithm, i.e. the loopy belief propagation. In order to reduce the computational cost, we fix the values of messages that depart from the nodes given evidence and omit the computation of these messages during iterations.

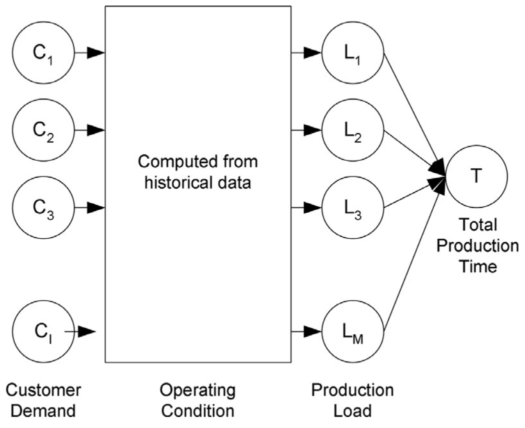


Fig. A.20. Bayesian network.

In conclusion, our method is able to handle the large domain discrete variables without increasing computational cost exponentially. In addition, in our approach, we can discretize the continuous parents as finely as needed, which does not increase the number of parameters in intermediate factors due to decision-tree structured CPTs. We assume that the continuous variables are Gaussian, which may limit the applicability of the proposed method. If the domain of the discrete child becomes extremely large (more than 100 domain cardinalities), the factors become large which may make the tree-CPT sparse because our approach cannot split the discrete child nodes. While this is a limitation, the proposed method increases significantly the size of the domain of discrete variables which can be used as a part of the system model.

Real life steel production process data have been used to examine the effectiveness of the proposed inference algorithm. The results demonstrate that the proposed algorithm accurately and very rapidly predicts the probability distributions of unobserved variables in large hybrid Bayesian networks.

Future work will focus on applying the presented Bayesian network model to the scheduling and planning for steelmaking plants such that both the profit and customer satisfaction are maximized.

Appendix A. Structure learning

The goal of structure learning is to find a network structure $\mathcal{G}$ that is a good estimator for the data **x**. The most common approach is *score-based structure learning*, which defines the structure learning problem as an optimization problem (Koller & Friedman, 2009). A score function score$(\mathcal{G} : \mathcal{D})$ that measures how well the network model fits the observed data $\mathcal{D}$ is defined. The computational task is to solve the combinational optimization problem of finding the network that has the highest score. In this paper, we employ BIC scores described as follows:

$$\text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}) = M\sum_{i=1}^{N}\mathbf{I}(x_i; \text{pa}_{\mathcal{G}}(x_i)) - M\sum_{i=1}^{N}H(x_i) - \frac{\text{Log}M}{2}\text{Dim}[\mathcal{G}] \qquad (A.1)$$

where $\text{pa}_{\mathcal{G}}(x_i)$ are the parent nodes of x_i given graph $\mathcal{G}$, $\mathbf{I}(x_i; \text{pa}_{\mathcal{G}}(x_i))$ is the mutual information between x_i and $\text{pa}_{\mathcal{G}}(x_i)$, and $H(x_i)$ is the marginal entropy of x_i. With this score function, the optimal graph $\mathcal{G}^*$ can be computed as follows:

$$\mathcal{G}^* = \text{argmax}_{\mathcal{G}}\text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}). \qquad (A.2)$$

Since this combinational optimization problem is known to be *NP*-hard (Chickering et al., 1997), it is difficult to compute the optimal graph for industrial plants which often include a large number of process variables.

In order to reduce the computational cost, we incorporate the fundamental process knowledge into the network structure learning framework. One can easily noticed that the nodes of variables belonging to the customer demands should be placed on the root nodes in the Bayesian network while production loads are affected by the customer demands, operating conditions or both and not vice versa. Taking into account above considerations, we propose the type of Bayesian networks as shown in Fig. A.20.

References

Bishop, C. (2006). *Pattern recognition and machine learning*. New York, NY: Springer-Verlag.
Boutilier, C., Friedman, N., Goldszmidt, M., & Koller, D. (1996). Context-specific independence in Bayesian networks. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-96* (pp. 115–123).
Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Belmont: Wadsworth International Group.
Buyukozkan, G., Kayakutlu, G., & Karakadlar, I. (2015). Assessment of lean manufacturing effect on business performance using Bayesian belief networks. *Expert Systems with Applications, 42*, 6539–6551.
Cai, Z., Sun, S., Si, S., & Yannou, B. (2011). Identifying product failure rate based on a conditional Bayesian network classifier. *Expert Systems with Applications, 38*, 5036–5043.
Cano, A., Gómez-Olmedo, M., Moral, S., & Pérez-Ariza, C. B. (2014). Extended probability trees for probabilistic graphical models. *Probabilistic graphical models* (pp. 113–128). Springer International Publishing.
Chavira, M., & Darwiche, A. (2007). Compiling Bayesian networks using variable elimination. In *Proceedings of the 20th international joint conference on artifical intelligence, IJCAI'07* (pp. 2443–2449).
Chickering, D., Heckerman, D., & Meek, C. (1997). A Bayesian approach to learning Bayesian networks with local structure. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-97* (pp. 80–89).
Cobb, B., Rumi, R., & Salmeron, A. (2007). *Bayesian network models with discrete and continuous variables*. Berlin: Springer-Verlag.
DesJardins, M., & Rathod, P. (2008). Learning structured Bayesian networks: combining abstraction hierarchies and tree-structured conditional probability tables. *Comput. Intell., 24*, 1–22.
Friedman, N., & Goldszmidt, M. (1996). Learning Bayesian networks with local structure. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-96* (pp. 252–262).
Gogate, V., & Domingos, P. (2013). Structured message passing. In *Proceedings of the twenty-ninth conference on uncertainty in artificial intelligence, UAI-2013* (pp. 252–261).
Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT Press.
Koller, D., Lerner, U., & Angelov, D. (1999). A general algorithm for approximate inference and its application to hybrid Bayes nets. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-99* (pp. 324–333).
Kozlov, A., & Koller, D. (1997). Nonuniform dynamic discretization in hybrid networks. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-97* (pp. 314–325).
Lauritzen, S. (1992). Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association, 87*, 1098–1108.
Lauritzen, S., & Jensen, F. (2001). Stable local computation with conditional gaussian distribution. *Statistics and Computing, 11*, 191–203.
Lauritzen, S., & Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statistics, 17*, 31–57.
Melo, A. C. V., & Sanchez, A. J. (2008). Software maintenance project delays prediction using Bayesian networks. *Expert Systems with Applications, 34*, 908–919.
Moral, S., Rumi, R., & Salmeron, A. (2001). *Mixtures of Truncated exponentials in hybrid Bayesian networks*. Berlin: SpringerVerlag.
Murphy, K., Weiss, Y., & Jordan, M. (1999). Nonuniform dynamic discretization in hybrid networks. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-99* (pp. 467–475).
Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. San Francisco: Morgan-Kaufmann.
Perkusich, M., & Soares, G. (2014). A procedure to detect problems of processes in software development projects using Bayesian networks. *Expert Systems with Applications, 42*, 437–450.
Poole, D., & Zhang, N. (2003). Exploiting contextual independence in probabilistic inference. *Journal of Artificial Intelligence Research, 18*, 263–313.
Rumi, R., & Salmeron, A. (2007). Approximate probability propagation with mixtures of truncated exponentials. *International Journal of Approximate Reasoning, 45*, 191–210.
Sharma, R., & Poole, D. (2003). Efficient inference in large discrete domains. In *Proceedings of conference on uncertainty in artificial intelligence, UAI-2003* (pp. 535–542).
Yuan, C., & Druzdzel, M. J. (2006). Hybrid loopy belief propagation. In *Proceedings of the third european workshop on probabilistic graphical models (PGM'06)* (pp. 317–324).

Chapter 6

Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables

Computers and Chemical Engineering 79 (2015) 113–134

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables

Junichi Mori, Vladimir Mahalec*

Department of Chemical Engineering, McMaster University, Hamilton, Ontario, Canada L8S 4L7

ARTICLE INFO

Article history:
Received 4 June 2014
Received in revised form 4 February 2015
Accepted 8 February 2015
Available online 5 March 2015

Keywords:
Estimation of production time
Hybrid Bayesian network
Structure learning
Large domain of discrete variables
Decision tree
Steel plate production

ABSTRACT

Knowledge of the production loads and production times is an essential ingredient for making successful production plans and schedules. In steel production, the production loads and the production times are impacted by many uncertainties, which necessitates their prediction via stochastic models. In order to avoid having separate prediction models for planning and for scheduling, it is helpful to develop a single prediction model that allows us to predict both production loads and production times. In this work, Bayesian network models are employed to predict the probability distributions of these variables. First, network structure is identified by maximizing the Bayesian scores that include the likelihood and model complexity. In order to handle large domain of discrete variables, a novel decision-tree structured conditional probability table based Bayesian inference algorithm is developed. We present results for real-world steel production data and show that the proposed models can accurately predict the probability distributions.

1. Introduction

Accurate estimation of the production loads and the total production times in manufacturing processes is crucial for optimal operations of real world industrial systems. In this paper, a production load represents the number of times that a product is processed in the corresponding process unit and a production time is defined as the length of time from production start to completion. Production planning and scheduling for short, medium and long term time horizons employ various optimization models and algorithms that require accurate knowledge of production loads and total production times from information at each of the processing steps. The approaches to predict the production loads and total production times can be either based on mechanistic model or can employ data-driven techniques. Model-based prediction methods may be applied only if accurate mechanistic models of the processes can be developed. First principal models require in-depth knowledge of the processes and still cannot take into consideration all uncertainties that exist in the processes. Therefore, mechanistic models may not work well for prediction of production loads and production times of the real-world industrial processes. On the other hand, data-driven approaches do not require in-depth process knowledge and some advanced techniques can deal with the process uncertainties.

The most straightforward approach is to use the classical statistical models (e.g. regression models) that estimate the values of new production loads and a production time from the past values in the historical process data. The relationships between the targeted variables and other relevant variables are used to compute the statistical model that can predict the production loads and production times. However, such models are too simple to predict the nonlinear behavior and estimate the system uncertainties. An alternative simple method is to compute the average of these values per each production group that has similar production properties and then utilize the average values of each production group as the prediction (Ashayeria et al., 2006). In this case the prediction accuracy significantly depends on the rules that govern creation of production groups and it may be challenging to find the appropriate rules from process knowledge only. To overcome this limitation, supervised classification techniques such as artificial neural networks (ANN), support vector machine, Fisher discriminant analysis and K-nearest neighbors (KNN) may be useful to design the rules to make production groups from historical process data. However, even though we can accurately classify the historical process data into an appropriate number of production groups, these methods do not consider model uncertainties and cannot handle missing

* Corresponding author. Tel.: +1 905 525 9140.
E-mail address: mahalec@mcmaster.ca (V. Mahalec).

http://dx.doi.org/10.1016/j.compchemeng.2015.02.005
0098-1354/© 2015 Elsevier Ltd. All rights reserved.

values and unobserved variables. In addition, we are typically forced to have multiple specific models tailored for specific purposes, e.g. for production planning or for scheduling, which causes model maintenance issues and lack of model consistency.

Bayesian network (BN) models offer advantages of having a single prediction model for predicting the production loads and total process times in planning and scheduling. Bayesian networks are also called *directed graphical models* where the links of the graphs represent direct dependence among the variables and are described by arrows between links (Pearl, 1988; Bishop, 2006). Bayesian network models are popular for representing conditional independencies among random variables under system uncertainty. They are popular in the machine learning communities and have been applied to various fields including medical diagnostics, speech recognition, gene modeling, cancer classification, target tracking, sensor validation, and reliability analysis.

The most common representations of conditional probability distributions (CPDs) at each node in BNs are conditional probability tables (CPTs), which specify marginal probability distributions for each combination of values of its discrete parent nodes. Since the real industrial plant data often include discrete variables which have large discrete domains, the number of parameters becomes too large to represent the relationships by the CPTs. In order to reduce the number of parameters, context-specific independence representations are useful to describe the CPTs (Boutilier et al., 1996). Efficient inference algorithm that exploits context-specified independence (Poole and Zhang, 2003) and the learning methods for identification of parameters of context-specific independence (Friedman and Goldszmidt, 1996; Chickering et al., 1997) have been developed. The restriction of these methods is that all discrete values must be already grouped at an appropriate level of domain size since learning structured CPTs is NP-hard. However, discrete process variables typically have large domains and the task of identifying a reasonable set of groups that distinguish well the values of discrete variables requires in-depth process knowledge. To overcome this limitation, *attribute – value hierarchies* (AVHs) that capture meaningful groupings of values in a particular domain are integrated with the tree-structured CPTs (DesJardins and Rathod, 2008). Such approach is not applicable in general process systems, since some discrete process variables do not contain hierarchal structures and thus AVHs cannot capture the useful abstracts of values in that domain. In addition, this model cannot handle the continuous variables without discretizing them. Furthermore, the authors do not describe how to apply AVH-derived CPTs to Bayesian inference. Therefore, this method has difficulty predicting probability distributions of production loads and total process time from observed process variables in the real-world industrial processes. Efficient alternative inference methods in Bayesian Networks containing CPTs that are represented as decision trees have been developed (Sharma and Poole, 2003). The inference algorithm is based on variable elimination (VE) algorithm. However, because the computational complexity of the exact inference such as VE grows exponentially with the size of the network, this method may not be appropriate for Bayesian networks for large scale industrial data sets. In addition, the method does not deal with application of the decision-tree structured CPTs to hybrid Bayesian networks, where both discrete and continuous variables appear simultaneously.

In hybrid Bayesian networks, the most commonly used model that allows exact inference is the conditional linear Gaussian (CLG) model (Lauritzen, 1992; Lauritzen and Jensen, 2001). However, the proposed network model does not allow discrete variables to have continuous parents. To overcome this limitation, the CPDs of these nodes are typically modeled as softmax function, but there is no exact inference algorithm. Although an approximate inference via Monte Carlo method has been proposed (Koller et al., 1999), the convergence can be quite slow in Bayesian Networks with large domain of discrete variables. Another approach is to discretize all continuous variables in a network and treat them as if they are discrete (Kozlov and Koller, 1997). Nevertheless, it is typically impossible to discretize the continuous variables as finely as needed to obtain reasonable solutions and the discretization leads to a trade-off between accuracy of the approximation and cost of computation. As another alternative, the mixture of truncated exponential (MTE) model has been introduced to handle the hybrid Bayesian networks (Moral et al., 2001; Rumi and Salmeron, 2007; Cobb et al., 2007). MTE models approximate arbitrary probability distribution functions (PDFs) using exponential terms and allow implementation of inference in hybrid Bayesian networks. The main advantage of this method is that standard propagation algorithms can be used. However, since the number of regression coefficients in exponential functions linearly grows with the domain size of discrete variables, MTE model may not work well for the large Bayesian networks that are required to represent the industrial processes.

Production planning and scheduling in the steel industry are recognized as challenging problems. In particular, the steel plate production is one of the most complicated processes because steel plates are high-variety low-volume products manufactured on order and they are used in many different applications. Although there have been several studies on scheduling and planning problems in steel production, such as continuous casting (Tang et al., 2000; Santos et al., 2003), smelting process (Harjunkoski and Grossmann, 2001) and batch annealing (Moon and Hrymak, 1999), few studies have dealt with steel plate production scheduling. Steel rolling processes manufacture various size of plates from a wide range of materials. Then, at the finishing and inspection processes, malfunctions occurred in the upstream processes (e.g. smelting processes) are repaired, and additional treatments such as heat treatment and primer coating are applied such that the plates satisfy the intended application needs and satisfy the demanded properties. In order to obtain successful plans and schedules for steel plate production, it is necessary to determine the production starting times that meet both the customer shipping deadlines and the production capacity. This requires prediction models that can accurately predict the production loads of finishing and inspection lines and the total process time. However, due to the complexity and uncertainties that exist in the steel production processes, it is difficult to build the precise prediction models. These difficulties have been discussed in the literature (Nishioka et al., 2012).

In our work, in order to handle the complicated interaction among process variables and uncertainties, Bayesian networks are employed for predicting of the production loads and prediction of the total production time. Since the steel production data have large domain of discrete variables, their CPDs are described by tree structured CPTs. In order to compute the tree structured CPTs, we use decision trees algorithm (Breiman et al., 1984) that is able to group the discrete values to capture important distinctions of continuous or discrete variables. If Bayesian networks include continuous parent nodes with a discrete child node, the corresponding continuous variables can be discretized as finely as needed, because the domain size of discretized variable does not increase the number of parameters in intermediate factors due to decision-tree structured CPTs. Since the classification algorithms are typically greedy ones, the computational task for learning the structured CPTs is not expensive. Then, the intermediate factors can be described compactly using a simple parametric representation called the canonical table representation.

As for Bayesian network structure, if the cause–effect relationship is clearly identified from process knowledge, knowledge based network identification approach is well-suited. Such identification of cause–effect relationships may require in-depth knowledge of the processes to characterize the complex physical, chemical and biological phenomena. In addition, it can be time-consuming to build precise graphical model for complex processes, and it is also challenging to verify the identified network structure. Therefore, data-driven techniques are useful to systematically identify the network structure from historical process data. The basic idea of learning the network structure is to search for the network graph so that the likelihood based score function is maximized. Since this is a combinational optimization problem which is known as *NP*-hard, it is challenging to compute the precise network structure for industrial plants where there are a large number of process variables and historical data sets. In order to restrict the solution space, we fix the root and leaf nodes by considering the properties of the process variables. In addition, we take into account two types of Bayesian network structures, which are: (i) a graph where a node of the total production time is connected with all nodes of the production loads and (ii) a graph where a node of the total production time is connected with all nodes of the individual production time on each production load and the nodes of these individual process time are connected to the nodes of the total production time. In the latter Bayesian network structure, the training data set that contains each production period of all production processes is needed in order to directly compute the parameters of conditional probability distributions of each production period.

We also propose the decision-tree structured CPT based Bayesian inference algorithm, which employs belief propagation algorithm to handle hybrid Bayesian networks with large domain of discrete variables. In order to reduce the computational effort required for multiplying and marginalizing factors during belief propagation, the operations via dynamically construct structured CPTs are proposed. Since the inference task is to predict the cause of production loads and the total production time from the effect of other process variables, we need the causal reasoning which can be carried out by using the sum-product algorithm from top node factors to bottom node factors. In addition, other types of inference such as diagnostic and intercausal reasoning are also investigated in the case when we need to infer some other process variables such as operating conditions given production loads.

The organization of this paper is as follows. Section 2 introduces the plate steel production problem. Section 3 proposes the structure identification algorithm for steel plate production processes. Section 4 describes the proposed decision-tree structured CPTs based Bayesian inference algorithm and Section 5 proposes the inference algorithm. The presented method is applied to the steel plate production process data in Section 6. Finally, the conclusions are presented in Section 7. Appendix A describes algorithm to find the network structure that maximizes the score function, Appendix B describes computation of causal reasoning, Appendix C deals with diagnostic and causal reasoning, and Appendix D describes belief propagation with evidence.

2. Problem definition

The plate steel mills studied in this paper consist of rolling machines, finishing and inspection processes. The purpose of the plate mills is to manufacture the steel plates, which are end products made from steel slabs via continuous casting processes. The slabs are sent to the rolling machines which include roughing mill, finishing mill and hot leveling machine through continuous or batch reheating furnace. To manufacture the high-strength steel plates, Accelerated Cooling (ACC) process which is based on the combination of controlled rolling and controlled cooling is used. After rolling or cooling, the plates are sent to the finishing and inspection processes, where defects that have occurred in the upstream processes such as smelting processes are repaired. Here, additional treatments such as heat treatment and primer coating are carried out as needed so that the plates have the properties required for their intended usages. The process is illustrated in Fig. 1.

One of the most important goals for production planning and scheduling is to use in the best possible manner the subsystems which are the bottlenecks, since the total plant capacity is determined by the bottleneck capacity. In the plate mills, bottlenecks typically occur in the finishing and inspection lines. Therefore, an effective way to approach the planning and scheduling in the plate mills is to determine the rolling starting time of each order in such a manner that the limiting capacities are used to their maximum and that the customer demanded shipment deadlines are satisfied. For this purpose, when sales department staff receives orders from the customers, they must take into account the plate mills capacities and production times to avoid creating supply bottlenecks and to meet the customer deadlines. Similarly, the staff in the manufacturing department prepares the production plans and schedules while considering the production capacities and production times. However, due to the complicated and stochastic processes of the finishing and inspection lines, there is no deterministic way to predict the production loads and production time that are required to

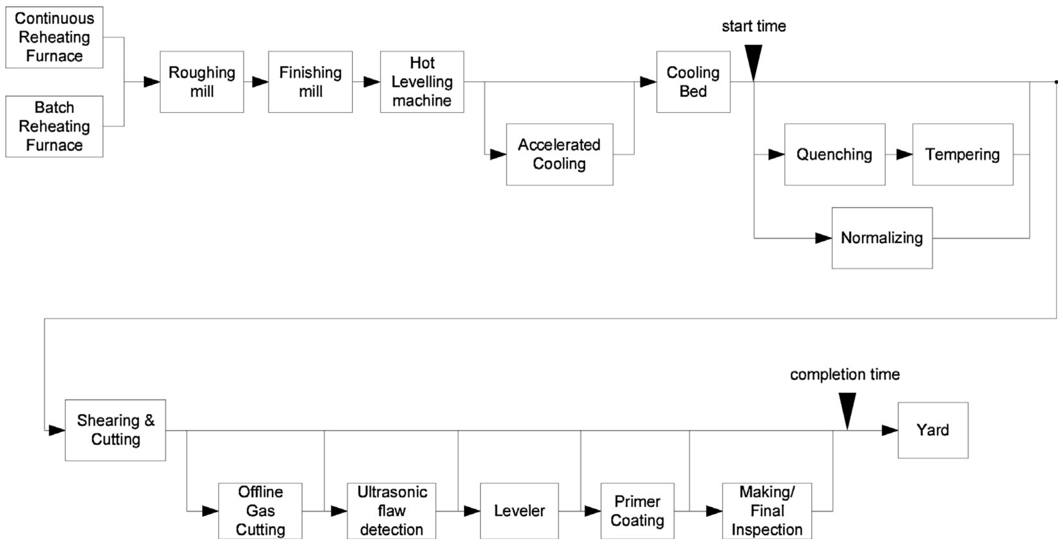


Fig. 1. The example process.

determine the production start times as required to avoid the bottlenecks and meet the customer deadline. In this paper, production loads represent the number of times that one plate or a set of total plates per day is processed at the corresponding production process unit. For instance, if 10 plates are processed at process A per day, then the production load of process A is computed as 10 per day. Production time for each plate or each customer order represents the time required to manufacture a product, starting from slabs to the finished products. In this paper, since the processing time in the inspection and finishing lines is a large part of the production time, the production time represents the period from production start to completion as shown in Fig. 1.

When a steel plate manufacturer receives orders from a customer, the product demand is known while the operating conditions, which are required to manufacture this specific product (order), are not available. On the other hand, at the manufacturing stage, all information about customer demands and detailed operating conditions is known.

Motivated by the above considerations, we can divide the tasks in plate steel production planning and scheduling into two parts: (i) accurate prediction of the production loads and production time and (ii) optimization of the manufacturing plans and schedules based on the production time prediction models. In this paper, we focus on the development of the prediction model for production loads and production time. The prediction model needs to enable:

1 Estimation of the probability distributions of production loads and production time, since the finishing and inspection lines include various sources of uncertainties and thus stochastic prediction models are desirable.
2 Dealing with unobservable (unavailable) variables, because it is desirable to have a single model and avoid multiple models that meet with specific problems.

Having a single model with above properties enables planning and scheduling based on the same model which improves consistency of decisions between planning and scheduling and also simplifies model maintenance.

3. Identification of Bayesian networks structure for steel plate production

In order to build a model which has the properties mentioned in the previous section, we develop the methods to compute probability distributions of unknown states of production loads and production times by using Bayesian network. Methodology to build the model will be illustrated by an example which has 21 discrete variables and 5 continuous variables as shown in Table 1. Each production load is assigned a binary variable having value of 1 if the corresponding production process is not needed, otherwise it is 2.

In this section, we propose the knowledge based date-driven structure learning algorithm to construct the Bayesian network structure. In the subsequent sections, we will propose the parameter estimation and inference approaches with the constructed Bayesian network structure.

The computational task for structure learning is to find the optimal graph such that the maximized Bayesian scores can be obtained, which is known as NP-hard problem. In order to reduce the computational effort, we allow some variables to be set at the root nodes or the leaf nodes and also impose some constraints about orders of subsets of system variables.

First and the most important task for Bayesian inference is to synthesize the precise network structure. The most straightforward method is to design the network structure from process knowledge. However, by-hand structure design requires in-depth process

Table 1
System variables of steel production process.

Variable No.	Variable description	Domain size
1	Customer demand A	12
2	Customer demand B	2
3	Customer demand C	20
4	Customer demand D	20
5	Customer demand E	20
6	Customer demand F	10 (discretized)
7	Customer demand G	10 (discretized)
8	Customer demand H	10 (discretized)
9	Customer demand I	10 (discretized)
10	Operating condition A	5
11	Operating condition B	2
12	Operating condition C	2
13	Operating condition D	2
14	Operating condition E	2
15	Operating condition F	2
16	Operating condition G	2
17	Production load A	2
18	Production load B	2
19	Production load C	2
20	Production load D	2
21	Production load E	2
22	Production load F	2
23	Production load G	2
24	Production load H	2
25	Production load I	2
26	Process time	Continuous

knowledge because identification of cause–effect relationships is needed to characterize the complex physical, chemical and biological phenomena in systems. On the other hand, data-driven based techniques such as score function based structure learning are useful when there is lack of process knowledge or considered processes are too complicated to analyze.

The goal of structure learning is to find a network structure $\mathcal{G}$ that is a good estimator for the data $\mathbf{x}$. The most common approach is *score-based structure learning*, which defines the structure learning problem as an optimization problem (Koller and Friedman, 2009). A score function score($\mathcal{G}:\mathcal{D}$) that measures how well the network model fits the observed data $\mathcal{D}$ is defined. The computational task is to solve the combinational optimization problem of finding the network that has the highest score. The detailed algorithm to find the network structure that maximizes the score function is explained in Appendix A. It should be noted that this combinational optimization problem is known to be *NP*-hard (Chickering, 1996). Therefore, it is challenging to obtain the optimal graph for industrial plants which often include a large number of process variables.

In order to reduce the computational cost, we incorporate the fundamental process knowledge into the network structure learning framework. As shown in the variable list of Table 1, all of our system variables belong to either customer demands or operating conditions or production loads or production time. One can immediately notice that customer demands can be considered "*cause variables*" of all system variables in the Bayesian network and are never impacted by either operating conditions, production loads or production time. Therefore, the nodes of variables belonging to the customer demands should be placed on the root nodes in the Bayesian network. Similarly, production loads are affected by the customer demands, operating conditions or both and not vice versa. Taking into account above considerations, we propose two types of Bayesian networks as shown in Fig. 2. In both Bayesian networks, the nodes related to the customer demands are placed at the root nodes followed by operating conditions, production loads and production time. The difference between two structures is that in the second BN structure, the nodes associated with the production time at each production process are added between the nodes of production loads and total production time while in

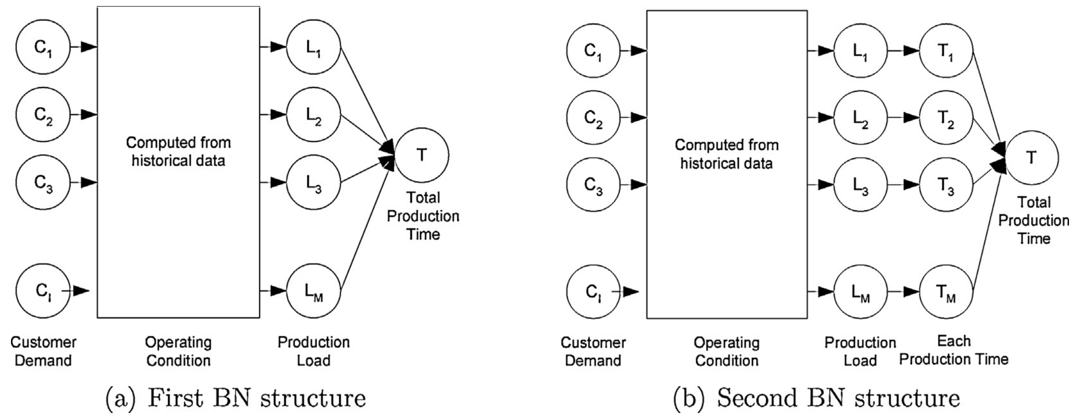


Fig. 2. Two types of Bayesian network structure.

the first BN structure, total production time is directly connected to the production loads.

In addition, we propose some predetermined variable ordering $\prec$ over x to reduce further the computational cost further. If the ordering covers order relationships over all variables as $x_1 \prec x_2, \ldots, x_I$ and the maximum number of parents for each node set to be at most d, then the number of possible parent sets for x_i is at most $\binom{i-1}{d}$ rather than 2^{I-1} (Koller and Friedman, 2009). In practice, nevertheless, these complete orderings are very difficult to identify from process knowledge. Instead of identification of complete orderings, we proposed *incomplete orderings* that do not cover order relationships over all variables but restrict the ordering over subsets of variables. The following is an example of the incomplete ordering.

$$\{x_1, x_2\} \prec \{x_3, x_4, x_5\} \prec \{x_6, x_7\} \tag{1}$$

This incomplete ordering suggests that a computed network should be consistent with the order relationships where x_1 and x_2 precede x_3, x_4, x_5, x_6 and x_7, but any constraints about orders within each subset $\{x_1, x_2\}$, $\{x_3, x_4, x_5\}$ and $\{x_6, x_7\}$ are not imposed. Although the incomplete orderings are optional to set for the structure learning, they can further restrict the solution space because they make the combinational search space much smaller. Therefore, our optimization task is to search for the optimal edges between the nodes of customer demands, operating conditions and production loads so that the BIC scores are maximized and the computed network is consistent with the fixed root and leaf nodes and the incomplete orderings. Consequently, we can further reduce the computational cost due to these constraints.

The search space in our optimization is restricted to the neighboring graphs that can be obtained by either adding one edge or deleting one edge or reversing the existing edge. It should be noted that the neighboring graphs should satisfy the constraints including fixed root and leaf nodes and graph to be acyclic. Therefore, the illegal graphs that violate these constraints are removed from neighboring solutions during search. Tabu search algorithm (Glover, 1986) is employed to make a decision whether the neighboring graphs are accepted. This algorithm uses a neighborhood search to move from one solution candidate to an improved solution in the search space unless the termination criterion is satisfied. In order to avoid local optima as much as possible, a solution candidate is allowed to move to a worse solution unless that solution is included in tabu list which is a set of solutions that have been visited over the last t iterations, where t is termed as tabu size (set to be 100 in this work). The detailed algorithm for network identification is presented in Table 2.

Table 2
Algorithm 1. Network identification algorithm.

```
       Procedure Network-Identification (
       $\mathcal{D}$ // training data set
       $\mathcal{G}_0$ // initial network structure
       $\mathcal{R}$ // a set of fixed root nodes
       $\mathcal{L}$ // a set of fixed leaf nodes
       $\prec$ // a set of incomplete orderings
       global TabuList // tabu list
       )
1      Tabulist $\leftarrow \emptyset$
2      $\mathcal{G} \leftarrow$ Search($\mathcal{G}_0, \mathcal{R}, \mathcal{L}, \prec$)
3      $\mathcal{G}_{best} \leftarrow \mathcal{G}$
4      for $i = 1, \ldots$ until convergence
5        $\mathcal{G} \leftarrow$ Search($\mathcal{G}, \mathcal{R}, \mathcal{L}, \prec$)
6        if score$_{BIC}(\mathcal{G} : \mathcal{D}) >$ score$_{BIC}(\mathcal{G}_{best} : \mathcal{D})$
7          $\mathcal{G}_{best} \leftarrow \mathcal{G}$
8      return $\mathcal{G}_{best}$
       Procedure Search (
       $\mathcal{G}$ // network structure
       $\mathcal{R}$ // a set of fixed root nodes
       $\mathcal{L}$ // a set of fixed leaf nodes
       $\prec$ // a set of incomplete orderings
       )
1      while new graph is not obtained
2        Select indices $i$ and $j$ ($i \neq j$) at random
3        if edge $i \rightarrow j$ exists in $\mathcal{G}$
4          Delete edge $i \rightarrow j$ or reverse edge $i \rightarrow j$ at random and then get $\mathcal{G}_{new}$
5        else if edge $j \rightarrow i$ exists in $\mathcal{G}$
6          Delete edge $j \rightarrow i$, add new edge $i \rightarrow j$ and then get $\mathcal{G}_{new}$
7        else
8          Add new edge $i \rightarrow j$ and then get $\mathcal{G}_{new}$
9        if $\mathcal{G}_{new}$ is cyclic
10         continue
11       if $\mathcal{G}_{new}$ is in TabuList
12         continue
13       if $\mathcal{G}_{new}$ conflicts either $\mathcal{R}$ or $\mathcal{L}$ or $\prec$
14         continue
15       Add $\mathcal{G}_{new}$ to TabuList
16       return $\mathcal{G}_{new}$
```

4. Parameter estimation for Bayesian network with large domain of discrete variables

The major issue that is encountered when applying the proposed Bayesian networks is that some discrete variables have large discrete domains, which causes less robust inference. Let us assume that we have the conditional probability distribution of the node of UST load whose parent nodes are customer demand A, B, C, D, F, G, H, I, and operating condition A. Since the product of cardinalities of all variables is $12 \times 2 \times 20 \times 20 \times 10 \times 10 \times 10 \times 10 \times 5 = 480$ million, it requires a huge amount of training data to compute the appropriate parameters for all elements in the table.

In order to overcome this issue, we propose the decision tree structured CPDs based Bayesian inference techniques that can handle a large number of discrete values. As for second issue of unobservable production time of each process, we utilize the maximum log-likelihood strategies to estimate the parameters of the probability distributions of each process time, as discussed in Section 6.

The network structures studied in this work include both continuous and discrete variable nodes, and hence are called hybrid Bayesian networks. We represents sets of random variables by $\mathcal{X} = \Gamma \cup \Delta$ where Γ denotes the continuous variables and Δ represents the discrete variables. Random discrete variables are denoted by upper case letters from the beginning of the alphabet (A, B, A_1) while continuous variables are denoted by upper case letters near the end of the alphabet (X, Y, X_1). Their actual values are represented by the lower case letters (e.g. a, x, etc.). We denote sets of variables by bold upper case letters ($\mathbf{X}$) and the assignments of those sets by the bold lower case letters ($\mathbf{x}$). We use $\mathrm{Val}(X)$ to represent the set of values that a random variable X can take.

The conditional linear Gaussian (CLG) model is widely used for representation of hybrid Bayesian networks. Let $X \subseteq \Gamma$ be a continuous node, $A \in \mathrm{pa}(X) \cap \Delta$ be its discrete parent nodes and $Y_1, \ldots, Y_k \in \mathrm{pa}(X) \cap \Gamma$ be its continuous parent nodes where $\mathrm{pa}(X)$ denotes a state of parent nodes of X. A Gaussian distribution of X for every instantiation $a \in \mathrm{Val}(A)$ can be represented in moment form as:

$$p(X|\ \mathrm{pa}(X)) = P(X|\ \mathrm{pa}(X); \Theta_a) = \mathcal{N}\left(X|\sum_{j}^{k} w_{a,j} y_j + b_a, \sigma_a^2\right) \tag{2}$$

where $w_{a,j}$ and b_a are parameters controlling the mean, σ_a is the standard deviation of the conditional distribution for X, and $\Theta_a = \{w_{a,j}, b_a, \sigma_a\}$ is the set of model parameters for instantiation a. This representation can also be rewritten in more convenient canonical form as follows (Lauritzen and Wermuth, 1989):

$$C(\mathbf{X}; K_a, h_a, g_a) = \exp\left(-\frac{1}{2}\mathbf{X}^T K_a \mathbf{X} + \mathbf{h}_a^T \mathbf{X} + g_a\right) \tag{3}$$

where K_a, $\mathbf{h}_a$ and g_a represent parameters of ath instantiation in canonical table representations. The canonical table representation can express both the canonical form used in continuous networks and the table factors used in discrete networks. Specifically, when $A = \emptyset$, only a single canonical form ($a = 1$) is obtained. Meanwhile, when $X = \emptyset$, parameters of K_a and $\mathbf{h}_a$ are vacuous and only canonical forms $\exp(g_a)$ remain for every instantiation a.

The limitation of the canonical table representation is that discrete child nodes cannot have continuous parents. For Bayesian networks of the steel plates manufacturing process (Figs. 10 and 11), since the variables related to size of plates such as height, width, length and weight are continuous nodes with discrete child nodes, they are converted into discrete variables. It is true that there is a trade-off between the accuracy of the approximation and the domain size of discretized variables, but due to the proposed decision-tree structured CPTs, we can deal with a large number of values in discrete variables when they are the root nodes in the Bayesian network. In steel plate manufacturing Bayesian networks, although the variables of height, width, length and weight are continuous ones with discrete child nodes, they do not have the parent nodes and thus we can discretize them as finely as we like.

Before computing the parameters of the canonical table, we need to define the instantiations of the set of discrete variables. The simplest approach is to use the traditional CPTs. However, the number of parameters required to describe such CPTs exponentially increases with the domain size. To overcome this limitation, it is useful to capture conditional independence that holds only in

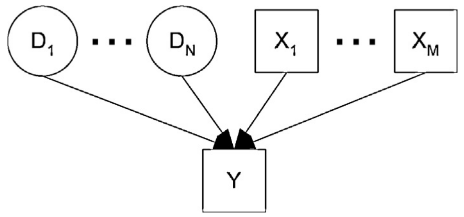


Fig. 3. Bayesian network that contains continuous child node.

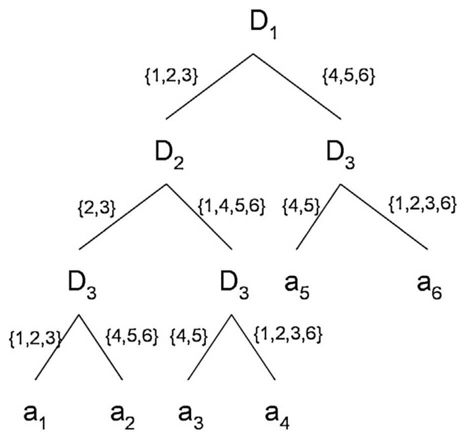


Fig. 4. Decision tree of a factor that contains continuous child node.

certain contexts in the Bayesian networks. Although several works about structured CPTs have been reported to capture this independence (Boutilier et al., 1996; Poole and Zhang, 2003), all of these methods assume that all discrete variables are already grouped at an appropriate level of domain size.

In this work, classification trees algorithm (Breiman et al., 1984) is employed to learn the structured CPTs. Classification trees can predict the values of the target variables by following decisions in the tree from the root nodes down to the leaf nodes. Their structures are identified by choosing a split so that the minimized statistical measures such as Gini impurity or entropy can be obtained.

First, we consider a factor that contains both continuous nodes $\mathbf{X}$ and discrete nodes $\mathbf{D}$ with continuous child node Y as shown in Fig. 3. Continuous variables are described as rectangles while discrete variables are ovals. The classification tree $\mathcal{T}$ classifies the discrete variable set $\mathbf{D}$ into a small number of subsets $A = \{a_1, a_2, \ldots, a_L\}$ in accordance with the decision $a = \mathcal{T}(\mathbf{d})$ where L is the number of the leaves in the tree $\mathcal{T}$. For example, the decision tree provided in Fig. 4 means that if $D_1 \in \{4, 5, 6\}, D_3 \in \{4, 5\}$, then the corresponding records are assigned the instantiation a_5 regardless of the value of

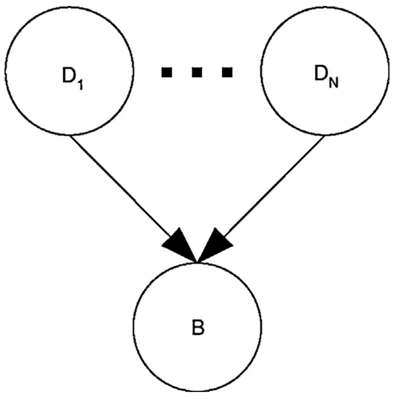


Fig. 5. Bayesian network that contains discrete child node.

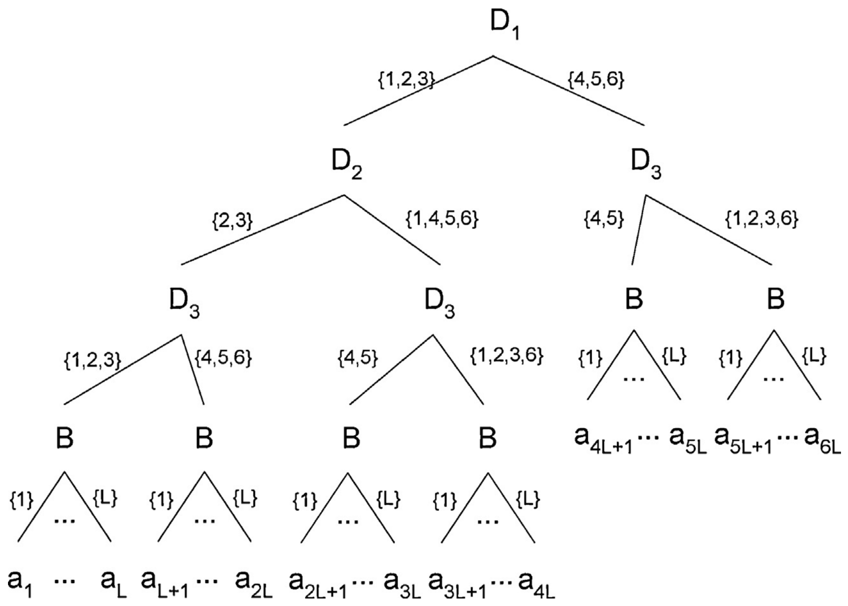


Fig. 6. Decision tree of a factor that contains discrete child node.

D_3. It should be noted that the tree representation requires only $|\text{Val}(A)| = 6$ parameters to describe the behavior of target variable Y. In the example, if the traditional CPTs are used, the number of parameters becomes $|\text{Val}(D_1)| \times |\text{Val}(D_2)| \times |\text{Val}(D_3)| = 216$, which is much larger than that of the tree structured CPTs. Therefore, the tree structured CPTs need much smaller number of training data to learn the parameters of the conditional probability distributions and also lead to more robust estimation compared to the traditional CPT representation.

After giving instantiations $a_i \in A$ to all samples in the training data by following the decision in the constructed tree, parameters of the canonical for each instantiations s_i can be computed as follows:

$$K_{a_i} = \Sigma_i^{-1}$$

$$\mathbf{h}_{a_i} = \Sigma_i^{-1}\boldsymbol{\mu}_i \tag{4}$$

$$g_{a_i} = -\frac{1}{2}\boldsymbol{\mu}_i^T \Sigma_i^{-1}\boldsymbol{\mu}_i - \log((2\pi)^{n/2}|\Sigma_i|^{1/2})$$

with

$$\Sigma_i = \text{cov}[\mathbf{Z}_i] \tag{5}$$

$$\boldsymbol{\mu}_i = \text{mean}[\mathbf{Z}_i] \tag{6}$$

where $\mathbf{Z} = \mathbf{X} \cup Y, \mathbf{Z}_i = \{z : \mathcal{T}(\mathbf{d}) = a_i\}$ and $\text{cov}[\mathbf{Z}_i]$ and $\text{mean}[\mathbf{Z}_i]$ are the covariance matrix and mean vector of $\mathbf{Z}_i$ respectively.

Next, let us consider the case when a factor contains discrete nodes $\mathbf{D}$ with discrete child node B as shown in Fig. 5. The classification tree classifies discrete variables $\mathbf{D}$ into a small number of subsets $A = \{a_1, a_2, \ldots, a_L\}$ by the following the rules $a = \mathcal{T}(\mathbf{d})$ with L being the number of leaves so that the values of a discrete child node B can be categorized well. While the decision tree can well categorize the discrete parent variables $\mathbf{D}$, a discrete child variable B that is used as a dependent variable in the tree $\mathcal{T}$ is not included in the nodes of the decision tree. As a result, the obtained decision tree $\mathcal{T}$ can classify values of discrete parent variables $\mathbf{D}$ only and it excludes a discrete child variable B from the branching nodes in the tree $\mathcal{T}$. Since we need to classify values of all discrete variables that the factor includes, the discrete child node is placed at all leaf nodes in the tree $\mathcal{T}$ to continue categorizing discrete variables by using their values. Specifically, all values of a discrete child variable $b_1 \ldots b_L \in B$ are used as categorical splits at the leaf nodes in the tree $\mathcal{T}$ and thus the number of branches at each leaf node is same as the domain size of $\text{Val}(B) = L$. Therefore, the total number of instantiations assigned in the new decision tree $\mathcal{T}_{\text{new}}$ is $\text{Val}(B)$ times as large as the number of leaves in the original tree $\mathcal{T}$. Fig. 6 shows the example of a decision tree when a factor contains a discrete child node. The decision tree gives us the rule where if $D_1 \in \{4, 5, 6\}, D_3 \in \{4, 5\}$ and $B \in \{1\}$, then the corresponding records are assigned the instantiation a_{4L+1} regardless of the value of D_2. The number of instantiations are $\text{Val}(B) \times 6 = 6L$ rather than $|\text{Val}(D_1)| \times |\text{Val}(D_2)| \times |\text{Val}(D_3)| \times |\text{Val}(B)| = 216L$ which is the number of parameters required to describe the traditional CPT of discrete variables $\mathbf{D} \cup B$.

Using the instantiations $a_i \in A$ assigned by following the decision $\mathcal{T}_{\text{new}}$, parameters g_{a_i} of the canonical table representation can be computed as follows:

$$g_{a_i} = \log\frac{1}{N_{a_i}}\sum_{z}\mathbf{1}(f(\mathbf{d}) = a_i) \tag{7}$$

where $\mathbf{1}$ represents an indicator function, N_{a_i} is the number of samples belonging to the instantiation a_i. The other canonical parameters K_{a_i} and $\mathbf{h}_{a_i}$ are vacuous since the factor does not include any continuous nodes.

5. Inference in Bayesian network with large domain of discrete variables

5.1. Types of reasoning

Bayesian inference answers the *conditional probability query*, $P(\mathcal{X}|\mathbf{E} = \mathbf{e})$ where the evidence $\mathbf{E} \subseteq \mathbf{X}$ is a subset of random variables in the Bayesian network and $\mathbf{e} \in \mathbf{Val(E)}$ is the set of values given to these variables. The four major reasoning patterns in Bayesian inference are *causal reasoning*, *diagnostic reasoning*, *intercausal reasoning* and *mixed reasoning*. The causal reasoning predicts the downstream effects of factors from top nodes to bottom nodes while the diagnostic reasoning predicts the upstream causes of factors from bottom nodes to top nodes. The intercausal reasoning is the inference where different causes of the same effect can interact. The mixed inference is the combination of two or more of the above reasoning. In order to implement the causal inference, the conditional probability of effects for given causes is computed by means of the sum-product algorithm from top node factors to bottom node factors.

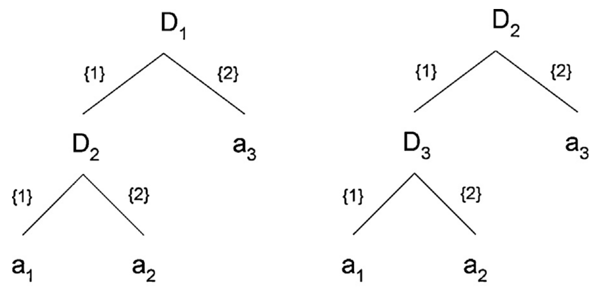
Inferences other than causal reasoning, e.g. diagnostic, intercausal and mixed reasoning cannot be carried out in logic which is monotonic. Therefore, the fundamental algorithm for exact inference for these reasons in graphical models is *variable elimination* (VE). VE can be carried out in a clique tree and each clique in the clique tree corresponds to the factors in VE. Since the computational cost of the clique tree algorithm exponentially grows with the width of the network, the inference cannot be achieved in large width networks that represent many real-world industrial problems. Hence, one needs to employ approximate inference algorithms such as loopy belief propagation (LBP) algorithm, since they are tractable for many real-world graphical models. Consequently, we will employ LBP algorithm to infer the probability distributions of unobserved variables, except for the causal reasoning. All of the above inference techniques require the computation of multiplying factors and marginalizing variables in factors.

Since our purpose for employing Bayesian inference is to estimate the downstream effects of production loads and production time from upstream causes of the customer demands or the operating conditions, we need to be able to carry out causal reasoning computations. We can carry out such Bayesian inference in the very straightforward way by applying the sum-product algorithm from top node factors to bottom node factors. However, it is also very useful to estimate the probability distributions of operating conditions given the customer demands and production loads in the case when we need to find the best operating conditions that satisfy both the customer demands and production capacities. Hence, besides the causal reasoning, in this work we take into account the other types of inference such as intercausal reasoning.

5.2. Multiplying factors and marginalizing over variables in factors

In order to compute potentials of final and intermediate factors including messages between clusters, the operations of multiplying and marginalizing factors are needed. These operations require us to divide the large domain of discrete variables by following the tree structure. In this section, we develop two main operations: (i) multiplying factors and (ii) marginalizing over variables in factors.

The simplest approach is to convert the tree structured CPTs into traditional conditional probability tables which can be analyzed by using standard multiplying and marginalizing techniques. However, the number of rows in the converted traditional CPT of a factor exponentially increases both with the domain size and with the number of discrete variables that a factor includes. Therefore, this approach easily runs out of available computer memory for large Bayesian networks where factors contain a large number of discrete variables (huge domain of discrete variables). Instead of converting to the traditional CPTs, we propose more compact form of conditional probability tables.



(a) Tree for Factor A (b) Tree for Factor B

Fig. 7. Example of two decision trees.

Table 3
Compact CPT for factor A.

D_1	D_2	Probability distribution
1	1	$C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1})$
1	2	$C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_3}, g^A_{a_2})$
2	1,2	$C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_2}, g^A_{a_3})$

Table 4
Compact CPT for factor B.

D_2	D_3	Probability distribution
1	1	$C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	2	$C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	1,2	$C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$

Let us consider the simple example of multiplying two factors described in Fig. 7. First we convert the tree structured CPTs into conditional probability tables where each row corresponds to each instantiation in the tree and each entry represents a probability distribution in the corresponding instantiation as shown in Tables 3 and 4. We named this conditional probability table as *compact conditional probability table* (compact CPT) since this representation is more compact than the traditional one in terms of the number of rows. We combine the two compact CPTs into a single CPT to be able to make all distinctions that original two CPTs make. Therefore, the compact CPT for factor B is combined with each row of the compact CPT for factor A as shown in Table 5. Since the combined CPT contains redundant rows whose cardinalities conflict each other, we eliminate these inconsistent rows. Finally, the canonical table representation of a new factor C after multiplying two factors is shown in Table 6. The canonical parameters of the probability distribution can be computed by following the

Table 5
Combined compact CPT of factors A and B.

Factor A		Factor B		Probability distribution
D_1	D_2	D_2	D_3	
		1	1	$C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	1	1	2	$C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
		2	1,2	Inconsistent
		1	1	Inconsistent
1	2	1	2	Inconsistent
		2	1,2	$C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
		1	1	$C(\mathbf{X}; K^C_{a_4}, \mathbf{h}^C_{a_4}, g^C_{a_4}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1,2	1	2	$C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
		2	1,2	$C(\mathbf{X}; K^C_{a_6}, \mathbf{h}^C_{a_6}, g^C_{a_6}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$

Table 6
Compact CPT for factor C.

D_1	D_2	D_3	Probability distribution
1	1	1	$C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
1	1	2	$C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}) = C(\mathbf{X}; K^A_{a_1}, \mathbf{h}^A_{a_1}, g^A_{a_1}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
1	2	1,2	$C(\mathbf{X}; K^C_{a_3}, \mathbf{h}^C_{a_3}, g^C_{a_3}) = C(\mathbf{X}; K^A_{a_2}, \mathbf{h}^A_{a_2}, g^A_{a_2}) \cdot C(\mathbf{X}; K^B_{a_3}, \mathbf{h}^B_{a_3}, g^B_{a_3})$
2	1	1	$C(\mathbf{X}; K^C_{a_4}, \mathbf{h}^C_{a_4}, g^C_{a_4}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_1}, \mathbf{h}^B_{a_1}, g^B_{a_1})$
2	1	2	$C(\mathbf{X}; K^C_{a_5}, \mathbf{h}^C_{a_5}, g^C_{a_5}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$
2	2	1,2	$C(\mathbf{X}; K^C_{a_6}, \mathbf{h}^C_{a_6}, g^C_{a_6}) = C(\mathbf{X}; K^A_{a_3}, \mathbf{h}^A_{a_3}, g^A_{a_3}) \cdot C(\mathbf{X}; K^B_{a_2}, \mathbf{h}^B_{a_2}, g^B_{a_2})$

obtained table. For instance, the canonical parameters with $D_1 = 1$, $D_2 = 2$, $D_3 = 1$ are computed as follows:

$$
\begin{aligned}
K^C_{a_1} &= K^A_{a_1} + K^B_{a_1} \\
\mathbf{h}^C_{a_1} &= \mathbf{h}^A_{a_1} + \mathbf{h}^B_{a_1} \\
g^C_{a_1} &= g^A_{a_1} + g^B_{a_1}.
\end{aligned} \tag{8}
$$

Next, let us consider summing out discrete variables D_3 in the compact CPT described in Table 6. We sum up probability distribution to make all distinctions except for D_3 that original table makes. In this example, since D_1 and D_2 of the first and second rows in the table take same values of $D_1 = 1$ and $D_2 = 1$, the corresponding two canonical forms are summed up. Nevertheless, if we need to marginalize factors that contain continuous variables by summing out discrete variables, the resulting distribution is not a Gaussian but a mixture of Gaussians. Therefore, a mixture of Gaussians model should be used to represent the marginalized factors, which can be achieved through Monte Carlo integration (Yuan and Druzdzel, 2006). The problem with this approximation method is that the number of Gaussian components grows with the number of times discrete variables are summed out. In order to avoid such a problem, the mixture of Gaussian is approximated by collapsing it into a single Gaussian.

The marginal distribution over D_1 and D_2 assigns probability distributions to specific events such as $P(D_1 = 1, D_2 = 1)$, $P(D_1 = 2, D_2 = 1)$, $P(D_1 = 1, D_2 = 2)$ and $P(D_1 = 2, D_2 = 2)$. For instance, the marginal probability distribution function $P(D_1 = 1, D_2 = 1)$ can be computed as follows:

$$
P(D_1 = 1, D_2 = 1) = C(\mathbf{X}; K^C_{a_1}, \mathbf{h}^C_{a_1}, g^C_{a_1}) + C(\mathbf{X}; K^C_{a_2}, \mathbf{h}^C_{a_2}, g^C_{a_2}). \tag{9}
$$

However, as mentioned before, this distribution is not a Gaussian but a mixture of Gaussian and thus it cannot be represented as a canonical form. Instead, we employ the weak discrete marginalization where the summing operation uses the collapsing operation (Koller and Friedman, 2009). The collapsing operation approximates a mixture of k Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$ by a single Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_j, \Sigma_j)$. Its mean vector $\boldsymbol{\mu}_j$ and covariance matrix Σ_j for new instantiation j are computed as follows:

$$
\boldsymbol{\mu}_j = \sum_{i=1}^{k} p(i|j)\mu_i \tag{10}
$$

$$
\Sigma_j = \sum_{i=1}^{k} p(i|j)\Sigma_i + \sum_{i=1}^{k} p(i|j)(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \tag{11}
$$

where $p(i|j)$ is the conditional probability of the original instantiation i given the new instantiation j and it satisfies $\sum_{i=1}^{k} p(i|j) = 1$. Therefore, the parameters (K_{new}, $\mathbf{h}_{\text{new}}$, g_{new}) of a canonical form $P(D_1 = 1, D_2 = 1)$ which is summed out variable D_3 can be computed as follows:

$$
K_{\text{new}} = \Sigma^{-1}_{\text{new}} \tag{12}
$$

$$
\mathbf{h}_{\text{new}} = \Sigma^{-1}_{\text{new}} \boldsymbol{\mu}_{\text{new}} \tag{13}
$$

where

$$
\begin{aligned}
\boldsymbol{\mu}_{\text{new}} &= p(D_3 = 1|D_1 = 1, D_2 = 1) K^{C-1}_{a_1} \mathbf{h}^C_{a_1} \\
&\quad + p(D_3 = 2|D_1 = 1, D_2 = 1) K^{C-1}_{a_2} \mathbf{h}^C_{a_2}
\end{aligned} \tag{14}
$$

$$
\begin{aligned}
\Sigma_{\text{new}} &= p(D_3 = 1|D_1 = 1, D_2 = 1) K^{C-1}_{a_1} \\
&\quad + p(D_3 = 2|D_1 = 1, D_2 = 1) K^{C-1}_{a_2} \\
&\quad + p(D_3 = 1|D_1 = 1, D_2 = 1)(K^{C-1}_{a_1}\mathbf{h}^C_{a_1} - \boldsymbol{\mu}_{\text{new}})(K^{C-1}_{a_1}\mathbf{h}^C_{a_1} - \boldsymbol{\mu}_{\text{new}})^T \\
&\quad + p(D_3 = 2|D_1 = 1, D_2 = 1)(K^{C-1}_{a_2}\mathbf{h}^C_{a_2} - \boldsymbol{\mu}_{\text{new}})(K^{C-1}_{a_2}\mathbf{h}^C_{a_2} - \boldsymbol{\mu}_{\text{new}})^T.
\end{aligned} \tag{15}
$$

In the case when a set of continuous variables $\mathbf{Y}$ needs to be integrated out, continuous marginalization is carried out after summing out discrete variables. Let us assume that our canonical table consists of a set of canonical forms $C(\mathbf{X}, \mathbf{Y}; K^C_{a_i}, \mathbf{h}^C_{a_i}, g^C_{a_i})$ indexed by a_i where:

$$
K_{a_i} = \begin{bmatrix} K_{\text{XX}} & K_{\text{XY}} \\ K_{\text{YX}} & K_{\text{YY}} \end{bmatrix} \tag{16}
$$

$$
\mathbf{h}_{a_i} = \begin{pmatrix} \mathbf{h}_{\mathbf{X}} \\ \mathbf{h}_{\mathbf{Y}} \end{pmatrix}. \tag{17}
$$

The integral over the variables $\mathbf{Y}$ becomes a canonical form $C(\mathbf{X}; K'^C_{a_i}, \mathbf{h}'^C_{a_i}, g'^C_{a_i})$ computed as (Lauritzen, 1992):

$$
\begin{aligned}
K'^C_{a_i} &= K_{\mathbf{XX}} - K_{\mathbf{XY}} K^{-1}_{\mathbf{YY}} K_{\mathbf{YX}} \mathbf{h}'^C_{a_i} = \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}} K^{-1}_{\mathbf{YY}} \mathbf{h}_{\mathbf{Y}} \\
g'^C_{a_i} &= g + \frac{1}{2}\left(\log|2\pi K^{-1}_{\mathbf{YY}}| + \mathbf{h}^T_{\mathbf{Y}} K^{-1}_{\mathbf{YY}} \mathbf{h}_{\mathbf{Y}} \right)
\end{aligned} \tag{18}
$$

where $K_{\mathbf{YY}}$ should be positive definite since these equations include the calculation of inverse matrix of $K_{\mathbf{YY}}$.

Above methodology enables us to compute product and marginalization of factors. Compared to the approach that converts tree structure CPTs into the traditional large CPTs, the proposed compact CPTs do not require large computer memory to carry out these operations. Instead, the summing out operations group rows in the CPTs so that all rows in the each group have the same discrete values except for summed out variables and the computational cost of grouping rows increases with $O(n \log n)$. One possible drawback is that the decision tree structures are no longer available once factors are multiplied or marginalized because the compact CPT based operations discard tree structures.

5.3. *Belief propagation algorithm*

In order to implement various kinds of reasoning, we apply the *belief propagation* (BP) algorithm to handle a large hybrid Bayesian network containing large domain of discrete variables. The detailed

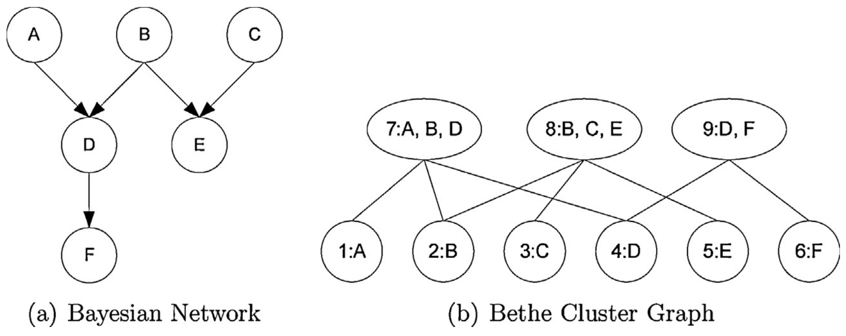


Fig. 8. Example of Bethe cluster graph.

Table 7
Algorithm 2. Hybrid loopy belief propagation algorithm.

```
    Procedure Hybrid-LBP (
    $\Phi$ // Set of factors
    e // Evidence
    $\mathcal{G}_{\text{cluster}}$ // Bethe cluster graph
    )
1   Set $\mathcal{E}$ to be set of edges in $\mathcal{G}_{\text{cluster}}$
2   Initialize all factors so that all of them are consistent with the evidence e
3   Initialize all messages as $\{K_{a_i}, \mathbf{h}_{a_i}, g_{a_i}\} = \{1, 0, 0\}$ for each instantiation $a_i$
4   while any one of messages are not converged
5     Select $(i, j) \in \mathcal{E}$ by following message schedule
6     if scope($\psi_i$) = scope($e$)
7       continue
8     Update message $\mu_{i\to j}$ from Eq. (25)
9   Compute final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ from (26)
10  Normalize $\tilde{P}_\Phi(\mathbf{C}_i)$ and get $P_\Phi(\mathbf{C}_i)$
11  return $P_\Phi(\mathbf{C}_i)$
```

algorithms of causal, diagnostic and intercausal reasoning are described in Appendices B and C. When entering evidence into Bayesian network, factors need to be modified so that they are consistent with the evidence. The method of inference given the evidence is shown in Appendix D.

It should be noted that the computational task for multiplying incoming messages exponentially increases with the number of incoming messages, when using the sum-product algorithm. This is the case when the diagnostic, intercausal or mixed inferences are carried out because the LBP requires a large number of calculations for factor multiplication and marginalization. To reduce the computational cost in LBP algorithm, we make use of the property of Bethe cluster graph. In a Bethe cluster graph, the scope of messages covers single variable since any large clusters are connected each other. If we give the evidence to some variable $X_i = e$, the message departed from the corresponding singleton cluster $\mathbf{C}_i$ remains the same during iterations. In other word, we can peg the message $\delta_{i\to k}$ to evidence e and thus we can skip the computation of Eq. (25). The proposed loopy belief propagation algorithm is shown in Table 7.

6. Application example

6.1. Comparison with SVM and ANN

A simulated example is used to evaluate the validity and performance of the Bayesian networks compared to ANN and SVM. The input and output data are generated from the simple system described in Fig. 9. In this example, $[X_1, X_2, X_3, Y_1, Y_2]$ are

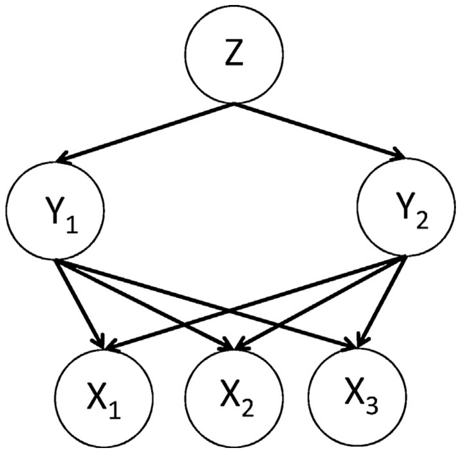


Fig. 9. Simple system.

Table 9
Results of BN, ANN and SVM for the simple system.

Known variables	BN	ANN	SVM
$[X_1, X_2, X_3, Y_1, Y_2]$ (fully observed)	0.970	0.958	0.958
$[X_1, X_2, X_3]$ (partially observed)	0.838	0.734	0.734

input variables while Z is an output variable. Variables are linearly dependent when they are connected with the arcs. Further, all variables are binarized and thus the domain size of each variable is two.

The radial basis kernel function is used for SVM and its parameters are determined automatically using the heuristic method described in the literature (Caputo et al., 2002). As for ANN, the number of hidden variables is determined from cross-validation. The classification accuracies, which are the proportion of true results, along with the number of hidden variables are shown in Table 8. It can be seen that accuracy remains the same even when we increase the number of hidden variables. We set the number of hidden variables to be four for ANN.

The computational results are shown in Table 9. When all variables can be observed except for output variables, the accuracies for test data of BN, ANN and SVM are greater than 0.9, which means all methods can accurately predict the output variables. On the other hand, when input data are partially known, both ANN and SVM predict the output variable for test data with lower accuracies of

Table 8
Results of the cross-validation (ANN).

The number of hidden variables	1	2	3	4	5	6	7	8	9	10
Accuracy	0.928	0.928	0.928	0.930	0.926	0.930	0.928	0.930	0.928	0.928

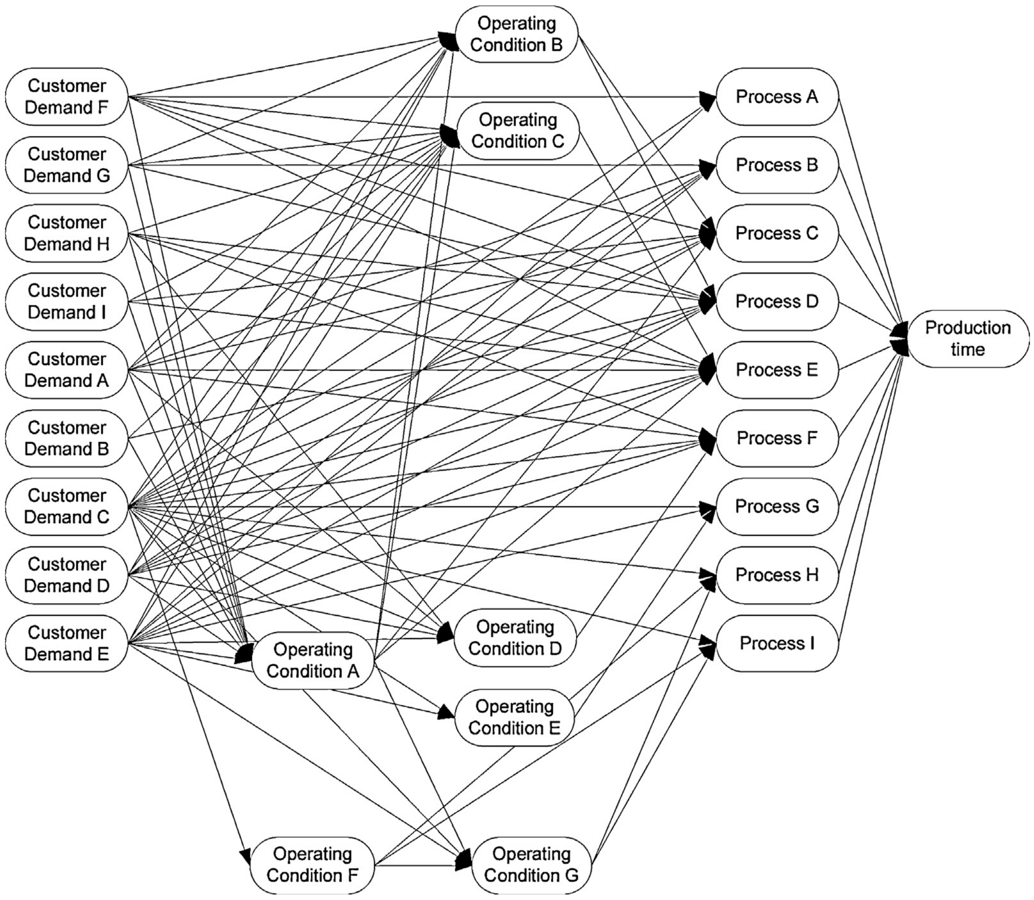


Fig. 10. Identified first type of Bayesian network for steel production process.

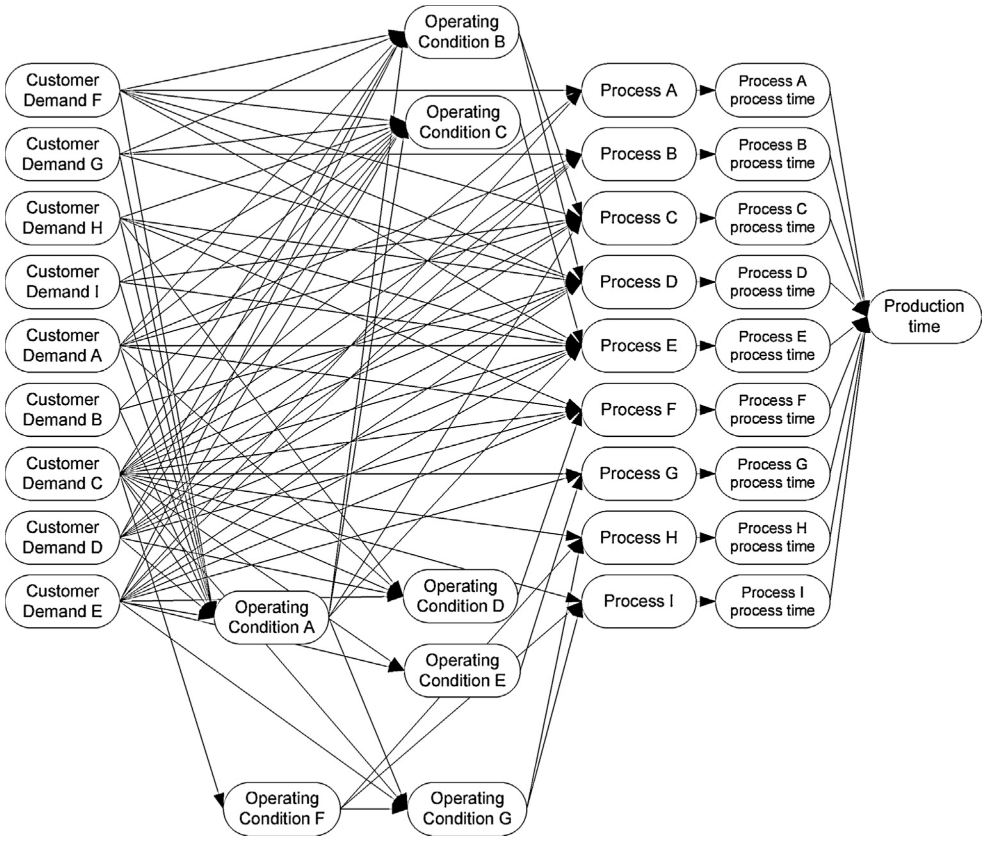


Fig. 11. Identified second type of Bayesian network for steel production process.

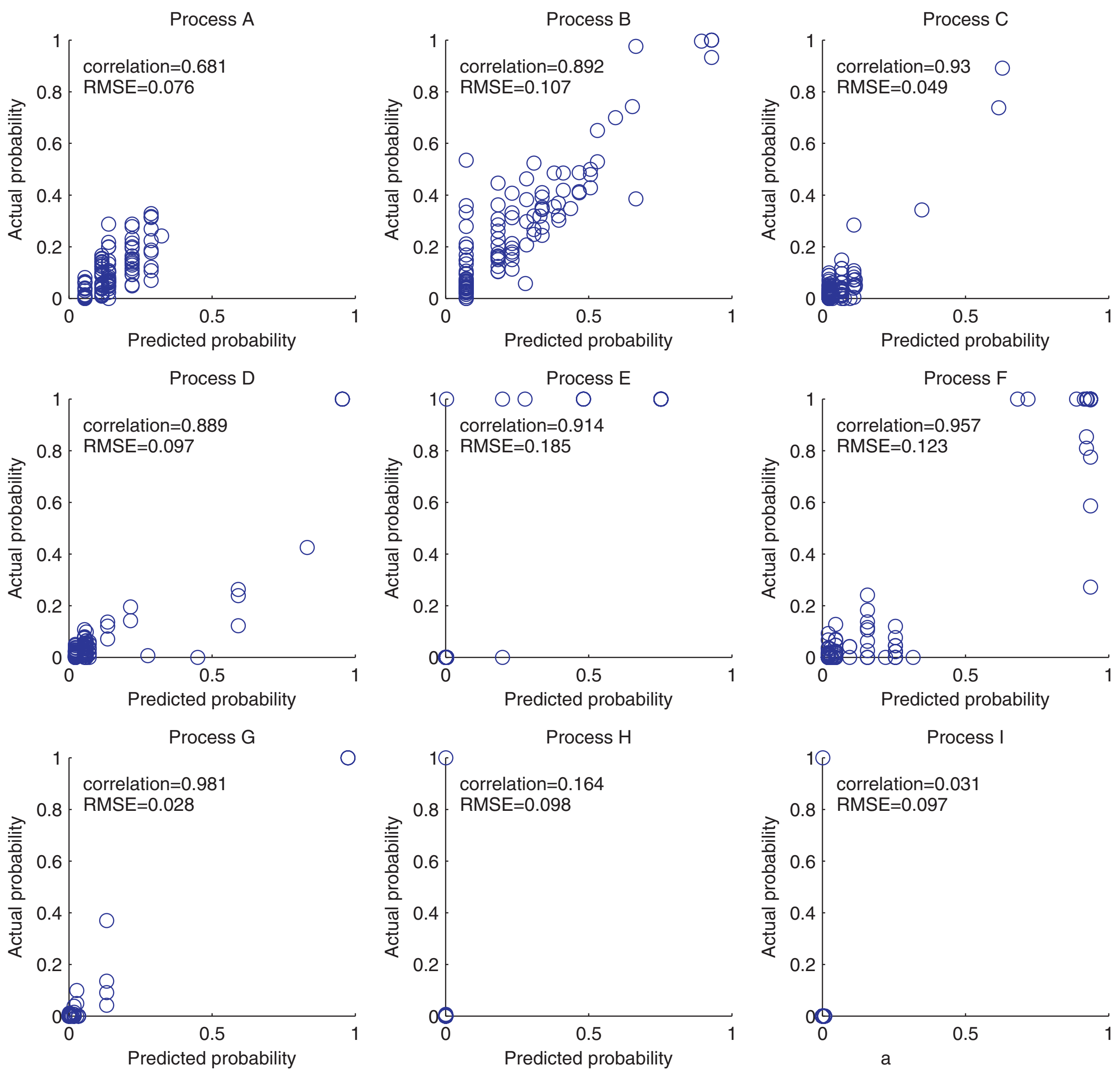


Fig. 12. Scatter plots of probability of production loads given customer demands in case 1.

0.734 and 0.734 respectively. Meanwhile, BN predicts the output variable with the high accuracy of 0.848 even though only partial input data can be observed. These computational results demonstrate that BN is superior to SVM and ANN in terms of accuracy with partially observed data.

It is true that SVM is known as one of the strongest classification machine and it works equally to or better than BNs for classification problems with fully observed data. However, as the computational results show, if only partial input data are observed, BN performs better than SVM. In addition, if we need to predict multiple variables, SVM requires multiple models which meet the specific situations, while BN can work for any situations (e.g. fully or partially observed) by using single model. Furthermore, SVM works only for classification problems and thus it is not be able to predict both continuous and discrete variables simultaneously.

6.2. *Bayesian network model identification*

The real-world plate steel production data are utilized to examine the effectiveness of the proposed Bayesian network based prediction models. All the case studies have been computed on a DELL Optiplex 990 (Intel(R) Core(TM) i-7-2600 CPU, 3.40 GHz and 8.0 GB RAM). First, we use the training data set consisting of 10, 000 steel plates to learn the network structure. The Bayesian network structures are identified by maximizing BIC scores as shown in Figs. 10 and 11. The execution time for the network identification was 1, 353[*s*]. Then, we use the training data set consisting of 100, 000 steel plates to learn the decision-tree structured CPTs while we use the test data set of another 100, 000 plates to evaluate prediction accuracy. The first and second Bayesian networks are mainly different in the form of the conditional probability of total production time.

6.3. *Prediction of production loads and total production time*

In the first test scenario, we assume that the variables corresponding to the customer demands are known and the other variables are unknown. This situation occurs when persons in the sales department receive orders from customers, since the sales planning staff need to know the production loads and the process time of these orders as required to satisfy the customer demands and within constraints of the production capacity. The task of this inference is to predict the effects of production loads and a process time given the causes of the customer demands, which means that the type of inference is causal reasoning. In order to compute the true probability distributions which are required

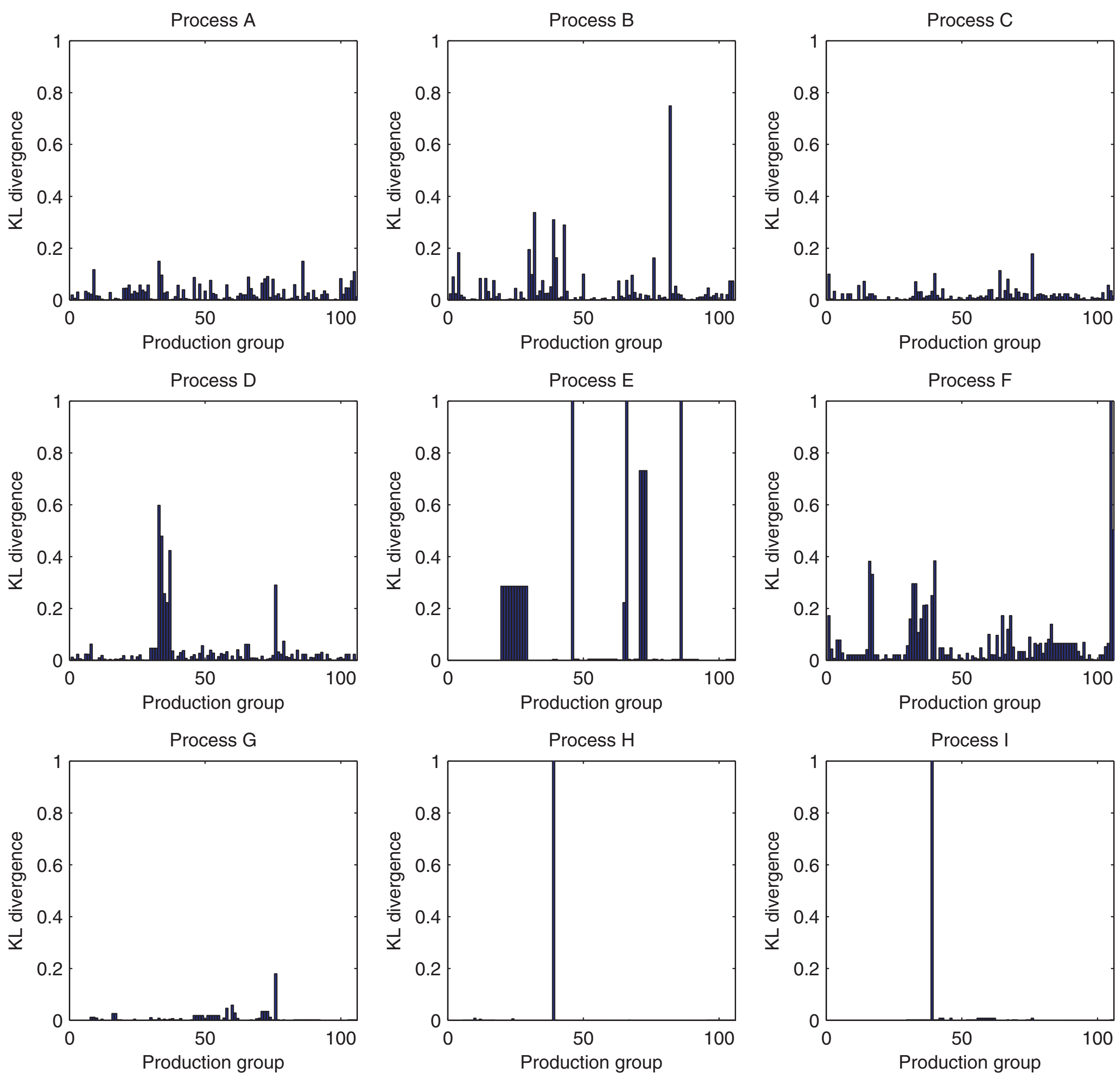


Fig. 13. KL divergences between the predicted and true probability distributions of production loads given customer demands in case 1.

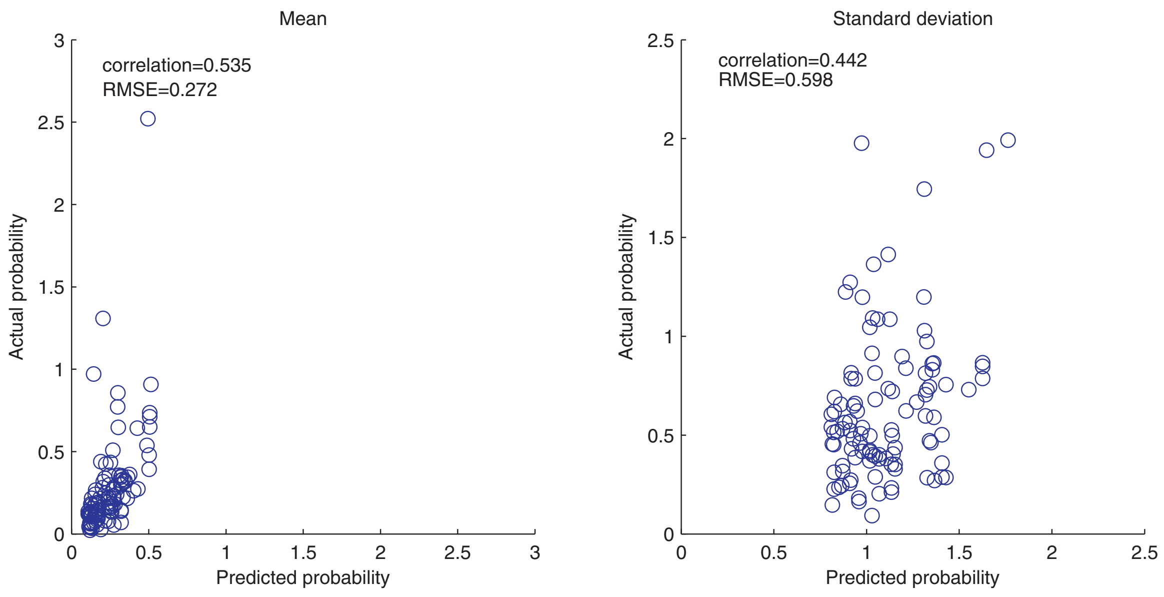


Fig. 14. Scatter plots of mean and standard deviation of production time given customer demands using the first Bayesian network in case 1.

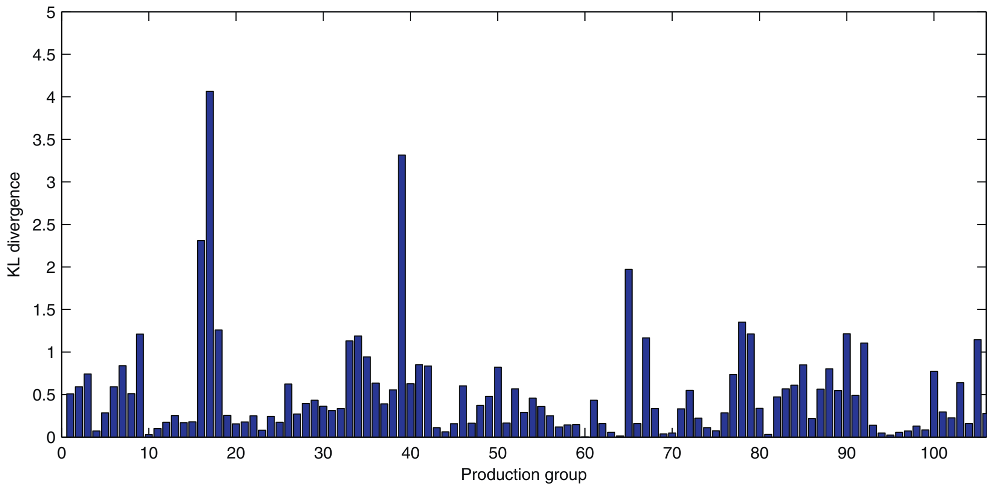


Fig. 15. KL divergences between the predicted and true probability distributions of production time given customer demands using the first Bayesian network in case 1.

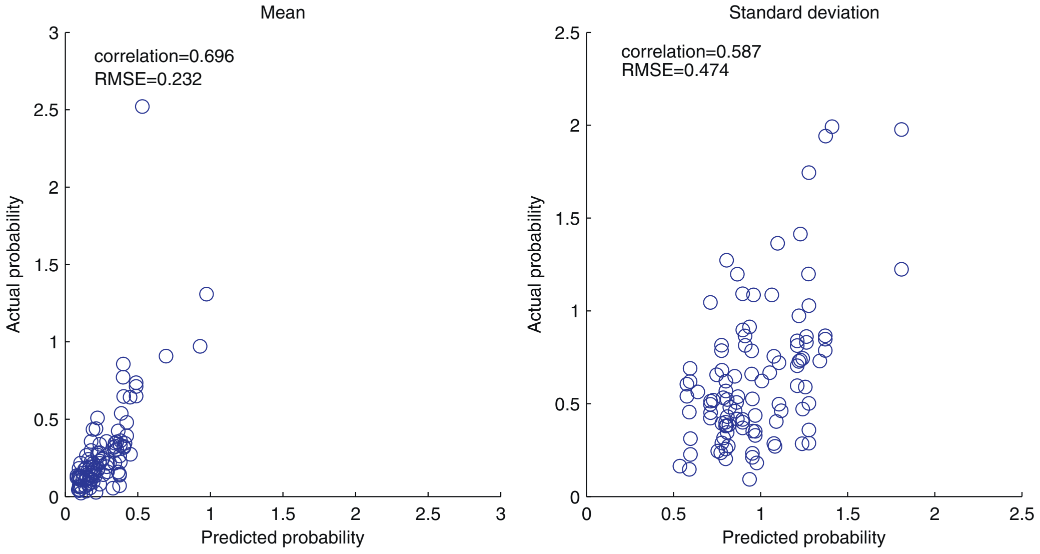


Fig. 16. Scatter plots of mean and standard deviation of production time given customer demands using the second Bayesian network in case 1.

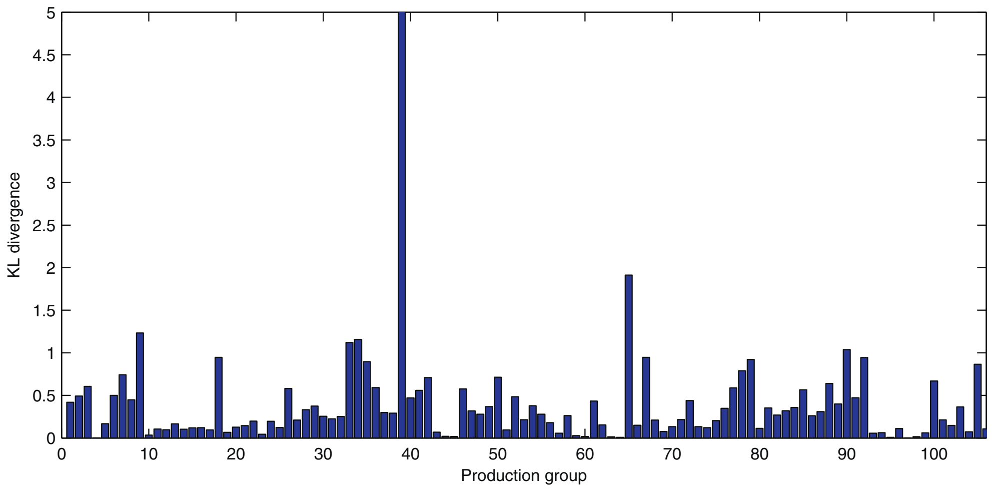


Fig. 17. KL divergences between the predicted and true probability distributions of production time given customer demands using the second Bayesian network in case 1.

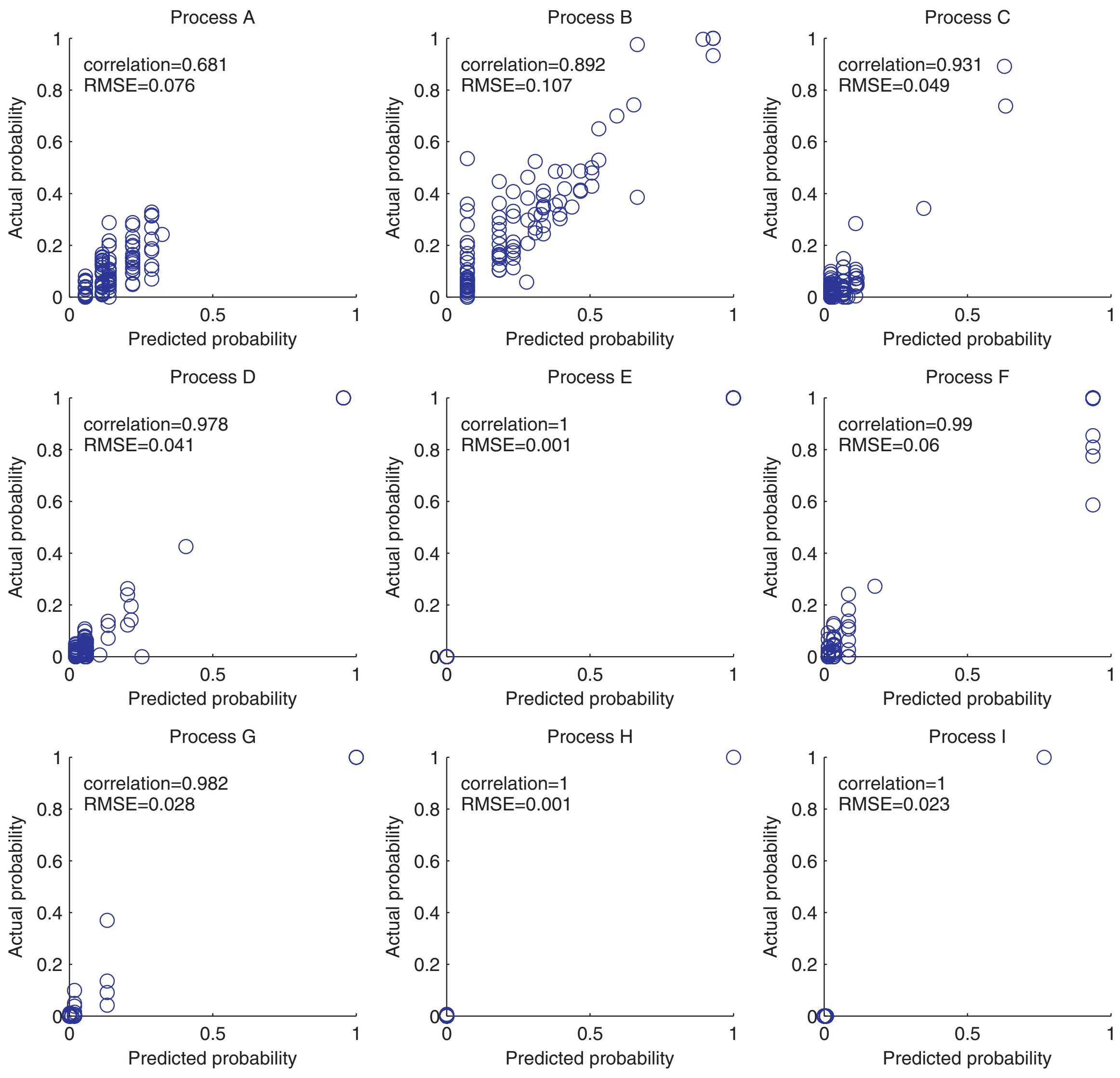


Fig. 18. Scatter plots of probability of production loads given customer demands and operating conditions in case 2.

for comparison to the prediction, we classify the test data into production groups which contain plates that have the same customer specifications and operating conditions and then compute the probability distribution for each production group. The production groups containing more than 100 plates are chosen for comparison while the other production groups are not selected because we cannot compute the reliable probability distributions from a small number of samples. The total number of the production groups that are used for comparison is 106.

First, we predict the production loads and the production time by using the first Bayesian network structure shown in Fig. 10. Fig. 12 shows the scatter plots between the predicted and true probabilities with production loads being 2. Further, the Kullback–Leibler (KL) divergence, which measures the difference between the predicted probability distributions and actual ones is shown in Fig. 13. It can be observed that the difference between the predicted and true probability distributions of the variables of process A, process B, process C, process D, process E, process F and process G are relatively small while the predicted probabilities with process H and process I values being 2 are zero in all production groups. Moreover, the scatter plots between the predicted and true parameters of means and standard deviations of the production time are shown in Fig. 14, and their KL divergences are provided in Fig. 15. It can be seen that the means of the predicted distributions are relatively close to the true means but the variances of the predicted distributions largely differ from the actual ones. Somewhat inaccurate prediction of the production loads of process E, process F, process H and process I can be considered a cause of these differences in the production time.

Next, the probability distributions of the production loads and the production time are computed via the second type of Bayesian network, shown in Fig. 11. The difference between the two structures is that the nodes associated with the production time at each production process are added between the nodes of production loads and total production time in the second type on Bayesian network. Similarly to the first model, we provide the scatter plots of the predicted and true parameters of means and standard deviations and their KL divergences in Figs. 16 and 17, respectively. It can be observed that the prediction accuracy does not improve at all from the first Bayesian network. The cause of inaccurate prediction of the variances of the total production time is the low accurate prediction of the production loads.

In the second test case, we consider the situation where the variables associated with both customer demands and operating conditions are known and the variables of both production loads and process time are unknown. In this situation, we want to predict

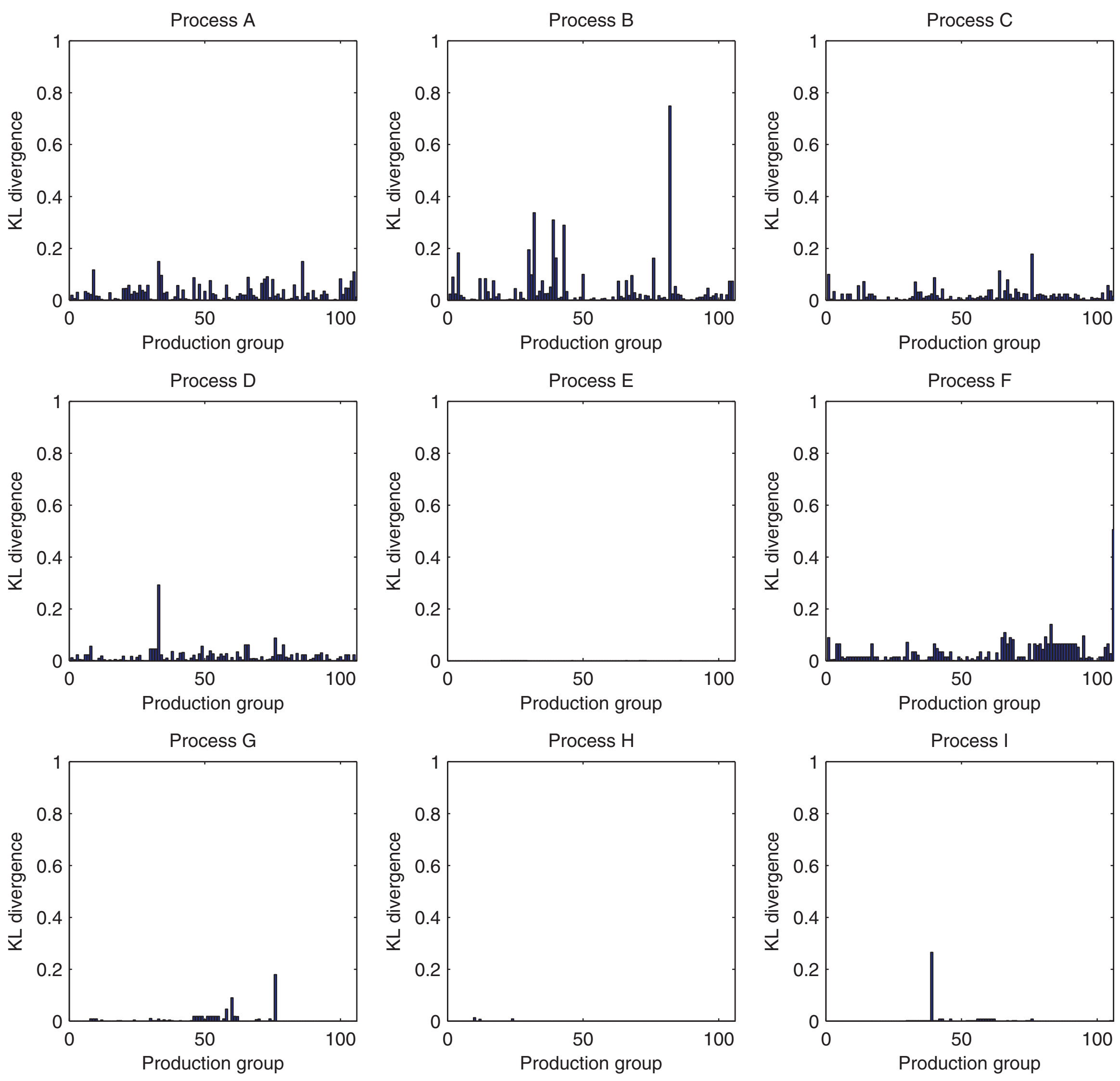


Fig. 19. KL divergences between the predicted and true probability distributions of production loads given customer demands and operating conditions in case 2.

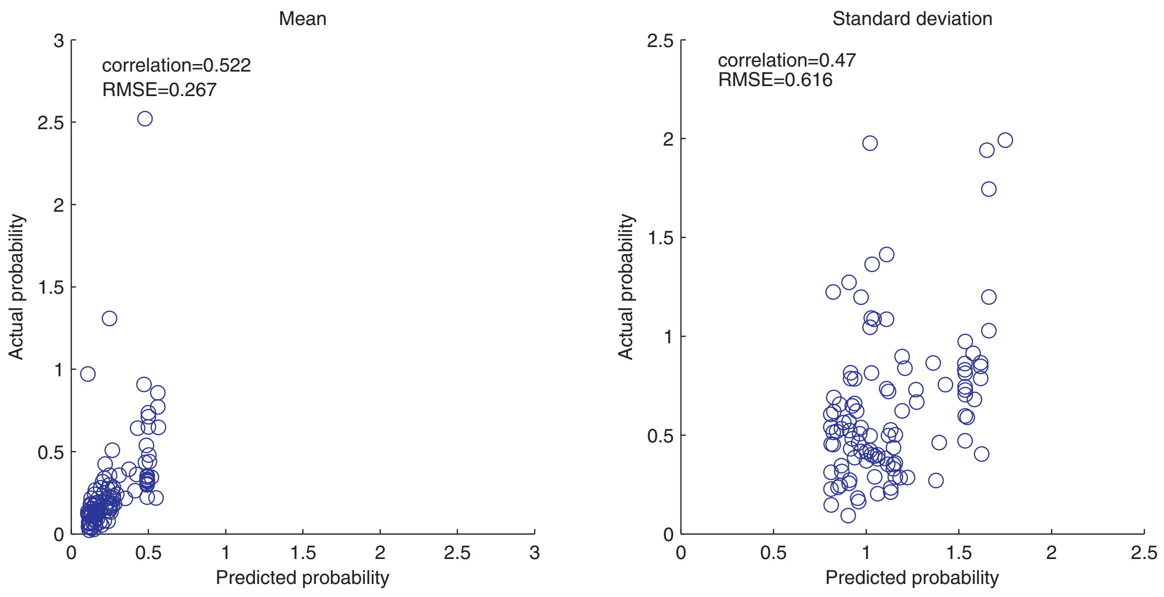


Fig. 20. Scatter plots of mean and standard deviation of production time given customer demands and operating conditions using the first Bayesian network in case 2.

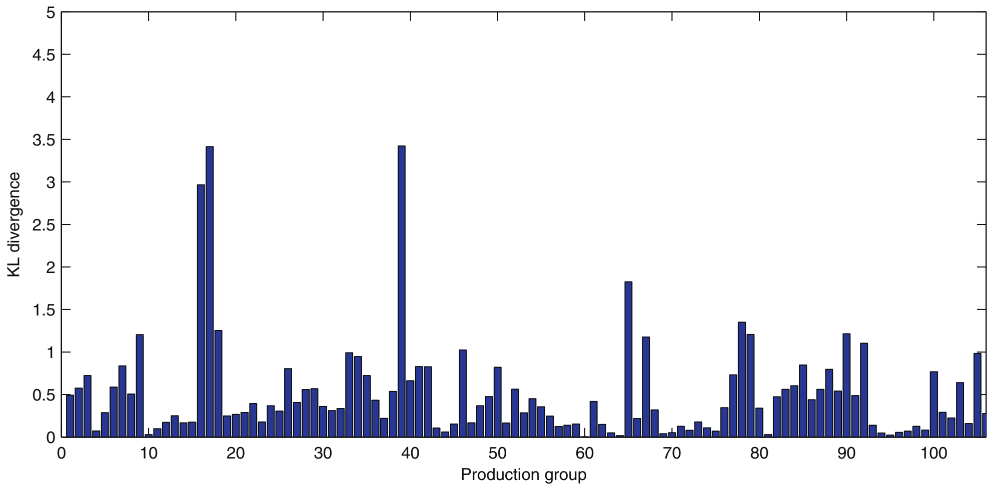


Fig. 21. KL divergences between the predicted and true probability distributions of production time given customer demands and operating conditions using the first Bayesian network in case 2.

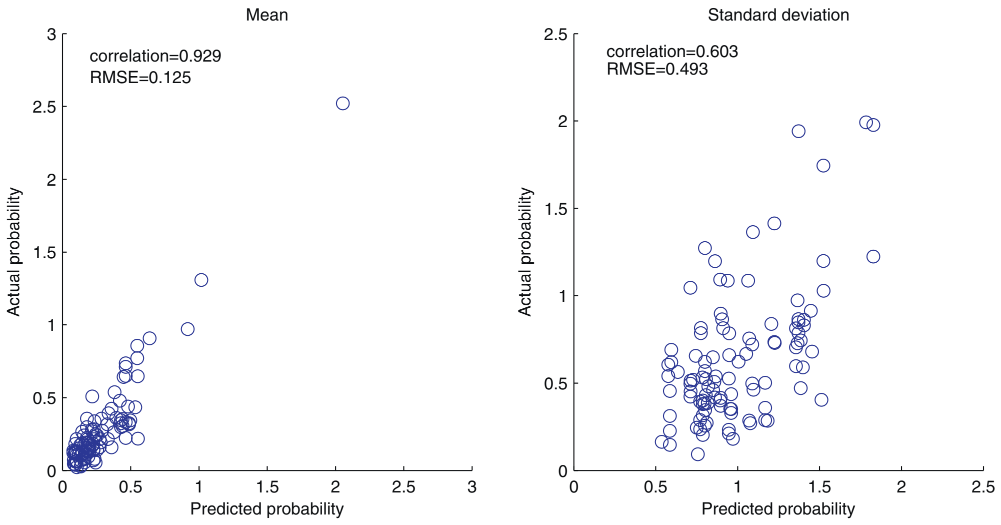


Fig. 22. Scatter plots of mean and standard deviation of production time given customer demands and operating conditions using the second Bayesian network in case 2.

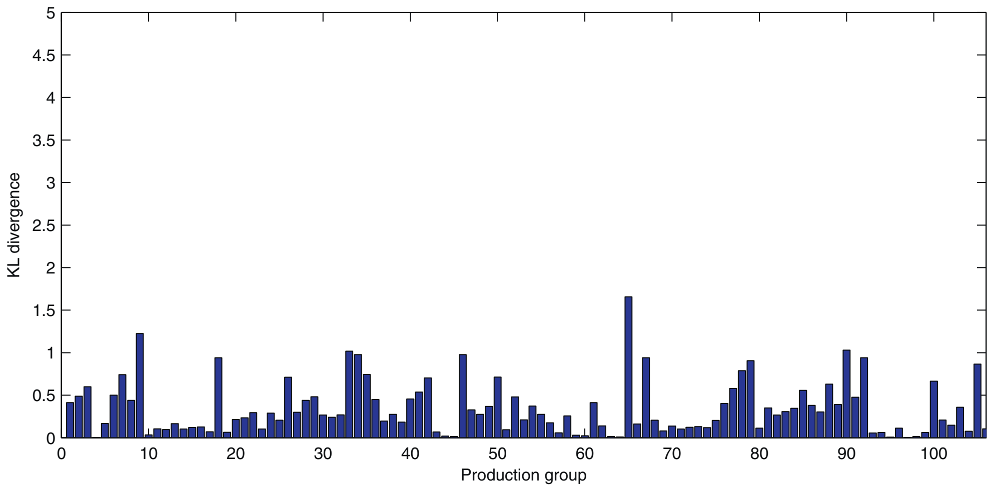


Fig. 23. KL divergences between the predicted and true probability distributions of production time given customer demands and operating conditions using the second Bayesian network in case 2.

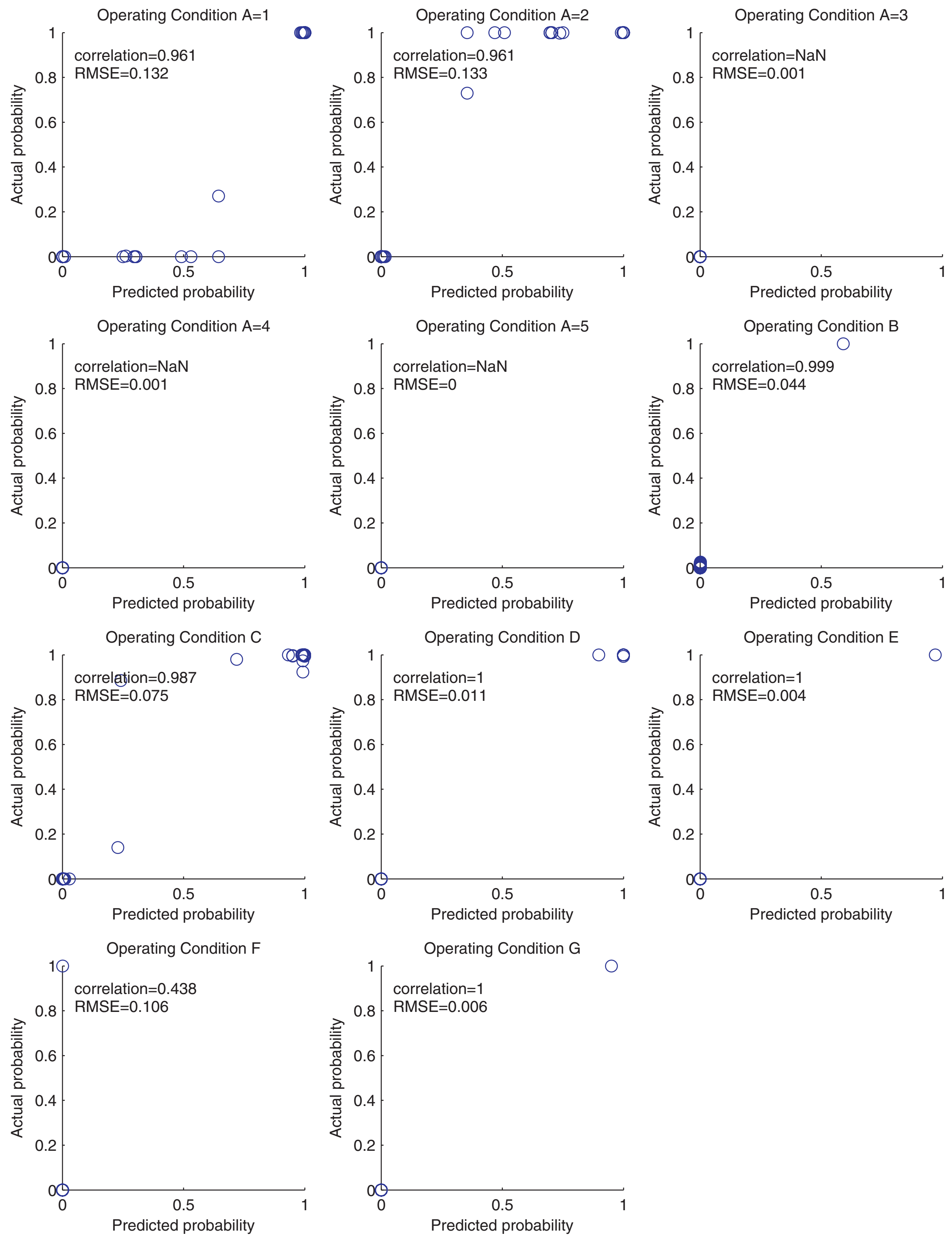


Fig. 24. Scatter plots of probability of each cardinality of operating conditions given customer demands and production loads in case 3.

the probability distributions of the production loads and the process time given all information except for these target variables. Such situation often arises at the production scheduling stage. Similarly to the previous test scenario, the types of inference is causal reasoning and we make use of the same decision-tree structured CPTs as the previous test case. Fig. 18 shows the scatter plots between the predicted and actual probability of the production loads. In addition, their KL divergences are provided in Fig. 19. It can be observed that the predicted and true probability distributions are almost same. In particular, the accuracy of the predicted distributions of loads of process D, process E, process F, process G, process H and process I loads are significantly improved compared to the first test scenario. All of their RMSEs are less than 0.1 and the correlations are also much higher than those in the first test scenario. This is mainly because we know the additional values of the operating conditions that are considered having a significant

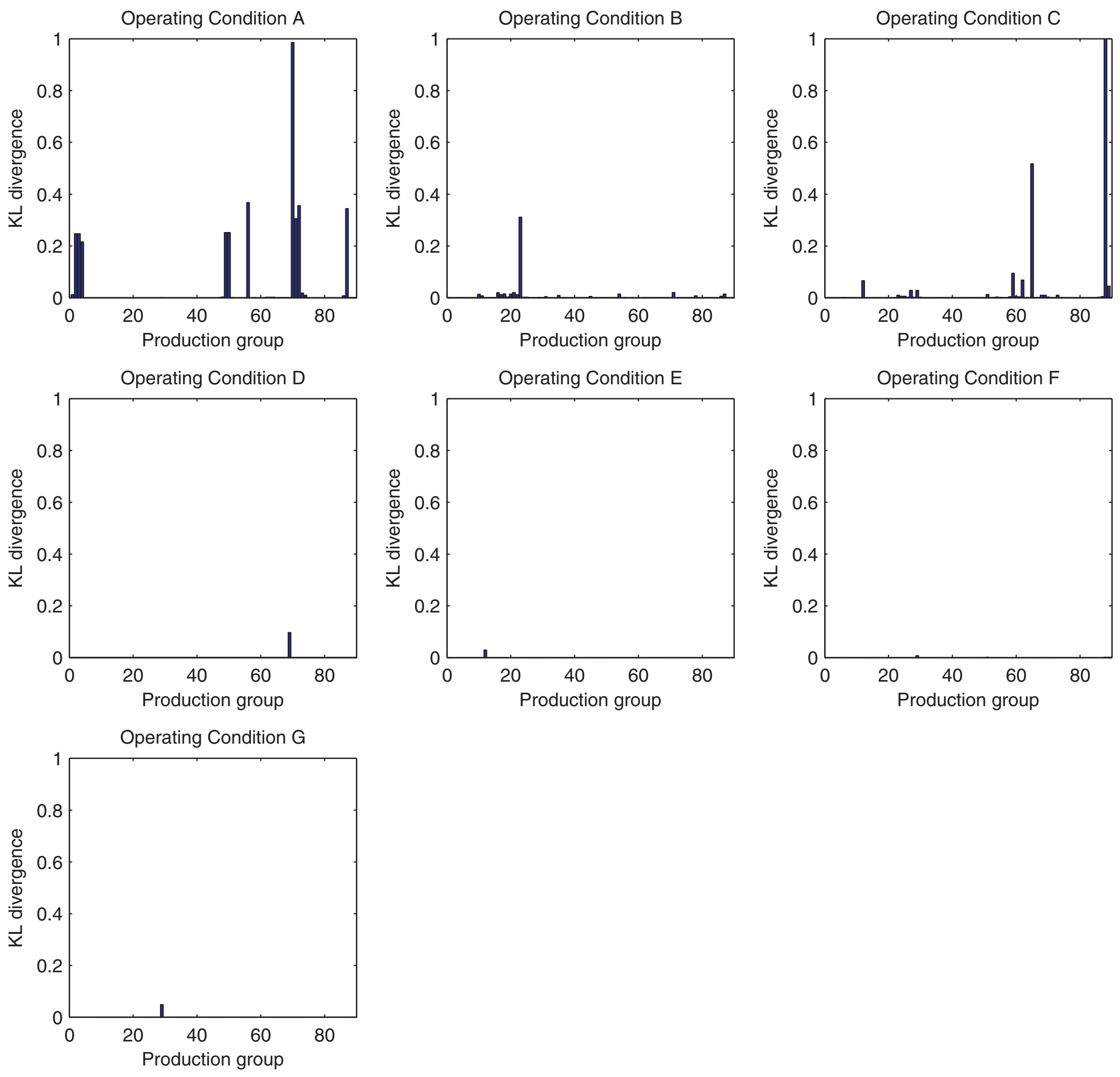


Fig. 25. KL divergences between the predicted and true probability distributions of operating conditions given customer demands and production loads in case 3.

influence on these production loads. However, in spite of the accurate prediction of production loads, the means of the predicted probability distributions of the production times are not improved as shown in Figs. 20 and 21.

The second type of Bayesian network shown as Fig. 11 is also used to predict the production time. The scatter plots between the predicted and actual parameters of means and standard variances and their KL divergences are given in Figs. 22 and 23. One can readily see that the KL divergences between the predicted and actual probability distribution are smaller than all the above results. In particular, the predicted means are very close to the actual ones. As for the accuracy of the predicted standard variances, the second Bayesian network model can accurately predict the standard deviations in most production groups. In some production groups, the differences between the predicted and actual standard deviations are somewhat large. In order to improve further, it may be desirable to add other nodes of variables such as the amount of in-process inventory, the number of workers and the failure rate of each machine in Bayesian networks, which should have an impact on the process time.

All the results indicate that even though our Bayesian network includes large domain of discrete variables, the causal reasoning can be carried out by using the proposed decision-tree structured conditional probability table representations. Moreover, a single Bayesian network model can carry out the most plausible reasoning and its accuracy depends on using the values of known variables. This means that we can avoid having multiple specific models tailored for specific purposes. Therefore, the proposed Bayesian network model satisfies the properties required for plate steel production scheduling and planning.

6.4. Prediction of operating conditions

Up to now we have dealt with prediction of the probability distributions of the production loads and the total production time. In this last test case, we assume that the values of the customer demands and production loads are known while the operating conditions are unknown. Such situation arises when we want to analyze the relationship between the production loads and the operating conditions, or we need to design the operating conditions so that both the customer demands and production capacity constraints can be satisfied. Unlike the previous test scenarios, the type of the inference is mixed reasoning and thus the loopy belief propagation algorithm is carried out. Similarly to the previous two scenarios, we evaluate the prediction accuracy for each production of steel plates with the same customer demands and production

Table 10
Average of execution time for inference (s).

Case No.	Reasoning type	First model	Second model
1	Causal	4.33	0.85
2	Causal	4.06	0.56
3	Mixed	1.52	

loads. We select the production groups containing more than 100 plates to compute the true probability distributions, which are then used for comparison with the predicted probability distributions. The number of production groups that are used in this comparison is 90. The probability of cardinality of the predicted and true operating conditions is shown in Fig. 24. In addition, their KL divergences are shown in Fig. 25. One can readily see that most KL divergences are small; hence, the proposed inference algorithm can accurately predict the probability distributions of unknown variables in the mixed reasoning. Above results demonstrate that the proposed Bayesian inference method can carry out the mixed reasoning even though Bayesian networks have large domain of discrete variables.

Average execution times for all the above test scenarios are provided in Table 10. Comparing the execution time of the first and the second Bayesian networks, we note that the average execution time of the second model is less than 1[s], which is much smaller than that of the first Bayesian network model. That is because in the first model, the node of total production time is connected to the all nodes of the production loads as their child and thus we need a large number of multiplication of factors to compute the final potential of the production time, which leads to high computational effort since the computational cost of factor multiplication increases with the number of variables contained in the factor. For the mixed reasoning, the average execution time is less than 2[s], which is small enough to use it in practical applications.

7. Conclusion

In this study, we propose the Bayesian network models for prediction of the production loads and the production time in manufacturing of steel plates. In order to identify the network structure from historical process data, we search for the graph such that the maximized Bayesian scores are obtained. Since the network contains both continuous and discrete system variables and some of discrete variables have large cardinalities, the decision-tree structured CPTs based hybrid Bayesian inference technique is proposed. Once the Bayesian network structure is constructed, we compute the context-specific conditional probability tables represented by decision trees from historical process data set by using classification tree algorithm. Then, operators for computing multiplication and marginalization of factors represented by decision-tree structure based CPTs are developed. The Bayesian inference is carried out using belief propagation algorithms with the proposed multiplying and marginalizing operators. The causal reasoning is carried out by means of the sum-product algorithm whose direction is downstream from the top node variables to the bottom nodes variables. Other types of reasoning are carried out by the loopy belief propagation.

Real life steel production process data have been used to examine the effectiveness of the proposed inference algorithms. Even though our Bayesian network contains large domain of discrete variable nodes, the results show that the proposed algorithm can be successfully applied to industrial scale Bayesian networks and it can predict the probability distributions of unobserved variables such as the production loads and the production times in the steel plate production.

Results of this work will be applied to production planning and scheduling in manufacturing of steel plates.

Appendix A. Structure learning

Several kinds of scoring functions have been developed. Most of them have a decomposable score function as follows:

$$\text{score}(\mathcal{G}) = \sum_{i=1}^{I} \text{score}(x_i,\ \text{pa}_{\mathcal{G}}(x_i)) \tag{19}$$

where $\text{pa}_{\mathcal{G}}(x_i)$ are the parent nodes of x_i given graph $\mathcal{G}$. Among various kinds of score functions, the BIC score (for Bayesian information criterion) is widely used; it is defined as

$$\text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}) = \ell(\hat{\theta}_{\mathcal{G}} : \mathcal{D}) - \frac{\text{Log}\, M}{2}\, \text{Dim}[\mathcal{G}] \tag{20}$$

where $\hat{\theta}_{\mathcal{G}}$ denotes the maximum likelihood parameters given a graph $\mathcal{G}$, $\ell(\hat{\theta}_{\mathcal{G}} : \mathcal{D})$ denotes the logarithm of the likelihood function, M is the number of samples, and $\text{Dim}[\mathcal{G}]$ is the model dimension that is equal to the number of parent nodes. The term of model dimension is introduced in the score function in order to handle a tradeoff between the likelihood and model complexity. Since the BIC scores are decomposable, they can be restated as follows:

$$\text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}) = M\sum_{i=1}^{N} \mathbf{I}(x_i;\ \text{pa}_{\mathcal{G}}(x_i)) - M\sum_{i=1}^{N} H(x_i) - \frac{\text{Log}\, M}{2}\, \text{Dim}[\mathcal{G}] \tag{21}$$

where $\mathbf{I}(x_i; \text{pa}_{\mathcal{G}}(x_i))$ is the mutual information between x_i and $\text{pa}_{\mathcal{G}}(x_i)$, and $H(x_i)$ is the marginal entropy of x_i.

Score decomposability is important property for network structure learning because if the decomposability is satisfied, a local change such as adding, deleting or reversing an edge in the structure does not change the score of other parts of the structure that remain same. It also should be pointed out that the mutual information term grows linearly in M while the model complexity term grows logarithmically, and thus the larger M leads to an obtained graph which better represents data (Koller and Friedman, 2009).

With the score function defined as above, the optimal graph $\mathcal{G}^*$ can be computed as follows:

$$\mathcal{G}^* = \text{argmax}_{\mathcal{G}}\ \text{score}_{\text{BIC}}(\mathcal{G} : \mathcal{D}). \tag{22}$$

Appendix B. Causal reasoning

The computational task of causal reasoning is to estimate the downstream effects of factors given the evidence of their ancestor variables. In our example, causal reasoning predicts the probability distributions of production loads and total production time from the evidence of customer demands, operating conditions or both. Conditional probability can be computed through a local message passing algorithm known as sum-product algorithm in downstream direction from the root nodes to the leaf nodes. Since our Bayesian network contains both discrete and continuous variables, a final potential $\tilde{P}_{\Phi}(\mathbf{C}_i)$ of factor $\mathbf{C}_i = \mathbf{A}_i \cup \mathbf{X}_i$ is computed using sum and integral operators as follows:

$$\tilde{P}_{\Phi}(\mathbf{C}_i) = \int \sum_{\mathbf{A}_i - A_i} \psi_i \prod_{k \in \text{pa}(i)} P_{\Phi}(\mathbf{C}_k) dx_1 dx_2 \ldots dx_M \tag{23}$$

where:

- ψ_i is the initial potential of $\mathbf{C}_i$, $P_{\Phi}(\mathbf{C}_k)$ is the normalized final potential of $\mathbf{C}_k$,
- $x_m \in \{\mathbf{X}_i - X_i\}$ are variables that we integrate out, and
- X_i or A_i is the child node of the factor $\mathbf{C}_i$ where if the child node variable is continuous, $A_i = \emptyset$, otherwise $X_i = \emptyset$.

First, we compute the initial potentials ψ_i for all factors. Each node in the Bayesian network is assigned one factor and thus the number of factors is equal to the number of nodes. A factor belonging to a set of top root nodes is assigned a singleton factor that contains a variable of the corresponding node. Since evidence is given to all singleton factors in the causal reasoning, their initial and final potentials have cardinality of one and their probabilities are always 1. Note that a factor corresponding to a node that is not placed on the top root node is assigned a non-singleton factor that contains a variable of the corresponding node and its all parent nodes. Therefore, the initial potentials for non-singleton factors are equal to the conditional probability distributions of its corresponding variable given its parent node variables. After computing initial potentials of all factors, the sum-product algorithm proceeds in the downstream direction from the top factors to the bottom factors in order to compute the final potentials of all factors. The causal reasoning algorithm searches for the non-singleton factor for which all parent nodes are assigned final potentials and then computes the final potential for that non-singleton factor by using Eq. (23). Since the final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ in Eq. (23) is not normalized, we need to normalize it and obtain $P_\Phi(\mathbf{C}_i)$ every time after computing the final potentials. This algorithm continues computing final potentials using the sum-product algorithm until final potentials of all factors are computed. Since the computation of final potentials is carried out from top node factors to bottom node factors, the probability propagation directions are the same as the arcs in Bayesian networks. The computed final potentials $P_\Phi(\mathbf{C}_k)$ are equal to the conditional probability distributions of the corresponding variable given the evidence.

Appendix C. Diagnostic and intercausal reasoning

While causal reasoning is carried out by simply applying the sum-product algorithm from the top nodes to the bottom nodes, conditional probability for diagnostic, intercausal and mixed reasoning can be computed by iteratively carrying out the sum-product algorithm until convergence in the loopy belief propagation (LBP). LBP schemes use a cluster graph rather than a clique tree that is used for VE (Murphy et al., 1999). Since the constraints defining the clique tree are indispensable for exact inference, the answers of message passing scheme are approximate queries in LBP.

First, we create the singleton factors for all nodes and then each singleton factor is assigned an initial potential of the prior probability of the corresponding variable. Also, we create the intermediate factors for all nodes that have parent nodes and then each factor is assigned an initial potential of the conditional probability distribution of the corresponding variable given its parent variables. Therefore, the number of factors is larger than the number of nodes. Given the cluster graph, we assign each factor ϕ_k to a cluster $\mathbf{C}_{\alpha(k)}$ so that $\mathrm{scope}(\phi_k) \subseteq \mathbf{C}_{\alpha(k)}$. Then the initial potentials ψ_i can be computed as follows:

$$\psi_i = \prod_{k:\alpha(k)=i} \phi_k. \tag{24}$$

In LBP, one factor communicates with another factor by passing messages between them. The messages have canonical form representation and their potentials are initialized as $\{K, \mathbf{h}, g\} = \{1, 0, 0\}$. A message from cluster $\mathbf{C}_i = \mathbf{A}_i \cup \mathbf{X}_i$ to another factor $\mathbf{C}_j = \mathbf{A}_j \cup \mathbf{X}_j$ is computed using sum and integral operators as follows:

$$\delta_{i\to j} = \int \sum_{\mathbf{A}_i - \mathbf{S}_{i,j}} \psi_i \prod_{k\in \mathrm{Nb}_i - \{j\}} \delta_{k\to i} dx_1 dx_2 \ldots dx_M \tag{25}$$

where:

- $x_m \in \{\mathbf{X}_i - \mathbf{S}'_{i,j}\}$ are integrated variables,
- $\mathbf{S}_{i,j} = \mathbf{A}_i \cap \mathbf{A}_j$ is the sepset of discrete variables,
- $\mathbf{S}'_{i,j} = \mathbf{X}_i \cap \mathbf{X}_j$ is the sepset of continuous variables, and
- Nb_i is the set of indices of factors that are neighbors of $\mathbf{C}_i$.

The messages are updated by using Eq. (25) in accordance with some message passing schedule until the canonical parameters of all messages are converged. Using all incoming messages, a final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ can be computed by multiplying them with the initial potential as follows:

$$\tilde{P}_\Phi(\mathbf{C}_i) = \psi_i \prod_{k\in \mathrm{Nb}_i} \delta_{k\to i}. \tag{26}$$

Similar to the causal reasoning, the final potential $\tilde{P}_\Phi(\mathbf{C}_i)$ needs normalization.

In order to carry out the LBP algorithm, we have to create a cluster graph in advance such that it satisfies the family preservation property. For this purpose, the *Bethe cluster* graph which is a bipartite graph and holds the family preservation property is widely used. The first layer of the Bethe cluster graph consists of large clusters $\mathbf{C}_k = \mathrm{scope}(\phi_k)$ while the second layer consists of singleton clusters X_i. Then, edges are placed between a large cluster $\mathbf{C}_k$ and a singleton cluster X_i if $X_i \in \mathbf{C}_k$. Fig. 8 shows an example of the Bethe cluster graph. In this example, the Bethe cluster graph has 6 singleton clusters $\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}$ and 3 large clusters $\{A, B, D\}, \{B, C, E\}, \{D, E\}$. For instance, due to $\{B\} \in \{A, B, D\}$ and $\{B\} \in \{B, C, E\}$, the edges between cluster 2 and 7 and between cluster 2 and 8 are placed.

Appendix D. Belief propagation with evidence

When initializing the factor ψ_i given the discrete evidence $e \in E \cap \Delta$, we delete all sets of canonical parameters K, $\mathbf{h}$, g in the factor ψ_i that conflict the evidence e.

Next, let us consider initializing the factor ψ_i given the continuous evidence $e \in E \cap \Gamma$. If the factor does not have any discrete variables, a canonical form of the factor ψ_i is reduced to a context representing the evidence e. In Eq. (16), we set $\mathbf{Y} = y$ with y being the value of the evidence e, then the new canonical form given continuous evidence is described as follows (Lauritzen, 1992):

$$\begin{aligned} K' &= K_{\mathbf{XX}} \\ \mathbf{h}' &= \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}} y \\ g' &= g + \mathbf{h}_{\mathbf{Y}}^T y - \frac{1}{2} y^T \mathbf{K}_{\mathbf{YY}} y. \end{aligned} \tag{27}$$

If the factor has discrete variables and does not have a continuous variable except for scope(E), the canonical parameter g_{a_i} is modified for each instantiation a_i as follows:

$$g_{a_i} = -\frac{1}{2} y K_{a_i} y + \mathbf{h}_{a_i}^T y + g_{a_i}. \tag{28}$$

The canonical parameters K_{a_i} and h_{a_i} become vacuous because the new factor contains no continuous variable. If the factor contains both discrete and continuous variables except for scope(E), the parameters of the canonical form are computed for each instantiation a_i using Eq. (27). After modifying all factors so that all of them are consistent with the evidence, the reasoning algorithms mentioned above can be carried out.

References

Ashayeria J, Heutsa RJM, Lansdaalb HGL, Strijbosch LWG. Cyclic production-inventory planning and control in the pre-deco industry: a case study. Int J Prod Econ 2006];103:715–25.

Bishop CM. Pattern recognition and machine learning. New York, NY: Springer-Verlag; 2006].

Boutilier C, Friedman N, Goldszmidt M, Koller D. Context-specific independence in Bayesian networks. In: Proceedings of UAI-96; 1996]. p. 115–23.

Breiman L, Friedman JH, Olshen RA, Stone CG. Classification and regression trees. Belmont: Wadsworth International Group; 1984].

Caputo B, Sim K, Furesjo F, Smola A. Appearance-based object recognition using SVMs: which kernel should I use? In: Proceedings of NIPS workshop on Statistical methods for computational experiments in visual processing and computer vision; 2002].

Chickering DM. Learning Bayesian networks is NP-complete. In: Learning from data: artificial intelligence and statistics. Springer-Verlag; 1996] [chapter 12].

Chickering DM, Heckerman D, Meek C. A Bayesian approach to learning Bayesian networks with local structure. In: Proceedings of UAI-97; 1997]. p. 80–9.

Cobb BR, Rumi R, Salmeron A. Bayesian network models with discrete and continuous variables. Berlin: Springer-Verlag; 2007].

DesJardins M, Rathod P. Learning structured Bayesian networks: combining abstraction hierarchies and tree-structured conditional probability tables. Comput Intell 2008];24:1–22.

Friedman N, Goldszmidt M. Learning Bayesian networks with local structure. In: Proceedings of UAI-96; 1996]. p. 252–62.

Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res 1986];13:533–49.

Harjunkoski I, Grossmann IE. A decomposition approach for the scheduling of a steel plant production. Comput Chem Eng 2001];25:1647–60.

Koller D, Friedman N. Probabilistic graphical models: principles and techniques. MIT Press; 2009].

Koller D, Lerner U, Angelov D. A general algorithm for approximate inference and its application to hybrid Bayes nets. In: Proceedings of UAI-99; 1999]. p. 324–33.

Kozlov A, Koller D. Nonuniform dynamic discretization in hybrid networks. In: Proceedings of UAI-97; 1997]. p. 314–25.

Lauritzen S. Propagation of probabilities, means and variances in mixed graphical association models. J Am Stat Assoc 1992];87:1098–108.

Lauritzen SL, Jensen F. Stable local computation with conditional Gaussian distribution. Stat Comput 2001];11:191–203.

Lauritzen SL, Wermuth N. Graphical models for associations between variables, some of which are qualitative and some quantitative. Ann Stat 1989];17:31–57.

Moon S, Hrymak AN. Scheduling of the batch annealing process-deterministic case. Comput Chem Eng 1999];23:1193–208.

Moral S, Rumi R, Salmeron A. Mixtures of truncated exponentials in hybrid Bayesian networks. Berlin: Springer-Verlag; 2001].

Murphy KP, Weiss Y, Jordan MI. Nonuniform dynamic discretization in hybrid networks. In: Proceedings of UAI-99; 1999]. p. 467–75.

Nishioka K, Mizutani Y, Ueno H, Kawasaki H, Baba Y. Toward the integrated optimization of steel plate production process: a proposal for production control by multi-scale hierarchical modeling. Synthesiology 2012];5:98–112 [in Japanese].

Pearl J. Probabilistic reasoning in intelligent systems: networks of plausible inference. San Francisco: Morgan-Kaufmann; 1988].

Poole D, Zhang NL. Exploiting contextual independence in probabilistic inference. J Artif Intell Res 2003];18:263–313.

Rumi R, Salmeron A. Approximate probability propagation with mixtures of truncated exponentials. Int J Approx Reason 2007];45:191–210.

Santos CA, Spim JA, Garci A. Mathematical modeling and optimization strategies (genetic algorithm and knowledge base) applied to the continuous casting of steel. Eng Appl Artif Intel 2003];16:511–27.

Sharma R, Poole D. Efficient inference in large discrete domains. In: Proceedings of UAI-2003; 2003]. p. 535–42.

Tang L, Liu J, Rong A, Yang Z. A mathematical programming model for scheduling steel making continuous casting production. Eur J Oper Res 2000];120:423–35.

Yuan C, Druzdzel MJ. Hybrid loopy belief propagation. In: Proceedings of the third European workshop on probabilistic graphical models (PGM'06); 2006]. p. 317–24.

Chapter 7

Planning and scheduling of steel plates production.

Part II: Scheduling of continuous casting

Contents lists available at ScienceDirect
Computers and Chemical Engineering
journal homepage: www.elsevier.com/locate/compchemeng

Planning and scheduling of steel plates production. Part II: Scheduling of continuous casting

Junichi Mori, Vladimir Mahalec*

Department of Chemical Engineering, McMaster University, 1280 Main St. West, Hamilton, ON L8S 4L8, Canada

A R T I C L E I N F O

Article history:
Received 7 August 2015
Received in revised form 27 January 2016
Accepted 28 January 2016
Available online xxx

Keywords:
Steel plates production planning and scheduling
Long term scheduling
Two level algorithm
Parallel simulated annealing with leaping

A B S T R A C T

Production planning and scheduling in the steel industry are challenging problems due to large number of products being produced. This work deals with scheduling of the continuous casting of steelmaking, i.e. determination of the number of fixed capacity pots of each grade of steel and the charge sequence in each casting machine. Since a huge number of binary variables make the full-space model mixed-integer linear programming model computationally intractable, we propose a two-level algorithm. At the top level, we solve the planning problem which determines the number of pots of each grade for every planning period by solving the relaxed mixed integer linear model. At the lower level, the scheduling problem is solved by an algorithm which combines ideas from parallel simulated annealing and shuffled frog-leaping algorithm. Real-world steel-plate production data are utilized to demonstrate the effectiveness of the proposed algorithm.

1. Introduction

Steel plates are high variety low-volume products manufactured on order since they are used to produce thousands of different products, e.g. parts of car bodies, housing enclosures for machinery, etc. Frequently a steel plant produces several hundred different SKUs within one or two weeks. Such large number of different products (SKUs) makes it difficult to compute schedules which will increase profit, reduce production costs and material consumption, satisfy the customer specifications and meet the shipping due dates. Simplified process flow of the steel plates production is shown in Fig. 1. A *charge* is the basic unit of steel making production and it consists of the same steel grade. In the continuous casting machine, the casting from charge to charge is carried out to transform molten steel into solid. The first solid form of the steel produced in these facilities is cut up into smaller solid form called *slab*. These slabs are rolled and cut into plates of the correct thickness and properties. Finally, these plates are handled in the finishing and inspection lines for satisfying the customer demands, and then they are sent to the distribution warehouse.

There are many scheduling and planning tasks to be completed in the successful production in steel mills. Among them, scheduling of steelmaking continuous casting (SCC) of steel slabs followed by rolling to steel plate is one of the most important tasks. This is due to the fact that SCC contributes the largest portion of the material loss in production; in addition, downstream production schedules such strongly depend on the schedules of the SCC process.

Schedules for steelmaking continuous casting (SCC) production should meet the following criteria:

(1) Each plate is completed before its due date.
(2) Amounts of leftover and contaminated steel which are total loss are minimized.
(3) Continuous casting utilization is maximized (i.e. the residual capacity at the continuous casting stage is minimized).
(4) Production constraints and inventory capacities at the downstream production stage are satisfied.

At the same time, scheduling of SCC must take into account the performance of the downstream processes. Since it is imperative to take into account the customer due date and production capacity of downstream processes such as finishing and inspection lines, it is necessary to predict as accurately as possible the production loads and production times and then to incorporate these prediction models into the mathematical models for optimization. In order to obtain the accurate knowledge of production loads and production times, in the previous work we developed Bayesian network based prediction models for estimation of probability distributions of production loads and process times from the observed variables such as customer demands and operating

* Corresponding author. Tel.: +1 905 525 9140x26386.
E-mail address: mahalec@mcmaster.ca (V. Mahalec).

http://dx.doi.org/10.1016/j.compchemeng.2016.01.020
0098-1354/© 2016 Elsevier Ltd. All rights reserved.

Nomenclature

Indices

I	index of grades
U	index of casts
V	index of charges
K	index of time period (day)
L	index of finishing and inspection processes

Parameters

W	cost coefficient in the objective function
$D_{i,k}$	demand of grade I on time period k
$C_{i,k}$	contamination cost when grade i is assigned after grade i'
P	maximum amount of charges
$MI_{\max}$	maximum inventory
$MI_{\min}$	minimum inventory
$Load_{l,i,m}$	production load of process unit l, grade i, and elapsed day m
$Load_l^{\max}$	production capacity of process unit l

Binary variables

$Y_{i,u,v,k}$	binary variables to determine if grade i is assigned in charge v of cast u on time period k
$Z_{i,i',u,v,k}$	binary sequence variables between grade i is assigned in charge v of cast u on time period k

Integer variables

$Ur_{i,k}$	integer variables for the number of charges of grade i on time period k

Continuous variables

$X_{i,u,v,k}$	continuous variables for amount of grade i, cast u, charge v on time period k
$M_{i,k}^{in}$	continuous variables for amount of grade i on time period k
$M_{i,k}^{out}$	continuous variables for demand of grade i on time period k
$M_{i,k}^{open}$	continuous variables for opening volume of grade i on time period k
$M_{i,k}^{close}$	continuous variables for closing volume of grade i on time period k
$Q_{l,k}$	continuous variables for production loads of process unit l on time period
$S1_{i,u,v,k}^{+}$	positive slack variables for production capacity at the continuous casting stage
$S1_{i,u,v,k}^{-}$	negative slack variables for production capacity at the continuous casting stage
$S3_{i,k}^{+}$	positive slack variables for maximum inventory capacity
$S3_{i,k}^{-}$	negative slack variables for minimum inventory capacity
$S4_{l,k}^{-}$	negative slack variables for production capacity at the finishing and inspection processes
$Xr_{i,k}$	continuous variables for amount of grade i on time period k
$Sr1_{i,k}^{+}$	positive slack variables for production capacity at the continuous casting stage
$Sr1_{i,k}^{-}$	negative slack variables for production capacity at the continuous casting stage
$S_{pr,L3}^{+}(j,n)$, $S_{pr,L3}^{-}(j,n)$	positive and negative slack variables for the product inventories

conditions (Mori and Mahalec, 2015). In this paper, we make use of the most likely production times computed via Bayesian network to optimize SCC scheduling of steel-plate production.

Since SCC process plays an important role in steelmaking industries, many optimization models and approaches have been proposed. In general, SCC production scheduling problems should determine charge sequencing in each casting machine (first level scheduling) and timing of the charges on continuous casting machines (second level scheduling).

Second level scheduling has been a subject of work by several research groups. Atighehchian et al. (2009) scheduled the second level by an algorithm (HANO) which is a combination of ant colony optimization (ACO) and non-linear optimization. Their objective was to reach production continuity, to increase productivity, and to cut down the total production costs. Just-In-Time (JIT) nonlinear models which consider punctual delivery and production operation continuity have also been proposed to eliminate the machine conflicts given cast sequences (Tang et al., 2000). Discrete-time mixed integer linear programming (MILP) formulation for scheduling of the SCC process has also been developed (Tang et al., 2002). Above methods find optimal timing of the charges on continuous casting machines while the charge sequence in each cast is assumed to be known a prior and determined from first level scheduling.

Although the second level scheduling problems have been well studied, few studies have dealt with the first level scheduling because it is intractable to solve the first level scheduling of the SCC process for a long time horizon (e.g. one week). Therefore, the first level schedules are usually determined empirically by skillful persons or computed based on the rule-based approaches.

One approach to solving the first level scheduling are decomposition approaches. For instance, Harjunkoski and Grossmann (2001) tackled the first level scheduling of a steel production using mathematical programming methods based on a decomposition scheme that generates smaller sub-problems. However, the presented method does not consider the individual charge which is a unit of production consisting only one grade of steel. Therefore, the method is not applicable for scheduling of general SCC processes. In addition, all of the existing approaches which deal with both the first and the second level scheduling methods do not take into account the production capacities of the downstream processes at all. As we discussed in the previous work (Mori and Mahalec, 2015), the production capacities of the downstream processes such as finishing and inspection lines should be taken into account while scheduling, because the downstream processes often create a bottleneck for the entire steel-plate production.

In order to overcome these limitations, we propose a novel approach to the first level scheduling of the SCC processes for steel plate production. Most likely processing times and production capacities for the steel plate manufacturing are first estimated from the Bayesian network of the process. These are then used in the scheduling of continuous casting. The remaining obstacle to solving this scheduling problem is a large number of binary variables that are needed to represent accurately the process. Therefore, a full space model mixed-integer linear programming (MILP) is not feasible for industrial applications due to excessive computational times and frequent failures to find even a single feasible solution within reasonable execution times.

In order to overcome this limitation, we propose a decomposition approach where the full-space model is divided into two levels: (i) top level which determines the number of pots per grade for each day by solving the relaxed mixed-integer linear model (MILP) that does not take into consideration the sequence-related penalties (e.g. contaminated steel) and (ii) lower level which optimizes the cast sequencing while taking into consideration the sequence penalties in the continuous casting machine by using meta-heuristic methods. Relaxed MILP model is obtained by

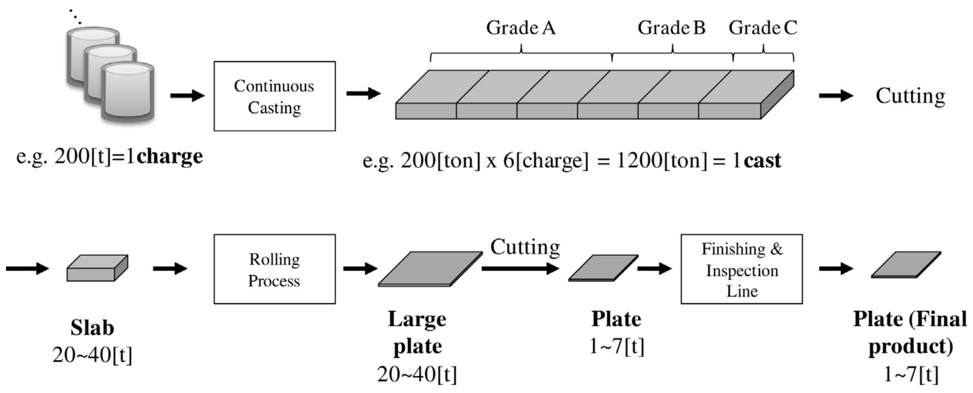


Fig. 1. Process flow of steel-plate production system.

disregarding the fact that switching from one grade of steel to another causes mixing of these two grades at the boundary between them, i.e. contamination of a part of the plate material. In order to ensure that each pot is filled to capacity, since pots are of fixed size, we formulate continuous constraints and penalize differences between the amount of material in the pot and its capacity.

Since meta-heuristic approaches are employed in this work, our approach is not guaranteed to yield the global solutions, but they assure computation of good schedules within reasonable execution times even for very large problems. Our assumption is that in the industrial practice schedules would be computed on a daily basis for a fixed length of the scheduling horizon. The scheduling horizon needs to be sufficiently long (e.g. one week) to avoid computation of schedules which may be optimal over e.g. two days but may be far from the optimum over longer horizons. For this reason we have not used the rolling horizon approach where one would schedule, e.g. two days, then move forward by two days and schedule them, etc.

Many kinds of meta-heuristic approaches have been proposed. The approaches fall into two categories, which are: single solution and population-based approaches. The single solution approaches such as simulated annealing (Aarts and Korst, 1989), tabu search (Glover, 1986) and iterated local search (Lourenco et al., 2003) focus on modifying and improving a single candidate solution. Meanwhile, the population-based approaches such as genetic algorithm (GA) (Lawrence, 1991), particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), ant colony optimization (ACO) (Marco et al., 2006) and shuffled frog-leaping algorithm (Eusuff et al., 2006) maintain and improve multiple-candidate solutions and often generate new solutions by population characteristics. In this work we introduce a new parallel meta-heuristic algorithm to schedule continuous casting.

The organization of this article is as follows. Section 2 provides the problem statement while Section 3 describes a Bayesian network based mixed-integer linear programming (MILP) model for the scheduling of SCC. In Section 4 we introduce the two-level scheduling approach that divides the original full space model into two level scheduling models. In order to solve the lower level scheduling problem efficiently, we propose the novel parallel SA that can avoid local optima as much as possible by communicating states more effectively than the traditional parallel SA by using concepts from shuffled frog algorithm. The presented methods are applied to the real-world steel plate production data in Section 5. Finally, the conclusions are presented in Section 6.

2. Problem statement

We assume that the steel plate manufacturing facility has received orders for the next 5–10 days or more. Since the number of different steel plate SKU can run into hundreds or thousands, we will group them into grades, based on the qualities of the steel from which they are made of. The task is to determine the amount of products in each charge of each continuous caster over period of 5–10 days, as well as the sequence of charges. This is called *long term scheduling*. Once the amount of products in each charge is determined, each plate is assigned to one slab such that the leftover can be minimized, which is called *slab design problem* (Schaus et al., 2010; Dash et al., 2007). Finally, the second level scheduling computes the timing of the charges on continuous casting machines from the results obtained in the first level scheduling. Our paper focuses on the first level scheduling. The diagram of the entire set of scheduling tasks for steel-plate production is shown in Fig. 2.

Our targeted scheduling problem of SCC for steel-plate production is stated as follows:

Given:

1. A scheduling horizon (e.g. one week).
2. The number of charges per cast (e.g. 6 charges per cast).
3. The number of casts per day (e.g. 3 casts per day).
4. The maximum processing capacity of each finishing and inspection process.
5. The size of charges (e.g. 200 tons).
6. The contamination cost when switching grades between consecutive charges.
7. A set of delivery orders for each grade along the scheduling horizon (e.g. demand profile).

Determine:

1. The amount of products on each charge of each continuous caster.
2. Sequence of charges.
3. The amount of products that each finishing and inspection line should process each day.
4. The inventory profile of products at a distribution warehouse.

While minimizing:

1. The cost of producing contaminated steel caused by the switching grades between consecutive charges.
2. The cost of leftover in each charge.
3. The amount of inventory at a distribution warehouse.

Subject to:

1. Only one steel grade can be used in a single charge. Once it begins charging, the same grade steel must be used and its amount should always be equal to the size of the charge.

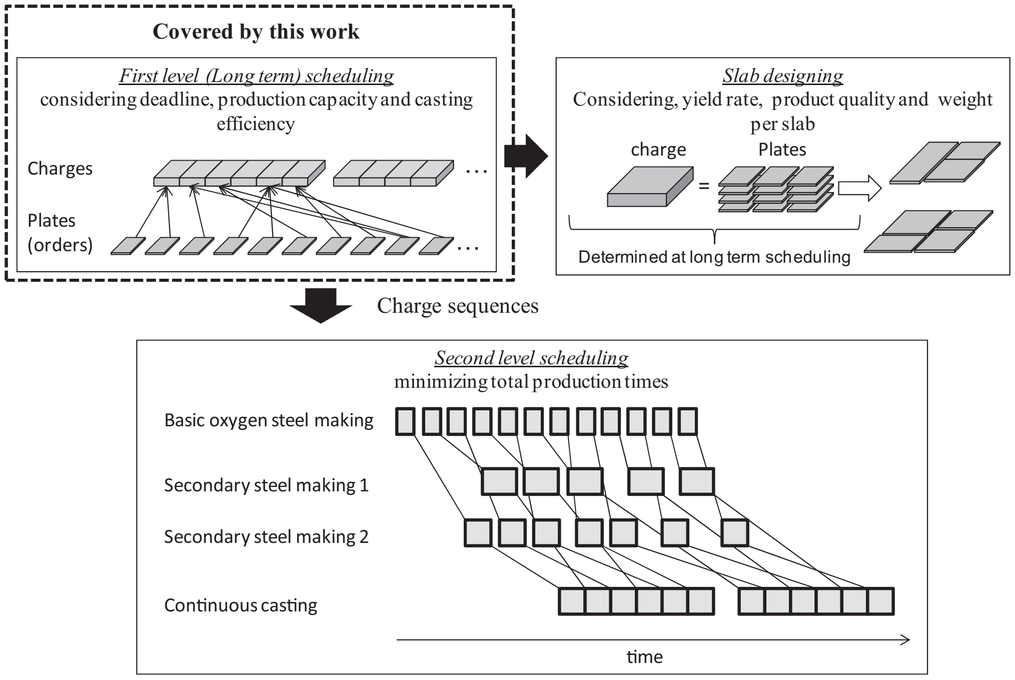

Covered by this work
First level (Long term) scheduling
considering deadline, production capacity and casting efficiency
Charges
Plates (orders)
Slab designing
Considering, yield rate, product quality and weight per slab
charge
Plates
Determined at long term scheduling
Charge sequences
Second level scheduling
minimizing total production times
Basic oxygen steel making
Secondary steel making 1
Secondary steel making 2
Continuous casting
time

Fig. 2. Illustrative diagram of SCC scheduling for steel-plate production.

2. Finishing and inspection processes can handle at most the same amount of steel as its processing capacity.

Assuming:

1. There is only one casting machine for steel plates and it cannot be used for other kinds of products such as steel sheet.
2. Processing order of the finishing and inspection lines is fixed.
3. Variables associated with both customer demands and operating conditions are known.
4. Variables associated with both production loads and process time are unknown.
5. Production loads and process times follow probability distributions are known (they are predicted by the Bayesian network model from variables associated with both customer demands and operating conditions.

3. Bayesian network based scheduling model

3.1. Bayesian network model

Model of the above production process deals with the allocation of grades to each charge such that each plate is completed by its due date, the total leftover in charges and contaminated steel in the continuous casting machine are minimized, and the production and the inventory at the downstream production lines are less than or equal to their capacities. In order to take into consideration the customer due dates and production capacities of the finishing and inspection processes, we need to predict the total production time of each plate. This time will be used for determining its ideal production starting time and the production loads of each plate for each production unit.

Since prediction of the probability distributions of production times for all plates are necessary, we utilize the Bayesian network based prediction model which has been developed in the previous work (Mori and Mahalec, 2015). Based on the Bayesian network model, the total demand $D_{i,k}$ of grade i on the ideal production starting day k and the expectation of production load of grade i at process unit l, elapsed day m can be computed.

Since we assume that the processing order of the finishing line is fixed, without loss of generality the process ordering can be described as {unit 1, unit 2, ..., unit L} with L being the number of the finishing and inspection processes. Gaussian distribution is appropriate for this manufacturing process, as described in part I (Mori and Mahalec, 2015). The parameters of probability distribution $DistA(e; p, l) = N(t; \mu^A_{p,l}, \sigma^A_{p,l})$ of elapsed time e between the times when plate p is started manufacturing and when it is arrived unit l can be computed as follows:

$$\mu^A_{p,l} = \mu^A_{p,l-1} + \mu_{p,l} \tag{1}$$

$$\sigma^A_{p,l} = \sqrt{\sigma^{A2}_{p,l-1} + \sigma^2_{p,l}} \tag{2}$$

with $\mu_{p,l}$ and $\sigma_{p,l}$ being means and standard deviations of production time of plate p at unit l. It should be noted that both of them can be estimated from variables related to the customer demands and operating conditions using the constructed Bayesian network model. The expectation of production load of plate p and unit l and elapsed day m is described below:

$$Expectation(m; p, l) = \int_m^{m+1} Prob(p, l) DistA(e; p, l) de \tag{3}$$

where $Prob(p,l)$ is a probability distribution of production load of plate p at unit l. The probability distribution $DistT(m,p)$ of total production time from starting manufacturing to arriving at the distribution warehouse can be computed as $DistA(m;p, L+1)$.

The proposed methodology for predicting the elapsed time from the start manufacturing is illustrated in Fig. 3. In this example, we assume that there are three units whose process ordering is Unit 1, Unit 2, Unit 3, and a distribution warehouse. First we need to derive the Bayesian network model that can predict probability distributions of the production time. We learn both the network structure and the model parameters from the historical process data. The detailed algorithms are described in the previous work (Mori and Mahalec, 2015). With the constructed Bayesian network model, we infer the probability distirbution of the production times of all units for each plate by using the information of customer demands and operating conditions. Then, we compute the elapsed time for each

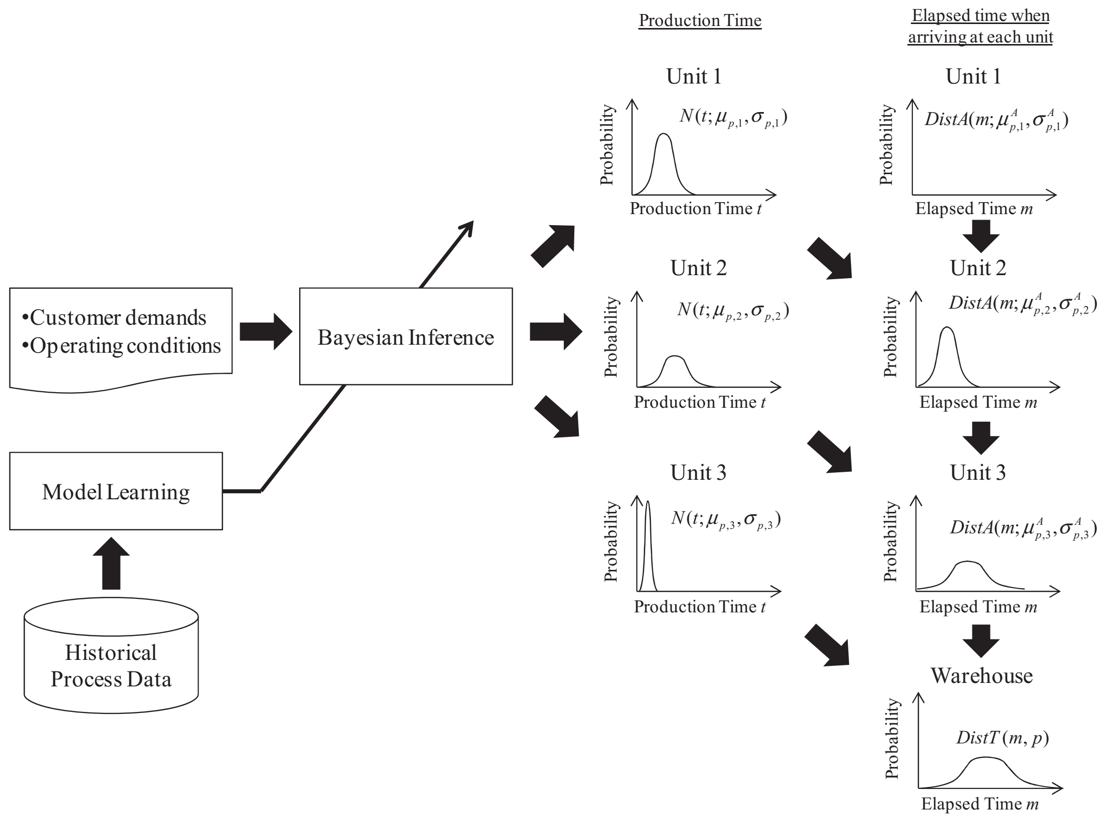

Production Time
Elapsed time when arriving at each unit
Unit 1
Unit 1
Probability
$N(t;\mu_{p,1},\sigma_{p,1})$
Production Time t
$DistA(m;\mu^A_{p,1},\sigma^A_{p,1})$
Elapsed Time m
Unit 2
Unit 2
$N(t;\mu_{p,2},\sigma_{p,2})$
$DistA(m;\mu^A_{p,2},\sigma^A_{p,2})$
•Customer demands
•Operating conditions
Bayesian Inference
Model Learning
Historical Process Data
Unit 3
Unit 3
$N(t;\mu_{p,3},\sigma_{p,3})$
$DistA(m;\mu^A_{p,3},\sigma^A_{p,3})$
Warehouse
$DistT(m,p)$

Fig. 3. Illustrative diagram of predicting the elapsed time when arriving at each unit.

unit between the times when the plate manufacturing is started and when it arrives at the corresponding unit. Finally, the total production time $DistT(m,p)$, which is same as the time from start manufacturing to arriving at the warehouse, can be computed from the probability distribution of both the elapsed time so far and the production time of Unit 3.

Given customer due date tc_p which is given for each plate p as demand profiles, an ideal production starting day at the confidence level of α can be obtained below:

$$ts_p = tc_p - r_p \tag{4}$$

where r_p is the estimated production time satisfying below:

$$\alpha = DistT(m \le r_p; p) \tag{5}$$

Finally, the total demand $D_{i,k}$ of grade i on the ideal production starting day k and the expectation of production load of grade i at process unit l, elapsed day m can be written as follows:

$$D_{i,k} = \sum\nolimits_{p \in grade(i), ts_p = k} weight_p \tag{6}$$

$$Load_{l,i,m} = \frac{1}{N_i} \sum_{p \in grade(i)} Expectation(m; p, l) weight_p \tag{7}$$

where $weight_p$ represents the weight of plate p, grade(i) is the set of plates whose grade is i and N_i is the total weight of all plates belonging to the steel grade i computed as follows:

$$N_i = \sum\nolimits_{p \in grade(i)} weight_p \tag{8}$$

3.2. Full space model MILP

The allocation of grades on each charge can be formulated as follows. The objective function Eq. (9) minimizes the total leftover in charges (f_1^+), the amount of molten steel that violates cast capacity (f_1^-), the amount of contaminated steel (f_2), the amount of steel-plate inventory (f_3), the amount of steel plates which violate maximum inventory capacity (f_3^+), the amount of steel plates which violate the minimum inventory capacity (f_3^-), and the amount of steel plates which violate the capacities of production unit (f_4^-). The values of the coefficients (W_1^+, W_1^-, W_2, W_3, W_3^+, W_3^-, W_4^-) depend on the problem, but generally W_1^-, W_4^- are set to be much greater than the other coefficients to ensure that the final solution satisfies the capacity constraints (if a physical feasible solution exists).

$$\min \; W_1^+ f_1^+ + W_1^- f_1^- + W_2 f_2 + W_3 f_3 + W_3^+ f_3^+ + W_3^- f_3^- + W_4^- f_4^- \tag{9}$$

All constraints considered in the full space MILP are presented below.

The indices, parameters and variables used in this work are shown in the Nomenclature section. In order to take into account the capacity of charge P and penalty of contaminated steel, two binary variables $Y_{i,u,v,k}$ and $Z_{i,i',u,v,k}$ are required. $Y_{i,u,v,k}$ is equal to one if grade i is assigned to charge v of cast u on time period k, and otherwise 0. Defining $X_{i,u,v,k}$ as the variables for the amount of steel of grade i assigned to charge v of cast i on time period k and slack variables $S1^+_{i,u,v,k}$ and $S1^-_{i,u,v,k}$, we can derive Eqs. (10)–(12) which represent the relation among these variables.

$$X_{i,u,v,k} + S1^+_{i,u,v,k} - S1^-_{i,u,v,k} = PY_{i,u,v,k} \quad \forall i, u, v, k \tag{10}$$

$$Z_{i,i',u,v,k} \ge Y_{i,u,v,k} + Y_{i',u,v-1,k} - 1 \quad \forall i, u, k, v \ge 2 \tag{11}$$

$$\sum_i Y_{i,u,v,k} = 1 \quad \forall u, k, v \tag{12}$$

$S1^+_{i,u,v,k}$ is equal to the amount of the leftover in charges while $S1^-_{i,u,v,k}$ becomes positive if the amount of steel products of the corresponding charge is greater than the size of charge, which means that there are no physical infeasible solutions. Then, the objective functions f_1^+, f_1^- and f_2 are presented below:

$$f_1^+ = \sum_i \sum_u \sum_v \sum_k S1^+_{i,u,v,k} \tag{13}$$

$$f_1^- = \sum_i \sum_u \sum_v \sum_k S1^-_{i,u,v,k} \tag{14}$$

$$f_2 = \sum_i \sum_{i'} \sum_u \sum_v \sum_k C_{i,i'} Z_{i,i',u,v,k} \tag{15}$$

where $C_{i,i'}$ is the contamination cost when grade i is assigned after grade i' and its diagonal elements are always zero. Although the variables $Z_{i,i',u,v,k}$ are binary variables, they can be treated as continuous variables because the left-hand side of Eq. (11) must be either −1, 0 or 1 and $Z_{i,i',u,v,k}$ is a positive variable.

The following constraints define the inventory balance for each time period. Eq. (16) means that the amount of input inventory $M^{in}_{i,k}$ is equal to the sum of amount of production $X_{i,u,v,k}$ in the same time period. Eq. (17) simply states that the total demand $D_{i,k}$ of grade i and ideal production time k is equal to the output inventory $M^{out}_{i,k}$. Eq. (18) defines that the closing inventory $MI^{close}_{i,k}$ should be equal to the opening inventory $MI^{open}_{i,k+1}$ on the next time period. Eq. (19) considers that the sum of input inventory $M^{in}_{i,k}$ and opening inventory $MI^{open}_{i,k}$ should be equal to the sum of $M^{out}_{i,k}$ and $MI^{close}_{i,k}$.

$$M^{in}_{i,k} - \sum_u \sum_v X_{i,u,v,k} = 0 \quad \forall i, k \tag{16}$$

$$M^{out}_{i,k} - D_{i,k} = 0 \quad \forall i, k \tag{17}$$

$$MI^{open}_{i,k+1} - MI^{close}_{i,k} = 0 \quad \forall i, k \tag{18}$$

$$M^{in}_{i,k} + MI^{open}_{i,k} - M^{out}_{i,k} - MI^{close}_{i,k} = 0 \quad \forall i, k \tag{19}$$

In order to avoid infeasible solutions, we define slack variables $S3^+_{i,k}$ and $S3^-_{i,k}$ which enables us to state the maximum and minimum inventory constraints as follows:

$$MI_{\max} - MI^{close}_{i,k} + S3^+_{i,k} \geq 0 \quad \forall i, k \tag{20}$$

$$MI_{\min} - MI^{close}_{i,k} - S3^-_{i,k} \leq 0 \tag{21}$$

$S3^+_{i,k}$ is equal to the amount of the inventory of grade i and time period k, while $S3^-_{i,k}$ is zero if there is no delivery delay, otherwise positive. The objective functions f_3, f_3^+ and f_3^- are described by Eq. (22)–(24).

$$f_3 = \sum_i \sum_k MI^{close}_{i,k} \tag{22}$$

$$f_3^+ = \sum_i \sum_k S3^+_{i,k} \tag{23}$$

$$f_3^- = \sum_i \sum_k S3^-_{i,k} \tag{24}$$

The last two constraints consider the production capacity of finishing and inspection processes. Eq. (25) states that sum of the product of the expectation of production load $Load_{l,i,m}$ defined in Eq. (7) and the amount of production $X_{i,u,v,k'}$ over grade i, cast u, charge v and time period k' satisfying $k' + m = k$ is equal to the production load $Q_{l,k}$ of process l on the time period k. Eq. (26) considers that the production load $Q_{l,k}$ should be less than the production capacity $Load^{\max}_l$.

$$\sum_i \sum_u \sum_v \sum_{k',m:k'+m-1=k} X_{i,u,v,k} Load_{l,i,m} - Q_{l,k} = 0 \quad \forall l, k \tag{25}$$

$$Q_{l,k} - S4^-_{l,k} \leq Load^{\max}_l \quad \forall l, k \tag{26}$$

where $S4^-_{i,k}$ is a slack variable and becomes positive value if the total amount of products that should be handed in the unit l and time period k is greater than the corresponding production capacity. The objective function f_4^- is shown in Eq. (27):

$$f = \sum_l \sum_k S4^-_{i,k} \tag{27}$$

The full space model given above includes a very large number of integer variables and thus it is tractable only for a small number of grades or a short planning horizon, as shown in Section 5.

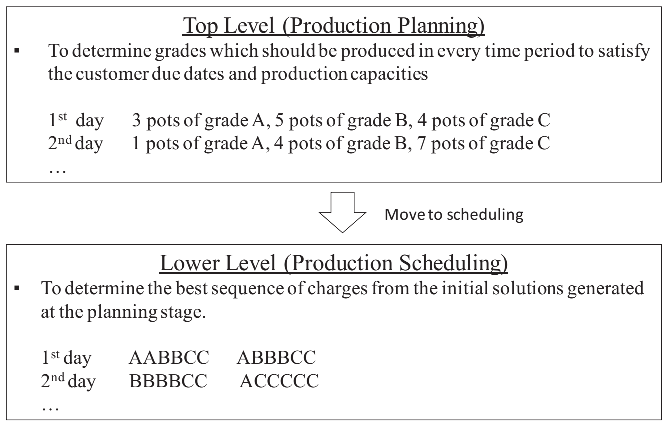
4. Long term scheduling of steel plates production by two level algorithm

Due to a large number of binary variables using the full space MILP approach described in section requires an expensive computational effort, even for a scheduling problem with only one day of period. In order to be able to compute good scheduling solutions within acceptable execution times, we decompose the problem in two levels (see Fig. 4):

- Top level determines the number of pots per grade for each day. The main purpose of this level is to assign products to charges so that the customer due dates, capacity constraints and inventory constraints are met. At this level we generate a relaxed integer formulation that does not take into account the sequencing penalty due to contaminated steel between different grades. Therefore, we call the top level *production planning*. Although there are still integer variables in the relaxed formulation, this planning problem is much easier to solve than the original full space straightforward formulation.
- At the lower level we optimize the sequence of these charges while taking into account the sequence penalty. We call the lower level *production scheduling*.

4.1. Planning model

Since the sequence penalty is not considered in the relaxed mixed integer linear model that will be described in this section, we introduce a new continuous variable $Xr_{i,k}$ and an integer variable $Ur_{i,k}$ instead of $X_{i,u,v,k}$ and $Y_{i,u,v,k}$. $Xr_{i,k}$ represents the amount of production steel of grade i in the time period k, while $Ur_{i,k}$ denotes the number of charges of grade i in the time period k. Eq. (28) states that the total amount of products per grade for each day should be equal to the pot size. To ensure there is always a numerical solution,


Top Level (Production Planning)
- To determine grades which should be produced in every time period to satisfy the customer due dates and production capacities

1st day 3 pots of grade A, 5 pots of grade B, 4 pots of grade C
2nd day 1 pots of grade A, 4 pots of grade B, 7 pots of grade C
…

Move to scheduling

Lower Level (Production Scheduling)
- To determine the best sequence of charges from the initial solutions generated at the planning stage.

1st day AABBCC ABBBCC
2nd day BBBBCC ACCCCC
…

Fig. 4. Two level algorithm for steel plate production scheduling.

we introduce slack variables $Sr1^{+}_{i,k}$ and $Sr1^{-}_{i,k}$, where $Sr1^{+}_{i,k}$ is equal to the amount of the leftover of grade i and time period k, while $Sr1^{-}_{i,k}$ becomes positive if the amount of steel products of grade i and time period k is greater than the total amount of all charges, which means that there are no physically feasible solutions.

$$Xr_{i,k} + Sr1^{+}_{i,k} - Sr1^{-}_{i,k} = PUr_{i,k} \quad \forall i, k \tag{28}$$

Eq. (29) constraints the number of pots for each day so that it is equal to the product of the number of pots per cast and the number of casts per day as shown below:

$$\sum_{i} Ur_{i,k} = UV \quad \forall k \tag{29}$$

with U and V being the number of casts per day and the number of charges per cast, respectively. Then, instead of f^{+}_{1} and f^{-}_{1}, we define new functions fr^{+}_{1} and fr^{-}_{1} which represent the total leftover and the degree of violation of the pot capacity, respectively, as follows:

$$fr^{+}_{1} = \sum_{i}\sum_{k} Sr1^{+}_{i,k} \tag{30}$$

$$fr^{-}_{1} = \sum_{i}\sum_{k} Sr1^{-}_{i,k} \tag{31}$$

If a physical solution is infeasible related to the total production capacity, fr^{-}_{1} becomes positive, otherwise it is zero. If all charges are equal to the pot size, then fr^{-}_{1} is zero. Moreover, since Eqs. (16) and (25) include $X_{i,u,v,k}$, we can rewrite these equations as follows:

$$M^{in}_{i,k} - Xr_{i,k} = 0 \quad \forall i, k \tag{32}$$

$$\sum_{i} \sum_{k',m:k'+m-1=k} Xr_{i,k} Load_{l,i,m} - Q_{l,k} = 0 \quad \forall l, k \tag{33}$$

Eq. (33) imposes the constraint that makes the amount of production $Xr_{i,k'}$ over grade i with k' satisfying $k' + \Delta m = k$ equal to the production load $Q_{l,k}$ of process l on the time period k. With these relaxations, the relaxed mixed integer model is rewritten as follows:

$$\begin{aligned} \min \quad & W^{+}_{1} fr^{+}_{1} + W^{-}_{1} fr^{-}_{1} + W_3 f_3 + W^{+}_{3} f^{+}_{3} + W^{-}_{3} f^{-}_{3} + W^{-}_{4} f^{-}_{4} \\ \text{s.t.} \quad & (10)\text{–}(24), \quad (17)\text{–}(33) \end{aligned} \tag{34}$$

4.2. Meta-heuristic approach to charge sequence optimization

4.2.1. Solution representation

One of the most widely used representations for sequence problem is job-to-position representation (Rahimi-Vahed and Mirzaei, 2007). Since the number of charges for each grade is fixed, we can use job-to-position representation by letting grade of charge be equal to “job”, and the order of charge be equal to “position”. The value of the first element represents a grade scheduled in the first charge. The second value shows a grade scheduled in the second charge and so on. Table 1 shows an example of solution representation. This example means that the first four charges are filled with products of grade A, and the following two charges are grade B. In the second cast, the first three charges are grade C, the next two charges are grade B and the last charge is grade D.

Table 1
Solution representation.

Cast	First cast						Second cast					
Location	1	2	3	4	5	6	7	8	9	10	11	12
Grade	A	A	A	A	B	B	C	C	C	B	B	D

4.2.2. Simulated annealing

In this work, we use simulated annealing (SA) in order to solve the scheduling problems. SA is a general meta-heuristic approach to combinatorial optimization (Aarts and Korst, 1989) and employs a neighbor search algorithm to move from one solution candidate to an improved solution in the neighborhood. Since the obtained solutions may be local optima, simulated annealing allows the objective value to worsen occasionally with a certain decreasing probability in order to avoid local minima as much as possible. That probability is specified by an acceptance probability function $P(f(\mathbf{x}), f(\mathbf{x}'), T)$ that depends on the two neighboring solutions $\mathbf{x}$ and $\mathbf{x}'$ and temperature T with f being the objective value. If $f(\mathbf{x}') \le f(\mathbf{x})$, we accept the solution $\mathbf{x}'$, otherwise we accept it with the probability of $\exp((f(\mathbf{x}') - f(\mathbf{x}))/T)$. T is decreased at each step following some annealing schedules such as $T \leftarrow T \times T_{Factor}$ with T_{Factor} being a temperature factor. In this way, SA employs stochastic decision elements to search for global optimal solution as much as possible. Another possible approach is a Bayesian heuristic algorithm which also employs stochastic decision so that selecting neighboring solutions can achieve the better value of objective function. The neighboring solutions are selected at some probability which is computed by Bayesian method (Mockus and Reklaitis, 1999). The key difference with respect to the Bayesian approach is to learn the parameters which control the probability. However, for parallel computing, the solution diversity is important factor to get better solutions and thus randomness is needed rather than learning parameters. Therefore, in this paper we employ SA method to solve the cast scheduling problems.

Since a reasonable solution is obtained from the relaxed MILP method, it is useful to use it as an initial state in the SA search. In order to obtain wide range of solutions, our search space is within the neighboring sequences that can be obtained by swapping two sets of charges not by swapping two single charges. Meanwhile, in order to avoid inefficient search as much as possible, we restrict the swapping as follows: (i) if two sets are swapped among different casts, their lengths are same; (ii) if swapped within the same cast, lengths of both sets are allowed to be different.

4.2.3. Initial solution

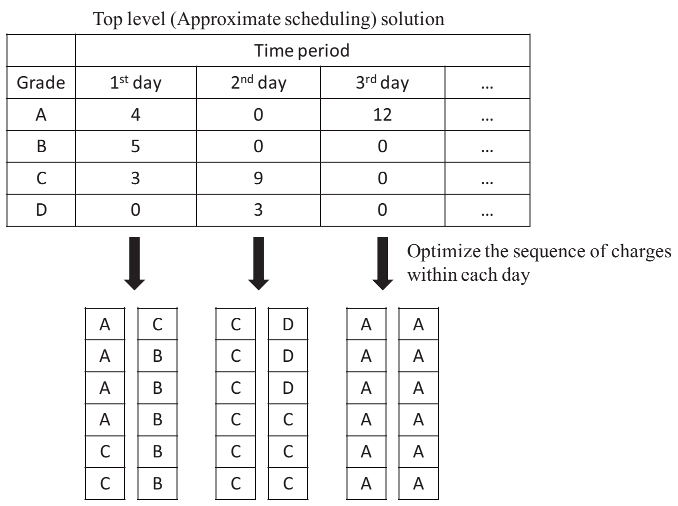
Since a reasonable solution is obtained from the relaxed MILP method, it is useful to use it as an initial state in the SA search. The solutions obtained from the relaxed MILP method denote the number of charges for each grade and each day and thus they do not represent the cast sequences. Therefore, we need to transform these solutions into the form of solutions that represent the sequence of charges. For this purpose, we optimize the sequence of charges for each day independently, and then all the obtained cast sequences are employed as an initial solution in the SA search as shown in Fig. 5.

4.2.4. Swapping operator

In order to obtain wide range of solutions, our search space is within the neighboring sequences that can be obtained by swapping two sets of charges not by swapping two single charges. Meanwhile, in order to avoid inefficient search as much as possible, we restrict the swapping as: (i) if two sets are swapped among different casts, their lengths are same; (ii) if swapped within the same cast, lengths of both sets are allowed to be different. Therefore our searching spaces are restricted the two operations as described in Fig. 6.

4.2.5. Evaluation of objective values in the SA search

It is computationally expensive to compute the value of the whole objective function every time after swapping sets of charges. Therefore, we re-compute only a part of objective values related to casts that are changed by a swapping operator, while the other


Top level (Approximate scheduling) solution
Optimize the sequence of charges within each day

	Time period			
Grade	1st day	2nd day	3rd day	...
A	4	0	12	...
B	5	0	0	...
C	3	9	0	...
D	0	3	0	...

A	C	C	D	A	A
A	B	C	D	A	A
A	B	C	D	A	A
A	B	C	C	A	A
C	B	C	C	A	A
C	B	C	C	A	A

Fig. 5. Generation of an initial solution.

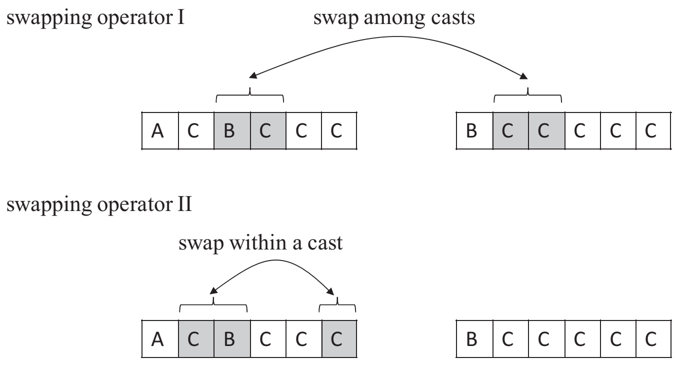


Fig. 6. Swapping operators.

objective values remain same. Table 2 shows the values of variables that should be re-computed after the swapping operator.

4.2.6. A novel parallel simulated annealing

Parallel simulated annealing has been introduced as a method for improving the search for optimum solution. Several researches have explored how to parallelize SA search (Ferreiro et al., 2013; Lee and Lee, 1996). The most straight-forward parallelization is to carry out SA on each worker independently and then to communicate their state at the end of process. This type of parallelization is called *asynchronous approach*. On the other hand, the *synchronous approaches* communicate states of all workers and states are exchanged among processors. It should be noted that the term "worker" represents a computational thread and has no physical meaning.

In this work, we improve the synchronous parallel SA in a manner that avoids local optima as much as possible by communicating states more effectively. The basic idea of this new version of parallel SA is that when the SA is converged in a worker, the new initial state is generated from a combination of the best solution among all workers and the solution in the corresponding worker. The new initial states are computed by using the regeneration idea employed in the shuffled-frog leaping algorithm. The shuffled frog leaping algorithm is a memetic meta-heuristic that is based on the evolution of memes carried by individual and a global exchange of information among the population (Eusuff et al., 2006). We use this evolutionary procedure in order to replace a local optimal solution with new one in each worker.

Let $\mathbf{P}_B$ be the best solution and $\mathbf{P}_C(w)$ be the current solution of worker w, step size $\mathbf{S}$ is computed as follows:

$$\mathbf{S} = \min\{\text{int}\,[\text{rand}(\mathbf{P}_B - \mathbf{P}_C(w))], \mathbf{S}_{\max}\} \quad \text{for a positive step} \tag{35}$$

$$\mathbf{S} = \min\{\text{int}\,[\text{rand}(\mathbf{P}_B - \mathbf{P}_C(w))], \mathbf{S}_{\max}\} \quad \text{for negative step} \tag{36}$$

where $\mathbf{S}_{max}$ denotes the maximum step which should be equal to the number of grades, rand is a random number in the range [0,1] and int returns the nearest integer. The new state is then computed by

$$\mathbf{P}_C^{new}(w) = \mathbf{P}_C(w) + \mathbf{S} \tag{37}$$

Fig. 7 shows a diagram of the proposed parallel simulated annealing method. First, we define K types of initial temperature and temperature factor and then assign them to each worker. K is the number of CPUs we can use. Then, the new solutions are computed by implementing the user specified number of steps (e.g. 2 or 3) of SA for each processor. Each processor performs the SA process asynchronously. In order to increase the diversity along with the searching, all the solutions that are not improved take part in the evolution. If a new solution is better than the previous one, then accept the new solution. Otherwise, generate a solution using Eqs. (35)–(37). In this example, the solution $\mathbf{P}_C(2)$ is not improved in Worker 2 after implementing SA; it is replaced with the new state obtained by adding the step size $\mathbf{S}^{2,3}$ which is computed from Eq. (37) using the current state $\mathbf{P}_C(2)$ and the global state $\mathbf{P}_C(3)$. Similarly the unimproved solution of $\mathbf{P}_C(K)$ is replaced with the new state by adding the computed step size. Since the other solutions such as $\mathbf{P}_C(1)$ and $\mathbf{P}_C(3)$ are improved, they remain the same at this step. Using this procedure we update the solutions in each worker synchronously unless all the solutions are not improved or execution time becomes larger than the specified time.

The new position itself does not represent a sequence because the elements are continuous values. Therefore, it is necessary to construct a sequence of grades from the new position. We employ the rule (Jarboui et al., 2008; Liu et al., 2007) where the

Table 2
Indices of the objective values that should be recomputed after a swapping operator.

	Swapping within same cast; time period = $k1$ cast = $u1$	Swapping among casts whose time periods are the same; time period = $k1$ casts = $u1$, $u2$	Swapping among different casts whose time periods are different; time period = $k1$, $k2$ casts = $u1$, $u2$
$M_{i,k}^{in}$	$\forall i$, period $k1$	$\forall i$, cast $u1$ and $u2$	$\forall i$, cast $u1$ and $u2$
$Sr1_{i,k}^{+}$	$\forall i$, cast $u1$	$\forall i$, cast $u1$ and $u2$	$\forall i$, cast $u1$ and $u2$
$Sr1_{i,k}^{-}$	$\forall i$, cast $u1$	$\forall i$, cast $u1$ and $u2$	$\forall i$, cast $u1$ and $u2$
$MI_{i,k}^{open}$	$\forall i$, period $k1$	$\forall i$, period $k1$	$\forall i$, from period $k1$ to $k2$
$MI_{i,k}^{close}$	$\forall i$, period $k1$	$\forall i$, period $k1$	$\forall i$, from period $k1$ to $k2$
$Sr3_{i,k}^{+}$	$\forall i$, period $k1$	$\forall i$, period $k1$	$\forall i$, from period $k1$ to $k2$
$Sr3_{i,k}^{-}$	$\forall i$, period $k1$	$\forall i$, period $k1$	$\forall i$, from period $k1$ to $k2$

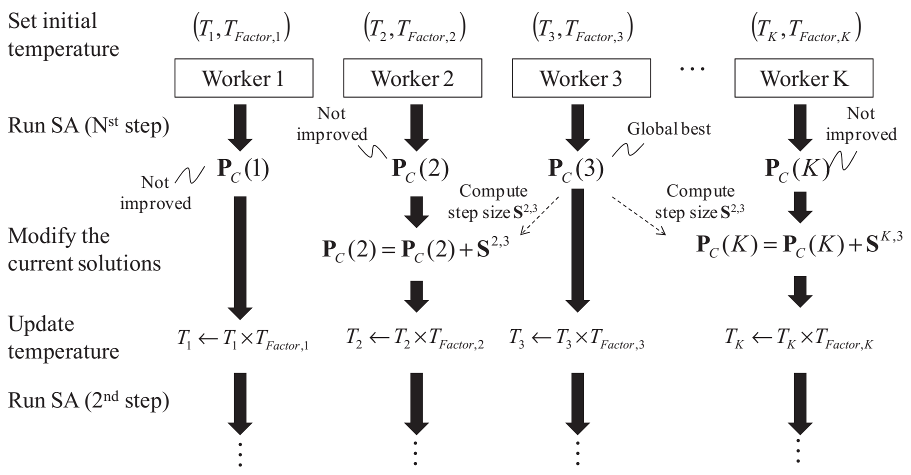

Set initial temperature
$(T_1, T_{Factor,1})$
$(T_2, T_{Factor,2})$
$(T_3, T_{Factor,3})$
$(T_K, T_{Factor,K})$
Worker 1
Worker 2
Worker 3
Worker K
Run SA (N^st step)
Not improved
Global best
Not improved
Not improved
$\mathbf{P}_C(1)$
$\mathbf{P}_C(2)$
$\mathbf{P}_C(3)$
$\mathbf{P}_C(K)$
Compute step size $\mathbf{S}^{2,3}$
Compute step size $\mathbf{S}^{2,3}$
Modify the current solutions
$\mathbf{P}_C(2)=\mathbf{P}_C(2)+\mathbf{S}^{2,3}$
$\mathbf{P}_C(K)=\mathbf{P}_C(K)+\mathbf{S}^{K,3}$
Update temperature
$T_1 \leftarrow T_1 \times T_{Factor,1}$
$T_2 \leftarrow T_2 \times T_{Factor,2}$
$T_3 \leftarrow T_3 \times T_{Factor,3}$
$T_K \leftarrow T_K \times T_{Factor,K}$
Run SA (2^nd step)

Fig. 7. Illustrative diagram of the proposed parallel simulated annealing.

Table 3
Solution generation (A: 3 charges, B: 4 charges, C: 2 charges).

Location	1	2	3	4	5	6	7	8	9
New solution (continuous value)	2.2	2.2	1.6	2	2	2	1.4	1.8	1.8
New solution (grade)	C	C	A	B	B	B	A	A	B

smallest position value of a charge is first picked and assigned grade A. Then, the second smallest position value of a charge is picked and assigned grade A if grade A is still available, otherwise assigned grade B, and so on. This procedure converts the position information into a sequence of grades (Table 3).

4.2.7. *Convergence criteria*

We have employed two types of convergence criteria. First one is that if current best solution among all workers is not improved after implementing SA on each worker, the algorithm stops (convergence criterion I). Second one is that if all current solutions of each worker are not improved after implementing SA, algorithm stops (convergence criterion II). Compared to the convergence criterion I, the second criterion forces search for new solutions to be deeper than the search carried out to meet criterion I; consequently, the computation times for criterion II and greater than the computational times for criterion I as shown by the case studies below.

The step-by-step procedure of the presented optimization approach is listed below.

(1) Build the Bayesian network model that can predict the probability distributions of the production loads and process times from historical process data.
(2) Estimate the probability distribution of production loads and process time for each order (plate) by means of Bayesian inference method based on the constructed Bayesian network model.
(3) Compute the total demand of each grade on the ideal production starting day and the expectation of production load of each grade at each process unit and each elapsed day.
(4) Determine grades which should be produced in every time period to satisfy the customer due dates and production capacities by solving the relaxed mixed integer problem at the planning stage through the commercial solver such as Gurobi.
(5) Set the initial sequence using the solution generated in step (4).
(6) Define K types of initial temperature and temperature factor and assign them to each worker with K being the number of CPU we can use.
(7) Find the sequence of charges from the current sequence by implementing specific number of steps of SA for each processor which performs a SA process asynchronously.
(8) If a new solution (sequence) is better than the previous one, then accept new solution. Otherwise, generate a solution based on the idea of frog flip leaping algorithm.
(9) Discretize the new solution (see last paragraph of Section 4.2.6).
(10) If all workers do not improve their solutions, then the algorithm is terminated. Otherwise, return to step 7).

5. Case studies

In this section, the results obtained by the proposed parallel SA with frog leaping are presented. All the case studies have been computed on a windows 2007 Professional (Intel® Core(TM) i7-4930 CPU @ 3.40 GHz and 16.0 Gb RAM). The full space and the top level production planning models were solved using Gurobi 5.6.3.

5.1. *Small test problems*

Three small test problems are used to see how far from the true optimum are the solutions obtained from our algorithm for the smaller size problems. The optimum is computed by solving the full space MILP model. Table 4 shows the size of full-space MILP model.

Table 5 compares the results from the full space model MILP, the single SA and the parallel SA methods and the proposed parallel SA with shuffled frog leaping algorithm. In these examples, the optimality gaps obtained from full space MILP models are 0% and the

Table 4
Size of three small test problems.

Example	Horizon (days)	Total term (days)	# Casts	# Grades	Full space MILP model	
					# Conts	# Bins
1	7	10	7	3	526	126
2	5	7	5	7	1537	210
3	6	8	6	7	1857	252

Horizon = length of days for which the production scheduling are made, total term = length of days for which the production scheduling deals with the orders, # Casts = number of casts, # Grades = number of grades, # Conts = number of continuous variables, and # Bins = number of binary variables.

Table 5
Results comparison between solution algorithms for small test problems.

Test sets	Algorithm	CPU time (s)	Obj. func. values	Lower bound	Gap (%)
1	Full space MILP	4	700.00	700.00	0.0
	Single SA[a]	19	700.00	–	–
	Parallel SA	13	700.00	–	–
	Single SA[a] (two level algorithm)	22	700.00	–	–
	Parallel SA with leaping algorithmConvergence Criterion I (two level algorithm)	97	700.00	–	–
2	Full space MILP	466	300.00	300.00	0.0
	Single SA[a]	16	500.00	–	–
	Parallel SA	14	500.00	–	–
	Single SA[a] (two level algorithm)	11	400.00	–	–
	Parallel SA with leaping algorithmConvergence Criterion I (two level algorithm)	89	300.00	–	–
3	Full space MILP	99,172	348.03	348.03	0.0
	Single SA[a]	29	400.00	–	–
	Parallel SA	20	400.00	–	–
	Single SA[a] (two level algorithm)	22	400.00	–	–
	Parallel SA with leaping algorithmConvergence Criterion I (two level algorithm)	227	348.03	–	–

[a] Initial temperature $T = 0.8$ and temperature factor T_{Factor} is set to be the best value among [0.5, 0.6, 0.7, 0.8, 0.9].

Table 6
Process capacity.

Mill	Processes	Production capacity
Steel making mills	# Casts	3 casts per day
	# Charges	6 charges per cast
	Size of a charge	200 tons/charge
Steel plate making mills	Process A	350 tons/day
	Process B	700 tons/day
	Process C	1200 tons/day
	Process D	500 tons/day
	Process E	400 tons/day
	Process F	100 tons/day
	Process G	1500 tons/day
	Process H	150 tons/day
	Process I	100 tons/day

# casts = number of casts and # charges = number of charges.

Table 7
Size of two industrial test problems.

Test sets	Horizon (days)	Total term	# Casts	# Grades	Full space MILP model	
					# Conts	# Bins
1	7	13	21	23	59, 270	2898
2	7	13	21	36	141, 866	4536

Horizon = length of days for which the production scheduling are made, total term = length of days for which the production scheduling deals with the orders, # Casts = number of casts, # Grades = number of grades, # Conts = number of continuous variables, and # Bins = number of binary variables.

objective function values of both full space MILP and the proposed parallel SA with shuffled frog leaping algorithms are same. In example 1, the execution time required by the full space MILP algorithm is smaller than the proposed method. However, SA the problem size increases, the proposed algorithm becomes much faster than the full space MILP, as shown by examples 2 and 3.

5.2. Real-world steel-plate production scheduling problems

The real-world steel-plate production data are utilized to examine the effectiveness of the proposed optimization algorithm. The two test sets we consider are based on one week plan of a steel-plate production and the number of grades being 23 and 36, respectively. The production capacities and the size of full space MILP models are shown in Tables 6 and 7, respectively.

Two-level algorithm both with single SA and the proposed parallel SA are used to compute solutions. As a comparison, the full space model MILP, the single SA, and the parallel SA methods are also carried out to compute the solutions. As for parallel computing in parallel SA and MILP, 12 parallel workers are used.

5.2.1. Test set #1

Test set #1 problem computes the first level SCC schedule for 23 kinds of steel grades and the scheduling horizon is one week. Table 8 compares the results from full space model MILP, single SA, parallel SA, two-level algorithm with single SA, and two-level algorithm with the new parallel SA. It can be seen that the full space model algorithm computes the worst objective values even though the computation time is set to be 24 h (86,400[s]). That is because the problem is NP-hard; clearly, the full space model MILP algorithm does not work for industrial-scale problems.

Table 8
Comparison of solution algorithms for real steel-plate production scheduling problems.

Test sets	Algorithm	Execution time (s)	Obj. func. values	Lower bound	Gap (%)
1	Full space MILP	86,400	5979.24	0	>100
	Single SA[a]	429	5152.06	–	–
	Parallel SA	399	4508.39	–	–
	Single SA[a] (two level algorithm)	466	4615.08	–	–
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	393	3831.97	–	–
	Parallel SA with leaping algorithmConvergence Criterion II (two level algorithm)	1905	3715.21	–	–
2	Full space MILP	86,400	21,201.46	0	>100
	Single SA[a]	368	18,619.67	–	–
	Parallel SA	628	18,376.50	–	–
	Single SA[a] (two level algorithm)	313	18,179.84	–	–
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	508	17,540.18	–	–
	Parallel SA with leaping algorithmConvergence Criterion II (two level algorithm)	2951	17,463.44	–	–

[a] Initial temperature $T = 0.8$ and temperature factor T_{Factor} is set to be the best value among [0.5, 0.6, 0.7, 0.8, 0.9].

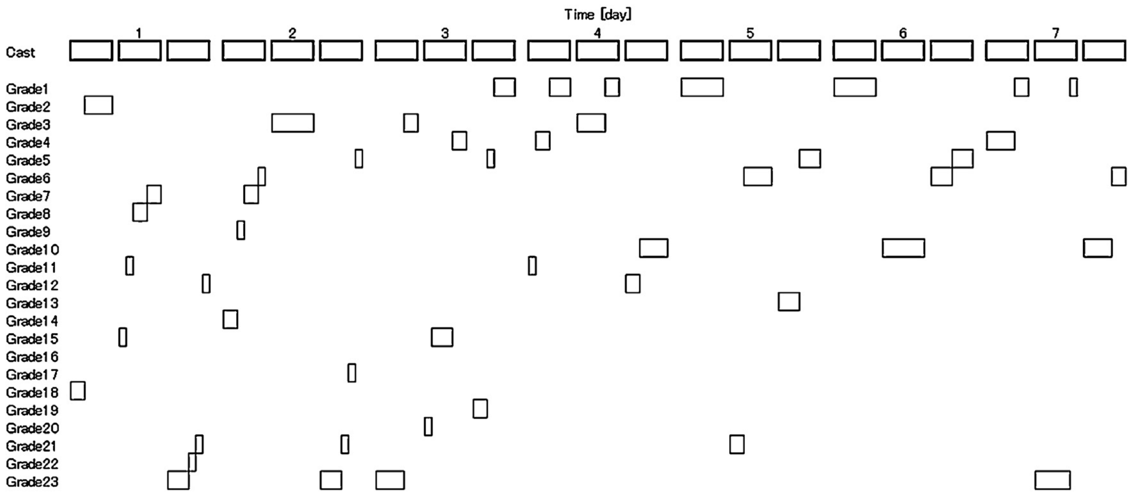


Fig. 8. Cast schedule computed by two-level algorithm with the proposed parallel SA for test set #1.

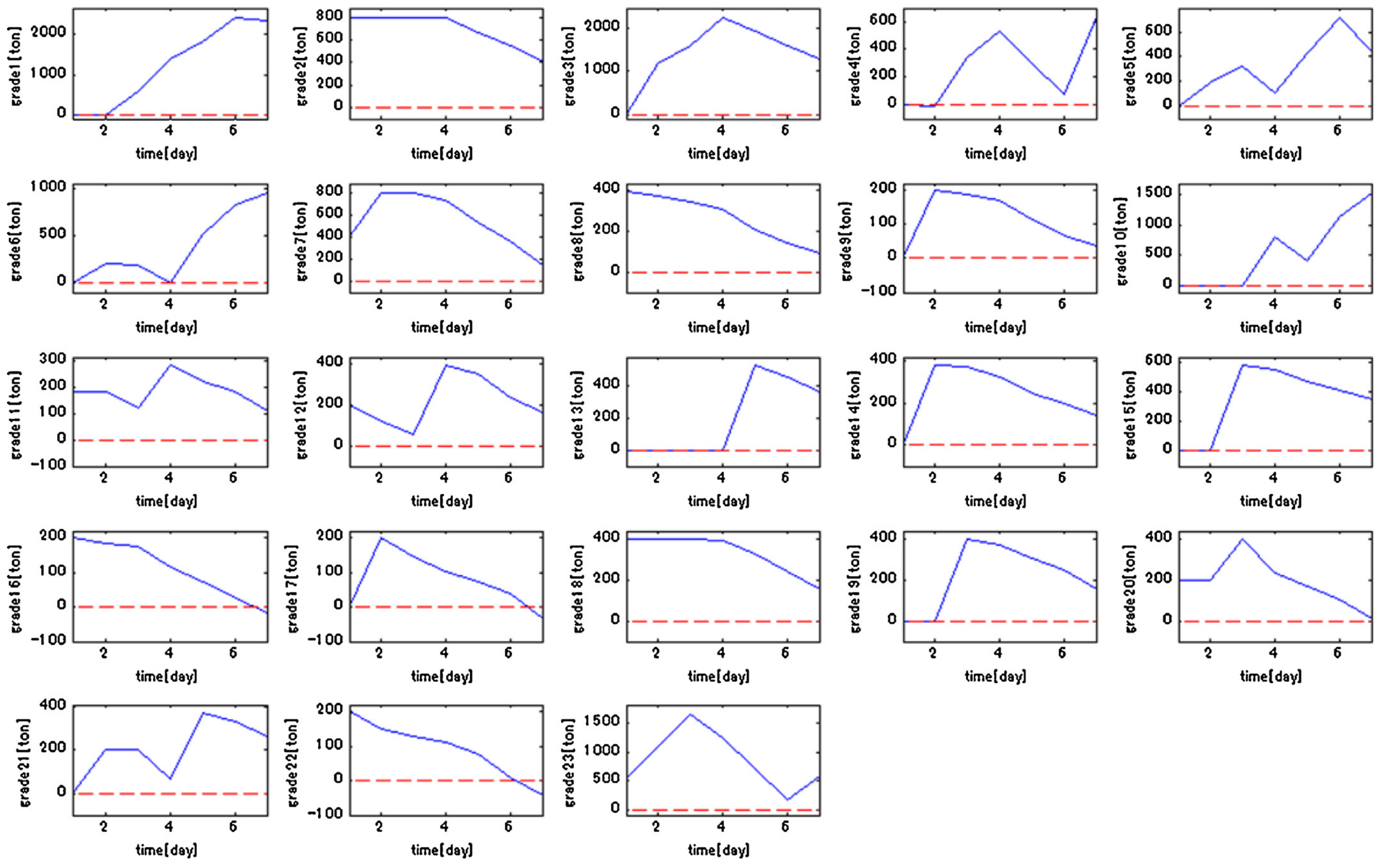


Fig. 9. Trend plots of inventory computed by two-level algorithm with the proposed parallel SA for test set #1. (For interpretation of the references to color in this sentence, the reader is referred to the web version of the article.)

On the other hand, meta-heuristic approaches give us much better solutions in terms of both values of objective function and execution times as shown in Table 8. For instance, the single SA method shows lower objective values and its computational time is much less than the solution of the full-space model MILP. Further, objective values of the proposed two-level algorithm with single SA are even better than those of the simple SA approach, since the relaxed model, which takes into account all the objective functions except for the sequence penalty, gives good initial solutions for the SA search. This comparison shows that the proposed two-level algorithm with single SA has ability to find better solutions than both the full space model MILP approach and the single SA method.

As for the parallel computing methods, we have investigated the effectiveness of the previously published parallel SA (Ferreiro et al., 2013), where all workers start from random initial solutions so that each worker runs independently SA until reaching the next level of temperature. The objective value of the parallel SA is better than all the above methods due to multi-initial solutions. The proposed two-level algorithm with a new parallel SA at the lower level has the elapsed time of 393 s when the convergence criterion I is satisfied; the objective value is 3813.97 (Table 8), which means that both objective value and execution time are less than all the above methods. In addition, when the convergence criterion II is used, the elapsed time is 1905 s elapsed and the objective value becomes 3715.21. This is the lowest among all algorithms evaluated in this work. The proposed two-level algorithm with the new parallel SA with leaping has the best performance in terms of both the objective value and the execution time.

The scheduling results obtained from the proposed parallel SA with frog leaping algorithm are shown in Figs. 8–10. Fig. 8 shows the cast scheduling where horizontal axis represents the time period and the vertical axis denotes the grade index. Each number represents a grade (e.g. 1 means grade A, 2 means grade B and so on). Fig. 9 shows the trend plots of inventories for each grade. In Fig. 9 the trend plots of production loads of finishing lines and their production capacities are drawn by blue solid lines and red dashed lines, respectively. It should be pointed out that all production loads are less than their production capacity for the entire planning period and almost all of inventory levels are greater than zero.

5.2.2. Test set #2

Test set #2 is different from test set #1 only in the number of grades which need to be scheduled, as shown in Table 7. The results from the full space model MILP, the single SA, the parallel SA, the two-level algorithm with single SA, and the two-level algorithm with the new parallel SA with frog leaping are shown in Table 8. Similarly to test set #1, the objective value of the full space model MILP is the worst among all the above methods even though its maximum execution time is set to be 24 h (86,400[s]).

As for meta-heuristic approaches, the objective value and the execution time of single SA are much lower than those of the full space model MILP approach. In addition, the proposed two-level algorithm with single SA performs better than those of the simple SA approach in terms of both objective value and its execution time than the single SA due to the same reason mentioned in the test set #1. This confirms the effectiveness of the two-level algorithm due to the relaxed model MILP creating good initial solutions for SA search.

The parallelization of SA given random initial solution set for each worker gives us slightly lower objective values than the single SA while its computational time is about twice as long as single SA. When applied within the two-level, parallel methods yield results shown in Table 8. When the convergence criterion I is satisfied at the time of 508 s, the solution obtained has the objective value of 17,540.18, which is the lower than the above methods. Furthermore, when the convergence criterion II applied,

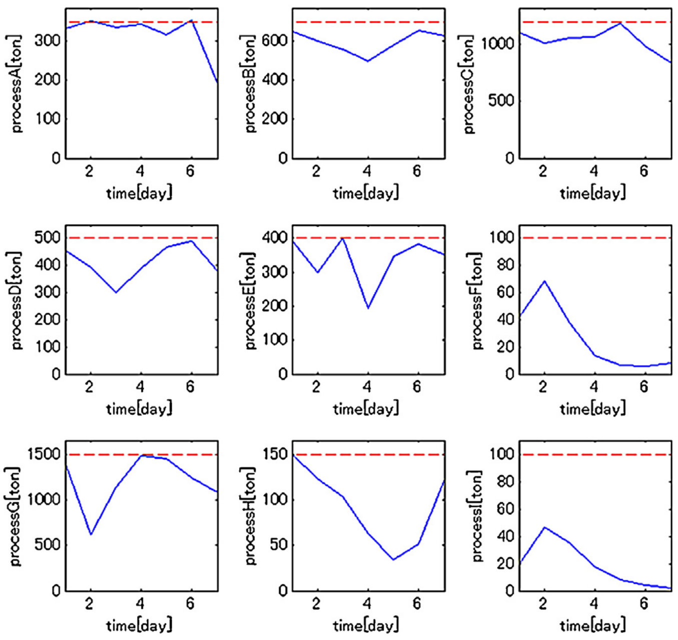


Fig. 10. Trend plots of production loads by two-level algorithm with the proposed parallel SA for test set #1.

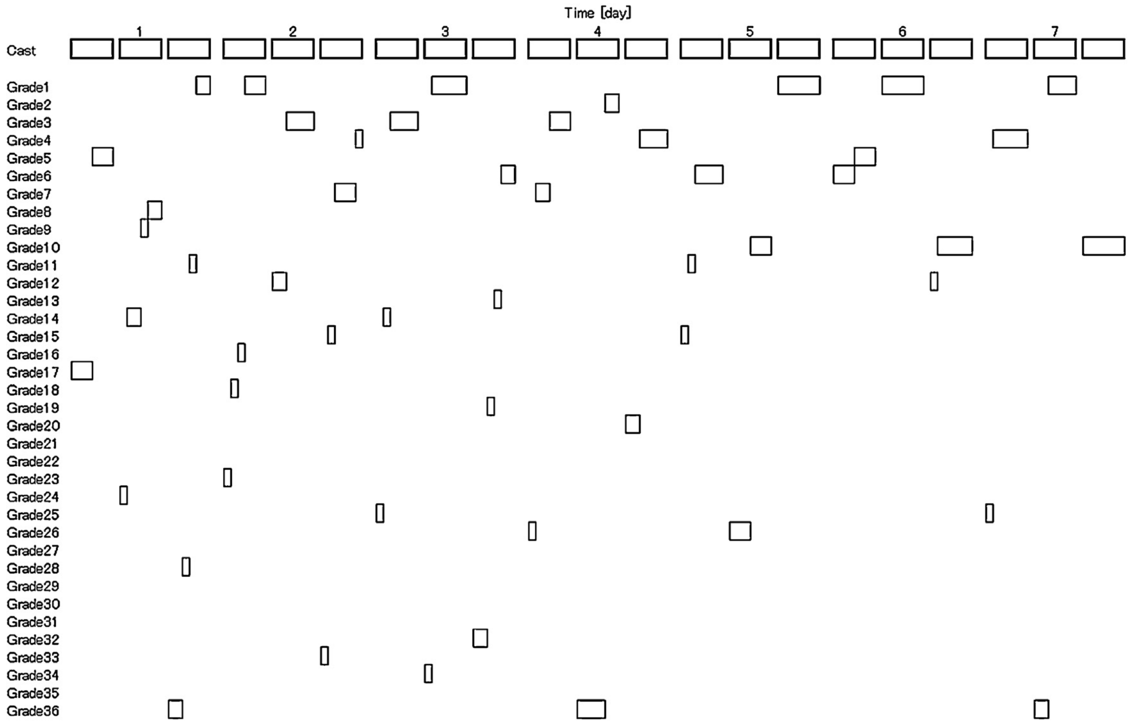


Fig. 11. Cast schedule computed by two-level algorithm with the proposed parallel SA for test set #2.

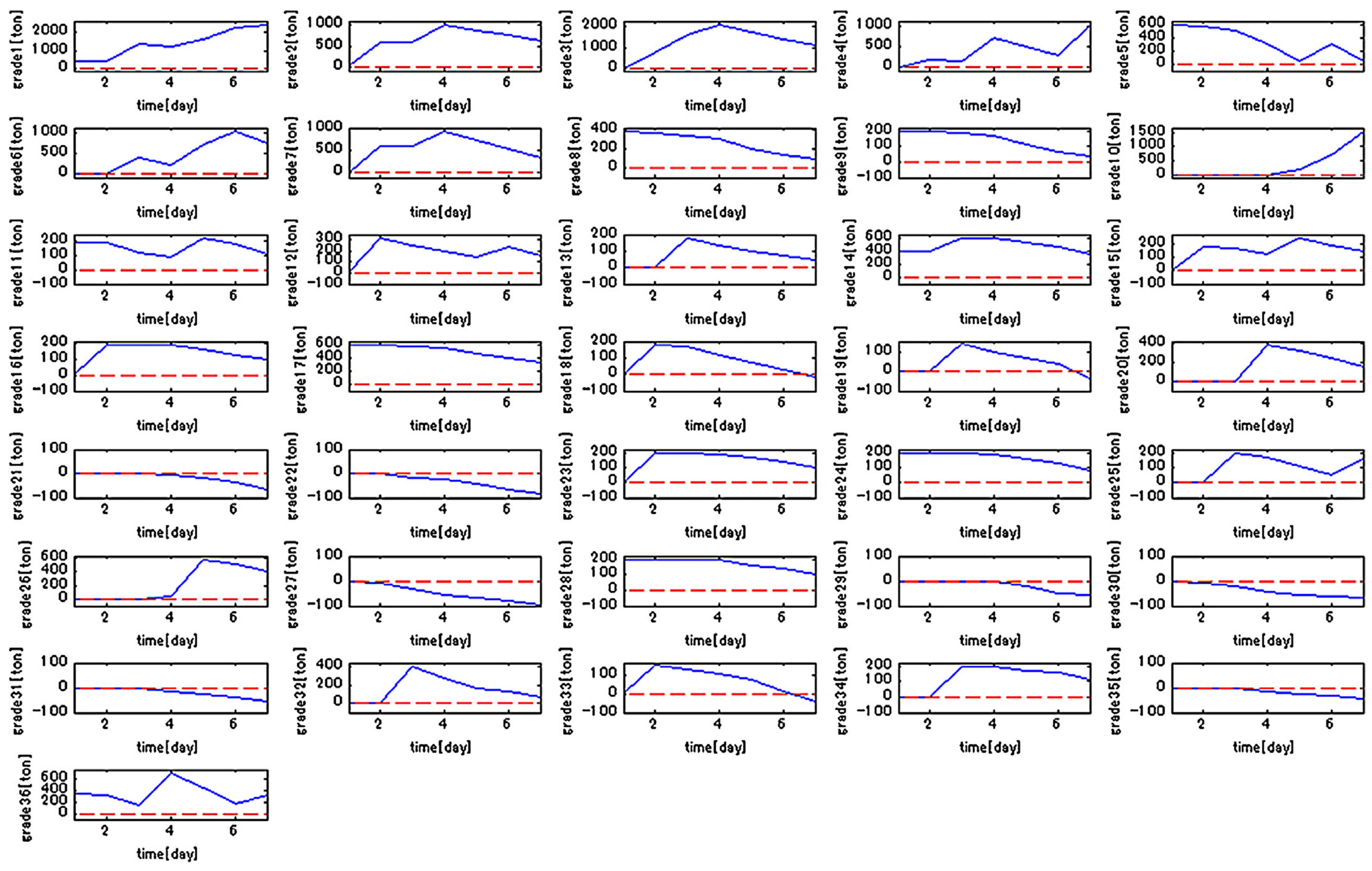


Fig. 12. Trend plots of inventory computed by two-level algorithm with new parallel SA for test set #2.

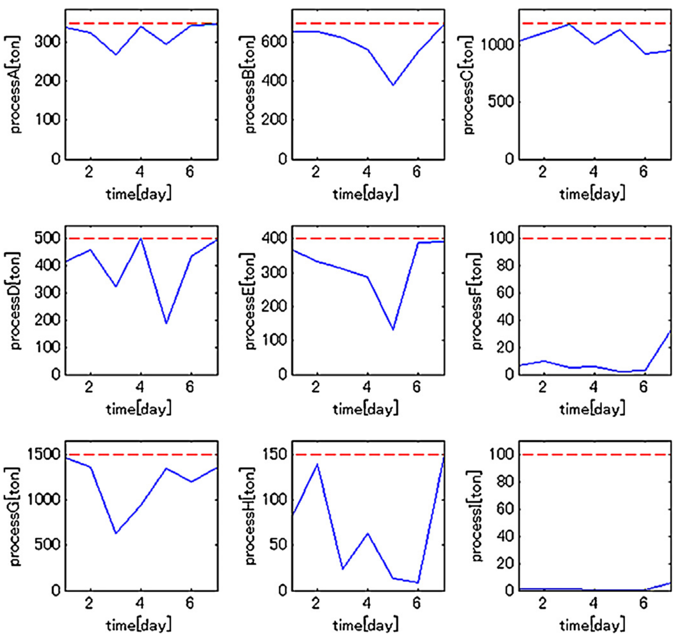


Fig. 13. Trend plots of production loads computed by two-level algorithm with new parallel SA for test set #2.

the objective function value becomes 17,463.44, which is the smallest in our experiments for test set #2. It should be noted that the proposed two-level algorithm with the new parallel SA with leaping provides the best scheduling performance in terms of both the objective value and the execution time. The scheduling results implemented by the two-level algorithm via new parallel SA are shown in Figs. 11–13. Similarly to the test set #1, the inventories are greater than zero and the production loads are less than their capacities for the whole planning horizon.

6. Conclusions

This work proposes a new optimization algorithm which can assign and sequence a large number of different grades of steel plates into a continuous casting machine for steel plate production. In order to consider the production loads of downstream processes and production times, Bayesian inference method is utilized to predict their probability distributions from customer demands and operating conditions. Bayesian inference is derived from the Bayesian network model developed from the historical process data as described in our preceding work. Continuous casting scheduling is a difficult optimization problem due to a large number of binary variables which are needed to represent exactly process characteristics in the optimization model. In order to reduce compute high quality schedules, we propose a new two-level algorithm.

At the top level, we solve a multi-period production planning (it can be considered as approximate scheduling) which is a mixed-integer linear model that does not take into account sequence penalties. Solution of this problem is guaranteed to be feasible at the period boundaries and it is easy to compute via present days MILP solvers. The lower level is tasked with computing the sequencing of charges such that there are a minimum number of switches. This is accomplished by permuting the sequences within the constraints of the top level solution.

We have tested several different variations of the two level algorithm. They differ with respect to implementation of the sequencing at the lower level. One version employs simulated annealing, while the other employ modified parallel simulated annealing combined with the idea of leaping from shuffled frog-leaping algorithm (we do not use shuffled frog-leaping algorithm; instead, we adopt the idea of leaping and implement it in the modified parallel simulated annealing). Simulated annealing has been selected to solve this sequence optimization problem because of its strong capability to avoid local optimal solutions as much as possible by allowing the objective values to worsen occasionally.

Proposed new parallelization approach to SA search aims to further improve the ability of SA to avoid local optimal solutions. The main concept of the proposed parallel SA is that unimproved solutions on each SA step are replaced with new solutions which are computed from the current solution in the corresponding worker and a global best solution among all workers. During the computation of the new solutions, we incorporate into the parallel SA the ideas that have been proposed in the shuffled-frog leaping algorithm.

For small scale test problems, which can be solved to 0.0% optimality gap via full space MILP model, the proposed algorithm computes the global optimum.

Real world steel production data have been used to examine the effectiveness of the presented optimization approach on large problems which arise in industrial practice. The computational results demonstrate that the proposed two-level algorithm generates better solutions and within shorter execution times than both the single SA and the existing parallel SA. Convergence Criterion II is recommended to be used, since it ensures deeper search and leads to better results. Even though the proposed meta-heuristic optimization method is not guaranteed to yield global optimal solutions, it can generate the much better solutions within practical execution times than the rigorous full-space MILP method over much longer execution times.

Industrial application of the proposed algorithm would have to include model maintenance and data preprocessing as needed for the algorithm. Bayesian network for computation of most likely production times would have to be updated (model maintenance) if there is a significant change in some production step (change which would impact the production time by an amount greater than e.g. two standard deviations of the corresponding production time). During preprocessing of data, orders would have to be grouped into plates made of same material, in order to reduce the size of the problem. Since customer orders change very frequently (daily, sometimes hourly), the schedule would have to be recomputed every time when there is a significant change in product orders or when shipment time changes due to a customer request. Proposed algorithm is sufficiently fast to use it on a daily basis for computation of steel plate continuous casting process.

References

Aarts E, Korst J. Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing. New York: John Wiley and Sons, Inc; 1989.
Atighehchian A, Bijari M, Tarkesh H. A novel hybrid algorithm for scheduling steel-making continuous casting production. Comput Oper Res 2009;36(8):2450–61.
Dash S, Kalagnanam J, Reddy C, Song SH. Production design for plate products in the steel industry. IBM J Res Dev 2007;51:345–62.
Eusuff M, Lansey K, Pasha F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Eng Optim 2006;38(2):129–54.
Ferreiro AM, García JA, López-Salas JG, Vázquez C. An efficient implementation of parallel simulated annealing algorithm in GPUs. J Glob Optim 2013;57(3):863–90.
Harjunkoski I, Grossmann IE. A decomposition approach for the scheduling of a steel plant production. Comput Chem Eng 2001;25(11–12):1647–60.
Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res 1986;13(5):533–49.
Jarboui B, Damak N, Siarry P, Rebai A. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Appl Math Comput 2008;195(1):299–308.
Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of IEEE conference on neural networks, vol. 4; 1995. p. 1942–8.
Lawrence D. Handbook of genetic algorithms. New York: Van Nostrand Reinhold; 1991.
Lee SY, Lee KG. Synchronous and asynchronous parallel simulated annealing with multiple Markov chains. IEEE Trans Parallel Distrib Syst 1996;7(10):993–1008.
Liu B, Wang L, Jin YH. An effective PSO-based memetic algorithm for flow shop scheduling. IEEE Trans Syst Man Cybern 2007;37(1):18–27.
Lourenco HR, Martin OC, Stützle T. Iterated local search. Handbook of metaheuristics international series in operations research & management science, vol. 57. Kluwer; 2003. p. 320–53.
Marco D, Birattari M, Stutzle T. Ant colony optimization. Comput Intell Mag IEEE 2006;1(4):28–39.
Mockus L, Reklaitis GV. Continuous time scheduling representation approach to batch and continuous process scheduling. 2. Computational issues. Ind Eng Chem Res 1999;38:204–10.
Mori J, Mahalec V. Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables. Comput Chem Eng 2015;79:113–34.
Rahimi-Vahed A, Mirzaei AH. A hybrid multi-objective shuffled frog-leaping algorithm for a mixed-model assembly line sequencing problem. Comput Ind Eng 2007;53(4):642–66.
Schaus P, Van HP, Monette JN, Coffrin C, Michel L, Deville Y. Solving steel mill slab problems with constraint-based techniques: CP, LNS, and CBLS. J Constraints 2010;16:125–47.
Tang L, Luh PB, Liu J, Fang L. Steel-making process scheduling using Lagrangian relaxation. Int J Prod Res 2002;40(1):55–70.
Tang L, Liu J, Rong A, Yang Z. A mathematical programming model for scheduling steelmaking-continuous casting production. Eur J Oper Res 2000;120(2):423–35.

Chapter 8

Evolutionary algorithms for multi-objective scheduling of continuous casting of steel plates

Abstract

This work introduces a two-step approach to optimization of schedules for continuous casting of steel plates production. In step one, a mixed integer relaxation of the scheduling problem computes how much of each grade of steel to produce in each period of a multi-period model of the production. The second step employs new variants of the non-dominated sorting genetic algorithm (NSGA-II) with jumping gene modification to multi-objective scheduling of the continuous casting. We modify traditional genetic algorithm operators so that only the feasible sequences of operations are examined and combine NSGA-II with jumping gene operations and local search. Initial population for the algorithm is generated via perturbations of the solutions of a weighted sum single objective schedule optimization (Mori and Mahalec, 2016). Comparison with a widely used shuffled frog leaping algorithm shows that the proposed algorithm produces significantly more non-dominated solutions, thereby enabling much better understanding of multi-objective trade-offs.

1. Introduction

Production of steel plates is a multi-step process (see Fig. 1) where slabs of different grades of steel are produced via continuous casting where charges of specific grades of steel are poured into the casting machine and then solidified into slabs. The slabs are then rolled into large plates, cut into required plate sizes and sent to the finishing and inspection lines. Scheduling of continuous casting determines the amount of each grade and the sequence of switching between the grades. It is desirable to minimize the number of switches between the grades, since mixing of different grades materials takes place at the interface, which leads to a loss of material. In addition, one needs to minimize tardiness with respect to the customer due dates. If the product demand cannot be met on by their target dates for all customers, then it is desirable to meet the orders by them most important customers first and deal with the remainder of the orders when there is available capacity.

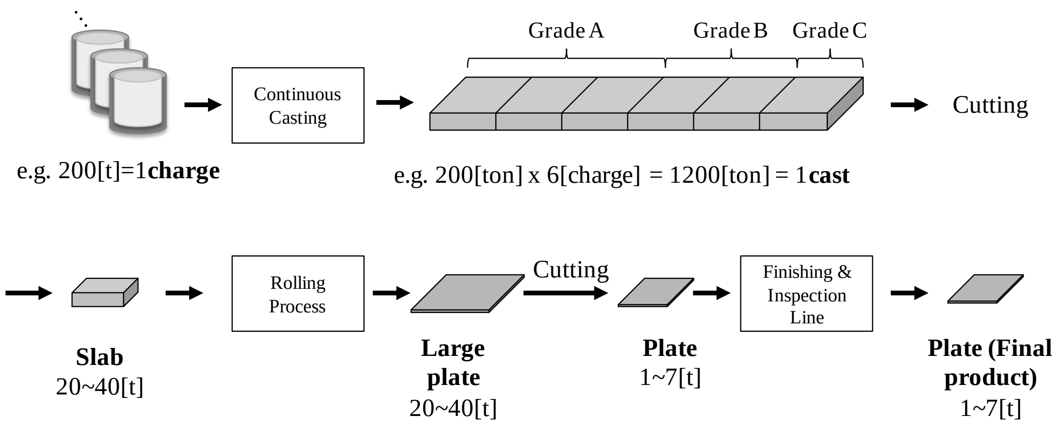


Fig. 1 Process flow of steel-plate production system

One approach to scheduling of continuous casting is to employ a single objective function which combines multiple objectives. Such approach leads to a high quality single solution point, which does not provide an insight into trade-offs between various objectives. This work introduces an algorithm for optimization of multiple objectives, without having to combine them into a single objective function, thereby enabling better understanding of the trade-offs between different objective. The algorithm combines non-dominated sorting genetic algorithm NSGA-II with ideas from jumping gene genetic algorithm and local search to optimize scheduling of manufacturing operations such as continuous casting, i.e. operations which produce many different distinct products while striving to minimize switching for one product grade to another and meet the customer due dates.

Without loss of generality, the multi-objective optimization problem (MOP) can be mathematically formulated as:

$$\min \quad \mathbf{f}(x) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_M(\mathbf{x})\} \tag{1}$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{x}) \geq 0, \mathbf{h}(\mathbf{x}) = 0 \tag{2}$$

where $\mathbf{x} \in \mathbf{\Omega}$ is the vector of decision variables, $\mathbf{\Omega}$ is the decision space and $\mathbf{f} \in \mathbf{F}^M$ consists of $\mathbf{M}$ objective functions $f_i : \mathbf{\Omega} \to R$. The functions $\mathbf{g}$ and $\mathbf{h}$ denote the sets of inequality and equality constraints. The objectives in Eq. (1) often conflict with each other and improvement of one objective may cause deterioration of another objective. Therefore, there is no single solution that can optimize all the objectives. There are three relationships among the solutions as follows (Goh and Tan, 2009):

Definition 1.1 *Weak Dominance:* $\mathbf{f}_1 \in \mathbf{F}^M$ *is said to weakly dominate* $\mathbf{f}_2 \in \mathbf{F}^M$ *, denoted as* $\boldsymbol{f}_1 \preccurlyeq \boldsymbol{f}_2$*, iff* $f_{1,i} \le f_{2,i}$, $\forall i \in \{1,...,M\}$.

Definition 1.2 *Strong Dominance:* $\mathbf{f}_1 \in \mathbf{F}^M$ *is said to strongly dominate* $\mathbf{f}_2 \in \mathbf{F}^M$ *, denoted as* $\mathbf{f}_1 \prec \mathbf{f}_2$ *, iff* $f_{1,i} \le f_{2,i}$, $\forall i \in \{1,...,M\}$ *and* $f_{1,j} < f_{2,j}$, $\exists j \in \{1,...,M\}$.

Definition 1.3 *Incomparable:* $\mathbf{f}_1 \in \mathbf{F}^M$ *is said to incomparable with* $\mathbf{f}_2 \in \mathbf{F}^M$ *, denoted as* $\mathbf{f}_1 \sim \mathbf{f}_2$*, iff* $f_{1,i} \le f_{2,i}$, $f_{1,j} < f_{2,j}$, $\exists i \in \{1,...,M\}$ *and* $f_{1,j} > f_{2,j}$, $\exists j \in \{1,...,M\}$.

The sets of all the best trade-off solutions called *Pareto optimal set* are desirable for decision makers. This concept, which was first proposed by Pareto (Stadler, 1979) are described as:

Definition 1.4 *Pareto Optimal Solution: A feasible solution* $\mathbf{x}^* \in \boldsymbol{\Omega}$ *is called a Pareto optimal solution, iif* $\nexists \mathbf{f}(\mathbf{x}_\mathrm{j}) \prec \mathbf{f}(\mathbf{x}^*)$, $\mathbf{f}(\mathbf{x}_j) \in \mathbf{F}^M$.

Definition 1.5 *Pareto Optimal Set: The Pareto optimal set, denoted as* PS^* *, is the set of non-dominated solutions such that* $PS^* = \{\boldsymbol{x}^* | \nexists \boldsymbol{f}(\boldsymbol{x}_j) \prec \boldsymbol{f}(\boldsymbol{x}^*), \boldsymbol{f}(\boldsymbol{x}_j) \in \boldsymbol{F}^M\}$.

Definition 1.6 *Pareto Optimal Front: The pareto optimal set, denoted as* PF^* *, is the set of non-dominated solutions in the objective space such that* $PF^* = \{f(\boldsymbol{x}) | \boldsymbol{x} \in PS^*\}$.

Without any clear decision maker's preference, the goal of multi-objective optimization is to discover as many non-dominated solutions as possible within the limits of the computational resources. In other words, the aim of multi-objective optimization is to minimize the distance to PS^* and to maximize the diversity within the approximation of PS^*.

The organization of this article is as follows. Review of prior work on multi-objective optimization is presented in Section 2. Section 3 provides the problem statement. Since non-dominated sorting genetic algorithm II, NSGA-II, is the basis for the algorithm we propose, the overview of NSGA-II algorithm is presented in Section 4. The proposed multi-objective optimization algorithm is presented in Section 5. It is comprised of NSGA-II modified by the jumping gene operators and by a local search. The presented methods are applied to the real-world steel plate production data in section 6. Finally, the conclusions are presented in section7.

2. Review of prior work on multi-objective optimization

Literature on multi-objective optimization abounds with various approaches to solve these difficult problems. There are two basic approaches: (i) mathematical programming and (ii) evolutionary optimization. We review briefly the prior work on both approaches, while focusing on multi-objective optimization. Since evolutionary optimization methods have received less attention in the process systems community, we will provide somewhat more extensive review of such methods.

The most straightforward approach to solve deterministic MOPs is the weighted sum method, The method combines all of the objectives into an aggregated single objective by using a weight vector. The objective function in Eq. (1) is transformed to the following form:

$$\min \quad f(\mathbf{x}) = w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \ldots + w_M f_M(\mathbf{x}) . \qquad (3)$$

The weight vector is usually normalized such that $\sum w_i = 1$ and $w_i \le 1$. The weighted sum method changes weights systematically and each single objective optimization determines one particular optimal solution on the Pareto front. Although this method is simple and easy to use, there are some inherent problems (Das and Dennis, 1997). Firstly, the weighted sum is a convex combination of objectives and the performance of the method depends on the shape of Pareto optimal front. As a result, it cannot find all the optimal solutions for problems whose Pareto optimal front is nonconvex. Secondly, it is typically difficult to select the weights such that the optimal solution distribution is uniform since the objectives usually have different magnitudes. To overcome the difficulty of nonconvexity, ϵ-constraint method has been proposed (Chankong and Haimes, 1983) and has been applied in some industries e.g. (Liu and Pistikopoulos, 2009; Subramanyana et al., 2011). In this method, only one objective is optimized while the other objectives are transformed into constraints. The parameter vector ϵ is used as the boundary for all of the objectives and thus different ϵ leads to a different optimal solution. Although this method can reduce the issues associated with nonconvexity, the performance of this method is highly dependent on ϵ since a bad choice of ϵ will lead to infeasible solutions.

In order to overcome these limitations, the adaptive weighted sum (AWS) method was proposed for bi-objective optimization (Kim and Weck, 2005). AWS carries out the typical weighted-sum multi-objective optimization only in the specific region which is designated as a feasible region for sub-optimization by imposing inequality constraints in the objective space. This method has been extended to the multi-objective optimization problems with more than two objective functions (Kim and Weck, 2005). An alternative method is Normal Boundary Intersection (NBI) developed (Das and Dennis, 1998). The basic idea of NBI is to find the Pareto frontier using the set of all convex combinations of the individual global minima of the objective functions as a starting point. NBI can produce a uniformly distributed PFA in non-convex regions, which the weighed sum method lacks.

The methods described above (i.e. weighted-sum method, ϵ -constraint method, AWS and NBI) have a drawback that only one Pareto optimal solution can be obtained by solving one optimization problem. They become computationally expensive with an increase in the number of variables and constraints. More recently, the NBI algorithm was modified so that Pareto frontiers were obtained by solving only a single optimization problem (Siddiqui et al., 2012). In this algorithm, the optimization problem is solved by using a quasi-Newton method and its calculation history is used to obtain the Pareto frontier.

Extensive review of mathematical programming applications in steel plants was presented by Dutta and Fourer (2001). In addition, Tang et al (2001) presented a review of various production planning and scheduling techniques for steel production. Hence, we will mention selected papers published since then. Harjunkoski and Grossmann (2001) presented a decomposition approach to scheduling of steel plates production. They employed a single objective function which combined the makespan, in-process times, and the positive slack-variables representing hold-time variations. They assigned the weights in a manner which ensures that the hold-time violations are minimized. Bellabdaoui and Teghem (2006) presented a mixed integer model linear programming model for continuous cast scheduling where they minimize total completion time for a given sequence of production. More references for single objective function scheduling of steel plates production are reviewed in our previous publication (Mori and Mahalec, 2016).

Mathematical programming approaches, as those described above, guarantee that a global optimum is found, provided that the problems do not become too large for the state of the art MILP solvers to solve in a reasonable time the underlying MILP models. If execution times become too large, one has to develop alternative approaches, such as multi-objective evolutionary optimization algorithms, MOEAs.

MOEAs have been developed to solve multi-objective optimization problems in various aspects of chemical engineering e.g. (Alam et al., 2013; Sharma and Rangaiah, 2013). MOEA is based on a population of solution candidates and the reproduction process which is able to combine existing solutions to generate new solutions.

Vector Evaluation Genetic Algorithm (VEGA) (Schaffer, 1985) selects the candidates solutions for each of *M* objectives separately. Thus *M* sub-problems of size *N/M* are generated assuming a total population size of *N*. These sub-problems are shuffled together and recombination and mutation are performed as normal MOEAs. The algorithm is simple and easy to implement. Since it employs proportional fitness assignment, its selection scheme is opposed to the concept of Pareto dominance and good compromise solution for all the objectives will be discarded. Fonseca and Fleming introduced a variation of multi-objective genetic algorithm (MOGA) where fitness assignment is performed based on a Pareto-based ranking procedure (Fonseca and Fleming, 2003). In Pareto ranking, for each individuals $\mathbf{i} \in \mathbf{P}$, its rank $\mathrm{r}(\mathbf{i}) = 1 + |\{\mathbf{j}|\mathbf{j} \in \mathbf{P} \wedge \mathbf{j} \preccurlyeq \mathbf{i}\}|$ is calculated. A raw fitness is assigned to individuals by interpolating from the best ($\mathrm{r}(\mathbf{i}) = 1$) to the worst ($\mathrm{r}(\mathbf{i}) \leq \mathrm{N}$). Then, the fitness of individuals with the same rank are averaged and shared. While this method is efficient, the performance of the method depends on the sharing factor.

The nondominated sorting genetic algorithm (NSGA) (Srinivas and Deb, 1994), niched-Pareto genetic algorithm (NPGA) (Horn and Nafpliotis, 1993), strength Pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1998), Pareto archived evolution strategy (PAES) (Knowles and Corne, 2000) and Pareto envelope-based selection algorithm (PESA) (Corne etal., 2000) use a tournament selection scheme.

In NSGA, all non-dominated solutions belong to the first level of non-domination in the population. Then, these solutions are temporary removed in order to find the solutions belonging to the second non-dominated front and the procedure is repeated until all solutions are assigned a level of non-domination. The fitness is assigned to each individuals based on its non-domination level. To maximize the diversity of the solutions, fitness sharing is done in decision space. In addition, an extended version of NSGA, called NSGA-II, has been proposed in order to keep diversity without specifying a sharing parameter (Deb et al., 2002).

An alternative approach is niched Pareto genetic algorithm (NPGA) (Horn and Nafpliotis, 1993). The basic idea of NPGA is to combine the tournament selection and the concept of Pareto dominance. Two random competitors $\mathbf{i}, \mathbf{j} \in \mathbf{P}$ are selected and compared against a subset $\mathbf{P}_{dom} \in \mathbf{P}$. If $\mathbf{i}$ is non-dominated with respect to $\mathbf{P}_{dom}$ and $\mathbf{j}$ is not, then $\mathbf{i}$ is the winner. If $\mathbf{j}$ is non-dominated regarding to $\mathbf{P}_{dom}$ and $\mathbf{i}$ is not, then $\mathbf{j}$ is the winner. When both $\mathbf{i}$ and $\mathbf{j}$ are either dominated or non-dominated, the winner is decided by fitness sharing, using niche counts as calculated in the objective space. Similar to NSGA-II, a revised version of NPGA, called NPGA-II, was developed (Erickson et al., 2001). This method uses tournament selection based on the Pareto ranking in order to determine tournament winners. Like NPGA, tournament ties are solved by fitness sharing.

Strength Pareto evolutionary algorithm (SPEA) uses a mixture of established techniques and new techniques (Zitzler and Thiele, 1998). This method uses an external set $\overline{\mathbf{P}}$ including non-dominated solutions found. A strength value is computed for each individual $\mathbf{i} \in \overline{\mathbf{P}}$ and it is used for computing the fitness of each individual $\mathbf{j} \in \mathbf{P}$. Then, Pareto-based niching method is utilized in order to keep diversity in the population. An improved version of SPEA was proposed (Zitzler et al., 2001). This method incorporates a fine grained fitness assignment strategy, a density estimation technique, and an enhanced archive truncation method in order to improve convergence and diversity.

Pareto archived evolutionary algorithm (PAES) is a (1 + 1) evolution strategy in which a single parent generates a single offspring (Knowles and Corne, 2000). It adopts local search in combination with a reference archive of solutions previously found in order to identify the Pareto dominance ranking. Each objective is divided into 2^d equal divisions with d representing the number of bisections of the space and thus the entire search space is divided into 2^{dM} grids. Each solution is placed in a certain grid location based on the value of its objective. If the offspring belongs to a less crowded grid than the parent, the offspring becomes the parent in the next generation. Otherwise, the parent solution continues to be the parent.

Pareto envelope-based selection algorithm (PESA) (Corne et al., 2000) incorporates ideas from SPEA and PAES. Similar to SPEA, PESA uses an external set that stores the current approximation to the Pareto front. Like PAES, PESA maintains diversity which consists of a crowding procedure that divides objective space. Zitzler et al. (2003) proposed a quality measure of Pareto front algorithm (PFA), the ϵ -indicator which gives the factor by which an approximation set is worse than another in terms of all objectives is proposed. They developed the ϵ -indicator based evolutionary algorithm (IBEA).

As another MOEA, differential evolution (DE) has been developed for solving MOPs. The Pareto differential evolution (PDE) algorithm has been proposed for multi-objective optimization (Abbass et al., 2001). Since DE algorithm is sensitive to the parameter values including crossover and mutation rates, a self-adaptive Pareto differential evolution (SPDE) was also proposed (Abbass, 2002). While SPDE adopts the nearest neighbor distance function for preserving the diversity of the population, adaptive Pareto differential evolution (APDE) uses the population variance based functions (Zaharie and Petcu, 2003). In addition, vector evaluated differential evolution (VEDE) was proposed, which is a multi-population differential evolution approach inspired by VEGA (Parsopoulos et al., 2004). Similar to VEGA, selection is conducted for each of objectives separately and does not consider the diversity of the population. Since DE is not efficient for fine-grain optimization, rough sets theory was adopted in order to improve the spread of the non-dominated solutions produced by a DE based MOEA (Hernandez-Daz et al., 2006). MOEA based on ϵ -dominance concept and efficient parent and archive update strategies was proposed by Deb and coworkers (Deb et al., 2003; Cai et al., 2007). Because the ϵ -dominance does not allow two solutions with a small difference in the objective space to be non-dominated to each other, it maintains the diversity of the populations automatically. Furthermore, a multi-objective evolutionary algorithm based on decomposition (MOEA/D) was developed (Zhang and Li, 2007). This method decomposes a MOP into a number of scalar optimization sub problems and solves them simultaneously by DE in order to reduce the computational complexity.

An alternative to PDE algorithm is differential evolution for multi-objective optimization (DEMO) method has been intrduced (Robič and Filipič, 2005). DEMO replaces the parent individuals with the candidate that dominates it for convergence to the true Pareto front. In addition, DEMO uses non-dominated sorting and crowding distance in truncation of the extended populations for maintaining the diversity. Similarly to SPDE, the self-adaption mechanism was incorporated with DEMO algorithm (Chow and Tsui, 2004). Sequential quadratic programming (SQP) has also been used in order to improve candidate solutions produced by DE (Zamuda et al., 2009).

Particle swarm optimization (PSO) (Kennedy and Eberhart, 1995) has been modified for multi-objective optimization. The most common approach for MOPs is to use the weighted aggregation in which all the objectives are summed to a weighted as in Eq. (3). If w_i is fixed, the optimization has to be repeated many times to obtain a desired number of Pareto optimal solutions. Bang-Bang weighted aggregation (BWA) approach is used in order to ensure that the solutions move along the Pareto front (Parsopoulos and Vrahatis, 2002).

Multi-objective particle swarm optimization (MOPSO) algorithms use external archives and store the non-dominated solutions among the population and external archive. The velocity and position of each particle are updated using information of both the population and external archive (Coello and Lechuga, 2002). The main drawback of this method is that the computational cost increases rapidly with the number of objectives, population size and archive size become large. To solve this issue, the concept of ϵ -domination is employed instead of Pareto domination when updating the external archive (Mostaghim and Teich, 2003). If large values of ϵ is used, computational cost can be reduced. However, this method is theoretically able to cover the approximated Pareto front only if using small values of ϵ, which can be computationally expensive. To overcome this limitation, a covering method is used for MOPSO (Mostaghim and Teich, 2004). First, the MOPSO runs with a restricted archive size that can be obtained ϵ -dominance strategy. After obtaining the initial solutions, the particles in the population are divided into sub swarms around each non-dominated solution. Then, the best local solutions in sub swarms are found.

Another variation is non-dominated sorting particle swarm optimizer (NSPSO) developed by Li (Li, 2003). NSPSO compares all particles' personal bests and their offspring in the entire population, instead of basing the comparison solely on a particle's personal best with its offspring. The concept of Pareto dominance has also been used for improving MOPSO (Pulido and Coello Coello, 2004; Sierra and Coello, 2005). In order to reduce the computational times, a correlation of the objective functions has been used (Chow and Tsui, 2004). Techniques which are able to deal with constraints or discrete decision variables were developed by (Tseng and Liao, 2008) and by (Venter and Haftka, 2010). Fuzzy clustering has been utilized to prune the size of the non-dominated set (Agrawal et al., 2008). More recently, strength Pareto approach was combined with MOPSO in order to maintain the diversity of the Pareto front (Elhossini et al., 2010). A comparison of MOPSO and MODE was published recently (Dominguez and Pulido, 2011). This work concludes that even though MOPSO searches solutions more aggressively than MODE, MODE generate better distributed PFA that leads to better Pareto fronts.

While various kinds of MOEA have been developed, all of them are population based algorithms which may not be suitable for sequence optimization problem such as scheduling of continuous casting. That is mainly because combinations of the solutions rarely create better solutions especially for sequence optimization problems. Single solution approaches such as simulated annealing (Aarts & Korst, 1989), tabu search (Glover, 1986) and iterated local search (Lourenco et al., 2003) work better since they focus on modifying and improving a single candidate solution. However, these single solution approaches are not able to compute Pareto optimal front by means of solving a single optimization problem. In this work, we develop a novel MOEA algorithm for sequence optimization. In our approach, we first we solve a set of single optimization problems described in Eq. (3) to compute a couple of solutions using a single solution approach such as parallel simulated annealing with frog leaping algorithm (Mori & Mahalec, 2015) in accordance with the weight vector $\mathbf{w}$. Then, we start multi-objective optimization by using these solutions as initial population. In order to explore the entire Pareto optimal set, it is very important to assign good fitness of an individual such that diversity is maintained. For this purpose, we employ the idea of NSGA-II algorithms to compute the fitness and select the parents' chromosomes. Furthermore in addition to the traditional genetic operations, we employ the idea of jumping gene operations (Tang et al., 2011) as well as shuffle frog leaping algorithm (Li et al., 2012) to enhance the searching ability. Since GA crossover of two feasible solutions does not result in a feasible solution in many cases, we introduce the operation that makes infeasible solutions into feasible ones. Finally, we optimize the charge sequence within each cast; optimization is carried out every time after jumping gene operations to avoid generating useless populations as much as possible.

Since steel plate scheduling leads to very large problems, in this work we have developed new version of MOEAs. In order to have reassurance about the quality of the solutions obtain by the proposed algorithms, we first solve small scale problems by using mathematical programming approach and also solve the same small scale problems by using the proposed algorithms. Then we apply the proposed algorithms to large scale steel scheduling problems which cannot be solved in a reasonable time by using state of the art MILP solvers.

Our prior work (Mori and Mahalec, 2016) was based on a single objective function approach, which enabled computation of a high quality, single solution. That solution does not provide an insight into trade-offs between different objectives. For that reason, this work pursues multi-objective scheduling of the steel plates production.

3. Problem formulation

In this study we assume that the steel plate manufacturing facility has received orders for the next 5 to 10 days and thus the number of received orders can run into tens of thousands. Therefore we will group the received orders into grades, based on the qualities of the steel from which they are made of. Our task is to determine the sequence of charges over period of 5 to 10 days, which is called *Long Term Scheduling*. The targeted scheduling problem of SCC for steel-plate production is stated as follows:

Given:

1. A scheduling period (e.g. 1 week).
2. The number of charges per cast (e.g. 6 charges per cast).
3. The number of casts per day (e.g. 3 casts per day).
4. The maximum processing capacity of each finishing and inspection process.
5. The size of charges (e.g. 200[ton]).
6. The contamination cost when switching grades between consecutive charges.
7. A set of delivery orders for each grade along the scheduling horizon (e.g. demand profile).

Determine:

1. The amount of steel for each grade on each charge of each continuous caster.
2. Sequence of charges
3. The amount of products that each finishing and inspection line should process each day.
4. The inventory profile of products at a distribution warehouse.

While Minimizing:

1. The cost of producing contaminated steel caused by the switching grades between consecutive charges.
2. The amount by which the utilization of the production capacity may exceed the actual capacity available.

   Note that we included this as a term in the objective function and not as a constraint, since the amount of steel which is not produced in a given period can be produced in one of the subsequent periods, if there is available plant capacity. If there is no available plant capacity in the subsequent periods, it means that some of the requested product cannot be delivered and the plant needs to decide which customer orders to decline.

3. Delay in delivery of products to the customers at the time beyond customer due date.

   This objective can be reformulated as the amount of steel plates below the required minimum, since insufficient amount of steel plates means that some customers will not receive their products on time.

Subject to:

1. Only one steel grade can be used in a single charge. Once it begins charging, the same grade steel must be used and its amount should always be equal to the size of a charge.
2. Finishing and inspection processes can handle at most the same amount of steel as its processing capacity.

Assuming:

1. After optimizing the sequence of charges, orders can be assigned to each charge permuted within the time prior to the deadline so that each charge contains the same grade.
2. There is only one casting machine for steel plates and it cannot be used for other kinds of products such as steel sheet.
3. Processing order of the finishing and inspection lines is fixed.

4. Variables associated with both customer demands and operating conditions are known.
5. Variables associated with both production loads and process time are unknown.
6. Probability distributions of production loads and process times are known (they are predicted by the Bayesian network model from variables associated with both customer demands and operating conditions).

Mori and Mahalec (2015) proposed a two level algorithm to solve continuous cast scheduling with a single objective function. The top level is a multi-period linear programming (LP) model. Length of each period is set in a manner which relates it to the production. In this work, length of a period equals to one day. Solution from the top level determines how many pots of each grade of steel need to be produced each period so that the customer due dates, capacity demand and inventory constraints are met. At the lower level they employ modified parallel simulated annealing to optimize the sequence of the pot charges while taking into account the sequence penalty.

In this study we solve multi-objective scheduling of continuous casting. We also employ a two level algorithm, where the top level is the same as on Mori and Mahalec (2015), while the lower level computes Pareto optimal solutions for the following three objectives:

Minimize material contamination due to switching

$$\min \quad f_1 = \sum_k \sum_u \sum_v \mathrm{C}(\mathrm{Charge}(k,u,v).\mathrm{grade},\ \mathrm{Charge}(k,u,v+1).\mathrm{grade}) \tag{4}$$

where C is the cost of switching between two grades.

Minimize amount of steel plate inventory below specified minimum

$$\min \quad f_2 = \sum_i \sum_k S2_{i,k} \tag{5}$$

where S2 is max(0, the difference between the actual and minimum amount of steel).

Minimize amount of production capacity utilization above the production capacity

$$\min \quad f_3 = \sum_l \sum_k S3_{l,k} \tag{6}$$

where S3 is max(0, the difference between max capacity and capacity required to produce the amounts required to meet the demands).

Equation (4) minimizes the amount of contaminated steel (f_1) due to switching from one grade to another, where $\mathrm{C}(i, j)$ is a quality loss matrix which returns the amount of contaminated steel between grade *i* and *j*. "$\mathrm{Charge}(k,u,v).\mathrm{grade}$" term is a grade of charge allocated at *v*-th position of *u*-the cast of time period *k*. The objective given by function Eq. (5) minimizes the amount of steel plates which violate the minimum inventory capacity (f_2) where $S2_{i,k}$ is a slack variable which ensure that the following constraint is satisfied:

$$MI_{\min} - MI_{i,k}^{close} - S2_{i,k} \leq 0 \qquad \forall i,k \tag{7}$$

where $MI_{\min}$ is the minimum inventory and $MI_{i,k}^{close}$ is the closing inventory of grade *i* at time period *k*. If the $S2_{i,k}$ is positive, the corresponding steel plates cannot be delivered to the customers by the due date. The objective function described by Eq. (6) minimizes the amount of steel plates which violate the capacities of production unit (f_3) where $S3_{l,k}^{-}$ satisfies the following constraint:

$$Q_{l,k} - S3_{l,k}^{-} \leq Load_l^{\max} \quad \forall l,k \tag{8}$$

where $Q_{l,k}$ is the production load of process *l* on the time period *k* and $S3_{l,k}^{-}$ is a slack variable which becomes positive if the total amount of products that should be handled by the unit *l* and time period *k* is greater than the corresponding production capacity $Load_l^{\max}$. In other words, the materials corresponding to $S3_{l,k}^{-}$ can be produced on the following day (k+1) or later. Detailed formulation can be found in our previous work (Mori & Mahalec, 2016).

4. Non-Dominated Sorting Genetic Algorithm with Jumping-Gene Transposition Operation

In this section we will describe NSGA-II (Deb et al., 2002) and then in the subsequent section we describe a modified version which we have developed in this work.

Non-Dominated Sort and Crowding Distance

The first step if to initialize the decision variables for each individual *p* in population *P* and then initial objective values of each individual p are computed. The second step is to sort the initialized population based on non-domination. Members which are not dominated by any other individuals are assigned rank 1. Rank denotes the level of non-dominance. Then the rank 1 solutions are temporarily disregarded and the non-dominated solutions are assigned rank 2. This procedure is continued till all solutions are assigned a level of non-dominance. Once the non-dominated sort is complete, the crowding distance is assigned to each individual. Crowding distance is assigned front wise and comparing the crowding distance within same level of non-dominance. Table 1 shows the procedure for computation of crowding distance.

Table 1 Procedure for computing crowding distance

```
    Procedure  Compute Crowding Distance
1   Initialize the distance $D_{r,j}$ to be zero for all individuals, where j corresponds to the j-th
    individual in rank r-th front
2   for each objective function $i$
3       Sort the individuals in r-th rank front in terms of i-th objective values.
4       Assign infinite distance to boundary values so that boundary points are always
        selected. i.e. $D_{r,1} = \inf$ , $D_{r,N_r} = \inf$ where $N_r$ is the number of individuals in r-th
        rank front.
5       for k=2 to $N_r - 1$
6           $D_{r,j} = D_{r,j} + (D_{r,j-1}.\mathrm{obj}(i) + D_{r,j-1}.\mathrm{obj}(i))/(f_i^{\max} - f_i^{\min})$ where $D_{r,j-1}.\mathrm{obj}(i)$ is
            the i-th objective values of j-th individuals of i-th rank front.
```

Selection

After all the individuals are sorted and assigned crowding distance, the binary tournament strategy is implemented to select parent individuals as follow:

Step 1: Select two individuals **p** and **q** at random

Step 2: If a rank **p** is smaller than that of **q**, then **p** is winner. Else if a rank of **q** is smaller than that of **p**, then **q** is winner. If rank(**p**) = rank(**q**), then the individual that has bigger crowding distance than the other is winner.

Typically a size of parent individuals is set to be a half of a population size.

Traditional Genetic Operators

Once the parent individuals are selected, children $\mathbf{c}_1$ and $\mathbf{c}_2$ are generated from randomly selected two individuals $\mathbf{p}_1$ and $\mathbf{p}_{2.}$ In traditional NSGA-II algorithm typically use *Simulated Binary Crossover* (SBX). SBX generates new individuals $\mathbf{c}_1$ and $\mathbf{c}_2$ with a specified probability as follows:

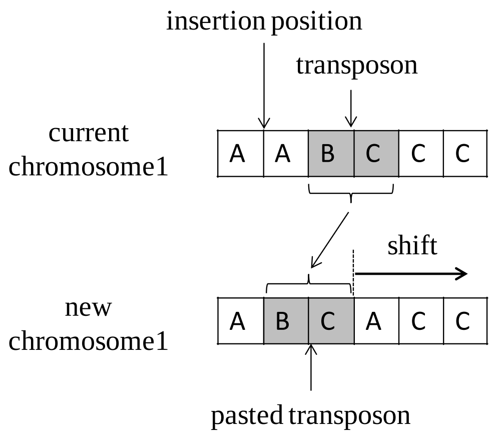
$$\mathbf{c}_1 = \frac{1}{2}\{(1-\beta)\mathbf{p}_1 + (1+\beta)\mathbf{p}_2\}, \quad \mathbf{c}_2 = \frac{1}{2}\{(1+\beta)\mathbf{p}_1 + (1-\beta)\mathbf{p}_2\} \tag{9}$$

where β is a random value. Furthermore, *Polynomial mutation* is implemented with a specified probability as follows:

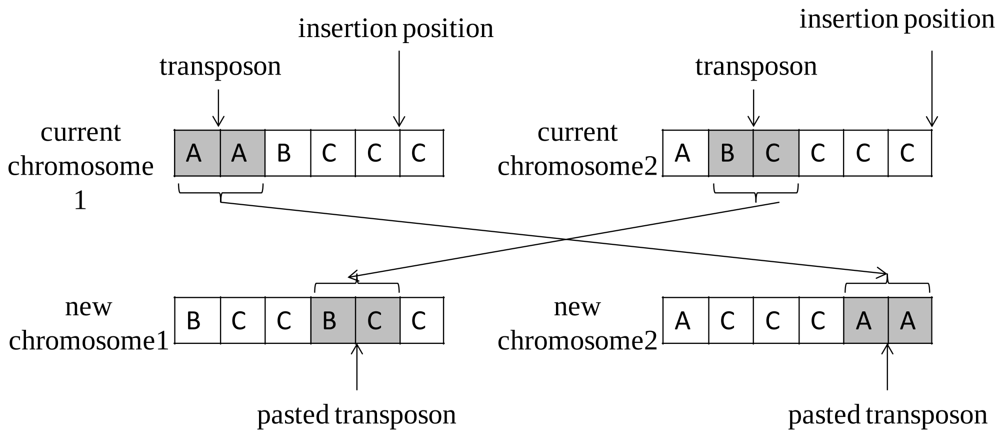
$$\mathbf{c} = \mathbf{p} + (\mathbf{p}^u - \mathbf{p}^l)\delta \tag{10}$$

where $\boldsymbol{c}$ is a child, $\mathbf{p}^u$ and $\mathbf{p}^l$ are upper and lower bounds on the parent component and δ is a small variation.

As an additional operation, Jumping-Gene (JG) transposition operator, is applied to Multi-objective optimization (Chan et al., 2008). There exist two types of transposons: 1) *cut-and-paste transposon* and 2) *copy-and-paste transposon,* as shown in Fig. 2 and 3, respectively. Cut and paste transopson, the element cut from an original position, is pasted into a new position of a chromosome, and the original chromosome remains same. Copy-and-paste operation (Fig. 3) replaces parts of the chromosome by the transoposon.

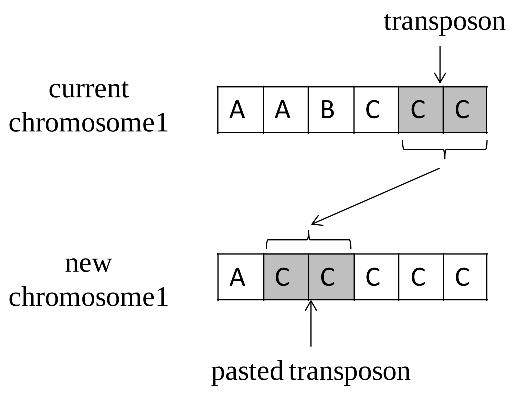


(a) Same chromosome

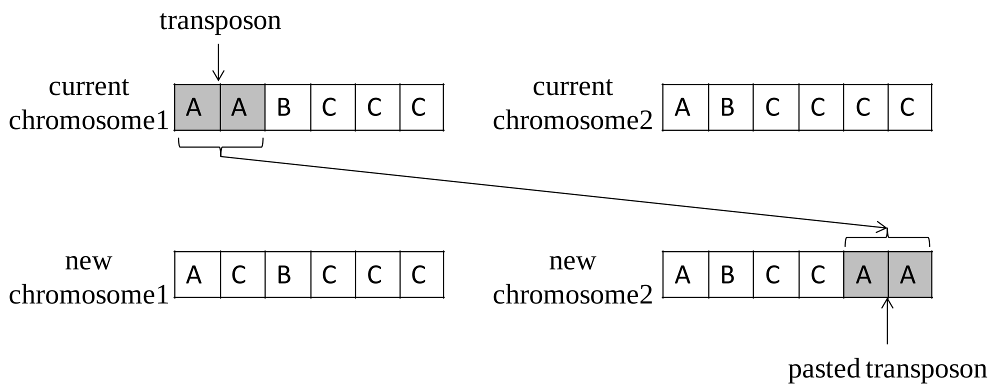


(b) Different chromosome

Fig. 2 Cut-and-Paste transposition



(a) Same chromosome



(b) Different chromosome

Fig. 3 Copy-and-Paste transposition

Recombination and Selection

The offspring population is combined with the current population and then the objective values of each individual are computed. Then, the rank assignment and selection are performed again to set the individuals of the next generation.

5. Modified non-dominated sorting genetic algorithm and modified jumping-gene shuffled frog leaping algorithm for cast sequence optimization, NSGA-II-JG

Solution representation

In order to apply the MOEA such as NSGA-II to the cast sequence optimization problem, first we need to represent sequence optimization problems. In this work, job-to-position representation is employed to represent the sequence problem (Rahimi-Vahed and Mirzaei, 2007). The value of the first element represents a grade scheduled in the first charge. The second value denotes a grade scheduled in the second charge and so on. Table 2 shows an example of solution representation. In this example, the first four charges are filled with products of grade A, and the following two charges are grade B. In the second cast, the first three charges are grade C, the next two charges are grade B and the last charge is grade D.

Table 2 Solution representation

Cast	**First cast**						**Second cast**					
Location	1	2	3	4	5	6	7	8	9	10	11	12
Grade	A	A	A	A	B	B	C	C	C	B	B	D

Initialized Population with the solution set computed from the Parallel SA method

Instead of initializing the decision variables for each individual *p*, the optimal solutions that can be obtained by using the weighted sum method is employed as some initial individuals. Therefore, first we minimize the following objective function by changing the weight vector $\mathbf{w} = [w_1, w_2, w_3]$:

$$\min \qquad f = w_1 f_1 + w_2 f_2 + w_3 f_3. \tag{11}$$

In order to maximize the diversity of initial solutions as much as possible, we set the weight vectors in Table 3 based on the sample weight vector pattern which is specified by user.

Table 3 Setting weight vector for weighted sum method

Weight vector pattern	w_1	w_2	w_3
1 (user specified)	$w_1^{balanced}$	$w_2^{balanced}$	$w_3^{balanced}$
2	$10w_1^{balanced}$	$w_2^{balanced}$	$w_3^{balanced}$
3	$w_1^{balanced}$	$10w_2^{balanced}$	$w_3^{balanced}$
4	$w_1^{balanced}$	$w_2^{balanced}$	$10w_3^{balanced}$

The weight sum method cannot find all the optimal solutions since it becomes computationally prohibitive with an increase in the number of optimal solutions that we need. In addition, the weight sum method cannot find Pareto optimal front for problems whose Pareto optimal front is nonconvex. Therefore in this work, initial solutions of problems #1, #2, # 3 and #4 are computed by solving weight sum optimization problems from Eq. (11), while the remaining individuals are initialized by JG transposition operation such as cut-and-paste operations or copy-and-paste operations (see Table 4).

Table 4 Computation of Initial Population

Individuals	Computation Strategy for Initial solutions
1	Minimize Eq. (11) with weight vector $\mathbf{w} = [w_1^{balanced}, w_2^{balanced}, w_3^{balanced}]$
2	Minimize Eq. (11) with weight vector $\mathbf{w} = [10w_1^{balanced}, w_2^{balanced}, w_3^{balanced}]$
3	Minimize Eq. (11) with weight vector $\mathbf{w} = [w_1^{balanced}, 10w_2^{balanced}, w_3^{balanced}]$
4	Minimize Eq. (11) with weight vector $\mathbf{w} = [w_1^{balanced}, w_2^{balanced}, 10w_3^{balanced}]$
5	Do cut-and-paste or copy-and-paste operation from individuals #1 to #4
6	Do cut-and-paste or copy-and-paste operation from individuals #1 to #4
7	Do cut-and-paste or copy-and-paste operation from individuals #1 to #4
…	…

Modified Traditional Genetic Operation for Sequence Optimization

After the traditional genetic operations are implemented in Eqs (9) and (10), the new position itself does not represent a sequence because the elements are continuous values. Similarly to our previous work (Mori and Mahalec, 2016), we employ the rule where the smallest position value of a charge is first selected and assigned grade A. Then, the second smallest position value of a charge is picked and assigned grade A if grade A is still available, otherwise assigned grade B, and so on. This procedure converts the position information into a sequence of grades.

Modified Jumping Gene Operation for Sequence Optimization

One of the major problems associated with using genetic operation for the sequence optimization problem is that solutions after genetic operations often do not meet the hard constraints. This is illustrated by the fact that GA has been very slow to gain acceptance for sequencing of operations since genetic operation involving one or two solutions very often does not result in a feasible solutions. Let us consider the very small scheduling problem as shown in Fig. 3(a). Now we have 1 charge of grade A, 1 charge of grade B and 4 charges of grade C and the current cast sequence is ACBCCC. After the copy and paste operation, we get the new cast sequence of ACCCCC. New cast sequence contains 1 charge of grade A, no charge of grade B and 5 charges of grade, which conflicts the number of charges for each grade we have. The simplest way to avoid infeasible solution is to discard the infeasible solution and then recompute a new chromosome after random reselection of a position of transposon. However, the regeneration of a new chromosome until obtaining feasible solution makes GA much slower and thus we should not do this approach for cast sequence optimization problem. To overcome this limitation, we propose a new method that makes chromosomes feasible by adding uniform random noise to all elements of solutions and deciding grades in accordance with the rule introduced by (Jarboui et al., 2008; Liu et al.,2007).

We will explain the algorithm by using the small example described by Fig. 2(b). In this example, we assume that we have 2 charges of grade A, 2 charges of grade B and 8 charges of grade C and the current sequence of two casts is ACBCCC and ABCCCC. After cut-and-paste transposition operation among these two casts, we obtain a new cast sequence of ACBCCC and ABCCAC. It is obvious that the new cast sequence violates the constraints since, for example, the new cast sequence includes 3 charges of grade A while we have only 2 charges of that grade. In order to avoid infeasible solutions, first we add uniform random noise [0 1] to each gene which may yield the following:

$$(1, 3, 2, 3, 3, 3, 3) \text{ and } (1, 2, 3, 3, 1, 3)$$
$$\rightarrow (1.23, 3.33, 2.57, 3.15, 3.72, 3.31) \text{ and } (1.24, 2.26, 3.32, 3.69, 1.41, 3.85) \qquad (12)$$

Then the smallest value of a gene (1.23) is first picked and assigned grade A. After that, the second smallest gene (1.24) of a charge is selected and assigned grade A if grade A is still available. Once grade A is exhausted, the smallest value of the remaining genes is selected (1.41) and assigned grade B if available, and so on. In this example, we finally obtain the new cast sequence ACCCCC and ABCCBC.

Local Search

Population based approaches such as MOEA give us useful information such as how many charges of each grade we should make for each cast, while the cast sequence is not optimal in many cases. Let us consider the example in Fig. 2 (b). After the cut-and-paste transposition operation, the new cast sequence becomes BCCBCC and ACCCAA, which is not a good sequence in terms of cast contamination cost, since same grade of charges should be continuously produced to reduce the amount of contaminated steel between different grades. As a result, solutions after genetic operators rarely become non-dominated solutions. In order to enhance the ability of genetic operation, a local search procedure is introduced after genetic operations. In the proposed algorithm, we optimize the sequence of charges within each cast such that contaminated steel is minimized. Although this optimization does not consider the other objectives, such as the minimum inventory capacity penalty and process capacity penalty, the change of charge sequence within same cast does not affect these objective values.

Traditional NSGA-II is shown in Fig. 4. Local search modifications to NSGA-II are depicted by Fig. 5. We designate as NSGA-II GO the algorithm obtained by adding local search to NSGA-II. The same ideas of local search can be included in other types of evolutionary algorithms, e.g. they can be applied to multi-objective shuffled frog leaping algorithm (Li et al., 2012) as shown in Fig. 6.

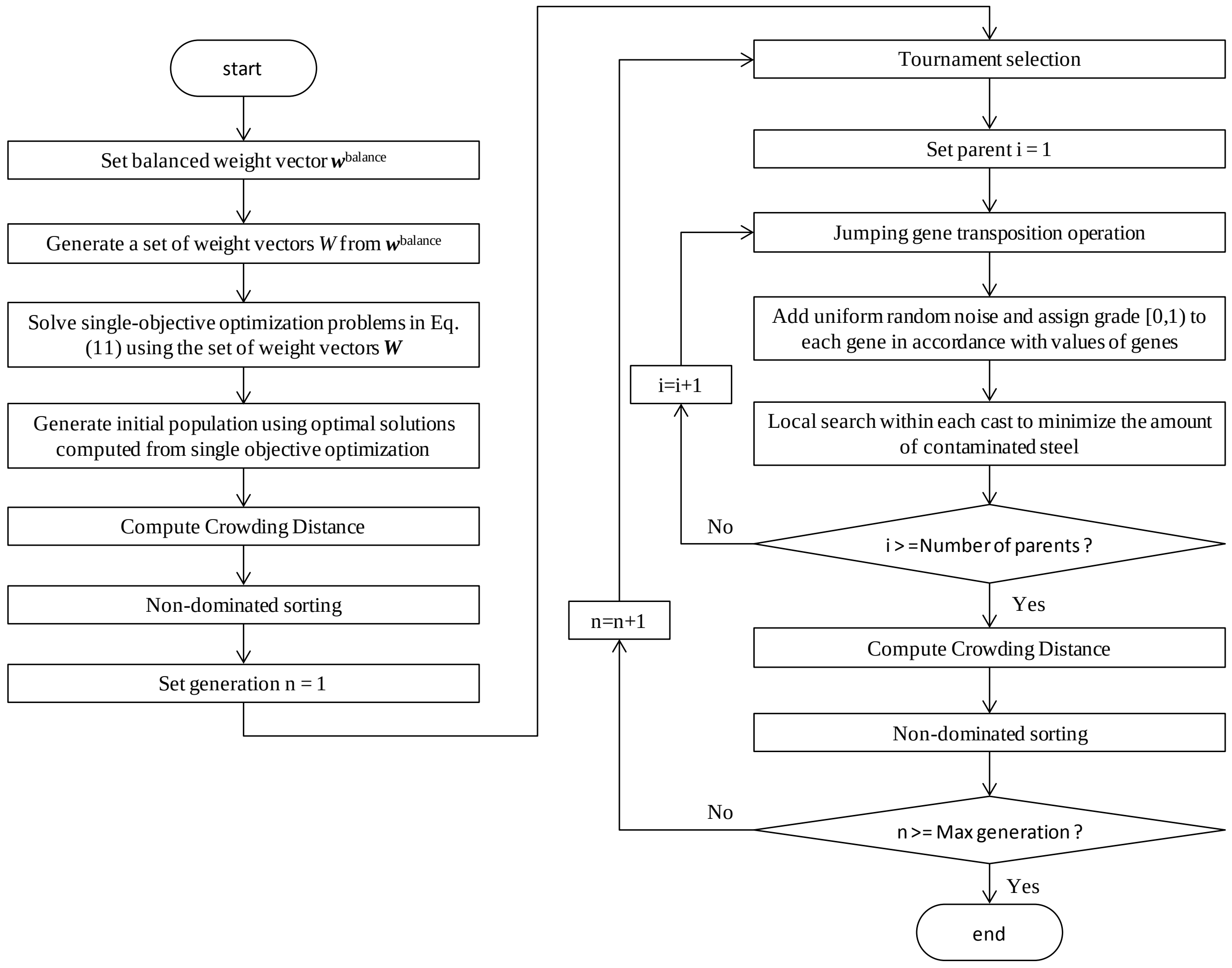

start
Set balanced weight vector $\boldsymbol{w}^{balance}$
Generate a set of weight vectors $\boldsymbol{W}$ from $\boldsymbol{w}^{balance}$
Solve single-objective optimization problems in Eq. (11) using the set of weight vectors $\boldsymbol{W}$
Generate initial population using optimal solutions computed from single objective optimization
Compute Crowding Distance
Non-dominated sorting
Set generation n = 1
Tournament selection
Set parent i = 1
Jumping gene transposition operation
Add uniform random noise and assign grade [0,1) to each gene in accordance with values of genes
Local search within each cast to minimize the amount of contaminated steel
i > =Number of parents ?
No
i=i+1
Yes
Compute Crowding Distance
Non-dominated sorting
n >= Max generation ?
No
n=n+1
Yes
end

Fig. 4 Flowchart of the proposed NSGA-II for a cast sequence problem (Algorithm 1, NSGA-II GO))

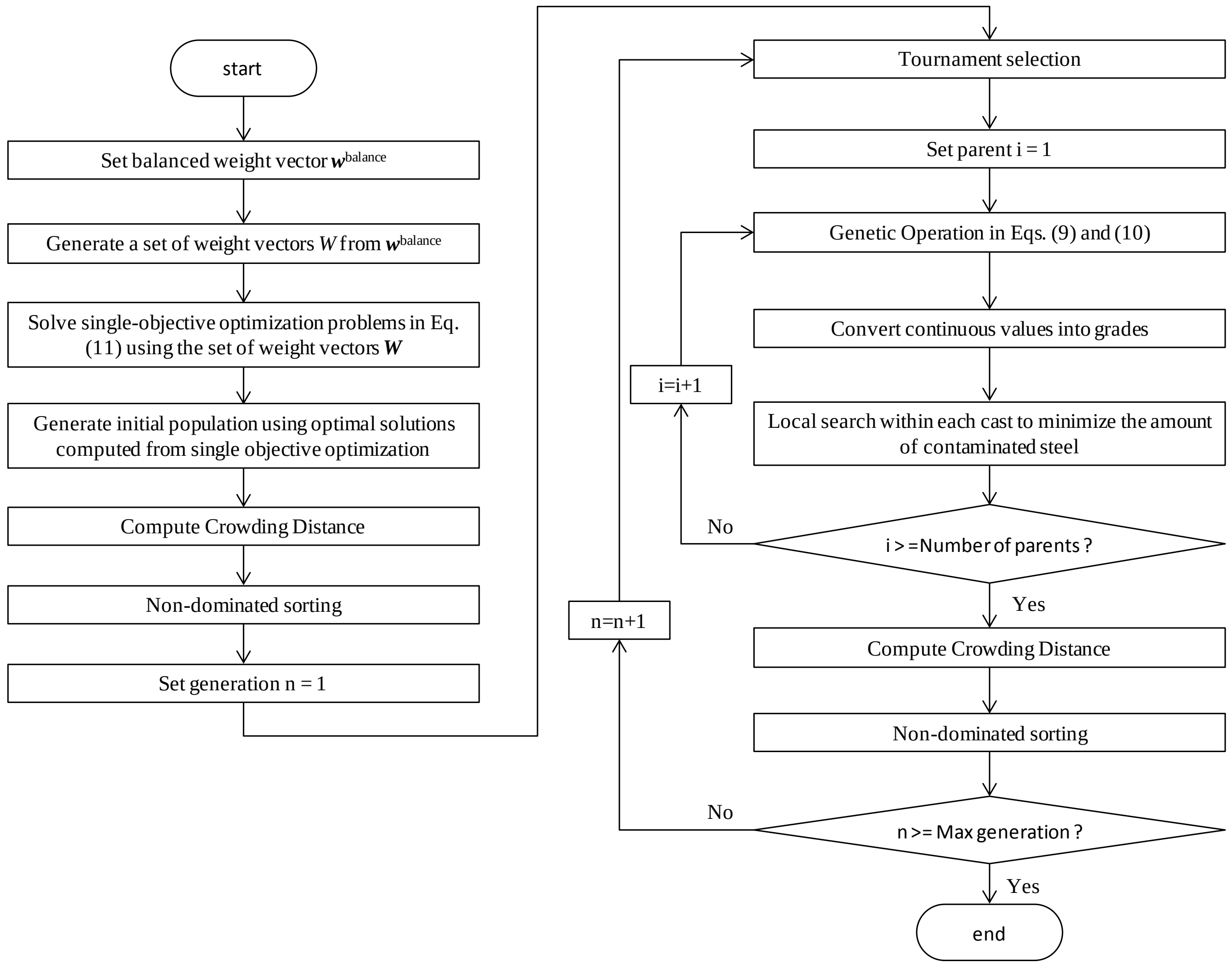

start
Set balanced weight vector $\boldsymbol{w}^{balance}$
Generate a set of weight vectors W from $\boldsymbol{w}^{balance}$
Solve single-objective optimization problems in Eq. (11) using the set of weight vectors $\boldsymbol{W}$
Generate initial population using optimal solutions computed from single objective optimization
Compute Crowding Distance
Non-dominated sorting
Set generation n = 1
Tournament selection
Set parent i = 1
Genetic Operation in Eqs. (9) and (10)
Convert continuous values into grades
Local search within each cast to minimize the amount of contaminated steel
i > =Number of parents ?
No
i=i+1
Yes
Compute Crowding Distance
Non-dominated sorting
n >= Max generation ?
No
n=n+1
Yes
end

Fig. 5 Flowchart of the proposed NSGA-II with JG transposition operation for a cast sequence problem (Algorithm 2, NSGA-II JP)

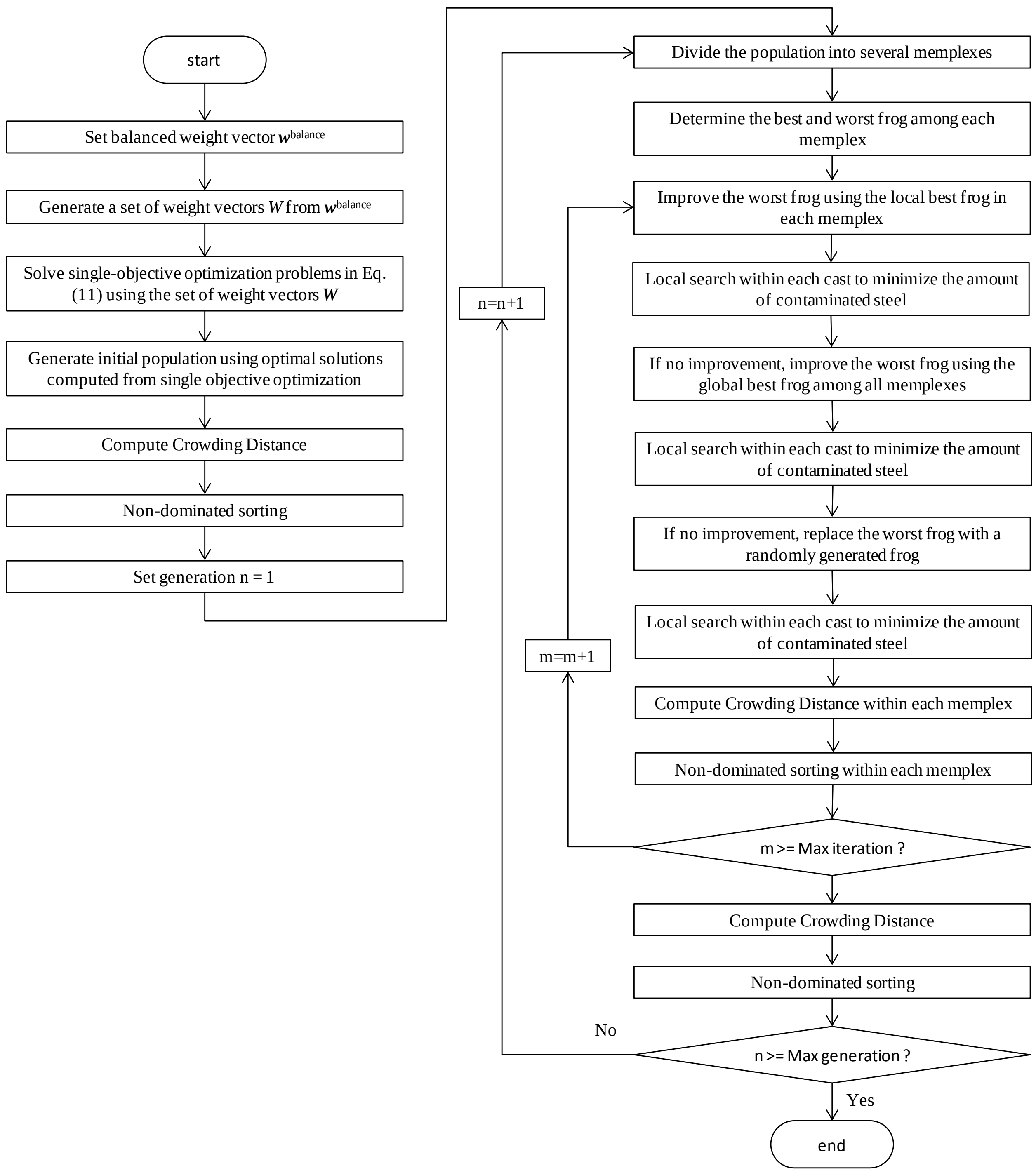

start
Set balanced weight vector $\boldsymbol{w}^{balance}$
Generate a set of weight vectors W from $\boldsymbol{w}^{balance}$
Solve single-objective optimization problems in Eq. (11) using the set of weight vectors $\boldsymbol{W}$
Generate initial population using optimal solutions computed from single objective optimization
Compute Crowding Distance
Non-dominated sorting
Set generation n = 1
Divide the population into several memplexes
Determine the best and worst frog among each memplex
Improve the worst frog using the local best frog in each memplex
Local search within each cast to minimize the amount of contaminated steel
If no improvement, improve the worst frog using the global best frog among all memplexes
Local search within each cast to minimize the amount of contaminated steel
If no improvement, replace the worst frog with a randomly generated frog
Local search within each cast to minimize the amount of contaminated steel
Compute Crowding Distance within each memplex
Non-dominated sorting within each memplex
m >= Max iteration ?
m=m+1
n=n+1
Compute Crowding Distance
Non-dominated sorting
n >= Max generation ?
No
Yes
end

Fig. 6 Flowchart of the proposed multi-objective shuffled frog leaping algorithm for a cast sequence problem (Algorithm 3, MOSFL)

6. Case Studies

In this section, we compare the results of the three algorithms presented in Figures 4, 5, and 6: (i) NSGA-II with local search, i.e. NSGA-II GO, (ii) NSGAS-II with gene transposition operation, i.e. NSGA-II-JG, and (iii) multi-objective shuffle frog leaping algorithm with local search (named MOSFL), respectively. All the case studies have been computed on a machine running Windows 2007 Professional (Intel® Core(TM) i7-4930 CPU @ 3.40GHz and 16.0 Gb RAM). All three algorithms use initial solutions which are obtained by solving 4 single optimization problems as shown in Table 4. Furthermore, as described in the previous section, the local search procedure which optimizes the sequence of charges within each cast is included in these three algorithms. The number of generations and populations size are set to be 300 and 300 respectively for all test examples we considered in this study.

All problems considered in this paper, including both small test problems and real-steel production scheduling problem, were solved by weighted sum methods in our previous paper (Mori & Mahalec, 2015).

6.1 Small test problems

The small test problems are used to compare three algorithms of NSGA-II-GO, MOSHL and NSGA-II-JG by investigating each solution one by one. Table 5 shows the size of full-space MILP model and Table 6 show the quality loss matrix $C(i, j)$.

Table 5 Size of three small test problems

Example	**Horizon [days]**	**Total Term [days]**	**# Casts**	**# Grades**	**Full space MILP model**	
					# Conts	**# Bins**
1	7	10	7	3	526	126
2	5	7	5	7	1,537	210
3	6	8	6	7	1,857	252

Horizon=length of days for which the production scheduling are made, total term=length of days for which the production scheduling deals with the orders, #Casts = number of casts, #Grades = number of casts, #Conts = number of continuous variables, and #Bins = number of binary variables

Table 6 Quality loss matrix $C(i, j)$

(a) Example #1

Grade	1	2	3
1	0	1	2
2	2	0	2
3	1	3	0

(b) Example #2 and #3

Grade	1	2	3	4	5	6	7
1	0	1	2	1	3	1	1
2	2	0	1	1	2	2	3
3	2	1	0	1	1	1	1
4	1	2	2	0	1	2	1
5	1	1	1	2	0	1	2
6	1	2	1	2	2	0	2
7	2	2	1	1	3	2	0

(c) Example #4

Grade	1	2	3	...	21	22	23
1	0	1	1	...	1	1	1
2	2	0	1	...	1	1	1
3	2	2	0	...	1	1	1
...	...	...	...	..	...	...	...
21	2	2	2	...	0	1	1
22	2	2	2	...	2	0	1
23	2	2	2	...	2	2	0

(c) Example #5

Grade	1	2	3	...	34	35	36
1	0	1	1	...	1	1	1
2	2	0	1	...	1	1	1
3	2	2	0	...	1	1	1
...	...	...	...	..	...	...	...
34	2	2	2	...	0	1	1
35	2	2	2	...	2	0	1
36	2	2	2	...	2	2	0

Example #1

Non-dominated solutions computed by NSGA-II-GO, MOSHL and NSGA-II-JG are shown in Table 7 to 9 respectively. Further, these non-dominated solutions are plotted in Figs. 7 and 8. It can be seen that NSGA-II-JG computes much better Pareto optimal frontier than the other two methods in terms of the number of non-dominated solutions and the solution diversity.

Table 7 Non-dominated solutions computed by NSGA-II-GO for example #1

No	f_1	f_2	f_3	**Initial Sol ?**
1	3	180	50	Yes
2	3	720	20	
3	4	180	40	
4	4	220	30	
5	4	580	20	
6	4	2070	10	
7	5	60	0	Yes
8	6	0	10	Yes
9	7	0	0	Yes

Table 8 Non-dominated solutions computed by MOSFL for example #1

No	f_1	f_2	f_3	**Initial Sol ?**
1	3	180	50	Yes
2	5	60	0	Yes
3	6	0	10	Yes
4	7	0	0	Yes

Table 9 Non-dominated solutions computed by NSGA-II- JG for example #1

No	f_1	f_2	f_3	**Initial Sol ?**
1	1	640	100	
2	1	940	40	
3	2	320	110	
4	2	420	100	
5	2	440	20	
6	2	1040	10	
7	2	1890	0	
8	3	180	50	Yes
9	3	240	30	
10	3	640	10	
11	3	710	0	
12	4	120	40	
13	4	210	30	
14	4	320	0	
15	5	60	0	Yes
16	6	0	10	Yes
17	7	0	0	Yes

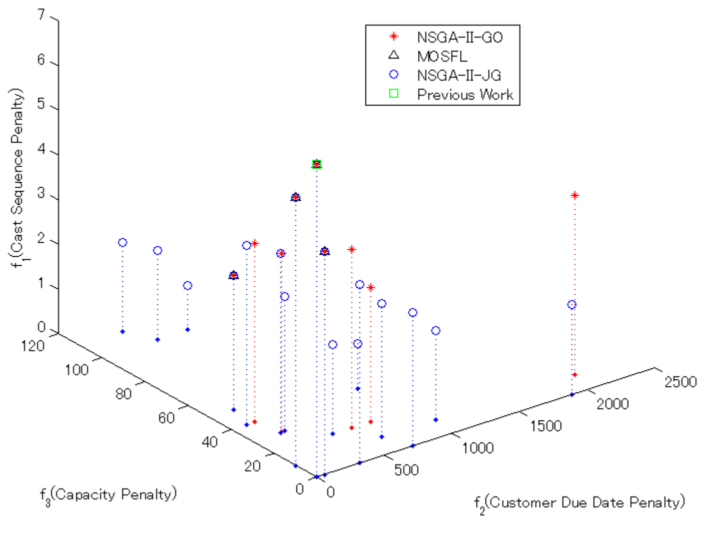


Fig. 7 3D-plot comparison of non-dominated solutions for example #1

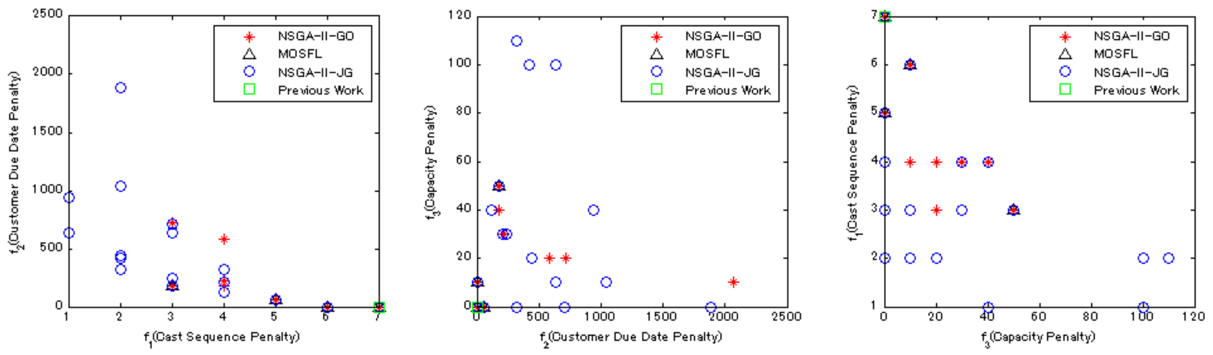


Fig. 8 2D-plot comparison of non-dominated solutions for example #1

When comparing NSGA-II-GO and NSGA-II-JG transposition operation, some non-dominated solutions in NSGA-II-GO are dominated by the solutions in NSGA-II-JG. For instance, #2 solution in NSGA-II-GO is dominated by #10 and #11 solutions. #3 solution in NSGA-II-GO is also dominated by #12 solution. These results mean that the NSGA-II-GO cannot find richer non-dominated solutions in terms of optimality while NSGA-II-JG can find better non-dominate solutions.

As for MOSFL, the number of non-dominated solutions is quite small. In fact, all non-dominated solutions are initial solutions computed by solving problems described in Table 4. Therefore, in this example MOSFL does not work at all.

In order to check the robustness of our proposed algorithm against random number generator, NSGA-II-GO, MOSHL and NSGA-II-JG are implemented with using different random seed. The results are shown in Fig. 9 and 10. As these figures indicate, the random number generator does not have much effect on the quality of the Pareto optimal solutions.

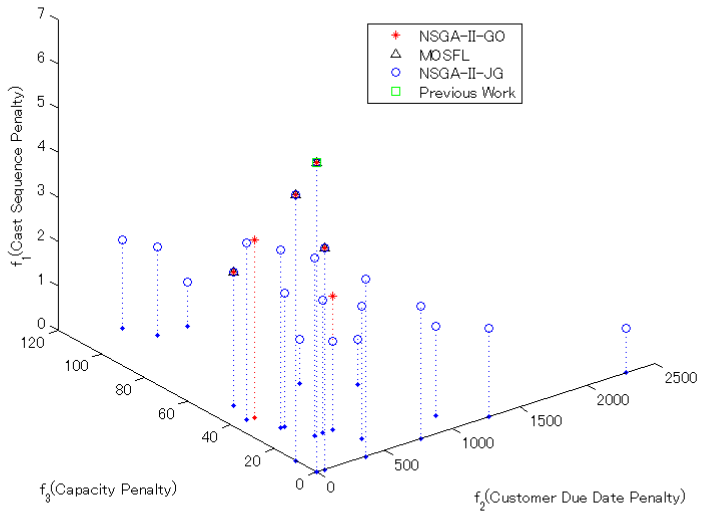


Fig. 9 3D-plot comparison of non-dominated solutions for example #1 (random seed = 2)

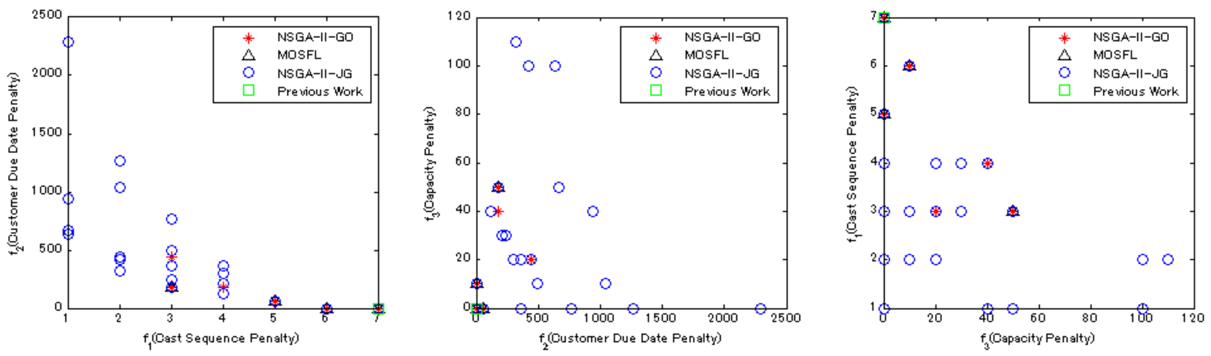


Fig. 10 2D-plot comparison of non-dominated solutions for example #1 (random seed = 2)

Sets of non-dominated solutions computed from above three algorithms with randomized initial solutions are plotted in Figs. 11 and 12. It can be seen that without good initial solutions the computed non-dominated solutions become worse. These results also indicate that NSGA-II-JG performs much better than the traditional genetic operation. In addition, it is clear that MOSFL is not able to find rich non-dominated solutions set compared with NSGA-II-GO and NSGA-II-JG for sequence optimization problems in terms of optimality. We also note that NSGA-II-JG also computes one solution which exactly equals to the result of the solutions obtained by using the parallel SA with shuffle frog leaping algorithm in our work with single objective function optimization (Mori & Mahalec, 2015).

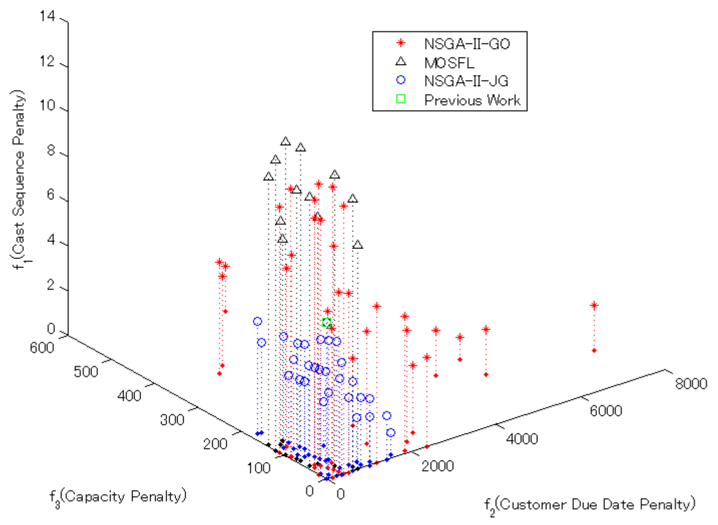


Fig. 11 3D-plot comparison of non-dominated solutions with randomized initial solutions for example #1

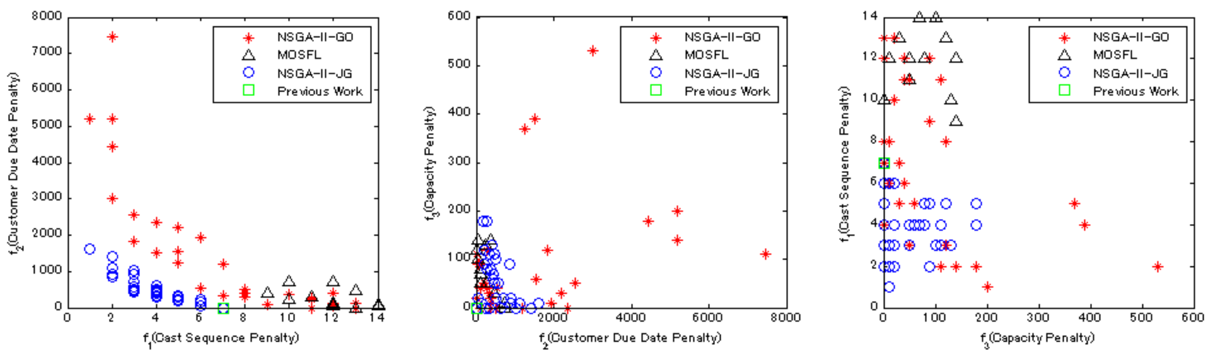


Fig. 12 2D-plot comparison of non-dominated solutions with randomized initial solutions for example #1

Example #2

Shown in Tables 10 to 12 are non-dominated solution obtained by NSGA-II-GO, MOSFL and NSGA-II-JG ; these non-dominated solutions are plotted in Figs. 13 and 14. It can be seen that all of these three methods are able to find 2 non-dominated solutions. That is because the number of non-dominated solutions in this example is very small. For example, in Table 4, f_1 of #2 solution is 3 while the other objective function values are zero. It means that the there are no non-dominated solutions whose f_1 is greater than 3.

Table 10 Non-dominated solutions computed by NSGA-II –GO for example #2

No	f_1	f_2	f_3	**Initial Sol ?**
1	2	162.69	0	Yes
2	3	0	0	Yes

Table 11 Non-dominated solutions computed by MOSFL for example #2

No	f_1	f_2	f_3	**Initial Sol ?**
1	2	162.69	0	Yes
2	3	0	0	Yes

Table 12 Non-dominated solutions computed by NSGA-II-JG for example #2

No	f_1	f_2	f_3	**Initial Sol ?**
1	2	162.69	0	Yes
2	3	0	0	Yes

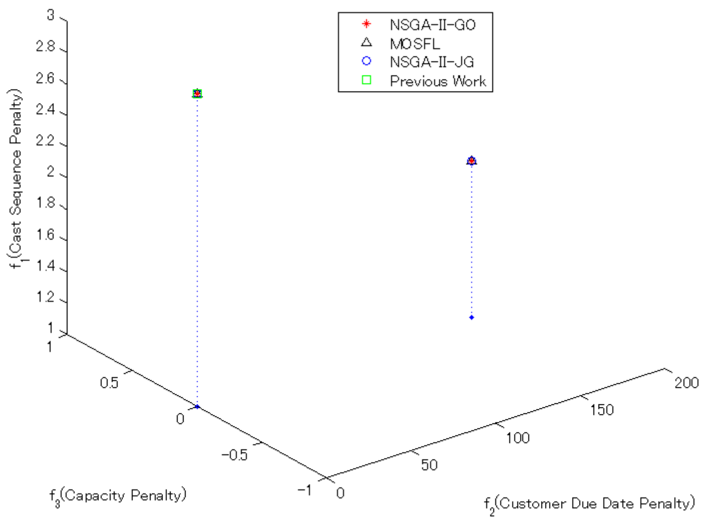


Fig. 13 3D-plot comparison of non-dominated solutions for example #2

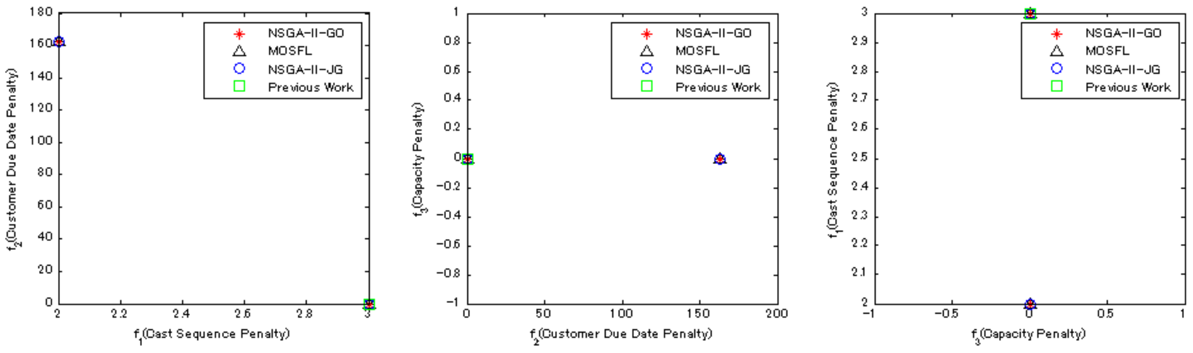


Fig. 14 2D-plot comparison of non-dominated solutions for example #2

Example #3

Non-dominated solutions computed by NSGA-II-GO, MOSFL and NSGA-II-JG are shown in Tables 13 to 15, respectively. Corresponding plots of these non-dominated solutions are presented in Figs. 15 and 16. Similarly to the previous test case, all three methods find two non-dominated solutions due to the same reasons as for example #2.

Table 13 Non-dominated solutions computed by NSGA-II-GO for example #3

No	f_1	f_2	f_3	**Initial Sol ?**
1	3	4.8029	0	Yes
2	4	0	0	Yes

Table 14 Non-dominated solutions computed by MOSFL for example #3

No	f_1	f_2	f_3	**Initial Sol ?**
1	3	4.8029	0	Yes
2	4	0	0	Yes

Table 15 Non-dominated solutions computed by NSGA-II-JG for example #3

No	f_1	f_2	f_3	**Initial Sol ?**
1	3	4.8029	0	Yes
2	4	0	0	Yes

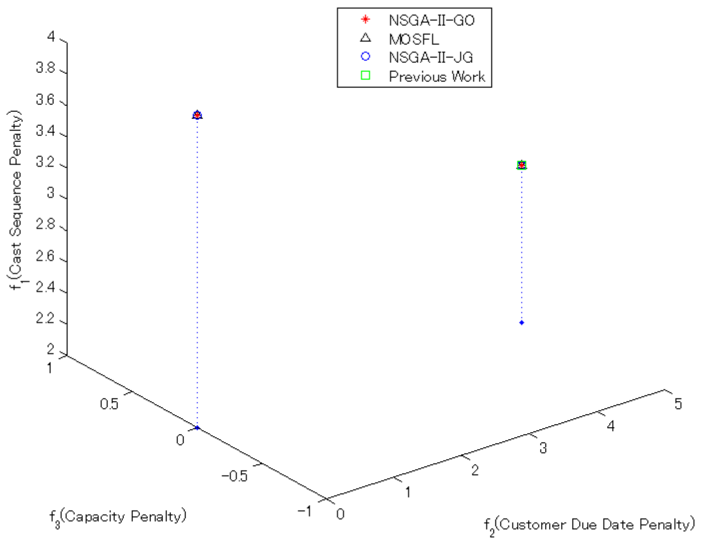


Fig. 15 3D-plot comparison of non-dominated solutions for example #3

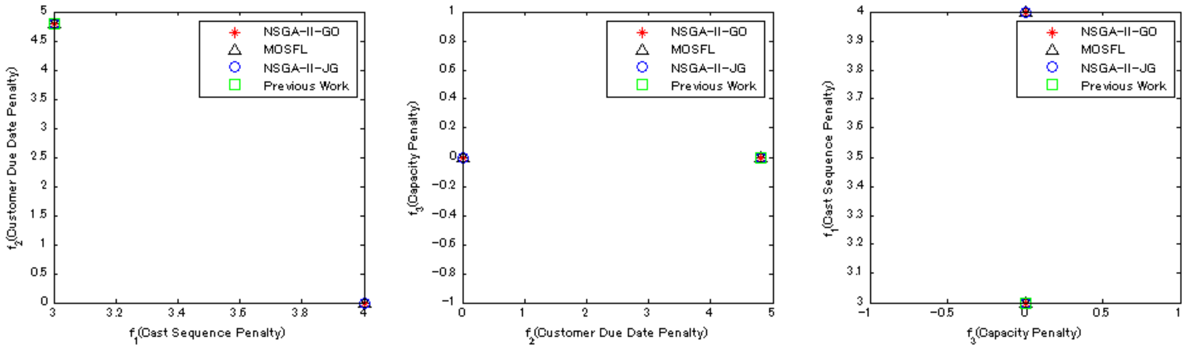


Fig. 16 2D-plot comparison of non-dominated solutions for example #3

CPU times and the number of non-dominated solutions for all the above small examples are provided in Table 16. There is no large difference in CPU times among three algorithms; the CPU times are from approximately 700 seconds to approximately 1000 seconds, which is much smaller than the times required by the weighted sum methods to obtain the same number of non-dominated solutions.

Table 16 Comparison of CPU time and number of non-dominated solutions for small examples

Example	Algorithm	CPU time [s]			# Non-Dominated Solutions
		Computation of Initial solution	MOEA	Total	
1 (random seed = 1)	NSGA-II-GO	97	817	914	9
	MOSFL		834	931	4
	NSGA-II-JG		746	843	17
1 (random seed = 2)	NSGA-II-GO	97	938	1035	6
	MOSFL		1085	1182	4
	NSGA-II-JG		887	984	21
2 (random seed = 1)	NSGA-II-GO	89	701	790	2
	MOSFL		721	810	2
	NSGA-II-JG		678	767	2
3 (random seed = 1)	NSGA-II-GO	227	784	1011	2
	MOSFL		1,028	1255	2
	NSGA-II-JG		803	1030	2

#Non-Domi Sol = number of non dominated solutions.

6.2 Industrial scale steel-plate production scheduling problems

The real-world steel-plate production data have been used to examine the effectiveness of the proposed optimization algorithms. Data used in the following two examples have been derived from the industrial data in order to protect the proprietary data while maintaining the structure of the original data. We will present two examples of scheduling one week of a steel-plate production having 23 and 36 steel grades, respectively. The production capacities and the size of full space MILP models are shown in Table 17 and 18 respectively. The number of pots per cast is 6, the size of the pot is 200 ton. The plan operates at full capacity of 3600 ton per day, therefore the number of casts per day is 3.

Table 17 Process capacity

Mill	Processes	Production Capacity
Steel making mills	# casts	3[casts per day]
	# charges	6[charges per cast]
	Size of a Charge	200[ton/charge]
Steel plate making mills	Process A	350[ton/day]
	Process B	700[ton/day]
	Process C	1,200[ton/day]
	Process D	500[ton/day]
	Process E	400[ton/day]
	Process F	100[ton/day]
	Process G	1,500[ton/day]
	Process H	150[ton/day]
	Process I	100[ton/day]

#casts = number of casts and #charges = number of charge

Table 18 Size of two industrial test problems

Test problem	Horizon [days]	Total term	# Casts	# Grades	Full space MILP model	
					# Cont. Var.	# Binary Var.
4	7	13	21	23	59,270	2,898
5	7	13	21	36	141,866	4,536

Horizon=length of days for which the production scheduling are made, total term=length of days for which the production scheduling deals with the orders, #Casts = number of casts, #Grades = number of casts, #Conts = number of continuous variables, and #Bins = number of binary variables

Example #4

Example #4 determines the cast schedule for 23 grades of steel for one week scheduling horizon. Solution from the top level LP model is presented in Fig. 17, where each distinctly colored quadrangle corresponds to a specific pot of some grade of steel. Further, in Fig 18 we show the cast schedule of the initial solution where $\mathbf{w} = [w_1^{balanced}, w_2^{balanced}, w_3^{balanced}]$.

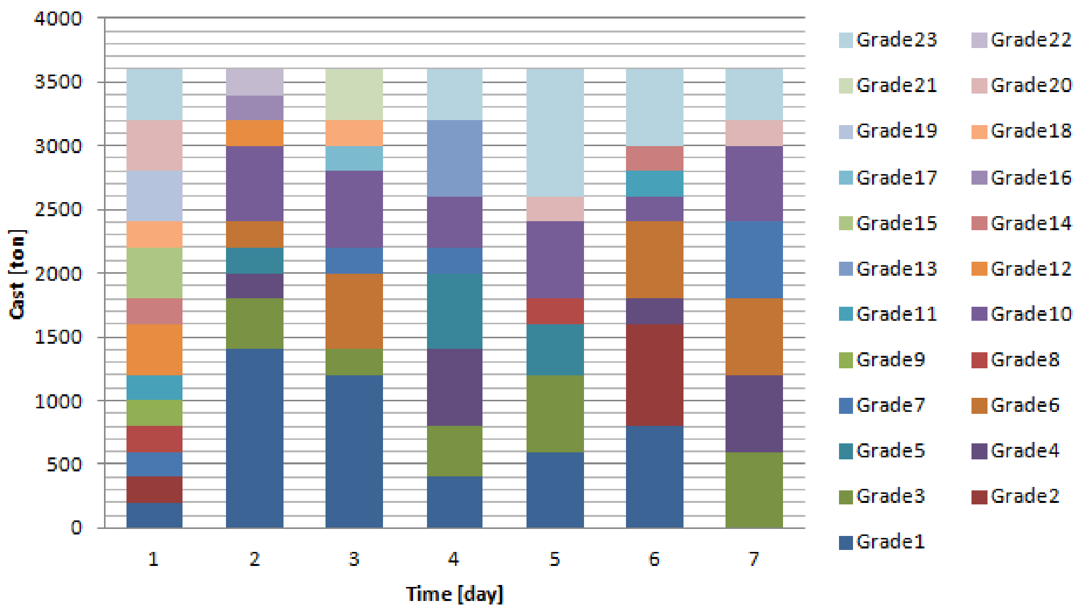


Fig. 17 Top level LP solution (Approximate scheduling)

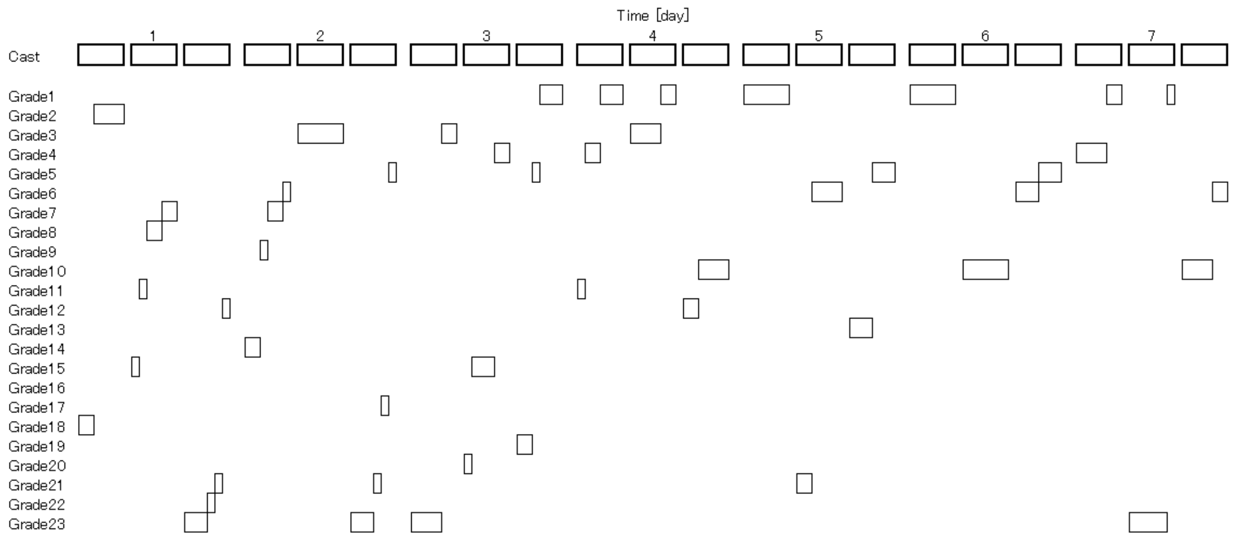


Fig. 18 Cast schedule computed by two-level algorithm with the modified parallel SA for example #4 ($\mathbf{w} = [w_1^{balanced}, w_2^{balanced}, w_3^{balanced}]$)

Scheduling task is to optimize the sequencing of these pots with respect to the three objective functions, Equations 4, 5 and 6.

During initialization, modified parallel simulated annealing modified with shuffled frog leaping algorithm (Mori and Mahalec, 2016) is employed to solve four single-objective optimization problems described in Table 4. These solutions are employed as initial solutions for the proposed multi-objective evolutionary algorithms for cast sequence problems, i.e.NSGA-II-GO, MOSFL and NSGA-II-JG. It should be noted that such initialization is essential for good performance in solving the multi-objective scheduling problem.

The non-dominated solutions from above three algorithms are plotted in Figs. 19 and 20. It can be observed that NSGA-II-JG is able to obtain a wide range of non-dominated solutions while the non-dominated solutions of NSGA-II-GO are distributed in the area with low cast sequence penalty and high customer due date penalty.

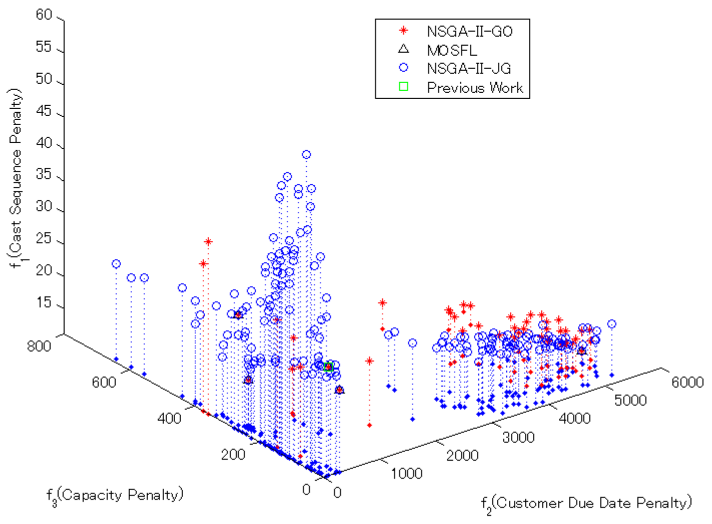


Fig. 19 3D-plot comparison of non-dominated solutions for example #4

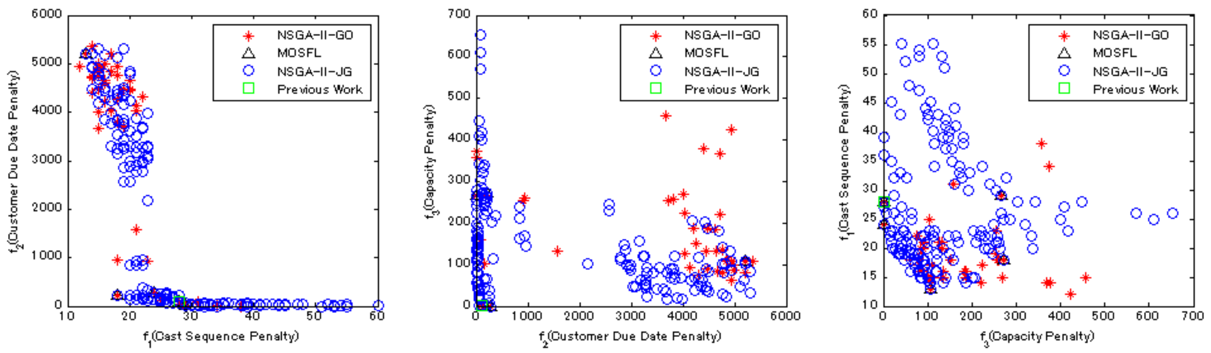


Fig. 20 2D-plot comparison of non-dominated solutions for example #4

CPU times and the number of non-dominated solutions are shown in Table 19. We note that 18 out of 45 solutions of NSGA-II-GO are dominated by NSGA-II-JG and thus NSGA-II-GO obtained only 27 non-dominated solutions which are non-dominated with the solution obtained by NSGA-II-JG. Meanwhile, since any solutions of NSGA-II-JG are not dominated by NSGA-II-GO, 163 solutions from NSGA-II-JG are non-dominated with the solution from NSGA-II with GO.

Table 19 Comparison of CPU time and number of non-dominated solutions for industrial examples

Test set	Algorithm	CPU time [s]			# of Non-Dominated Solutions
		Computation of Initial solution	MOEA	Total	
4	NSGA-II-GO	1905	2647	4552	43
	MOSFL		1362	3267	5
	NSGA-II-JG		2072	3977	163
5	NSGA-II-GO	2951	2356	5307	79
	MOSFL		1239	4190	6
	NSGA-II-JG		2044	4995	173
	NSGA-II-GO & NSGA-II-JG		-	-	250

Finally, let's note that the solution obtained by for the weighted sum single objective via parallel simulated annealing modified by shuffled frog algorithm (Mori and Mahalec, 2016) is one of the non-dominated solutions.

Example #5

Example #5 has 36 grades of steel which need to be scheduled, which increase the computational difficulties in comparison to test example #4. Similarly to example #4, we employ parallel simulated annealing with shuffle frog leaping algorithm in order to obtain rich initial solutions described in Table 4. Then, these initial solutions are employed for NSGA-II-GO, MOSFL and NSGA-II-JG. Figs. 21 and 22 show the non-dominated solutions from above three algorithms. The number of non-dominated solutions is provided in Table 20. It can be observed that NSGA-II-GO and NSGA-II-JG are able to obtain richer non-dominated solutions than MOSFL. Comparing the results from NSGA-II-GO and NSGA-II-JG, 23 out of 79 solutions of NSGA-II-GO are dominated by the solutions of NSGA-II-JG and thus NSGA-II-JG obtained 56 non-dominated solutions which are non-dominated by the solution obtained by NSGA-II-JG. Since only 6 solutions of NSGA-II-JG are dominated by NSGA-II-GO, we see that 167 solutions from NSGA-II-JG are non-dominated by the solutions from NSGA-II with GO operations. These results show that NSGA-II-JG obtains significantly richer set of non-dominated solutions than NSGA-II-GO. It should be noted that NSGA-II-JG cannot find some non-dominated solutions which can be obtained by NSGA-II-GO. In fact, NSGA-II-GO is able to find the solutions whose objective value f_1 is smaller than any solutions obtained by NSGA-II-JG. In this case, we can easily combine these two sets of non-dominated solutions and run non-domination sorting of the combined solution sets. The combined solutions (total of 250) produced by NSGA-II-GO and NSGA-II-JG are plotted in Figs. 23 and 24 while CPU times and the number of non-dominated solutions are shown in Table 20. Once again, the solution obtained by for the weighted sum single objective via parallel simulated annealing modified by modified shuffled frog algorithm is one of the non-dominated solutions.

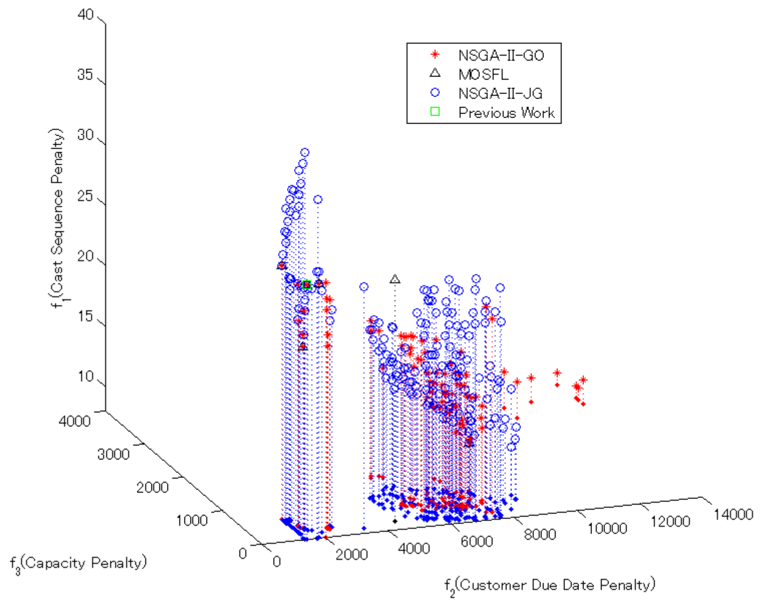


Fig. 21 3D-plot Comparison of non-dominated solutions for example #5

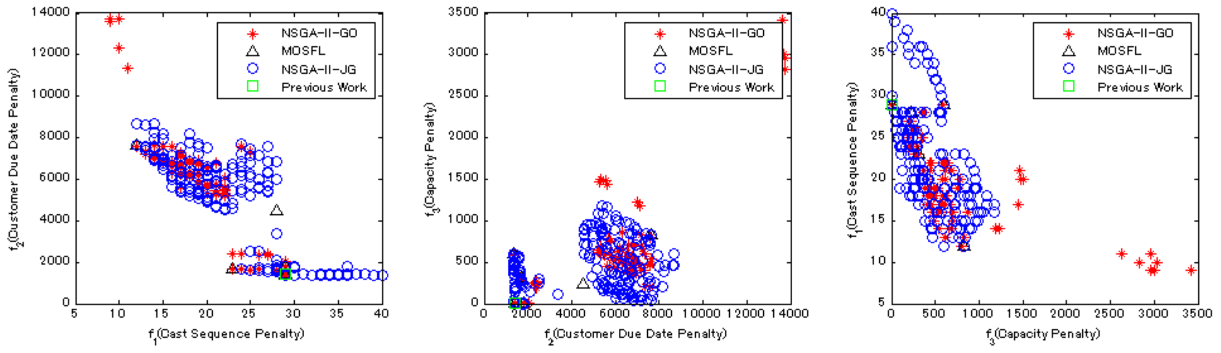


Fig. 22 2D-plot Comparison of non-dominated solutions for example #5

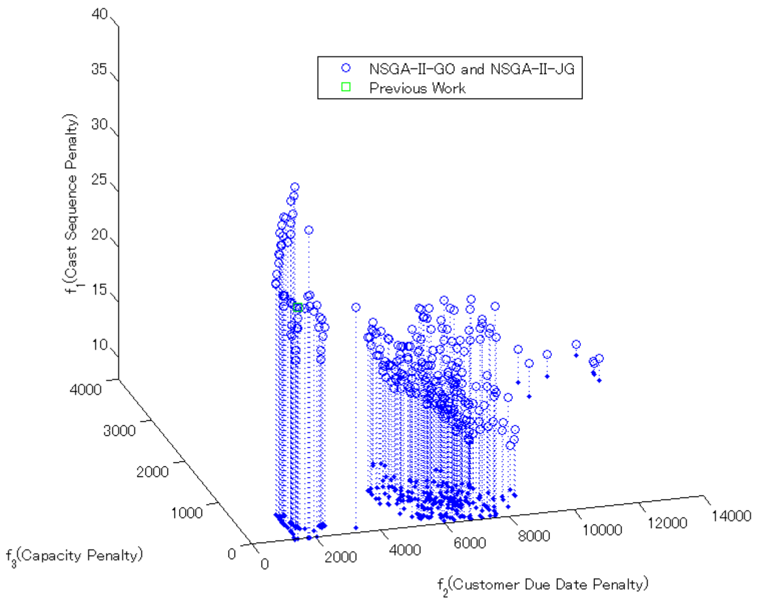


Fig. 23 3D-plot of non-dominated solutions computed by NSGA-II-GO & NSGA-II-JG for example #5

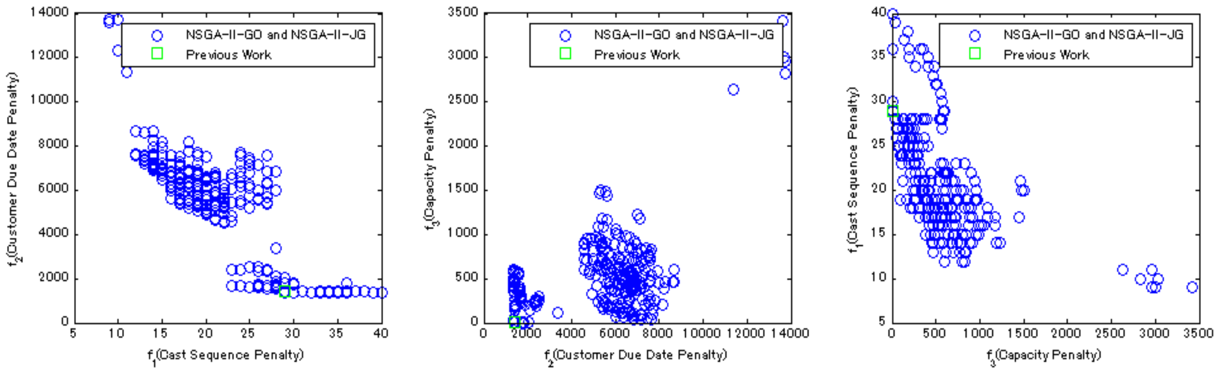


Fig. 24 2D-plot of non-dominated solutions computed by NSGA-II-GO & NSGA-II-JG for example #5

7. Conclusions

Continuous casting of steel plates produces a large number of different grades of steel which are rolled into plates per customer orders. Each grade of steel is produced in fixed size (pot size) batches. In order to eliminate contamination at the interface of two steel grades, it is desirable to continue producing the same grade of steel as long as possible. Production should meet desired delivery dates and also maintain inventories of steel plates above some minimum levels. Our previous work (Mori and Mahalec, 2016) has shown that the scheduling problem considered here cannot be solved for industrial scale problems by using MILP solvers. Moreover, further reflection on our prior work has lead us to conclude that a scheduler needs to understand the trade-offs between various scheduling objectives. Considerations of the trade-off between the cast sequence penalty, customer due date penalty and production capacity penalty is enabled via non-dominated set of solutions of the multi-objective formulation of the schedule optimization. Similarly to the single objective version, the multi-objective version of the industrial scale problem cannot be solved by MILP solvers. Hence, we have introduced in this work two new multi-objective evolutionary algorithms (MOEA) for solving the multi-objective scheduling of the steel plates scheduling:

(i) NSGA-II GO is non-dominated sorting genetic algorithm NSGA-II modified by including the local search, and

(ii) NSGA-II-JG is a non-dominated sorting genetic algorithm NSGA-II modified by including the local search and jumping gene transposons.

In order to apply the population based MOEA, job-to-position representation is employed to represent the sequence. In order to generate a large diverse Pareto optimal set, instead of using the traditional genetic operators such as SBX, we employ the cut-and-paste and copy-and-paste transposons. In addition, local search procedure within a cast is used after genetic operations in order to improve exploration of the neighboring solutions. Since good initialization is important for large scale problems, we have employed the modified parallel simulated annealing with shuffled frog leaping to generate the initial solutions (Mori and Mahalec, 2016).

Real life steel production process data have been used to examine the effectiveness of all three multi-objective scheduling algorithms. The computational results demonstrate that the proposed NSGA-II-JG algorithm finds the largest set of non-dominated solutions. NSGA-II GO algorithm finds a smaller set of non-dominated solutions, while MOSFL exhibits significantly worse performance than both NSGO-II GO and NSGA-II-JG. Our computational results show that both NSGA-II-JG and NSGA-II GO perform much better than the widely used multi-objective shuffled frog leaping algorithm and can solve problems of the size not tractable by the MILP solvers. Since some of the non-dominated solutions computed by NSGA-II GO are different that those computed by NSGA-II JG, the best approach is to compute the schedules by using both of these new algorithms and combine their sets of non-dominated solutions.

Proposed approach is able to compute a large set of non-dominated solution of the industrial scale multi-objective scheduling of steel plates production, thereby enabling good understanding of the trade-offs between different objectives.

.

References

Mori J, Mahalec V. Planning and scheduling of steel plates production. Part II: Scheduling of Continuous Casting. Comput Chem Eng. 2016, doi:10.1016/j.compchemeng.2016.01.020

Goh CK, Tan K. C. Evolutionary Multi-objective Optimization in Uncertain Environments: Issues and Algorithms. Studies in Computational Intelligence. Berlin Heidelberg: Springer-Verlag. 2009.

Stadler W. A survey of multicriteria optimization or the vector maximum problem, part I: 1776–1960. J. Optim. Theory Appl. 1979; 29(1): 1-52.

Das I, Dennis J. E. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. Structural optimization. 1997; 14(1): 63-69.

Chankong V, Haimes Y. Y. Multiobjective Decision Making: Theory and Methodology. New York: Elsevier Science. 1983.

Liu P, Pistikopoulos E. N, Li Z. A multi-objective optimization approach to polygeneration energy systems design. AIChE J. 2009; 56(5): 1218-1234.

Subramanyana K, Diwekara U, Zitney S. E. Stochastic modeling and multi-objective optimization for the APECS system. Comput. Chem. Eng. 2011; 35(12): 2667-2679.

Kim I. Y, Weck O. L. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. Struct Multidiscip O. 2005; 29(2): 149-158.

Kim I. Y, Weck O. L. Adaptive weighted sum method for multiobjective optimization: a new method for Pareto front generation. Struct Multidiscip O. 2005; 31(2): 105-116.

Das I, Dennis J. E. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. Structural optimization. 1998; 8(3): 631-657.

Siddiqui S, Azarm S, Gabriel S. A. On improving normal boundary intersection method for generation of Pareto frontier. Struct Multidiscip O. 2012; 46(6): 839-852.

Dutta, G., Fourer, R. A survey of mathematical programming application in integrated steel plants, Interactive trans. ORMS, 2001; 4: 1

Tang, L., Liu, J., Rong, A., Yang, Z. A review of planning and scheduling systems and methods for integrated steel production. European J of Oper. Res. 2001; 133: 1-20

Harjunkosi, I, Grossman, I. E. A decomposition approach for the scheduling of a steel plant production. Comput. Chem. Eng. 2001; 25(11-12): 1647-1660

Bellabdaoui, A., Teghem, J. A mixed-integer linear programming model for the continuous cast planning, Int. J. Production Economics, 2006 ; 104(2) 260-270

Alam M. S, Hossain M. A, Algoul S, Majumader M. A. A, Al-Mamun M. A, Sexton G, Phillips R. Multi-objective multi-drug scheduling schemes for cell cycle specific cancer treatment. Comput Chem Eng. 2013; 58(11): 12-32.

Sharma S, Rangaiah G. P. An improved multi-objective differential evolution with a termination criterion for optimizing chemical processes. Comput Chem Eng. 2013; 56(13): 155-173

Schaffer J. D. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. Proceedings of the 1st International Conference on Genetic Algorithms. 1985:93-100

Fonseca C. M, Fleming P. J. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In: Proceedings of the 5th International Conference on Genetic Algorithms. 2003: 416-423.

Srinivas N, Deb K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol Comput. 1994; 2(3): 221-248.

Horn J, Nafpliotis N, Goldberg D. E. Multiobjective Optimization Using the Niched Pareto Genetic Algorithm. Illinois Genetic Algorithms Laboratory Report No. 93005.

Zitzler E, Thiele L. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Computer Engineering and Communication Networks Lab Report. 1998; 43.

Knowles J. D, Corne D. W. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evol Comput. 2000; 8(2): 149-172.

Corne D. W, Knowles J. D, Oates M. J. The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization. In: Proceedings of the Parallel Problem Solving from Nature VI Conference. 2000:839-848.

Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput. 2002; 6(2): 182-197.

Erickson M, Mayer A, Horn J. The Niched Pareto Genetic Algorithm 2 Applied to the Design of Groundwater Remediation Systems. In: Proceedings of First International Conference on Evolutionary Multi-Criterion Optimization. 2001: 681-695.

Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. Computer Engineering and Communication Networks Lab Report. 2001; 103.

Zitzler E, Thiele L, Laumanns M, Fonseca C. M, Fonseca V. G. Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput. 2003; 7: 117-132.

Abbass H. A, Sarker R, Newton C. PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems. In: Proceedings of the Congress on Evolutionary Computation 2001. 2001; 2: 971-978.

Abbass H. A. The self-adaptive Pareto differential evolution algorithm. In: Proceedings of the Congress on Evolutionary Computation 2002. 2002; 1: 831-836.

Zaharie D, Petcu D. Adaptive Pareto Differential Evolution and Its Parallelization. In: Proceedings of 5th International Conference, PPAM 2003. 2003: 261-268.

Parsopoulos K. E, Tasoulis D. K, Pavlidis N. G, Plagianakos V. P, Vrahatis M. N. Vector evaluated differential evolution for multiobjective optimization. In: Proceedings of the Congress on Evolutionary Computation 2004. 2004; 1: 2014-211.

Hernández-Díaz A. G, Santana-Quintero L. V, Coello C. C, Caballero R, Molina J. A new proposal for multi-objective optimization using differential evolution and rough sets theory. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. 2006: 675-682.

Deb K, Mohan M, Mishra S. Towards a quick computation of well-spread Pareto-optimal solutions. Lecture Notes in Computer Science. 2003; 2632: 222-236.

Cai Z, Gong W, Huang Y. A novel differential evolution algorithm based on domination and orthogonal design method for multiobjective optimization. Evolutionary Multi-Criterion Optimization. 2007; 4403: 286-301.

Zhang Q, Li H. A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput. 2007; 11: 712-731.

Robič T, Filipič B. DEMO: Differential evolution for multiobjective optimization. Lecture Notes in Computer Science. 2005; 3410: 520-533.

Chow CK, Tsui HT. Autonomous agent response learning by a multi-species particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation 2004. 2004; 1: 778-785.

Zamuda A, Brest J, Boškovič B, Zumer V. Differential Evolution with Self-adaptation and Local Search for Constrained Multiobjective Optimization. In: Proceedings of the Congress on Evolutionary Computation 2009. 2009:195-202.

Kennedy J, Eberhart R. Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks 1995. 1995; 4: 1942-1948.

Parsopoulos K. E, Vrahatis M. N. Recent approaches to global optimization problems through particle swarm optimization. Natural Computing. 2002; 1(2-3): 235-306.

Coello Coello C. A, Lechuga M. S.MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation 2002. 2002; 2: 1051-1056.

Mostaghim S, Teich J. The Role of ε-dominance in Multi Objective Particle Swarm Optimization Methods. In: Proceedings of IEEE 2003 Congress on Evolutionary Computation 2003. 2003: 1764-1771.

Mostaghim S, Teich J. Covering Pareto-optimal fronts by subswarms in multi-objective particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation 2004. 2004: 1404-1411.

Li X. A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. 2003; 2723: 37-48.

Pulido G. T, Coello Coello C. A. Using Clustering Techniques to Improve the Performance of a Multi-objective Particle Swarm Optimizer. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. 2004; 3102: 225-237.

Sierra M. R, Coello Coello C. A. Improving PSO-based multi-objective optimization using crowding, mutation and ε-dominance. Evolutionary Multi-Criterion Optimization. 2005: 505-519

Tseng C. T, Liao CJ. A discrete particle swarm optimization for lot-streaming flowshop scheduling problem. Eur J Oper Res. 2008; 191(2): 360-373.

Venter G, Haftka R. T. Constrained particle swarm optimization using a bi-objective formulation. Struct Multidiscip O. 2010; 40(1-6): 65-76.

Agrawal S, Panigrahi K. B, Tiwari M. K. Multiobjective Particle Swarm Algorithm With Fuzzy Clustering for Electrical Power Dispatch. IEEE Trans Evol Comput. 2008; 12(5): 529-541.

Elhossini A, Areibi S, Dony R. Strength Pareto particle swarm optimization and hybrid EA-PSO for multi-objective optimization. Evol Comput. 2010; 18(1): 127-156.

Dominguez J. S. H, Pulido G. T. A comparison on the search of particle swarm optimization and differential evolution on multi-objective optimization. In: Proceedings of the Congress on Evolutionary Computation 2011. 2011:1978-1985.

Aarts E, Korst J. Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing. New York: John Wiley and Sons, Inc. 1989.

Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res. 1986; 13(5): 533-549.

Lourenco H. R, Martin O. C, Stützle T. Iterated Local Search. Handbook of Metaheuristics International Series in Operations Research & Management Science Kluwer. 2003; 57: 320-353.

Tang W. K. S, Yeung C. S. H, Man K. F. A Jumping Gene Evolutionary Approach for Multiobjective Optimization. ICT Innovations 2011, AISC. 2012; 150: 1-14

Li J, Pan Q, Xie S. An effective shuffled frog-leaping algorithm for multi-objective flexible job shop scheduling problems. Appl. Math. Comput. 2012; 9353-9371

Chan T.M, Man K. F, Kwong. S, Tang K. S. A Jumping Gen Paradigm for Evolutionary Multiobjective Optimization. IEEE Trans Evol Comput. 2008; 12: 143-159.

Rahimi-Vahed A, Mirzaei A. H. A hybrid multi-objective shuffled frog-leaping algorithm for a mixed-model assembly line sequencing problem. Comput Ind Eng. 2007; 53(4): 642-666.

Jarboui B, Damak N, Siarry P, Rebai A. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Appl Math Comput. 2008; 195(1): 299-308.

Liu B, Wang L, Jin Y. H. An effective PSO-based memetic algorithm for flow shop scheduling. IEEE Trans on Syst Man Cybern. 2007; 37(1): 18-27.

Chapter 9 Additions based on Comments during Defense

9.1 Tennessee Eastman Chemical process data generations

In chapter 2 and 4, the Tennessee Eastman Chemical process is used to examine the effectiveness of the proposed methods [1]. This process has six major unit operations in this process including a feed mixer, a exothermic 2-phase reactor, a product condenser, a vapor-liquid separator, a recycle compressor and a product stripper. Two liquid products, G and H, are produced from four gaseous reactants A, C, D and E along with a byproduct of F. An inert B is fed into the reactor where G and H are formed. The reactor product stream passes through a partial condenser to cool the reactor product and then the product is fed to a vapor-liquid separator. While non-condensed components are recycled to the reactor feed stream through a compressor, condensed components are fed to a stripper to remove remaining reactants. The problem statement defines operational process constraints that the control system should respect, 20 types of process disturbances and six operating modes at different G/H mass ratios in the product stream. Mode 1 (basic operating mode) of 50/50 G/H mass ratio is used in chapter 2 and 4, while mode 3 of 90/10 G/H mass ratio is used in chapter 4.

There are 41 measurements variables and 12 manipulated variables in the process. A decentralized control strategy is adopted to 1) maintain process variables at desired values, 2) keep process operating constraints, 3) minimize variability of product rate and quality in the product stream during disturbance, 4) minimize movement of values which affect other processes and 5) recover from disturbances such as production rate changes or product mix changes. As for process operating constraints, reactor pressure, reactor level, reactor temperature, product separator level and base level are controlled to keep them within normal operating ranges. Although high and low shutdown limits are employed to shutdown the process when the process conditions become out of control, in all test cases in this thesis, even after disturbances are added, all process variables never get out of their shutdown limits and they are controlled to keep themselves within normal operating ranges.

9.2 Information of Nelder-Mead algorithm

Since the complex integrals Eq. (23)-(25) in chapter 2 for mutual information are difficult to calculate analytically, a numerical optimization method termed as Nelder-Mead algorithm is adopted to compute a decomposition matrix [2]. In the Nelder-Mead algorithm, the worst vertex x_w is 1) reflected, 2) reflected and expanded or 3) contracted to the new point x_{new} as below:

$$x_{new} = x_c + \alpha(x_c - x_w) \tag{1}$$

where x_c is the centroid of the best side which is the one opposite the worst vertex x_w and α is a user-specified parameter. The last operation is that 4) all the points except for the best point shrink to the best point as below:

$$x_{new,j} = x_l + \alpha(x_j - x_l) \tag{1}$$

where $x_{new,j}$ is a new vertex corresponding to the j-th vertex x_j and x_l is the best vertex. In this thesis, the parameters α are set as shown in Table 1.

Table 1: Parameters of Nelder-Mead algorithm in chapter 2

Operator	Reflect	Reflect and expand	Contract	Shrink
Parameter α	2.0	0.5	-0.5	0.5

Further, since the objective function in the mutual information based optimization problem may have strong nonlinearity and multi-peak feature, the multi-start optimization strategy is used. The number of initial solutions is set to be 50 and thus 50 runs of Nelder-Mead algorithm are carried out to compute a decomposition matrix. The CPU time for each worker is about 5s to 20s.

9.3 References

[1] Downs JJ, Vogel EF, Plant-wide industrial process control problem. *Comput Chem Eng.* 1993;17:245-255

[2] Nelder A, Mead R. A simplex method for function minimization. *Comput J.* 1965;7(4):308-313

Chapter 10 Conclusions and Future Work

10.1 Conclusions

In this thesis, several statistical methods of process systems are developed for process monitoring, fault diagnosis and planning and scheduling. In chapter 2, a novel QNGLSP method is proposed for industrial process monitoring and fault detection by taking into account both measurement and quality variables. In order to retain the non-Gaussian features and relationships between measurement and quality variables, the developed decomposition algorithm employs the higher-order statistics of mutual information as the objective function when searching latent directions. After decomposition, I^2 and SPE indices are developed to detect process faults within non-Gaussian latent subspaces. Compared to the PLS-based monitoring method, the presented QNGLSP method adopts mutual information-based objective function and hence it can effectively extract the non-Gaussian relationship between measurement and quality variables. Even though the ICA-based process monitoring method can extract non-Gaussian features, it cannot take into account the relationship between measurement and quality variables, which can be achieved in the proposed QNGLSP-based monitoring method. The Tennessee Eastman Chemical process has been used to examine the effectiveness of the proposed method. The result demonstrates that the proposed method is superior to the PLS-based monitoring method.

In chapter 3, the QNGLSP approach is extended for nonlinear batch process monitoring. In order to identify nonlinear dynamics, the kernel principal components are extracted from unfolded and scaled measurement and quality data sets. Then, the QNGLSP method is implemented in the high-dimensional kernel feature space. Finally, a set of new monitoring indices is developed to capture abnormal behavior of batch process within non-Gaussian latent subspace. In this way, this new method can identify nonlinear dynamics and non-Gaussian relationships between measurement and quality variables. The proposed approach is applied to the fed-batch penicillin fermentation process and the process monitoring results of the presented method are demonstrated to be superior to those of the MKPCA, MKICA and MKPLS based process monitoring approaches.

In chapter 4, the new probabilistic graphical model based networked process monitoring is proposed for the identification of the fault propagation pathways and root-cause variables. The most probable probabilistic graphical model is constructed from historical process data and incidence matrix which does not require in-depth process knowledge. A set of monitored variables is broken into smaller subsets in accordance with the incidence matrix which is created from the process flow diagram. Then the most probable subgraph corresponding to each subset is computed by using graph samples generated from the Markov chain Monte Carlo simulation. After combination of all the computed subgraphs into the one entire network, network parameters are calculated from the historical process data. Furthermore, an abnormal likelihood index is developed to detect the faulty operating condition. For the identification of root-cause variables as well as fault propagation pathways, the posterior probability of the edge is computed. The proposed method is compared to the PCA/ICA based process monitoring and causal map of transfer entropy method in the 12 test cases of the Tennessee Eastman Chemical process. The results demonstrate that the proposed method is superior to the other methods in terms of accurate identification of fault propagation pathways and root-cause variable of process upset.

In chapter 5 and 6, the decision-tree structure CPTs based hybrid Bayesian inference technique is developed in order to deal with the network containing both continuous and discrete variables, some of which have large cardinalities. The context-specific CPTs are computed by using classification tree algorithm. Then, operators for computing multiplication and marginalization of factors represented by decision-tree structured based CPTs are developed. Belief propagation algorithm with the proposed operators is used for the inference in hybrid Bayesian networks. The proposed algorithm is applied to estimation of production times in the steel plate production. Even though the constructed Bayesian network contains large domain discrete nodes, the results show that the proposed algorithm can predict the probability distribution of the production times.

In chapter 7, the results of chapter 6 are applied to production planning and scheduling in manufacturing of steel plates. Concretely, the probability distributions of the production loads of downstream processes and production times are utilized to construct the scheduling model of the continuous casting. Since this problem is difficult to solve due to a large number of binary variables which are needed to represent exactly process characteristics. Therefore, a new two-level algorithm is proposed to reduce the computational cost. At the top level, a multi-period production planning problem is solved. Since this problem is formulated as a mixed integer linear model that does not consider sequence penalties, it is much easier to solve than the original full-space MILP model. Then at the lower level, the task is to determine the sequences of charges such that there are a minimum number of switched by using parallel SA search. In order to improve the ability of parallel SA algorithm to avoid local optimal solution, this algorithm is combined with the idea of leaping from shuffled frog-leaping algorithm. Real life steel production data have been used to examine the effectiveness of the proposed two-level algorithm. The computational results show that the proposed optimization algorithm is sufficiently fast and generates reasonable solutions for computation of steel plate continuous casting process.

In chapter 8, a new multi-objective evolutionary algorithm for the continuous casting scheduling problems mentioned in chapter 7 is developed. The set of solutions that is generated by means of the algorithm presented in chapter 7 is used as initial solutions of MOEA. The developed algorithm, NSGA-II JG is a non-dominated sorting algorithm NSGA-II that is modified by including local search and jumping gene transpons. Real world steel production data have been used to examine the effectiveness of the proposed algorithm. The computational results demonstrate that the proposed NSGA-II JG is the preferred choice for solving large scale multi-objective scheduling of continuous casting.

10.2 Future work

Process monitoring and diagnosis methods are proposed in chapters 2, 3 and 4. The process fault can be detected and the root-cause variables can be identified by using the models which are constructed from the normal process data. When applying the proposed models to real-world industrial processes, the accuracy and precision of those models should be verified regularly and then the models should be rebuilt from recent historical process data if necessary. However, it takes time and effort for process engineers to maintain the model accuracy. Therefore, maintenance free models are strongly desired in industrial practice. Future research may focus on development of the sequential and online learning algorithm in order to avoid the maintenance of the models. In addition, the proposed method will be applied to real industrial process data, especially for process monitoring of blast furnace operations in steel plants, which has been studied for years but are still very difficult to detect fault operations and isolate the root-cause variables.

Inference algorithm in hybrid Bayesian networks with large discrete and continuous domains is developed and applied for prediction of the process times and the production loads in steel plants. To handle Bayesian networks which include continuous parent nodes with discrete child nodes, the corresponding continuous variables are discretized as finely as needed, which can be achieved because the domain size of discretized variable does not increase the number of parameters due to the proposed decision-tree structured CPTs. While discretization of variables captures rough characteristics of the distribution of continuous variables, it can be efficiently used for probabilistic inference. A set of threshold values that partitions continuous variables into a finite number of intervals has strong effect on inference performance. However, in this thesis, continuous variables are partitioned at equal intervals in accordance with the user-specified number of intervals. Therefore, it is desirable to choose the threshold values that can enhance the inference performance in Bayesian networks.

In the present work of planning and scheduling of steel plate production, near optimal solutions can be obtained by decomposing the original problem into two levels. At the lower level, the sequence of charges is optimized using meta-heuristic approach with job-to-position representation. Meanwhile, if the lower level problem can be formulated as the travelling salesman problem (TSP), this sequence optimization problem may be solved much faster than the proposed algorithm since heuristic algorithms for TSP such as greedy, 2-opt, 3-opt, genetic algorithms and neural network approach have been well studied for years and some of them show good performance. In addition, future research may also focus on the multi-objective TSP for which the aim is to obtain the set of efficient solutions.