COMBINATORIAL OPTIMIZATION FOR DATA CENTER OPERATIONAL COST REDUCTION

COMBINATORIAL OPTIMIZATION FOR DATA CENTER OPERATIONAL COST REDUCTION

BY

SOMAYYE ROSTAMI, M. Sc. Electrical Engineering

A Thesis

submitted to the Department of Computing and Software

and the School of Graduate Studies

of McMaster University

in partial fulfilment of the requirements

for the degree of

Doctor of Philosophy

Doctor of Philosophy (2023) McMaster University

(Department of Computing and Software) Hamilton, Ontario, Canada

TITLE:	Combinatorial Optimization for Data Center Operational Cost Reduction
AUTHOR:	Somayye Rostami M. Sc. (Electrical Engineering/Telecommunication Systems), Iran University of Science and Technology, Tehran, Iran
SUPERVISOR:	Dr. Douglas Down, Dr. George Karakostas
NUMBER OF PAGES:	xiv, 154

Lay Abstract

Data centers, each hosting as many as tens of thousands of IT devices, contribute to a considerable portion of energy usage worldwide (more than 1 percent of global power consumption). They also encounter other operational costs mostly related to reliability of devices and maintenance. One of the key places to reduce energy consumption is through addressing the thermal heterogeneity in data centers by thermal-aware workload distribution for the servers. This prevents hot spot generation and addresses the trade-off between IT and cooling power consumption, the two main power consumption contributors. The corresponding optimization problem is challenging due to its combinatorial nature and the complexity of thermal models. In this thesis, we present a holistic approach for thermal-aware workload distribution in data centers, using linearization to make the problem model-independent and simpler to study. Two quite general nonlinear optimization problems are defined. The results confirm that the proposed approach completed by a proposed heuristic solves the problems efficiently and with high precision. Finally, we address a problem in inventory management related to data center maintenance, where we develop an efficient algorithm to solve a lot-sizing problem that has a goal of reducing data center operational costs.

Abstract

This thesis considers two kinds of problems, motivated by practical applications in data center operations and maintenance. Data centers are the brain of the internet, each hosting as many as tens of thousands of IT devices, making them a considerable global energy consumption contributor (more than 1 percent of global power consumption). There is a large body of work at different layers aimed at reducing the total power consumption for data centers. One of the key places to save power is addressing the thermal heterogeneity in data centers by thermal-aware workload distribution. The corresponding optimization problem is challenging due to its combinatorial nature and the computational complexity of thermal models. In this thesis, a holistic theoretical approach is proposed for thermal-aware workload distribution which uses linearization to make the problem model-independent and easier to study. Two general optimization problems are defined. In the first problem, several cooling parameters and heat recirculation effects are considered, where two red-line temperatures are defined for idle and fully utilized servers to allow the cooling effort to be reduced. The resulting problem is a mixed integer linear programming problem which is solved approximately using a proposed heuristic. Numerical results confirm that the proposed approach outperforms commonly considered baseline algorithms and

commercial solvers (MATLAB) and can reduce the power consumption by more than 10 percent. In the next problem, additional operational costs related to reliability of the servers are considered. The resulting problem is solved by a generalization of the proposed heuristics integrated with a Model Predictive Control (MPC) approach, where demand predictions are available. Finally, in the second type of problems, we address a problem in inventory management related to data center maintenance, where we develop an efficient dynamic programming algorithm to solve a lot-sizing problem. The algorithm is based on a key structural property that may be of more general interest, that of a just-in-time ordering policy.

To all the influential individuals in my life

Acknowledgements

I would like to first express my gratitude to McMaster University for providing me with a cooperative, dynamic, and responsive learning environment. I am thankful for the support and attention given to my various needs and concerns during my time there. Additionally, I am truly grateful for the opportunity to study under the guidance of Dr. Douglas Down. Throughout the duration of my studies, I consistently felt supported and understood by him. Working with him felt like an excellent fit, as he exemplifies reasonableness, intelligence, supportiveness, patience and professionalism. While I had the autonomy to work independently, I always received valuable advice and feedback from him. I would also like to extend my appreciation to Dr. George Karakostas, my co-supervisor, whose deep enthusiasm and guidance helped shape my work. Under his supervision, I not only received support but also gained a deeper understanding of discipline. Additionally, I appreciate the valuable insights I received from the committee members, Dr. Ashtiani and Dr. Todd.

I am also grateful for the opportunities that my home country, Iran, has provided me, which have led me to the position I am in today. Finally, I would like to express my heartfelt appreciation to my family and all the important individuals in my life

who have had a profound impact on me and have inspired me to persevere.

Contents

List of Figures

List of Tables

Declaration of Academic Achievement

The author of this thesis is the main author and contributor of the three manuscripts described in Chapters 2, 3 and 4. For the third manuscript, "Single-item Lot-sizing with Quantity Discount and Bounded Inventory", the author list is in alphabetical order.

Her contributions to these works consist of writing the manuscript, formulating the optimization problems, developing algorithms and approximation heuristics, designing baseline algorithms, conducting the experiments, implementing the framework, and generating the numerical results.

Chapter 1

Introduction

This chapter starts with a discussion of data center operational costs followed by motivations and an outline of this thesis. In Section 1.1, an overview of data centers is presented, some popular models are introduced and related works are discussed. Section 1.2 summarizes the remainder of this thesis.

Data centers play a crucial role in today's internet and online services. They host Information and Communication Technology (ICT) devices whose functions are processing, storing and transmitting information. Whether small or large, data centers operate in a complex and interactive environment with a variety of costs. According to [1], the cost of ownership is divided into five main expenses: infrastructure, server, power, network and maintenance costs. Infrastructure cost includes the capital costs such as the cost of land acquisition and building construction. The server and network costs encompass server acquisition and the cost of network hardware such as switches and cables. Power cost is the main operational cost, covering the power consumed

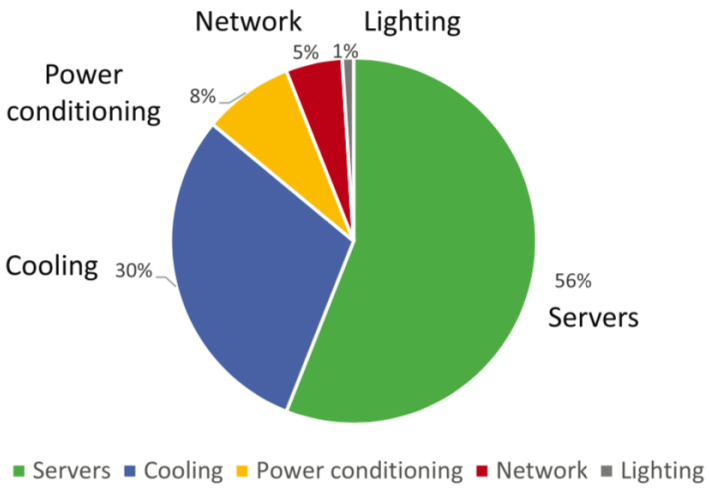

Figure 1.1: Typical data center power consumption breakdown [2]

by electrical devices, including servers, network and cooling components. According to Figure 1.1, servers and the cooling system are the main power consumers in a data center. Power Usage Effectiveness (PUE), the ratio of total power consumption to computing (IT) power consumption, is a popular metric for new developments as it is a measure of operational efficiency. The main goal is reducing the total power consumption, which may not be guaranteed by lowering PUE, but PUE is a generally accepted metric. Maintenance costs are impacted by the reliability of the devices, for example reduced equipment lifetimes may lead to more frequent replacements. Reducing maintenance costs has begun to receive more attention in the literature.

With the increased demand for internet services such as search engines, online banking and social media, and with the rapid development of cloud computing and artificial intelligence, data center operational costs have increased dramatically [3]. In 2013, data center energy consumption in the United States was about 91 billion kilowatt-hours, enough to supply all of the households in New York City for two

years. An estimate for 2020 was 140 billion kilowatt-hours [2]. In [4] two estimation models are compared. The most authoritative bottom-up study estimated between 1 to 1.5 percent of global energy use for data centers in 2010. Extrapolating, a rapid growth in demand would correspond to a rapid increase in energy consumption, a phenomenon that has often been touted in the literature. However, the current bottom-up models estimate only a 6 percent increase in energy consumption from 2010 to 2018 [4]. Three efficiency effects explain the fact that consumption is rising more slowly than demand: the improved energy efficiency of new computing and networking devices, virtual machine migration techniques which allow multiple tasks to be run on a single physical machine, and the transition to cloud- and hyperscale-class data centers that use ultra-efficient cooling systems. Having said that, due to the huge energy consumption of data centers with many tens of thousands of IT devices, decreasing the energy consumption even by a small percentage is important from both the energy and climate policy perspectives [4].

Data centers operate in a very dynamic environment and there are several interactions among their components. This makes power consumption management of a data center a complicated task. The dynamics are related to fluctuating IT workload, thermal effects of servers and cooling equipment due to their locations, and a time-varying ambient environment. Ignoring such dynamics in a data center can lead to irregular air flows and hot spot generation, leading to a very poor PUE and high operational costs [2].

To minimize operational costs, optimization and management of cooling and IT power consumption (as the two main power consumers in data centers) should be

done jointly. Because the cooling system is a main power consumer in a data center, over-provisioning of the cooling facilities leads to high operational costs, while under-provisioning may lead to unsafe operational temperatures for the servers leading to high failure rates and maintenance costs. There may also be a trade-off between cooling and IT power consumption. Another important factor in the optimization is Quality of Service (QoS). Workload distribution should guarantee QoS. The optimization depends on the models that are used for inlet temperatures, IT workload, and cooling and IT power consumption. Overall, joint optimization of cooling and IT power consumption while satisfying QoS and inlet temperature constraints is a challenging problem [2]. In addition to unsafe ranges for the operational temperatures, frequent temperature changes (due to workload fluctuations) are also correlated with high failure rates [5]. In general, reliability of devices is a recent perspective considered in the literature. Frequent switching of the servers between on and off states can also have adverse effects on QoS due to imposing delay [6].

There are several possibilities to reduce the energy consumption in a data center. Energy consumption reduction can be performed at different levels including chip level, server level, rack level and room level. At each level, IT and/or cooling power consumption can be improved. At the chip level, for example, Dynamic Voltage and Frequency Scaling (DVFS) can reduce the CPU's power consumption. Liquid cooling of CPUs is another technique at the chip level that is seeing greater levels of implementation. At the server level, server fans can help air circulation and intelligent workload distribution can avoid hot spot generation. At the rack level, some techniques such as holes in the rack front door and heat exchangers in the rack rear door can influence the air flow around the servers. At the room level, as an example, a

technique named hot aisle and cold aisle containment can reduce hot air recirculation in the room [7].

The power consumption optimization considered in this thesis is at the server and the room levels for traditional air-cooled data centers. Several techniques are proposed in the literature at both levels. A classification of the techniques is presented in [2]. Three main techniques are introduced for the server level. The first technique is workload dispatching (also known as job scheduling or task placement). It assigns incoming load to the servers (computing nodes) such that performance and temperature requirements are met, while preventing hot spots and minimizing the overall power consumption. At a (different) lower layer, using virtual machines allows workload migration between the physical machines (servers). In this way, multiple tasks can be run on a single machine and so with the increased number of idle servers, a subset of servers can be powered down to save idle power consumption (which can be more than 50% of the peak power consumption [6]). This is referred to as server consolidation. Workload migration can also balance the load of servers for performance requirements and thermal efficiency. Finally, server fan control can be used to reduce the fan power consumption. Room-level techniques are mostly related to cooling power consumption and include adaptive vent tile openings, Computer Room Air Conditioner (CRAC) fan speed control, CRAC supplied temperature control, and outside air cooling. System layout planning for server placement in racks to leverage the fact that servers have different characteristics is another possibility that can reduce cooling power consumption.

This thesis has two parts. The focus of the first part is on thermal-aware workload distribution while considering thermal and performance constraints, along with additional reliability aspects. There is a significant body of literature with similar perspectives but with different power consumption, thermal or performance models. The thermal model specifically is the main difference and challenge for optimization problems in this area. Different techniques to solve the optimization problem are proposed such as greedy heuristics or meta-heuristics, for example genetic algorithms. Similar approaches in defining the power consumption optimization problems motivated us to propose a general approach to solve them which can be applicable in different settings. Our aim is to build a theoretical framework for some general combinatorial optimization problems in this area and propose potential algorithmic approaches to solve them. Most work in this area has appeared in the systems literature but there is a lack of theoretical analysis of these optimization problems. The theoretical analysis becomes more important when one notices that the optimization problems defined in the literature have similarities in terms of the cost function and the constraints. We start with a general nonlinear problem proposed in a related work where there is no accompanying discussion about the solution. Our approach is to first linearize the problem. Depending on the type of decision variables (continuous or integral), linearization leads to a linear programming problem or a (mixed) integer linear programming problem. In the case of a mixed integer programming problem a heuristic is proposed to approximate the solution. Linearization is performed with different aims, first, studying the inherent difficulty of the problem and second proposing a time-efficient approach to solve the problem. The latter is important for online operation especially when the thermal models are computationally

expensive to evaluate. For example these models may not be closed-form and may involve solving a set of differential equations. Defining an optimization problem that considers the key challenging aspects is helpful to design algorithms for solving the original problem and its extended versions. We expect that this approach can help build a theoretical framework for optimization problems in this area.

For the second part of the thesis, a combinatorial optimization problem related to multi-period lot-sizing with quantity discount is studied. The application of this problem is also related to reducing operational costs for a data center. Some items used in a data center are purchased and stored in the inventory. The goal is to minimize the summation of ordering and holding costs when a quantity discount is available. We study the case of a single item with a quantity discount and bounded inventory. We also show that the case of two items is an NP-hard problem.

1.1 Background and Literature Review

This section starts with a general review of the literature, including different cooling strategies and data center related models. It ends with a review of the works most closely related to this thesis.

1.1.1 Overview of Data Centers

A traditional data center consists of the components shown in Figure 1.2. The ICT equipment includes server racks and networking components such as switches. The main consumers of ICT power are the servers. The total power consumption of servers depends on their technology, the amount of workload and the workload distribution.

The other main contributor to the power consumption in a data center is the cooling system. According to [3], the main cooling methods are active cooling and passive cooling.

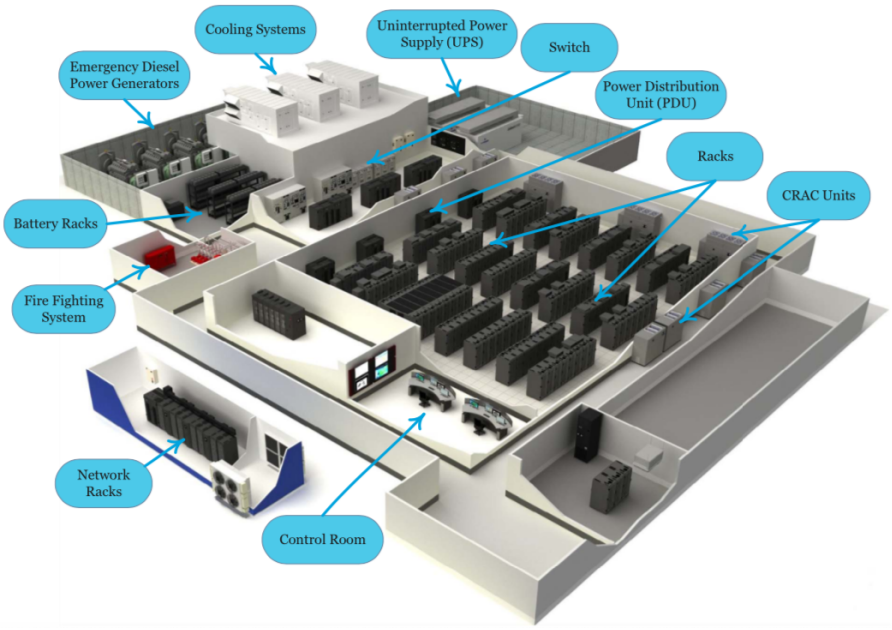

Figure 1.2: Typical layout of a data center [7]

Active air cooling is based on using CRACs. Figure 1.3 shows a common active air cooling approach in a data center. ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) and ETSI (European Telecommunications Standards Institute) have defined some climatic norms including air temperature and humidity ranges for a data center environment. A good circulation of air flow is the core requirement for respecting the norms and having an efficient cooling system. Hot aisle/cold aisle is a common technique in which the cold air comes through a raised floor. The air flow is from the fronts of the racks to their rears; the air moving from front to back absorbs heat from the servers. The heat is then extracted by the CRAC

units and finally rejected to the exterior environment [3]. This method reduces the mixing of hot and cold air. However, a portion of hot air in the rear of racks will return to the front of racks, instead of being extracted by the CRAC unit [2]. This effect is called heat recirculation, and is shown in Figure 1.4. Heat recirculation has a harmful effect, because it increases the server inlet temperatures. To compensate for this effect the supplied temperature should be set to a lower level than if heat recirculation were not present. Subsequently, the supply fans then work with higher speeds due to the lower temperature [3]. This is because when air is colder, the air flow rate becomes slower.

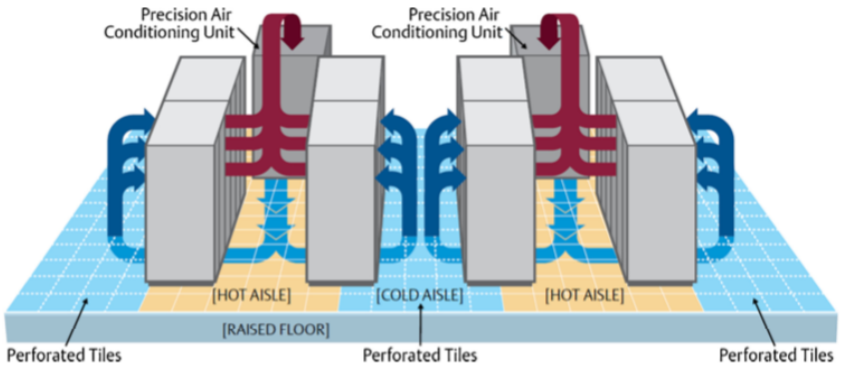

Figure 1.3: The hot aisle/cold aisle configuration [3]

To reduce the power consumption in the active cooling method and avoid heat recirculation, passive cooling methods have been proposed. The main solutions in the past few years are: free cooling, liquid cooling, immersion cooling, two-phase cooling and improving the influence of the building envelope [3]. While examining such techniques in the future may be helpful, the primary focus of this thesis will be on active, air-cooled systems.

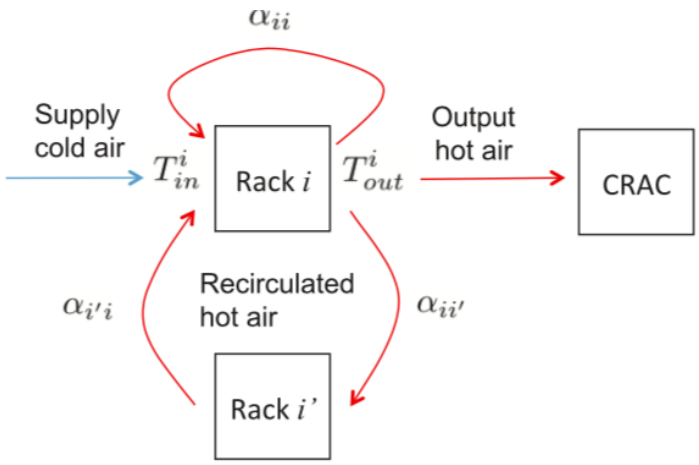

Figure 1.4: The heat recirculation effect [2]

1.1.2 Models of Power, Temperature and Workload

In this section several models considered for power consumption, temperature and workload are provided. These models are used in the formulation of the corresponding optimization problems.

IT Power Models

The most common IT power models are presented in [8]. The main consumer of computing power in a server is the CPU. According to [8], there are two popular IT power models:

1. In a DVFS-enabled processor the power consumption of a server, $P(f)$, is a cubic function of its processor's working frequency, f:

$$P(f) = \beta_1 + \beta_2 f^3 \tag{1.1.1}$$

where β_1 is the idle power consumption. When the server is in stand-by mode,

the power consumption is a constant that is much less than β_1. The constant β_2 is related to the average switching activity and capacitance. It also includes a linear relation between voltage and frequency of the CPU [9][10].

2. In a non-DVFS processor there is no voltage and frequency scaling and the CPU's frequency is equal to a constant, f^{max}. Given the server utilization, u, the fraction of time that the CPU is busy, then from [8], the power consumption is given by

$$P(u) = \beta_1 + \beta_3 u. \tag{1.1.2}$$

In [11] a more complex model is proposed, relating the performance state (P-state) of a core and its power consumption. More accurate models have been proposed in [12]. In this thesis, we use non-DVFS power models.

Cooling Power Models

Figure 1.5 shows the coupling of IT and cooling power consumption. According to [13], each thermal node in a data center has an input and output temperature and also a power consumption. In [6] the power consumption of a CRAC unit is given as

$$P_c = \begin{cases} K_l \frac{T_{in} - T_{out}}{COP(T_{out})} & T_{in} \geq T_{out} \\ 0 & T_{in} < T_{out} \end{cases} \tag{1.1.3}$$

where T_{in} and T_{out} are the input and output temperatures of the CRAC unit, respectively. K_l is a positive coefficient related to the volume of air passing through the CRAC unit (which depends on the air flow rate and the mass density of the

air) and the air heat capacity [8]. There is also an adjustable reference or supplied temperature, T_{ref}, for CRACs. T_{out} tends to the minimum of T_{ref} and T_{in} [13].

The universal coefficient of performance for each CRAC unit is defined in [13] as

$$COP(T_{out}) = 0.0068T_{out}^2 + 0.0008T_{out} + 0.458 \tag{1.1.4}$$

When there is only one CRAC unit, its power consumption is related to the total computing power consumption as [14]

$$P_c = \frac{P_{IT}}{COP(T_{out})} \tag{1.1.5}$$

There may also be other cooling facilities in a data center, for examples fans, computer room air handlers, chiller plants and air-side economizers. Some models are proposed in [15][16]. For example fan power consumption is a cubic function of fan speed. In this thesis, we do not assume a particular cooling facility for a data center, instead the model includes several cooling parameters on which the cooling power consumption within the data center depend.

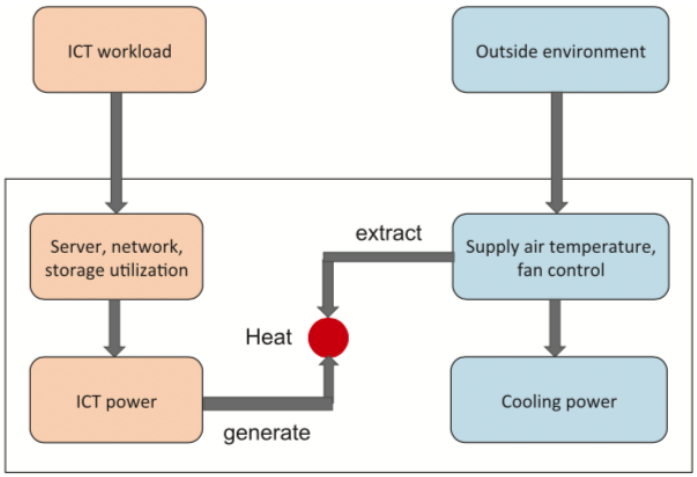

Figure 1.5: The coupling of IT and cooling power consumption [2]

Thermal Models

A thermal model determines the server (computing nodes) temperatures and is used in the temperature constraints of power consumption optimization problems. The thermal model can be considered as the main challenge for power consumption optimization in a data center. Providing a precise model that can also quickly calculate the temperatures is an important requirement for online optimization. There are white, gray and black box approaches for temperature models, based on physics, physics and learning (data-driven), and learning, respectively [2][17]. Computational Fluid Dynamics (CFD) simulations can precisely estimate the temperature and air flow distributions but are computationally expensive and may require many parameters. They are useful for initial data center layouts but are infeasible for real-time tasks [17]. Data-driven methods rely on data read from physical sensors. They are fast and they also can adapt to updates in the data center layout. One data-driven method is presented in [17][18]. According to [19], an appropriate method is a combination

of physical characteristics and learning from data. Such methods can be fast and sufficiently precise to capture dependencies of temperatures on workload variations, cooling parameters and the layout of the data center. Two simplified zonal-based models are presented in [19][20] that assume uniform conditions within different abstract zones in a data center, where each zone may contain several servers. However, the time to simulate these models (they are based on solving a set of differential equations) is still on the order of seconds even for a small data center with 25 zones (this data center is the basis for our experiments described later in the thesis). This makes using such models for online power consumption optimization problematic. A deep learning approach is presented in [21] for temperature predictions.

On the other hand, some simplified parametric linear models are common in the literature. A key differentiation between models is whether they are steady-state models or transient models. A steady-state model that is frequently used is [22][23]

$$T_{in,i} = T_{sup} + \sum_{j=1}^{n} D_{ij} P_j^{comp} \tag{1.1.6}$$

where there is one CRAC unit, T_{sup} is the supplied (reference) temperature of the CRAC unit and D is the heat recirculation matrix. $T_{in,i}$ is the inlet temperature for server (computing node) i, n is the total number of servers and P_i^{comp} is the power consumption of server i. The term $\sum_{j=1}^{n} D_{ij} P_j^{comp}$ represents the heat recirculation effect.

A common transient model is [6]

$$\begin{aligned} T_{out}(k+1|k) &= A_d T_{out}(k|k) + B_d U(k|k) \\ T_{in}(k+1|k) &= C_d T_{out}(k+1|k) \end{aligned} \tag{1.1.7}$$

where T_{in} and T_{out} are the input and output temperatures for the CRAC units or the inlet and outlet temperatures for the racks, respectively. U is the input vector consisting of the rack power consumptions and the CRAC units' reference temperatures. The notation $(j|k)$ means the value at time step j based on the information up to time step k.

In this thesis we use steady-state abstract thermal models, including several cooling parameters and heat recirclation effect, where a zonal-based thermal model is linearized for example.

Workload Models

Workload should be measured and predicted based on historical data. Two approaches are time series prediction and machine learning [2][24][25][26][27][28]. In [14] a data center system is modelled as a $GI/G/n$ queue and the arrival rate is estimated by use of a Kalman filter. The maximum arrival rate is estimated using regression. In [6] an $M/M/n$ queuing model is used. We use time series prediction. One possibility is to use Auto-Regressive Integrated Moving Average (ARIMA) models. A non-seasonal ARIMA model, $AR(p, q, d)$, for forecasting the demand at time period k, y'_k, is as follows:

$$y'_k = c + \phi_1 y'_{k-1} + \cdots + \phi_p y'_{k-p} + \theta_1 \epsilon_{k-1} + \cdots + \theta_q \epsilon_{k-q} + \epsilon_k \tag{1.1.8}$$

where p is the order of the auto-regressive part, q is the order of the moving average part, d is the degree of differencing (differences between consecutive values) involved in the calculation of y'_k, and ϵ_k is white noise [29].

1.1.3 Joint ICT and Cooling Power Optimization

There is a significant body of literature on optimization approaches to reduce the power consumption in a data center. The proposed cost functions are directly or implicitly related to the power consumption. As mentioned in several works, there is a trade-off between IT power consumption and cooling power consumption. The strategy to reduce the IT power consumption is to decrease the idle power consumption by distributing the workload on the least possible number of servers and switching the idle servers to stand-by mode or turning them off. However, this may increase the temperature of the active servers and generate hot spots that require increased cooling effort to reduce the temperature to the safe range. In other words, to reduce the cooling power consumption the maximum temperature of servers must be reduced.

As mentioned previously, optimization is typically performed in different layers. One possibility to reduce the power consumption is optimizing the placement of servers in racks, known as layout planning [2]. As the servers may have different power efficiencies and the locations in the racks are also different in terms of heat recirculation, the placement of the servers affects the cooling power consumption. In [30] power profiles are set using historical data and heuristics are proposed to map

the servers to the rack slots. In [31] varying workload is considered and an optimization problem is formulated that minimizes the weighted summation of maximum temperatures over a set of workload sizes.

Another possibility for optimization is to leverage virtualization of data centers. Establishment of virtual machines is a technique used in data centers to control the workload distribution. The arriving tasks have different system requirements such as number of processors and the size of memory. The tasks are assigned to virtual machines that are in turn assigned to physical machines (servers). As the workload changes over time, virtual machines can be migrated between physical servers in order to perform performance optimization. Virtualization with the aim of joint IT and cooling power consumption optimization while respecting the QoS and temperature constraints is performed to avoid under-loaded and over-loaded servers. Under-loaded servers increase IT power consumption while over-loaded servers may generate hot spots. Procedures typically consist of two parts: VM placements and VM migrations. A comprehensive explanation is given in [32]. Optimization in the virtual machine layer results in complicated optimization problems. The optimization problems are variations on bin-packing problems that are commonly approximated with greedy algorithms [32][33]. A multidimensional best-fit algorithm is proposed in [34]. An ant colony algorithm is proposed in [35]. A mixed integer programming problem is presented in [36] and approximated by a heuristic. A simulated annealing based algorithm is presented in [37]. Some theoretical results are given in [38].

The other layer is workload distribution which focuses on the total size of the workload and distributes it among the servers (computing nodes) while optimizing

the total power consumption. This is beneficial because it captures the effect of workload distribution without considering the details of the virtualization layer. The focus of this thesis is on thermal-aware workload distribution which is discussed in the next section.

1.1.4 Thermal-aware Workload Distribution

A typical thermal-aware workload distribution problem has the form

$$\begin{array}{ll} \min & \text{power consumption} \\ s.t. & \\ & \text{performance (QoS) constraints} \\ & \text{server inlet temperature constraints} \\ & \text{system dynamics representing transient thermal model (optional)} \\ & \text{bounds on decision variables} \end{array} \tag{1.1.9}$$

With respect to the choice of the cost function, many previous works ([14][15][39] for example) have considered the summation of IT and cooling power consumption. While server consolidation (in which idle servers go to stand-by mode or are turned off) is optimal for IT power consumption as it minimizes idle power consumption, it may not be optimal for cooling power consumption because it may generate hot spots. So, in general there is a trade-off between IT and cooling power consumption. This is explained in detail in [14].

Depending on the IT power consumption model employed, the decision variables

are the server utilizations or their CPU's working frequencies. If the workload distribution is performed at a higher level such as the chassis or rack level, the decision variables are the workload assigned to each computing node. Regarding the cooling configuration, the main decision variables are the CRAC units' reference (or supplied) temperatures. In some papers, there are other decision variables such as fan speeds [15]. Server utilization has a continuous value upper bounded by 1. CPU frequency or reference temperature can be continuous and bounded from below and above, or may be chosen from a predefined set.

One of the main constraints in this type of optimization problem is related to the QoS. A common criterion is average response time. The arriving workload must be divided among the servers and their utilizations or their CPU working frequencies must be set in a way that QoS will be guaranteed. The form of the performance constraints depends on which model for QoS is considered. Some works propose a queueing-theoretic model for performance [6][14][40]. In [14], the QoS requirement is translated to the minimum number of servers required. In [6], it is translated to the maximum racks' blocking rate and in [39] it is used for setting an upper bound for server utilizations. There are other works that perform job scheduling to satisfy QoS constraints [41][42][43].

Another critical constraint is on the server temperatures. The server inlet temperature should be maintained below a red-line temperature for safe operation. Thermal models were explained in Section 1.1.2. The works [14][15][39][44] have used steady-state models while transient models have been considered in [6][10][45]. Heat

recirculation is not taken into account in [15]. This reduces the complexity of the optimization problem because the server temperatures are not dependent on the power consumption of neighbouring servers. All references except [39] consider the red-line inlet temperature as a constant. In [39] this value is a function of a server's utilization.

A linear steady-state thermal model is used in many works including [46] and [47]. The model is used to calculate the inlet temperatures of the racks that contain a set of servers. The workload distribution is also performed at the rack level. In [46], they consider homogeneous data centers while in [47], heterogeneous data centers are also discussed. They propose a genetic algorithm and a genetic simulated annealing algorithm to approximate the solution, however these approaches may not be sufficiently fast. The combination of a genetic algorithm with simulated annealing is used in order to prevent the solution from being trapped into a local optimum. In general, the thermal model may not be at the server level. The thermal model might be zonal-based, for example where each zone contains a set of servers. In this case, the size of the problem reduces but the decision variables for workload distribution may not be 0-1 variables. A chassis consolidation approach is proposed in [48] and formulated as an integer programming problem where the solution of the relaxed linear problem is rounded.

In [14] a two-step optimization scheme is proposed. The first step uses coarse time intervals, where the optimal set of servers that can accommodate the maximum arrival rate during the interval is specified. The step of finding the set of active servers is called server provisioning in the literature. For this step the server utilizations are assumed to be identical. The decision variables are integral and determine which

servers are active. They show that in a homogeneous data center, the problem takes the form of an integer programming problem provided that the number of working servers is fixed. At the second step, solved for finer intervals, the utilization of servers determined in the first step are calculated to satisfy the current arrival rate. They show that the first homogeneous problem is NP-hard, while the second problem for the homogeneous case has a linear form. They then propose heuristics to rank and select the active servers. The second phase for the case of a heterogeneous data center results in a nonlinear problem. To solve this, they could set the supplied temperature of the CRAC and by using binary search and solving the resulting linear problem find a near optimal solution. In general, the heterogeneous data center case (servers have different capacities and power models) makes the optimization problem more complicated. In [49], homogeneous data centers are considered as a practical scenario due to the possibility of bulk purchase of servers in order to reduce maintenance costs. In some other works such as [39][50] heterogeneous data centers are assumed to be more common in practice.

In [39], the maximum allowable inlet temperature of a server is calculated as a function of red-line temperature for the CPU and the server utilization, while in [14] the bound on the server temperature is assumed to be a constant. However, the heat recirculation effect has not been taken into account. The supplied temperature is simply set as the minimum of the maximum allowable server inlet temperatures. They then propose a heuristic to solve the problem. They also mention that the problem can be solved by using sequential quadratic programming (SQP). The optimization problem in [44] is close to the problem proposed in this thesis. There are n servers and m CRAC units. Server inlet temperatures are a linear function of CRAC unit supplied

temperatures and server workloads. The cost function is the power consumption of CRAC units. This is assumed to be a linear function of CRAC unit supplied temperatures. There are discussions on how to estimate the coefficient matrices. They then propose a heuristic to solve the optimization problem. However, this method does not seem to work well when $m < n$, which is the case of interest for a data center.

The works that have used steady-state models for temperature cannot address the temporal dynamics of a data center. Because workload changes frequently and server temperatures should always be in the safe range, transient models are a better reflection of operational conditions. In [6][10][45], a transient model for temperature is considered and the system dynamics are included in a Model Predictive Control (MPC) optimization problem. The MPC approach is common in the presence of transient models to increase the accuracy of the solution (and potentially decrease the size of the problem). In this approach, the problem is solved for a fixed time horizon and the solution for the first time slot is implemented. This procedure is repeated continuously. To reduce the complexity of solving the problem online, in [10] they break the solution into two steps. In the first step, frequency settings and task assignment are optimized using the steady-state thermal model. The resulting mixed integer programming problem is solved after relaxation and the solution is rounded. In the second step, the supplied temperatures are optimized using the transient thermal model. In [6] they also propose a method to reduce the problem's complexity. Instead of the main decision variables being the number of servers working in each rack and the proportion of arrivals assigned to each rack, they use the rack power consumptions as the decision variables. Then, they set the blocking rate (the

probability that an arriving job is forced to wait) equal to the maximum allowed value and calculate the remaining decision variables in terms of rack power consumptions. To this aim, they use look up tables and linearize some relations. In [45] they also optimize IT power and cooling power separately.

The presented approaches in [40][49] are closer to control theory. In [49], some theoretical results are presented about the proposed problem assuming that it is convex. The problem for the case of homogeneous data centers seems to be linear. In [40] instead of finding the set of active servers they propose an MPC approach to find the number of active servers. They add two terms to the cost function that penalize input fluctuations. The solution is then found by a particle-based algorithm. An MPC approach considering input fluctuations is also presented in [51] and the nonlinear problem is solved by an interior-point algorithm which appears to be sufficiently fast for online calculations. The control interval is about 20 seconds. The cost of switching the servers between on and off states is considered in [52] but the cooling power consumption is not addressed.

In [50] a two-step approach is proposed. In the first step, the servers are classified into four groups according to their CPU utilization. In the second step, a meta-heuristic is proposed for thermal-aware workload distribution. A trichotomic search is proposed in [53] where there is a multivariable cost function that is nonlinear with respect to a single continuous variable. A tensor-based approach is proposed in [54] to reduce the dimensionality of the optimization problem. In [55], failure of equipment is considered while performing the workload distribution. A game-based thermal-aware resource allocation strategy is presented in [56]. In [57] an optimization

problem for total power consumption with performance and temperature constraints is formulated as a mixed integer nonlinear programming problem and is solved by a greedy algorithm. Similar problems have been proposed in [48][58]. Multi-objective cost functions are defined in [59][60]. They are optimized using Markov decision processes and swarm optimization, respectively.

In [61] some challenges for traditional optimization approaches are discussed and a deep learning approach is proposed for thermal-aware workload distribution. In [62] two optimization problems are defined with a data-driven thermal model solved by the *fmincon* function in MATLAB. The coupling of machine learning with optimization theory is a challenge for learning-based holistic control of power consumption in a data center [2].

As explained, the optimization problems defined in the literature have a number of similarities and it would be valuable to perform a theoretical analysis of solution techniques in this area.

1.1.5 The Lot-Sizing Problem

In this section, we present a brief background for Single-Item Lot-Sizing Problems (SILSP). A comprehensive survey that explains several aspects in a clear way is provided in [63]. We summarize the most important and related points presented in this survey, followed by a brief literature review. The definition of an SILSP is as follows. There is a horizon of T time periods where the demand for a single item is known in each period. The goal is to determine the periods in which orders are placed and order quantities to minimize the total cost while satisfying the demands

as they occur. The cost includes the cost of production and the cost of holding items in the inventory. The production cost itself consists of the cost of producing the items plus a constant setup cost for the periods in which an order is placed. SILSPs are valuable from both theoretical and practical aspects. From the practical view, they cover several interesting cases even in the simplest version. Even if they do not directly correspond to practical scenarios, studying SILSPs helps analyse more complicated problems if they can be decomposed into simpler problems. They are also helpful to gain insight about the structural properties of the optimal solution for more complicated problems, such as multi-item problems. The increasing importance of SILSPs led to the creation of the annual International Workshop on Lot-Sizing (IWLS) and the establishment of the EURO Working Group on Lot-Sizing (LOT).

The corresponding optimization problem is:

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} f_t^p(X_t) + f_t^h(I_t) \\ \text{s.t.} \quad & I_{t-1} + X_t = d_t + I_t \qquad \forall t = 1, ..., T \\ & X_t, I_t \geq 0 \qquad\qquad \forall t = 1, ..., T \end{aligned} \tag{1.1.10}$$

where d_t and X_t are the demand and order quantity at period t, respectively, I_t is the inventory level at the end of period t, and $f_t^p(\cdot)$ and $f_t^h(\cdot)$ are the production and holding cost functions at period t, respectively. The first constraint relates the current inventory level to the previous inventory level and the current order and demand quantities. The constraint $I_t \geq 0$ means that demand cannot be backordered. Without loss of generality, $I_0 = 0$. This general formulation is referred to as an uncapacitated SILSP, where there is no limit on production capacity.

When the cost functions are linear, which is discussed as the baseline problem in this area in [64], the problem becomes

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} s_t Y_t + p_t X_t + h_t I_t \\ \text{s.t.} \quad & I_{t-1} + X_t = d_t + I_t \qquad \forall t = 1, ..., T \\ & X_t \leq Y_t d_{t,T} \qquad \forall t = 1, ..., T \\ & Y_t \in \{0, 1\} \qquad \forall t = 1, ..., T \\ & X_t, I_t \geq 0 \qquad \forall t = 1, ..., T \end{aligned} \tag{1.1.11}$$

where $d_{t,k} = \sum_{i=t}^{k} d_i$, s_t is the setup cost, Y_t is the binary setup variable for period t, and p_t and h_t are the production and holding costs per item for period t, respectively. The problem is solved in $O(T^2)$ time by a dynamic programming algorithm developed in [64]. The algorithm is based on a structural property (zero inventory ordering property) such that there is an optimal solution in which if $X_t > 0$ then $X_t = d_{t,k}$ for some $k \geq t$. By the end of the 1980s, the complexity was improved to $O(TlogT)$ by three groups of researchers independently [65]. If the condition $p_{t-1} + h_{t-1} \geq p_t$ holds, which means it is always optimal to order as late as possible (known in the literature as the case without speculative motives to hold inventory or the case with Wagner–Whitin costs), the problem is then solvable in $O(T)$ time [66].

Regarding the complexity of SILSPs, most extensions are NP-hard in the ordinary sense which means there is a pseudo-polynomial algorithm to solve them. This is the reason for not using advanced methods such as meta-heuristics to solve different versions of SILSPs. Regarding solution techniques, the most common approach is dynamic programming. Other approaches include polyhedral approaches, branch and

bound and branch and cut methods, and dual algorithms. There are also some works proposing simple heuristics and approximation schemes even for the basic SILSP that are useful for decomposing more complicated problems.

Extensions of the basic SILSP are defined based on several practical scenarios, some of which are explained as follows.

- **Backlogging:** In the backlogging scenario the demand is allowed to be satisfied later than it occurs. This may be because of profitability considerations or capacity constraints. The uncapacitated SILSP can be easily generalized for backlogging. If b_t is the cost of backlogging an item from period $t-1$ to period t, by introducing new non-negative variables Z_t, the accumulated backlog at the end of period t, the term $\sum_{t=1}^{T} b_t Z_t$ is added to the cost function and the main constraint is converted to $I_{t-1} - Z_{t-1} + X_t = I_t - Z_t + d_t$, where $Z_0 = 0$. The backlogging flow is the opposite of the inventory flow.

- **Lost sales:** Similar to the backlogging case, part of the demand may be lost due to production capacity constraints or profitability considerations. Lost sales and backlogging can be considered together or separately. The uncapacitated SILSP can be generalized to consider lost sales. If l_t is the cost of each unit of lost sales at period t, and $S_t \geq 0$ is the value of unmet demand at the end of period t, then the inventory balance constraint is converted to $I_{t-1} + X_t + S_t = I_t + d_t$ and the term $\sum_{t=1}^{T} l_t S_t$ is added to the cost function, where $S_0 = 0$.

- **Time windows:** In these problems, there is a time window such that if the demand is satisfied within that time window there is no penalty in terms of inventory or backlogging costs.

- **Stochastic and elastic demands:** It is more realistic to consider the demand as a stochastic variable that must be predicted (with an associated error). The lot-sizing model including the cost parameters can also be stochastic. Elastic demand happens when the demand is a function of production unit price. So, the demand increases or decreases depending on the unit price.

- **Production capacity constraints:** The capacitated SILSP is considered in many works. These problems are harder than uncapacitated versions. In [67], the problem is characterized in terms of the structure of the unitary costs and the capacity over time. Each case corresponds to a quadruple $\alpha/\beta/\gamma/\delta$ that represents the special structure of unitary setup cost s_t, unitary production cost p_t, unitary holding cost h_t and capacity, respectively. The parameters are denoted by G, C, NI, ND, and Z which correspond to General structure, Constant, Non-Increasing, Non-Decreasing, and Zero, respectively. The case $G/G/G/G$ is NP-hard in the weak sense and pseudo-polynomial time algorithms have been designed to solve it. The polynomial cases include the $NI/G/NI/ND$, $NI/G/NI/C$, $C/Z/C/G$, $ND/Z/ND/NI$, $G/G/G/C$ models. The cases $Z/G/G/G$ and $NI/G/G/C$ have been solved by $O(TlogT)$ algorithms.

- **Inventory constraints:** In these uncapacitated SILSPs, a lower bound and/or an upper bound are set for the inventory level. In [68], they showed that the problem can be solved by a $O(T^3)$ dynamic programming algorithm where the concept of regeneration intervals is used for decomposing the solution. The production and holding costs are assumed to be concave. The complexity is

reduced to $O(T^2)$ in [69]. When all costs are linear and production variables are integer, the complexity has been further reduced to $O(TlogT)$.

- **Perishable inventory:** In these problems, there is a deterioration rate for the products and the holding cost generally depends on the time duration the product is stored in the inventory. Examples are storing food, pharmaceuticals, chemicals, and blood.

- **Cost structures:** The structure of the functions $f_t^p(\cdot)$ and $f_t^h(\cdot)$ is a main factor in the complexity of SILSPs. In [64] it is shown that problem (1.1.11) can be solved in $O(T^2)$ time. In [70], it is shown that if the cost functions are generally concave the problem can still be solved with the same time complexity, $O(T^2)$. In [68], an $O(T^3)$ algorithm is presented for the case of bounded inventory and concave cost function which was subsequently improved in [69] to $O(T^2)$. A further generalization is considering a piecewise concave cost function which covers several practical scenarios of interest such as quantity discounts, minimum order quantity, capacitated production, and bounded inventory [71]. Figure 1.6 shows some examples of concave production and holding costs. A concave production cost can be used to represent a different setup cost and quantity discount for larger orders. Concavity can also represent different holding costs corresponding to using a different warehouse for greater inventory levels, for example. A valuable work that integrates capacitated production, bounded inventory, and backlogging while considering piecewise concave cost functions with several breakpoints for the production and holding costs is proposed in [72]. The results about regeneration intervals presented in [68] are generalized,

proving that there is an optimal solution in which between two consecutive breakpoint levels for the inventory values, there is at most one order which is not equal to one of the production breakpoints. They use this property to develop a pseudo-polynomial dynamic programming algorithm for solving the general problem. Our work is a special case of their problem where there is one fixed breakpoint for the production cost which represents a quantity discount, the production and holding costs are linear (non-increasing in time), and there is no setup cost. By defining special regeneration intervals and proving three structural properties for the optimal solution, an $O(T^2)$ algorithm is proposed to solve the problem. It is also proved that the case of two items is NP-hard.

There is an extensive literature for extensions of SILSPs. The number of publications has doubled since 2007. We mention some of the most relevant and recent works. In [73] the complexity of SILSPs with different cost structures is discussed. An SILSP with quantity discount is presented in [74] and solved in $O(T^3)$ and $O(T^2)$ time for the two cases of all-unit and incremental discount structures, respectively. A fully polynomial time algorithm is designed in [75] for some quite general cost structures. In [76] the problem considered has a concave holding cost and piecewise concave production cost and it is proved that if the breakpoints are time-invariant and the number of breakpoints is fixed, the problem is solvable in polynomial time with a complexity of $O(T^{2m+3})$, where m is the number of breakpoints for the production cost. The complexity is reduced to $O(T^{m+2}logT)$ in [77]. The production cost is defined as a set of connected linear segments in [78] and with a concave holding cost, they show that the capacitated problem is solvable in $O(mT^3)$ time, where m is the average number of breakpoints per period. Another work closely related to ours,

presented in [79], is the case with capacitated production, one fixed breakpoint for the production cost, and linear holding cost, which is solved in $O(T^4)$ time. A pseudo-polynomial dynamic programming algorithm is developed in [80] for the case of two items with capacitated production. A swarm optimization approach is proposed for capacitated multi-item lot-sizing in [81].

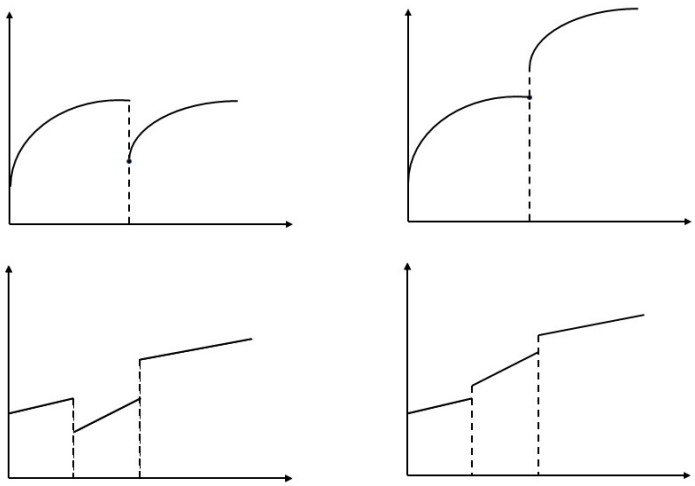

Figure 1.6: Examples of piecewise concave cost functions

1.2 Research Summary

As explained, regarding the first part of the thesis, there are many works in different layers to reduce the power consumption in a data center. They typically report a considerable reduction in power consumption, for example, more than 20%. However, due to the lack of a theoretical framework, the results are not guaranteed to be reproducible for different system instances. The optimization problems also have

some assumptions and structure that may not be applicable to other scenarios. On the other hand, because even a small reduction in power consumption is valuable from the energy and climate perspectives, building a theoretical framework for exploring the possible improvements is desirable. The other important aspect is time efficiency for online operation which is also connected with finding efficient algorithms for thermal-aware workload distribution. In addition, it is a challenge to tackle thermal models in optimization problems, due to their complexity and the fact that they may not be available as a closed-form function, for example, they might be data-driven models or models based on solving differential equations. The problems are NP-complete and usually take the form of mixed integer programming or integer programming problems. Due to complicated trade-offs, a greedy heuristic may not guarantee a near-optimal solution. Meta-heuristics such as genetic algorithms are also slow due to their iterative nature, especially for constrained optimization. Insight about the solution while designing and revising a meta-heuristic is also challenging and problem-specific. All of the mentioned challenges and the similarity of the proposed problems in the literature in terms of the type of cost function and the constraints motivated us to build a bottom-up framework for the theoretical analysis of optimization problems in this area.

In the first part of this thesis, we propose a general approach for tackling thermal-aware workload distribution problems. We start with studying a general nonlinear problem proposed in [51] where two general multivariable functions are considered for the cooling power consumption and the thermal model. The problem includes the effect of different cooling variables and heat circulation. According to [39] and to investigate the effectiveness of our approach, two red-line temperatures are considered

for idle and fully-utilized servers which allows less cooling effort for the idle servers, which in turn increases the cooling efficiency. This makes the problem a mixed integer programming problem which is proven to be NP-complete. To simplify the problem, there is the assumption that the servers are identical. This still allows the fundamental difficulty of the problem, the trade-off between the server temperatures, the cooling effort and the role of workload distribution, to be considered. So, the problem is general enough to consider several important effects and at the same time simple enough to be seen as a basic optimization problem for this area. Our approach is to first linearize the problem and then design a heuristic for approximation of the solution of the mixed integer linear programming problem. The heuristics are based on rounding the solution of the relaxed problem while considering the increase in the dominant cooling variable (the cooling variable with highest cooling effect) for each server. Linearization allows one to study the inherent difficulty of the problem while also being a time-efficient approach. For evaluation of our approach, we use a data center modeled in [20] where the thermal model is based on solving a set of differential equations. We perform linear regression on the cooling power consumption and the thermal model which was verified to be sufficiently precise and time efficient to generate near-optimal solutions. Two baseline algorithms are considered for comparisons, simple rounding and a genetic algorithm for an efficient search around the simple rounding solution. Two bad cases for simple rounding are also analyzed. The details are presented in Chapter 2. In the next effort, we consider the effect of server workload fluctuations that may be harmful to the reliability of the servers (due to frequent changes in temperature) and also for the QoS (due to imposing delay). To do this, we generalize the problem to include the cost of workload fluctuations. In other words,

the first problem can be seen as a single time slot problem while the second problem considers several time slots with added switching costs. We generalize the proposed heuristic accordingly. An MPC approach is also proposed for the case when workload (demand) predictions are available. The MPC approach helps to decrease the size of the problem and adapt to the fact that predictions generally become more accurate over time. The baseline algorithm is simple rounding. The evaluations verify the effectiveness of the proposed heuristic and the ability of the MPC approach to find a near-optimal solution. The details are explained in Chapter 3.

In the second part of the thesis (Chapter 4), an SILSP with bounded inventory and quantity discount is defined and solved in $O(T^2)$ time by a dynamic programming algorithm. To receive the discount it is ideal to make bulk purchases but it may not be possible because of the bounded inventory and a purchase in a given period may also prevent some future bulk purchase possibilities (because of limited remaining space in the inventory). We first define the problem with some structure for the costs, there are two price levels for each time period, there is a quantity threshold for the discount, the prices are non-increasing over time, there is a linear holding cost and there is no setup cost. To solve the problem, we first show some structural properties for an optimal solution while using the notion of regeneration intervals. Three lemmas are given followed by an efficient dynamic programming algorithm to solve the problem. The algorithm is based on a key structural property that may be of more general interest, that of a just-in-time ordering policy. The case of two items is also proved to be NP-hard by a reduction to a partitioning problem.

The concluding remarks for this thesis are presented in Chapter 5.

Chapter 2

Linearized Data Center Workload and Cooling Management

The content of this chapter is reproduced from: "Linearized data center workload and cooling management", S. Rostami, D. G. Down, and G. Karakostas, submitted to IEEE Transaction on Automation Science & Engineering, which is under the first round of revision. The author of this thesis is the first author and the main contributor of this manuscript.

Abstract

With the current high levels of energy consumption of data centers, reducing power consumption by even a small percentage is beneficial. We propose a framework for thermal-aware workload distribution in a data center to reduce cooling power consumption. The framework includes linearization of the general optimization problem and proposing a heuristic to approximate the solution for the resulting Mixed Integer Linear Programming (MILP) problems. We first define a general nonlinear power optimization problem including several cooling parameters, heat recirculation effects, and constraints on server temperatures. We propose to study a linearized version of the problem, which is easier to analyze. As an energy saving scenario and as a proof of concept for our approach, we also consider the possibility that the red-line temperature for idle servers is higher than that for busy servers. For the resulting MILP problem, we propose a heuristic for intelligent rounding of the fractional solution. Through numerical simulations, we compare our heuristics with several existing algorithms. In addition, we evaluate the performance of the solution of the linearized system on the original system. Finally, the results show that the proposed approach can reduce the cooling power consumption by more than 10 percent compared to the case of continuous utilizations and a single red-line temperature.

Note to Practitioners

The approach proposed in this paper can be used as a baseline approach for tackling different power optimization problems for data centers. It proposes a general and efficient approach to solve the nonlinear problems especially when the closed-form

functions, for example for the thermal model, are not available. A general nonlinear power optimization problem is defined. Linearization is then used to analyze the general problems and to propose a corresponding heuristic for approximation of the solution for ILP problems. The heuristic can be modified to solve similar problems. Scalability is an important factor that has been addressed in this approach because the linearized problem can be solved efficiently and calls to implicit functions are eliminated. So, this approach can be efficiently used for time sensitive calculations. The simulation results also show that the accuracy is maintained for larger problem sizes. For use in practice, better linearization approaches than regressing on a single linear function may be required. Our approach is adaptable to such settings.

Keywords: data center, thermal-aware workload distribution, integer programming, linearization, power consumption, red-line temperatures

2.1 Introduction

Data centers have been deployed to perform a large volume of computing tasks at considerable operational costs. With the increasing demand for cloud computing and internet services, data center operational costs have increased dramatically [1]. To reduce the power consumption of a data center, a number of techniques have been proposed at different levels including chip, server, rack and room levels [2]. This work focuses on thermal-aware workload distribution to reduce cooling power consumption, a main contributor to overall power consumption in a data center [1]. Decreasing the power consumption by even a small percentage may generate significant energy

savings. There have been a wide variety of optimization problems posed for such problems. Often, strong theoretical support is lacking. A general framework for systematic analysis in this area would be desirable.

In terms of solution techniques, a deep learning approach is used in [3] and [4]. A game-based strategy is presented in [5]. Proposing a heuristic (often a greedy heuristic) is also common [6][7][8]. However, analysis of such heuristics is somewhat lacking. Some theoretical results are given in [9]. Overall, we feel that a deeper analysis of the optimization approaches for such data center problems would be useful to guarantee an acceptable power reduction in different system instances. One way to do this is to adopt an approach that is standard in the area of control systems, for example. We propose a holistic approach for solving thermal-aware work distribution problems. This involves linearizing a nonlinear system, with the idea that the linear system is model-independent and easier to reason about. We find that this is the case in our setting, and we expect that this viewpoint can be of more general applicability. We believe that developing approaches to solve the resulting linear problem gives the promise of a single approach to solve a range of problems, rather than having to develop different heuristics for different (nonlinear) problems. If the decision variables are continuous, the linear problem can be solved by standard algorithms, otherwise if there are integral variables due to some energy saving or practical considerations, a single heuristic may address a range of similar problems. The linearization approach is especially efficient when closed-form functions are not available, for example evaluating the thermal model requires the numerical solution of a set of differential equations, which can become computationally quite expensive. Linear regression can then be used to estimate the functions. We also show that

the proposed heuristic is more time efficient compared to popular approaches such as genetic algorithms and other meta-heuristics, for example presented in [10][11][12] (which typically also require ad hoc customization to perform well).

There are similarities and differences between the power optimization problems defined in the literature for thermal-aware workload distribution. The choices of power consumption model, performance and temperature constraints, and thermal models are the key points in the power optimization problems defined. Most of the problems are aimed at minimizing the total power consumption, consisting of the summation of IT (Information Technology) and cooling power consumption. Different models for cooling power consumption are used, depending on which cooling facilities are present, i.e., CRAC (Computer Room Air Conditioner) units, fans, chillers [6][13][14]. IT power consumption is also a function of utilization of a computing node. In some works, frequency scaling is considered so that IT power consumption is also a function of the operating frequency of a server [14][15]. Performance and temperature constraints are also commonly considered in the existing literature. In terms of performance constraints, it is typical to require that the summation of workload assignments to the servers (computing nodes) reaches a target demand level. There may also be a constraint on the maximum workload assigned to each server [7][9][15]. For the temperature constraints, there is typically a red-line temperature that the inlet temperature of each server must not exceed. Most of the literature assumes that the red-line temperature is a constant. However, in [7] the red-line temperature for each server is a function of its utilization. In terms of thermal models, both steady-state and transient models are considered in the literature. Transient models are useful for real-time control settings [12][16][17][18]. Steady-state models

are also common [6][7][8][14]. With respect to thermal models, most of the literature considers the effect of heat recirculation on the server temperatures [10]. In this case, the inlet temperatures are a function of the servers' power consumption and the cooling parameters [6]. However, the models presented in [7] and [14] do not consider recirculation effects which makes solving the optimization problem easier by removing the thermal interactions among the servers. The effect of computing node failure on heat recirculation patterns is also considered in [19]. In this work, we formulate a thermal-aware workload distribution problem that can be seen as a basic problem, integrating the main aspects of the problems defined in the literature. A steady-state model with heat recirculation effects, several cooling parameters, and two red-line temperatures corresponding to idle and fully-utilized servers is considered.

Server consolidation is another possibility that is considered in the literature [6][8][14]. The goal is to reduce the total idle power consumption by turning idle servers off. However, when turning idle servers off, the thermal model may change because the heat recirculation pattern among the servers changes. In this case, linear dependence of the temperatures on the servers' power consumption may be problematic. However, linear dependence is assumed in several papers, including [6] and [8]. In this work, we assume that idle servers are not turned off, which corresponds with practice. However, our approach can also be applied to the case of server consolidation.

Heterogeneous and homogeneous data centers are considered in the literature [6][9][10][20]. Homogeneous data centers are a special case of heterogeneous data centers, where the servers are identical. In this case, the power optimization problem

is simplified as can be seen in [6]. If the number of servers chosen to be working is fixed, the total IT power consumption due to servers is a constant, leaving only the cooling power consumption to be minimized. In [7], the data center model is heterogeneous and differences among the servers are highlighted. However, in [9], a homogeneous data center is considered as a practical case. In this work we focus on homogeneous data centers as a first step towards tackling more complicated problems.

In this paper, we define a power optimization problem for a data center that is general enough to consider nonlinear dependencies and take into account different cooling parameters and heat recirculation effects. For our initial explorations, we make some assumptions. We assume that servers are identical and they are either idle or fully-utilized. According to the temperature model in [7], we study the effect of considering different server red-line temperatures as a function of server utilizations to see if exploiting this difference can reduce the total power consumption. To simplify these dependencies, we consider the case of having two red-line temperatures corresponding to idle and fully-utilized servers. In this case the power optimization problem becomes a mixed integer linear programming problem. A physical model of a data center involves general nonlinear functions for the cooling power consumption and server temperatures. We then linearize these functions. After showing that the resulting linear optimization problem is NP-complete, we introduce simple rounding as a baseline approximation algorithm. Next, we introduce two problematic instances for simple rounding. An approximation scheme is then proposed to perform a more intelligent rounding. For comparison, a generic genetic algorithm is also introduced. In the last section, we evaluate the proposed schemes for synthetic systems (to test the limits of our heuristics) and a model developed from a working, experimental data

center housed at McMaster University. We also evaluate the quality of the solution of the linearized problem by substituting it into the original, nonlinear model for the data center. Finally, as a concrete demonstration of our approach, we compare the case of considering two different red-line temperatures with the case of continuous utilizations but with one red-line temperature. Our contributions can be summarized as follows:

- Proposing a time-efficient holistic approach for solving thermal-aware workload distribution problems, including linearization of the problem and using the solution for the original system
- Formulating a basic thermal-aware workload distribution problem, integrating the main aspects of the problems introduced in the literature
- Demonstrating the effectiveness of solving the general problem through linearizing the underlying thermal and power consumption models and formulating a corresponding optimization problem for the linearized system
- Theoretical analysis of the linear version including demonstrating that common approaches have the potential to be problematic
- Evaluation of the proposed heuristic on synthetic and practical instances
- Evaluating the quality of the proposed solution approach by comparing with solving the original nonlinear problem directly
- Using the proposed approach to evaluate the energy savings possible for different red-line temperatures for idle and busy servers

The remainder of the paper is organized as follows. Section 2.2 gives the underlying nonlinear system model and corresponding power optimization problem. Section 2.3 discusses the linearized version of the nonlinear problem. Section 2.4 is concerned with the solution of the linearized problem. Section 2.5 provides experimental evaluation of our approach, both in terms of the limits of our heuristics and the applicability to a problem using a model from an operational data center. Finally, Section 2.6 provides concluding remarks and future research directions.

2.2 System Model and Motivation

Our starting point is a general power optimization problem introduced in [21]. We consider n servers located in racks and cooled with a cooling system parameterized by m adjustable cooling parameters, i.e., CRAC units' reference temperatures, air flows, and chilled water setpoint temperature. The cooling parameters are denoted by $v_1, v_2, ..., v_m$. The aim is to minimize the total power consumption for the data center consisting of the summation of cooling and IT power consumption. The cooling power consumption is a function F of the values of cooling parameters. The IT power consumption is a function of server utilizations, denoted by $\rho_1, \rho_2, ..., \rho_n$. So, the decision variables are $v_1, ..., v_m$ and $\rho_1, ..., \rho_n$. We will also represent the decision variables by the vectors v and ρ. We assume that the servers are identical.

The cooling variables are continuous and have a lower and upper bound given by $V_{LB}^{(j)}$ and $V_{UB}^{(j)}$ for v_j, $j = 1, ..., m$, i.e., input vectors V_{LB} and V_{UB} are of size m. For the utilization, we assume that servers are either idle or fully-utilized, so that the utilization of server i, denoted by ρ_i, is 0 or 1, $i = 1, ..., n$. To satisfy performance

constraints, the maximum utilization of each server may be required to be strictly less than 1. However, a utilization of 0 is common in the literature because it also allows server consolidation by turning the idle servers off. All these cases can be easily transformed to a similar 0-1 optimization problem. Our focus is on the server provisioning problem, where a set of working servers is chosen and load balancing is performed, while there are two red-line temperatures.

While minimizing power consumption, thermal constraints must also be respected to ensure reliable operation of servers. The inlet temperature of a server should not exceed a red-line temperature. The vector of inlet temperatures is denoted by T with size n. We consider a steady-state model for the inlet temperatures. In this case, the inlet temperatures depend on the value of cooling parameters and the servers' IT power consumptions (which are a function of server utilizations). This dependence is given by the function $M(v, \rho)$. Referring to [7], the red-line temperature for a server can also be a function of its utilization. In fact, more lightly utilized servers can have higher red-line temperatures. As we have assumed that server utilizations are either 0 or 1, we consider two red-line temperatures for each server, T_{idle} and T_{busy}, corresponding to the utilizations 0 and 1, respectively. According to the model presented in [7], $T_{idle} > T_{busy}$. This allows a lower cooling effort for idle servers, as they can tolerate a higher temperature.

There is also a performance constraint, related to quality of service. Because the servers are either idle or fully-utilized and have the same computational capacities, the performance criterion is translated to a required number of servers working, denoted by D. This means at least D servers should be working (note that the value of

D may be higher than required to support desired throughput, for example to satisfy latency requirements). As an increase in the number of busy servers tends to increase the inlet temperatures resulting in higher cooling and IT power consumption, the performance constraint is tight for the optimal power consumption, so exactly D servers are working. In this case, the IT power consumption is fixed and minimizing the cooling power consumption is equivalent to minimizing the total power consumption.

Based on the preceding discussion we can formulate the following optimization problem:

$$\begin{aligned}
\min \quad & F(v) && \text{(P0)}\\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i \geq D && \text{(2.2.1a)}\\
& M(v,\rho) \leq T_{idle} 1_{n\times 1} - (T_{idle} - T_{busy})\rho && \text{(2.2.1b)}\\
& v \geq V_{LB} && \text{(2.2.1c)}\\
& v \leq V_{UB} && \text{(2.2.1d)}\\
& \rho_i \in \{0,1\} \quad \forall i = 1,...,n && \text{(2.2.1e)}
\end{aligned}$$

where $F(v)$ is the cooling power consumption corresponding to the cooling variable vector v and $M(v,\rho)$ is the vector-valued function corresponding to the thermal model whose ith element is the inlet temperature of server i. Constraint (2.2.1a) is the performance constraint, and constraint (2.2.1b) limits the inlet temperatures to be less than the corresponding red-line temperatures. The right hand side of constraint (2.2.1b) is the vector representation of red-line temperatures corresponding to server

Table 2.1: Notations and Definitions

Notation	Definition
n	total number of servers in the system
m	total number of cooling variables
D	the current demand in terms of required number of working servers
ρ	the vector of server utilizations with size n
v	the vector of cooling variables with size m
V_{LB}	the vector of lower bounds for the cooling variables with size m
V_{UB}	the vector of upper bounds for the cooling variables with size m
T_{idle}	the red-line temperature for idle servers
T_{busy}	the red-line temperature for fully-utilized servers

utilization ρ, where depending on whether ρ_i is 0 or 1, the ith red-line temperature is T_{idle} or T_{busy}, respectively. The term $1_{n\times 1}$ represents a vector of size n with all elements equal to 1. The notations and their description are summarized in Table 2.1.

As explained, the performance constraint is tight for an optimal solution and, therefore, we need to decide which D servers should work to minimize the cooling power consumption while satisfying the temperature constraints.

As an example, we consider the data center modeled in [22] [23], an experimental data center housed at McMaster University. Similar to their model, there are five racks. Each rack consists of five servers. So, there are 25 servers in total. The cooling variables are the chilled water setpoint temperature and total air flow generated by two fans located at both ends of the five racks (the air flow is divided equally between the two fans). Thus, there are two cooling variables. Fig. 1 shows the top view for this system. To calculate the server temperatures in terms of the two cooling variables and the server utilizations, we use a model from [22][23]. This is the function M in problem (P0). Because M is not explicitly given and its calculation is based on

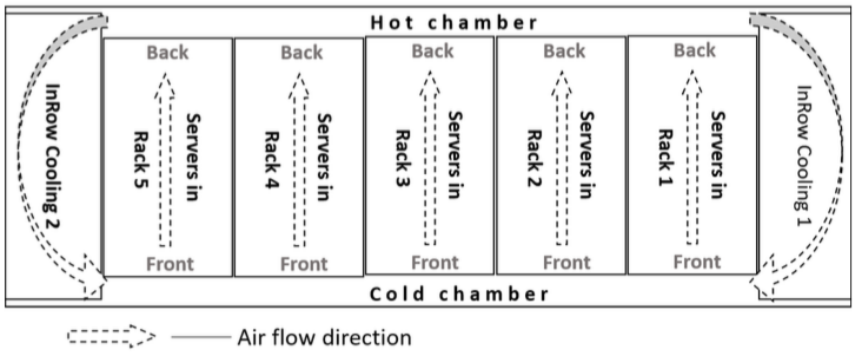

Figure 2.1: The data center's top view according to [22][24]

numerically solving a set of differential equations, it is expensive to evaluate. For example, using the platform MATLAB R2021b running on a 64-bit system with an i7-1185G7 processor and 8-GB RAM, each call takes about 2 seconds. So, we can infer that for any solver, the bulk of the execution time is evaluation of the function M at each iteration while searching the feasible space of the problem. For the function F in problem (P0), we also use the models developed in [22][23] for fan and chiller power consumption.

The explained inefficiency is the main motivation for our proposed approach, which is described in the following sections. We will show that even the linearized version of the problem is NP-complete.

2.3 Problem Definition

To find efficient heuristics to approximate the solution of problem (P0), we first make some simplifying assumptions, which allow us to transform problem (P0) to the version of the problem considered in this paper. The assumptions are used for developing heuristics in Section 2.4.3.

- Our main assumption is that F and M are linear functions. This allows us to achieve two goals. The first is to check the validity of the following procedure: performing regression and linearizing the system, solving the linear optimization problem and using the solution for the original system. A secondary goal is to assess the difficulty of problem (P0) by first studying its simplest version.

- F is an increasing function of the cooling variables. This is without loss of generality for two reasons. First, the cooling power consumption is monotone with respect to cooling system parameters such as fan speeds and reference temperatures of CRAC units. Second, for some variables such as reference temperatures of CRAC units, the function F is decreasing, however by transforming the corresponding cooling variables, we can convert F to an increasing function.

- M is a non-increasing function in terms of the cooling variables. This follows from the fact that to decrease the inlet temperatures, the cooling power consumption must be increased by increasing the value of cooling variables (recall that F is an increasing function).

- M is a non-decreasing function in terms of the server utilizations. This means when a server's utilization increases (equivalent to an increase in the server power consumption), the inlet temperatures also increase. This is also a reasonable assumption in practice.

- The problem is feasible for every vector ρ, i.e., if we fix the vector ρ, the feasible space for v is not empty. This is a result of the assumption that, in practice, maximizing the cooling effort is enough to prevent all of the servers from exceeding their red-line temperatures.

Using our assumptions, the following notation will simplify our exposition:

- $M = -Av + B\rho + E$, where $A_{i,j}, B_{i,j} \geq 0$ (all entries are non-negative). $A_{n\times m}$ is the cooling matrix, $B_{n\times n}$ is the heat recirculation matrix and $E_{n\times 1}$ is the constant part. $B_{i,j} \geq 0$ allows the optimal solution to have exactly D servers working. $A_{i,j} \geq 0$ will allow us to determine the dominant cooling variables in the proposed heuristic in Section 2.4.3.

- $b = T_{idle}$, and $a = T_{idle} - T_{busy} > 0$

- The cooling power consumption is $F = \sum_{j=1}^{m} c_j v_j$, where the coefficients c_j relate the value of cooling parameters to the power consumption of the corresponding cooling facilities. We normalize the cooling variables so that $F = \sum_{j=1}^{m} v_j$. This changes the values of A, V_{LB} and V_{UB} (the jth column of A is divided by c_j and the jth element in V_{LB} and V_{UB} is multiplied by c_j) but for convenience we use the same notation for the formulation.

As a result, our problem can be formulated in scalar form as follows:

$$\begin{aligned}
\min \quad & \sum_{j=1}^{m} v_j \\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i \geq D \\
-\sum_{j=1}^{m} A_{l,j} v_j + \sum_{i=1}^{n} B_{l,i} \rho_i + a\rho_l \leq b - E_l \quad & \forall l = 1, ..., n \\
v_j \geq V_{LB}^{(j)} \quad & \forall j = 1, ..., m \\
v_j \leq V_{UB}^{(j)} \quad & \forall j = 1, ..., m \\
\rho_i \in \{0, 1\} \quad & \forall i = 1, ..., n
\end{aligned} \tag{2.3.1}$$

The problem is a mixed integer programming problem. The first constraint can be seen as a covering constraint, but considering that exactly D servers are working in the optimal solution, it is better seen as an assignment constraint that assigns the workload to a set of D servers. The second set of n constraints can be interpreted as packing constraints in terms of ρ and covering constraints in terms of v. As we will see in Sections 2.4 and 2.5, the problem is NP-complete and solving it does not scale well using existing packages (in our case MATLAB). This is the motivation for proposing approximation schemes in the next section.

2.4 Approximation of the Optimal Solution

We first show that problem (2.3.1) is NP-complete. We then introduce simple rounding as a base algorithm to approximate (2.3.1). We show that simple rounding may be problematic by constructing two bad cases for the algorithm. Then we propose an approximation scheme to perform more intelligent rounding. Finally, we present a generic genetic algorithm as a standard approximation approach. The main purpose of presenting simple rounding and a genetic algorithm is to better understand how well the proposed scheme performs as compared to standard schemes.

2.4.1 NP-completeness

To show the NP-completeness of the problem, we assume a particular instance where $m = 1$, $A = 1_{n\times 1}$ and $E = 0_{n\times 1}$. In this case, there is only one cooling variable, denoted by v. We also set $V_{LB}^{(1)} = 0$ and $V_{UB}^{(1)} = \infty$. Then the problem becomes

$$\begin{aligned} \min \quad & v \\ \text{s.t.} \quad & \sum_{i=1}^{n} \rho_i \geq D \\ -v + \sum_{i=1}^{n} B_{l,i}\rho_i + a\rho_l \leq b \qquad & \forall l = 1, ..., n \\ \rho_i \in \{0,1\} \quad & \forall i = 1, ..., n \\ v \geq 0 & \end{aligned} \tag{2.4.1}$$

An equivalent formulation of (2.4.1) is

$$\begin{aligned} \min \quad & \max \ B'\rho \\ \text{s.t.} \quad & \sum_{i=1}^{n} \rho_i = D \\ & \rho_i \in \{0,1\} \quad \forall i = 1, ..., n \end{aligned} \tag{2.4.2}$$

where $B' = B + aI_{n\times n}$ and I is the identity matrix. The problem is obviously in NP, since, given an objective value V, a verifier can check in polynomial time whether a provided solution achieves V. In [6], problem (2.4.2) is shown to be NP-complete by a reduction to a partitioning problem.

2.4.2 Bad Cases for Simple Rounding

We first consider simple rounding as a baseline approach. In this scheme, we solve the LP relaxation of (2.3.1) (by allowing the utilizations to be fractional, $0 \leq \rho_i \leq 1$) and round the D largest values to 1 and the remaining values to 0. We will see that simple rounding may have unacceptable performance. To show this, let us consider two special instances of (2.3.1):

- **Case 1:** There are $p \leq n$ entries equal to 1 for each row of B; the remaining entries are 0. For each row of A one entry is equal to a constant $q > 0$ and the others are 0. Also, $V_{LB}^{(j)} = v_L$, $\forall j = 1, ..., m$, and $E = 0_{n\times 1}$.

- **Case 2:** In B the entries are 0 or 1. If $A_j \neq A_i$, where A_i is the ith row of A, then $B_{i,j} = B_{j,i} = 0$ which means that servers i and j are *isolated* from each other. Assume S is a set of non-isolated servers (those with equal corresponding rows in A), then for B_S (the sub-matrix of B corresponding to the servers in S) the number of ones in each row and each column is p. Finally, $E = 0_{n\times 1}, V_{LB} = 0, V_{UB} = \infty$.

Lemma 1 *If in Case 1,* $\frac{D}{n}(p + a) \leq b + qv_L$*, then there is an optimal solution for the LP relaxation of* (2.3.1) *that distributes the load D uniformly among the servers and satisfies $v^* = V_{LB}$. In this case, simple rounding does not guarantee a bounded approximation factor (the ratio of the solution found by the algorithm to the optimal solution).*

Proof: The inequality $\frac{D}{n}(p + a) \leq b + qv_L$ shows that in Case 1 by distributing the load D uniformly and setting $v = V_{LB}$ we can satisfy all the constraints in the problem. So an optimal solution is $v^* = V_{LB}, \rho^* = \frac{D}{n}1_{n\times 1}$ and $c^* = mv_L$, where c^* is the optimal fractional cost. With respect to the approximation factor, it is enough to show that it is not bounded for one specific example of Case 1. We define $\hat{B}$ as follows. We set $\hat{B}_{i,i} = 1, i = 1, ..., D$, while the remaining entries that are set to one (each row has a total of p ones according to the definition of Case 1) are chosen from the columns $D + 1$ to n. We show that when $B = \hat{B}$, the ratio of the cost for simple rounding to the optimal integral cost can grow arbitrarily large.

Because of symmetry, for simple rounding it does not matter which ρ_i^* values are rounded up to 1. Assume that the ρ_j^* values with $B_{i,j} = 1$ for some row i in B are rounded. Depending on which of D or p is smaller, D or p ρ_j^* values are rounded, respectively (potentially including ρ_i^*). Now suppose that

$$\frac{D}{n}(p+a) \leq b + qv_L < min(D,p) + a, \tag{2.4.3}$$

and define $s = \frac{min(D,p)+a}{b+qv_L}$ ($s > 1$ because of the second inequality). To satisfy the constraints in (2.3.1) for the rounded ρ^*, we must have $b + qv' \geq (b + qv_L)s = b + (sq + \frac{(s-1)b}{v_L})v_L$, where v' is the value of the corresponding cooling variable needed for cooling server i (that with non-zero entry in the ith row of A). So, $\frac{v'}{v_L} \geq \frac{(s-1)b}{qv_L}$ and

$$\frac{\hat{c}}{c^*} \geq \frac{v' + (m-1)v_L}{mv_L} \geq \frac{v'}{mv_L} \geq \frac{(s-1)b}{mqv_L} \tag{2.4.4}$$

where $\hat{c}$ is the cost of the solution for simple rounding. By reducing v_L or q in $\frac{(s-1)b}{mqv_L}$ (while maintaining (2.4.3), for example by increasing n), $\frac{\hat{c}}{c^*}$ can grow unboundedly. For example, set $p = 3, a = 1, b = 2, q = 1, D > p, m = 1$, then $\frac{\hat{c}}{c^*}$ is at least $\frac{2(2-v_L)}{(2+v_L)v_L}$, which can be made arbitrarily large by decreasing v_L (and decreasing $\frac{D}{n}$ to maintain (2.4.3)). We show that the optimal integral solution can also be equal to c^*. It is possible to have a total load of at least D by choosing at most $\lfloor b + qv_L - a \rfloor$ servers in each row of B (the specific choice depends on the structure of B), hence the optimal fractional and integral solutions are equal in terms of cooling variable values (setting the cooling variables equal to V_{LB} is enough to cool the servers). Let us set $b - a = 1$ ($\lfloor b + qv_L - a \rfloor \geq 1$) and $B = \hat{B}$. By choosing the servers $1, ..., D$, at most one server

is chosen at each row of B. □

Lemma 2 *In Case 2, there is an optimal solution for the LP relaxation of* (2.3.1) *that distributes the load uniformly among the servers that are not isolated. In this case, simple rounding does not guarantee a bounded approximation factor.*

Proof: In Case 2, assume that S with size $n' \leq n$ is a set of non-isolated servers with the corresponding row A_i in A. If the total load for S is D', it is not difficult to see that in this case distributing the load D' uniformly among the servers in S is an optimal workload distribution because based on the conditions for Case 2, the average of the values in $B_S \times \rho_S$ is fixed and equal to $\frac{D'p}{n'}$ (for the numerator, each element of ρ_S is repeated p times in the summation because there are p ones in each column of B_S), and for optimality we want to minimize the maximum of the values in $B_S\rho_S$. To be more precise, $max(B_S\rho_S + a\rho_S) - b$ determines the increase needed for the cooling variables to cool the servers in S. Similar to Lemma 1, we show that the approximation factor can be made arbitrarily large for $B = \hat{B}$, where in the ith row of $\hat{B}$, the ith to $(i+p-1)$th (mod n) entries are 1 (this is according to the definition of Case 2). For the proof, we assume $n' = n$ and the rounding is similar to what was done in the previous lemma. When $n' = n$, the optimal cost for ρ is proportional to $max(B\rho + a\rho) - b$, i.e., $c^*(\rho) = \alpha(max(B\rho + a\rho) - b)$ where α is a constant (if $max(B\rho + a\rho) \leq b$ then the optimal cooling variables are $V_{LB} = 0$). We show that the ratio of the approximate solution to the optimal integral solution can grow unboundedly. Note that according to the rounding in Lemma 1 and proportionality of the optimal cost to the max term we explained earlier, the optimal cost for the rounded solution, $\hat{c}$, is proportional to $min(D, p) + a - b$. Let us set $\frac{D}{n} = \frac{1}{p}, p < D$

and $B = \hat{B}$. If we choose the 1st, $(p+1)$th, $(2p+1)$th, ..., $((D-1)p+1)$th servers, then in each row exactly one server is chosen and the optimal integral cost, c^*, is proportional to $1 + a - b$ (assuming that $1 + a > b$). So $\frac{\hat{c}}{c^*} = \frac{p+a-b}{1+a-b}$, which can be made arbitrarily large by increasing p.

□

2.4.3 Proposed Approximation Schemes

We propose two variants of an approximation scheme for problem (2.3.1). The approximation schemes are based on assumptions that are appropriate for practical systems. The other approach is a standard genetic algorithm which is used for comparison purposes.

Maxmin Scheme

In this section, we propose a more intelligent rounding than simple rounding. Let us denote the optimal fractional solution for the LP relaxation of (2.3.1) by (ρ^*, v^*). In the proposed scheme, instead of just rounding ρ^*, the values in $B'\rho$ (recall that $B' = B + I_{n\times n}$ where I is the identity matrix) are also considered. Assume $\hat{\rho}$ is an integral workload distribution with total load D. If we replace ρ^* with $\hat{\rho}$, and set the cooling variables equal to v^*, the temperature constraints for some servers are violated. Thus, some values in v^* must increase to compensate for these violations. The sizes of the violations depend on the values in $B'\hat{\rho}$. We compute the violations for $\hat{\rho}$ as follows:

$$(\sum_{i=1}^{n} B'_{l,i}\hat{\rho}_i - (\sum_{j=1}^{m} A_{l,j}v_j^* + b - E_l))^+ \quad l = 1, ..., n \tag{2.4.5}$$

For the power consumption optimization, the total increase in v^* must be minimized. Now, if the maximum violation is small for some vector $\hat{\rho}$, the total increase needed in v^* to compensate should also be small. For the other violations corresponding to $\hat{\rho}$, there are two possibilities: they are much smaller than the maximum violation or they are on the order of the maximum violation. The first case can be fixed by a small increase. In the second case, due to correlations, we would expect at least some of the violation to be corrected when the maximum violation is addressed. In this case fixing the maximum violation will partly fix the other correlated violations. Two servers i and j are correlated if there are similarities between their corresponding rows in A and B' (servers that are near to each other may be correlated). Thus, with a modification (considering weights w_l defined below), the problem we solve is:

$$\begin{aligned} \min \quad & \max_{1 \le l \le n} \frac{[\sum_{i=1}^{n} B'_{l,i}\rho_i - (\sum_{j=1}^{m} A_{l,j}v_j^* + b - E_l)]}{w_l} \\ \text{s.t.} \quad & \sum_{i=1}^{n} \rho_i = D \\ & \rho_i \in \{0, 1\} \quad \forall i = 1, ..., n \end{aligned} \tag{2.4.6}$$

where w_l is the maximum entry in the lth row of A. The index of w_l in the lth row of A determines the dominant cooling variable corresponding to server l. The w_l values translate the value of violations to the increases needed in the cooling variables. We need this translation to compare the violations in terms of the increase needed to

compensate. To be more precise, we assume that a violation is mostly compensated for by the dominant cooling variable for each server and the ratio in the cost function measures the increase needed in this case.

However, the assumption that the violations that are on the order of the maximum violation are correlated may not be true in general. Problem (2.4.6) minimizes the maximum increase needed among all dominant cooling variables. This means it considers the maximum increase instead of the total increase in all dominant cooling variables. If there are $K \leq m$ dominant cooling variables, denoted by d_k, $k = 1, ..., K$, and S_k is the set of servers with dominant cooling variable d_k, the problem corresponding to minimizing the total increase is the following:

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \max_{l \in S_k} \frac{[\sum_{i=1}^{n} B'_{l,i}\rho_i - (\sum_{j=1}^{m} A_{l,j}v_j^* + b - E_l)]^+}{w_l} \\ \text{s.t.} \quad & \sum_{i=1}^{n} \rho_i = D \\ & \rho_i \in \{0, 1\} \quad \forall i = 1, ..., n \end{aligned} \tag{2.4.7}$$

The max term is for calculating the increase needed for the cooling variable d_k. So, the cost function is the total increase needed for all dominant cooling variables. Problem (2.4.7) better captures the correlation between the servers.

We still need to present a heuristic for solving (2.4.6) and (2.4.7) because they are also NP-complete (problem (2.4.1) is a special case for (2.4.6) and (2.4.7) in which there is only one cooling variable and A is a constant matrix). The heuristic consists of three different phases, specified as follows.

Phase 1 is for rounding ρ^* and generating $\hat{\rho}$. First note that ρ^* is also the optimal solution for the LP relaxation of (2.4.6) and (2.4.7), for which there is no need to increase the cooling variables. The idea is to gradually round the values in ρ^* by considering the cost values for (2.4.6) and (2.4.7). We know that the total workload for ρ^* is D and we need to find an integral solution with total load D for (2.4.6) and (2.4.7). Let us denote such a workload distribution by $\hat{\rho}$. To go from ρ^* to $\hat{\rho}$, some server workload modifications are required. We perform these modifications as follows. When a server that has positive load in ρ^* has its load reduced to zero, its load is distributed among the remaining servers proportional to their current load. This way, the new distribution imitates the previous distribution. If the resulting loads of some servers exceed 1, the extra load is again distributed among the remaining loaded servers. This is the base step in the following greedy heuristic. The heuristic selects the servers to be idle one by one. To select the next server to be idle, the heuristic successively reduces one of the $\rho_i > 0$ to zero and by performing the same redistribution process a new distribution $\rho^{(i)}$ results. The cost for $\rho^{(i)}$ is then calculated. Finally, the server that is selected to be idle is the one that minimizes the cost of (2.4.6) (or (2.4.7)). This process is repeated until there are exactly D servers with $\rho_i = 1$. The procedure is shown in Algorithm 1. This gradual rounding can also be performed for problem (2.3.1) directly. The issue is to calculate the cost function for (2.3.1), where a linear programming problem must be solved for each step. This increases the complexity of the heuristic as compared to problems (2.4.6) and (2.4.7).

In Algorithm 1, the servers that are idle in ρ^* remain idle. However, they may be working in the optimal integral solution. To allow these servers to work, in phase 2, after calculation of $\hat{\rho}$, there is another step that idles one of the working servers in

Algorithm 1 Calculation of $\hat{\rho}$

Solve the relaxed form of (2.3.1) and call the solution (v^*, ρ^*)
$S = \{i \in \rho^* | 0 < \rho_i \leq 1\}$
$l = |S| - D$
$\hat{\rho} = \rho^*$
while $l \neq 0$ **do**
    for $i \in S$ **do**
        $\rho^{(i)} = \hat{\rho}$
        $S' = S - \{i\}$
        $r = \rho_i^{(i)}$
        $\rho_i^{(i)} = 0$
        for $j \in S'$ **do**
            $\rho_j^{(i)} = \rho_j^{(i)} + \frac{\rho_j^{(i)}}{\sum_{k \in S'} \rho_k^{(i)}} r$
        end for
        while $\exists k \in S', \rho_k^{(i)} > 1$ **do**
            $r = \rho_k^{(i)} - 1$
            $\rho_k^{(i)} = 1$
            $S' = S' - \{k\}$
            for $j \in S'$ **do**
                $\rho_j^{(i)} = \rho_j^{(i)} + \frac{\rho_j^{(i)}}{\sum_{l \in S'} \rho_l^{(i)}} r$
            end for
        end while
        $x_i =$ value of the cost function of (2.4.6) (or (2.4.7)) for $\rho^{(i)}$
    end for
    Remove i with the smallest x_i from S and $\hat{\rho} = \rho^{(i)}$
    $l = l - 1$
end while
return $\hat{\rho}$

$\hat{\rho}$ and then switches it with each of the servers that are idle. It finally chooses the server to work as the one that minimizes the cost of (2.4.6) (or (2.4.7)). For problem (2.4.6) if more than one server minimizes the cost, then the server whose corresponding ρ minimizes the summation of violations, $\sum_{l=1}^{n} \frac{(\sum_{i=1}^{n} B'_{l,i}\rho_i - (\sum_{j=1}^{m} A_{l,j}v_j^* + b - E_l))^+}{w_l}$, is chosen. This final ranking is not needed for problem (2.4.7), because there is already a summation in the cost function.

Finally, in phase 3, the procedure explained so far is repeated for a specific number of iterations. At each iteration, a small perturbation of A, B and E is applied and the procedure is repeated. At the end, the solution that gives a lower cost for (2.3.1) is chosen (by fixing $\hat{\rho}$, (2.3.1) becomes a linear programming problem). The reason for these perturbations is the optimal fractional solution ρ^* is more sensitive to small changes in A, B and E in comparison to the optimal integral solution. We assume these small changes do not affect the optimal integral solution but when starting from multiple initial ρ^* values, we increase the chance of $\hat{\rho}$ values being close to the optimal integral solution by choosing the best solution ($\hat{\rho}$) among the perturbations.

Genetic Algorithm

Genetic algorithms are meta-heuristics that try to search the solution space of a problem efficiently. They have been previously used in works closely related to our problem. A genetic algorithm is used in [10] for thermal-aware task scheduling. An enhanced genetic algorithm is proposed in [11] for reducing cooling power consumption.

We also implement a genetic algorithm to see how such standard approaches work for this problem. The steps are similar to the algorithms presented in [25] and [26]

for multiple knapsack and set covering problems, respectively. Each solution is represented by the 0-1 vector of workload distribution with size n. The costs are the value of the objective function for problem (2.3.1). There is intensification around the simple rounding solution when generating the initial population. The algorithm's phases are as follows.

- **Population generation:** The initial population includes the simple rounding solution and the solutions where 10 percent of their ones (or zeros if the number of zeros is smaller, i.e., when $n - D < D$) are different from the simple rounding solutions. The perturbed utilizations are chosen uniformly at random. As assumed in Section 2.3, the cooling effort can always be increased enough to cool all the servers, so as long as there are D ones in a solution, it is feasible.

- **Parent selection:** This is similar to [25]. It is a tournament with size two. Two sets of members with size two are chosen randomly from the population and the member with smaller cost from each set is chosen.

- **Child generation (cross over):** This is similar to [26]. In the places that both parents have 1 or 0, the child imitates them. If f_1 and f_2 are the cost values for parent1 and parent2, respectively, a proportion $\frac{f_2}{f_1+f_2}$ of the remaining ones (there are D ones) are chosen from parent1 and a proportion $\frac{f_1}{f_1+f_2}$ are chosen from parent2. In this way, the child is also a feasible solution with D ones.

- **Mutation:** According to [25], one of the ones is randomly switched with one of the zeros.

- **Replacement:** This is similar to [26]. The algorithm rates the solutions in

terms of cost values. It then selects a solution uniformly from the second half of the rating and compares its cost with the cost of the child. If the child has a lower cost, it will replace the selected solution.

The parameters of the algorithm are the population size and the number of iterations for the last four phases in the algorithm. In Section 2.5, we will specify these parameters.

2.5 Evaluation

In this section, there are two main parts. In the first part, the performance of the proposed schemes will be evaluated for linear systems, in comparison to simple rounding, the proposed genetic algorithm and the Least Recirculated Heat (LRH) algorithm proposed in [6], which we will explain in greater detail later. We evaluate performance while considering scalability and running time of the schemes. The performance is the ratio of calculated cost of problem (2.3.1) for the algorithms to the optimal cost calculated with MATLAB. In the second part, the performance of our proposed approach is evaluated for the real nonlinear system and energy savings are calculated. All the calculations were performed in MATLAB. As explained in Section 2.2, the platform was MATLAB R2021b running on a 64-bit system with an i7-1185G7 processor and 8-GB RAM. We specify the functions used below.

In the first part, the first set of systems is artificial, corresponding to the bad cases for simple rounding as explained in Section 2.4. These cases may be challenging for the algorithms, and so this set of experiments is designed to test the limits of the proposed heuristics. The second set of results corresponds to the model of the

operational data center with 25 servers as described in Section 2.2. This is a five-rack air-cooled DC equipped with two in-row cooling units. The thermal model presented and validated for this system is zonal-based which is a standard alternative for full-CFD (Computational Fluid Dynamics) simulations [22][23]. We perform a linear regression on samples generated from the model of the operational data center to generate the linear system described by problem (2.3.1). To explore scalability, we generate two systems with sizes of 50 and 75 servers by scaling the matrices for the system with 25 servers.

In the second part, for the model of the operational data center (the 25 server model), we evaluate using the approximate and exact solution of the linear system in the original system described by problem (P0). The energy saving is evaluated by performing a comparison with the solution of the nonlinear system. We also compare these results with the case of continuous server utilizations and a single upper bound for the server temperatures, as an example of how our proposed approach can be used to generate useful operational insights.

2.5.1 Evaluation of the Proposed Schemes for Linear Systems

The first two systems correspond to the two bad cases for simple rounding as explained in Section 2.3. For Case 1, there are 25 servers and three cooling variables. For the cooling matrix A, for each row the entries are 0, except for one that is chosen randomly, and its value is also uniformly distributed on the interval [0,1]. For the heat recirculation matrix B, each server is affected by five servers including itself. Thus

there are five non-zero entries for each row of B, corresponding to the assumptions of Case 1. We set these five entries equal to 1. In this example, for the first row, the first five entries are equal to 1. For the second row we shift the ones to the right by one place. We continue in a similar manner for the remaining rows. To be precise:

$$B_{i,j} = 1 \Leftrightarrow j = (k + i - 2 \quad mod \quad 25) + 1, \quad k \in \{1, 2, 3, 4, 5\} \tag{2.5.1}$$

So, there are five ones in each row and each column of B. We also set $T_{idle} = 2$, $T_{busy} = 1$, $V_{LB} = (10^{-3} \quad 10^{-3} \quad 10^{-3})$, $V_{UB} = (10^{8} \quad 10^{8} \quad 10^{8})$. With these settings, when $D \leq 5$ the optimal cooling values are V_{LB}, which is a condition for Lemma 1. V_{LB} has small values to generate large approximation factors, as explained in the proof of Lemma 1. V_{UB} has large values to guarantee the feasibility of the problem. In Case 2, we only change A to correspond to the assumptions for Case 2. So, A has equal rows whose entries are randomly chosen on the interval [0,1]. In this case, when $D \leq 5$, V_{LB} is also the optimal cooling setting. Next, we generate Case 3 based on the previous cases to generate settings that may better reflect practice. Case 3 can be seen as a combination of Case 1 and Case 2 with smoother behaviour for A and B. For A, the entries are randomly chosen to be 0, 1, 2 or 3 with the constraint that the summation of the values in each row is 3. For B, we assume each server is affected mostly by itself, which is a reasonable assumption in practice. We also assume that four other servers have higher effects on a specific server. So, we consider three uniform distributions. For each row, the entry in the main diagonal is chosen from the interval [2,5], there are four other entries randomly chosen from [1,2], and the other entries are selected from [0,0.5].

Table 2.2: Performance of the algorithms over 100 runs for the synthetic linear models and over a selected range of demand D

Case 1												
	SR			GA			H1			H2		
D	avg	wrc	pop	avg	wrc	pop	avg	wrc	pop	avg	wrc	pop
4	560	5688	0.20	36	375	0.90	51	792	0.87	1	1	1
5	901	5218	0.02	334	546	0.13	408	1480	0.17	108	549	0.71
6	4.60	27.89	0	1.68	3.30	0.05	2.09	4.44	0.10	1.25	2.33	0.20
7	5.48	34.91	0	1.95	3.73	0.05	2.07	5.09	0.05	1.41	3.05	0.17
8	12.84	248	0	1.97	4.10	0	1.78	4.12	0.03	1.27	2.38	0.29
Case 2												
4	514	2015	0	64	553	0.84	1	1	1	1	1	1
5	509	1779	0	496	1779	0.02	472	1779	0.06	184	1121	0.62
9	2.16	3	0.02	1.55	2	0.45	1 .01	2	0.99	1	1	1
10	2.25	3	0	1.99	2	0.01	1.79	2	0.21	1.36	2	0.64
11	1.21	2	0.60	1	1	1	1	1	1	1	1	1
Case 3												
1	1.61	3.56	0.17	1.17	2.04	0.47	1	1	1	1	1	1
2	1.79	3.66	0.04	1.26	2.37	0.24	1.31	2.75	0.31	1.12	2.04	0.47
3	1.60	3.03	0.02	1.24	2.06	0.13	1.35	2.79	0.12	1.11	1.64	0.36
4	1.34	2.47	0.11	1.15	1.60	0.18	1.21	1.70	0.14	1.10	1.43	0.27
5	1.28	2.50	0.06	1.12	1.46	0.18	1.16	1.49	0.15	1.09	1.45	0.25

Table 2.2 and Table 2.3 report the performance and running times of four algorithms for these different cases. The algorithms are simple rounding (SR), genetic algorithm (GA), proposed scheme based on solving problem (2.4.6) (H1) and proposed scheme based on solving problem (2.4.7) (H2). The results are the average over 100 runs of the algorithms for randomly generated systems. For GA, the population size and the number of iterations are set to $5 \times min(D, n-D)$ and $10 \times min(D, n-D)$, respectively. Theses choices are based on trial and error while trying to balance the performance and the running time of GA. For H1 and H2, the number of perturbations, as explained in the last part of Section 2.4, is $min(5, D, n-D)$. In Table 2.2, there are three metrics, averaging the ratio of the calculated cost by the algorithm to the optimal cost, avg, the worst case ratio of the calculated cost to the optimal cost, wrc, and the proportion of times generating the optimal cost, pop, over a total of 100 runs. Due to lack of space we only report the results for five values of D to highlight the performance differences between the algorithms. H2 has the best performance

Table 2.3: Average running time (in seconds) of the algorithms over 100 runs for the synthetic linear models and over a selected range of demand D

Case 1					
D	SR	GA	H1	H2	OPT
10	0.018	1.221	0.115	0.149	0.165
11	0.018	1.420	0.119	0.156	0.168
12	0.020	1.643	0.128	0.159	0.160
13	0.020	1.676	0.128	0.163	0.136
14	0.019	1.538	0.122	0.153	0.101
Case 2					
11	0.016	1.553	0.132	0.212	2.555
12	0.020	1.691	0.127	0.203	1.119
13	0.020	1.707	0.130	0.203	0.552
16	0.021	1.278	0.130	0.194	1.842
17	0.021	1.126	0.129	0.195	0.464
Case 3					
10	0.017	1.160	0.108	0.133	0.131
11	0.018	1.348	0.112	0.139	0.146
12	0.018	1.513	0.114	0.144	0.152
13	0.018	1.469	0.115	0.147	0.156
14	0.018	1.337	0.111	0.144	0.142
Case 3 with 50 servers					
14	0.022	1.933	0.195	0.464	6.000
15	0.024	2.165	0.209	0.508	8.183
16	0.024	2.319	0.213	0.506	10.363
17	0.022	2.331	0.206	0.486	11.744
18	0.023	2.534	0.213	0.508	15.560

for the entire range of D. The performance for the middle range of D ($5 \leq D \leq 15$) is more important because the search space for the solution is larger (and hence the decisions for how to reduce power consumption are more challenging).

Table 2.3 reports the running times. Table 2.3 has an extra column, OPT, which reports the running time of solving the problem exactly (using the *intlinprog* function). According to Tables 2.2 and 2.3, H2 has the best performance and a reasonable running time. For Case 1, as expected from Lemma 1, when $D \leq 5$ the approximation factor is large for SR. This is also true for Case 2, as expected from Lemma 2. In these two cases, the other algorithms work better and the best results are clearly for H2 throughout. For Case 3, H2 still has the best performance but with less of an advantage over the other algorithms. The results in Table 2.3 show the best running

times are for SR and the worst are for GA. H1 and H2 have reasonable running times. The running times for H2 are more than for H1 and H1's running times scale more smoothly. The reason is H2 performs some calculations for determining the dominant cooling variables, along with the differences in the cost functions of (2.4.6) and (2.4.7). Finally, from the results for Case 3 with 50 servers we can see that the MATLAB function cannot solve the problem efficiently.

The other set of inputs comes from the linear regression of a model for the data center described in Section 2.2. We also set $T_{idle} = 35, T_{busy} = 27, V_{LB} = (1300 \quad 10), V_{UB} = (2300 \quad 20)$. We performed linear regressions on the functions M and F (using the *regress* function). F is a closed-form function but evaluating M requires the solution of a set of differential equations [22][23]. The data points are chosen uniformly at random from the defined ranges for the cooling variables and server untilizations (utilizations are used for regression of M and they are continuous in this case). We checked the value of R^2 statistics for the both functions M and F and it was at least equal to 0.95 in all cases. The results for regression of two components of F, the fan and the chiller power consumption, are shown in Fig. 2.2. After performing the modifications explained in Section 2.3, the standard form of problem (2.3.1) is generated. For evaluation of scalability, we consider three systems with 25, 50 and 75 servers. For 25 servers we use the regression results (omitting the constant biases for F) but for 50 and 75 servers we change the cooling and heat recirculation matrices as follows. To extend the model to more cascading units, we first change the cooling matrix for the case of 25 servers. Instead of one cooling variable corresponding to the total air flow of the fans, we assume that each fan has a separate air flow that affects closer servers more. For example, the leftmost servers are affected 90 percent

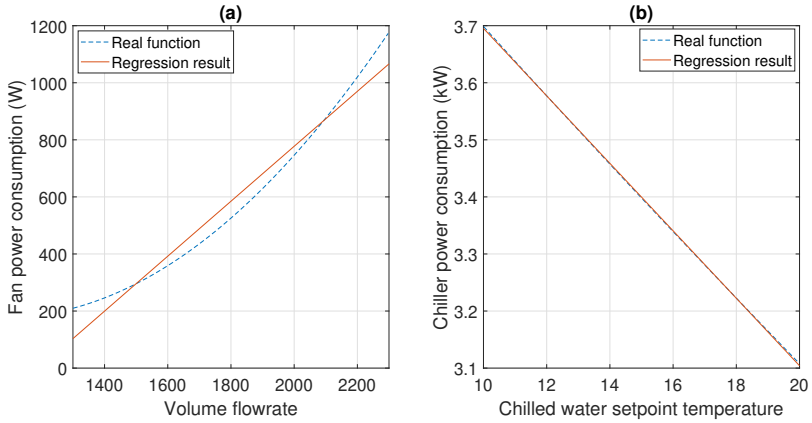

Figure 2.2: a) The actual and linearized fan power consumption, b) The actual and linearized chiller power consumption

by the left fan and 10 percent by the right fan. By considering these changes, we obtain a new cooling matrix with size 25×3 (two fans and one chiller). For the case of 50 and 75 servers both the cooling and heat recirculation matrices are extended. The details of how these changes are made are provided in the next paragraph.

For scaling from 25 servers to 50 servers, we consider two sets of five racks (each consisting of 25 servers) adjacent to each other. Instead of two fans, there are three fans, one between the two sets of racks, with two more at either end. We assume the airflow of the middle fan is spread equally between the two sets of racks and each set is affected by the fans at each of its ends. This way we can construct the cooling matrix (we did not change the coefficients for the chiller). For the heat recirculation matrix, the effects of servers in the two sets of racks on each other are considered as follows. The leftmost rack in the left set of racks is affected the least by the right set of racks. We assume the effects are 15 percent of the effects of the servers in the same set of racks. For the next rack (the second rack from the left) the effects are similar but larger due to the shorter distance. So, we assume the effects are twice the effects

Table 2.4: Performance of the algorithms over 100 runs for the linear data center models and over a selected range of demand D

Data center with 25 servers										
	SR		GA		H1		H2		LRH	
D	avg	wrc	avg	wrc	avg	wrc	avg	wrc	avg	wrc
2	1.22	1.50	1	1.03	1.02	1.45	1.01	1.31	1	1.03
4	1.28	1.45	1.01	1.33	1.01	1.32	1.01	1.33	1.02	1.05
6	1.33	1.53	1.08	1.35	1.03	1.31	1.02	1.38	1.03	1.07
8	1.35	1.50	1.14	1.37	1.08	1.40	1.08	1.43	1.34	1.55
10	1.32	1.42	1.24	1.37	1.14	1.39	1.13	1.39	1.35	1.48
12	1.03	1.14	1.02	1.09	1.01	1.03	1.01	1.09	1.06	1.17
14	1.02	1.09	1.01	1.04	1	1.03	1.01	1.05	1.04	1.13
16	1.02	1.06	1.01	1.04	1	1.02	1.01	1.04	1.11	1.22
18	1.02	1.07	1.01	1.03	1	1.03	1	1.04	1.09	1.17
20	1.01	1.11	1.01	1.04	1	1.02	1	1.03	1.06	1.15
22	1.03	1.08	1.01	1.04	1	1.03	1	1.04	1.04	1.13
24	1.01	1.05	1.01	1.05	1	1	1	1	1.03	1.14
Data center with 50 servers										
15	1	1.02	1	1.02	1	1.19	1	1.01	1.02	1.06
20	1.01	1.02	1.01	1.02	1.01	1.04	1	1.02	1.03	1.06
25	1.01	1.04	1.01	1.02	1	1.02	1	1.02	1.02	1.06
30	1	1.02	1	1.01	1	1.03	1	1.01	1.20	1.24
Data center with 75 servers										
40	1.05	1.09	1.05	1.07	1.04	1.07	1.03	1.05	1.10	1.13
50	1.02	1.05	1.02	1.04	1.01	1.03	1.01	1.03	1.10	1.12

calculated for the leftmost rack. For the next racks, the effects are three, four and five times the effects calculated for the leftmost rack, respectively. Finally, for the case of 75 servers, we assume the racks that are not adjacent are isolated from each other in terms of heat recirculation.

Table 2.4 reports the results. There are additional results for the LRH algorithm [6]. LRH tries to minimize the heat recirculation by choosing the servers with lowest values for summation of columns in the heat recirculation matrix B. The problem is solved 100 times. Each time the matrices A, B and E are perturbed by multiplying each of their entries with a random value drawn from a uniform distribution on the interval [0.98, 1.02]. The results show that H1 and especially H2 have the best performance in terms of avg. In the case of 25 servers and smaller D values, GA and especially SR clearly perform worse than the other algorithms. For some D values,

LRH also reports the worst performance for all three systems. Similarly to the results in Table 2.3, we also checked the running times in this case and scalibility was also confirmed in terms of running times.

2.5.2 Evaluation of Energy Savings for the Original System

We now evaluate the effectiveness of our proposed approach versus attempting to compute the optimal solution directly for the original system. The system considered corresponds to the operational data center of 25 servers and two cooling variables as described in the previous sections. We solve problem (P0) (using the *surrogateopt* function) with the actual nonlinear functions M and F and then compare the results with the results obtained by using the solution of the linearized system. We also implement three algorithms from the literature, UniD, minTemp, and LRH. UniD and minTemp are similar to the UniformWorkload and CoolestInlets algorithms presented in [27]. In UniD the workload is distributed uniformly among the servers. In minTemp, the next server to be chosen is the server with minimum inlet temperature (we set the cooling variables to their lower bounds). However, knowing the workload distribution is not enough to compute the cooling power consumption. We also need to calculate the cooling parameters. This step is performed by solving the linearized problem for the specific workloads, which is a standard linear programming problem (this is also the case for LRH). In Table 2.5, there are four extra columns, OPTL, OPT27, OPTi100, and OPTi1000. OPTL, OPTi100, and OPTi1000 correspond to solve the linear problem and solve problem (P0) with 100 and 1000 number of iterations, respectively (using the *intlinprog* and *surrogateopt* functions). We explain OPT27 below. All the solutions are evaluated for the original system. The results

reported in Table 2.5 are the ratio of the calculated cost to the best cost value found among all the columns for each value of D. According to Fig. 2.2, the maximum possible improvement for this system is $\frac{3700+1200}{3100+200} = 1.48$, which means an algorithm can reduce the power consumption by at most 48% when compared to another algorithm. We also checked the feasibility of the solution for the original system. For OPTL and the proposed algorithms, the initial solutions were not feasible for some values of D. To solve this issue, we reduced T_{idle} and T_{busy} for the linearized problem to 33.5 and 25.5, respectively.

As reported in columns OPTi100 and OPTi1000, the MATLAB function cannot return the optimal solution in most of the cases. At some cases, the difference with the best solution is about 14% even for OPTi1000. It is also very time consuming due to evaluation of the function M at each iteration (each call takes about 2 seconds) as explained in Section 2.2.

The results show that the linear regression and the corresponding approximation schemes, H1 and H2, have close to the best performance for the whole range of D, emphasizing also their time efficiency. The performance of UniD is also not acceptable for the whole range, particularly for the smaller values of D. While LHR reports the best solution for some values of D, its performance is not acceptable for some other values. Finally, minTemp reports the best solution for the whole range of D but there were some challenges. First, as explained, evaluation of inlet temperatures is time consuming. Second, for minTemp and LHR the initial solution was not feasible for some values of D. To solve this issue, we gradually increased the fan's flow rate by 5% until a feasible solution was reached. This step is also time consuming and

Table 2.5: Performance of different algorithms over the whole range of demand D, with results applied to the original 25-server system

Original 25-server system											
D	SR	GA	H1	H2	OPTL	OPT27	OPTi100	OPTi1000	UniD	minTemp	LRH
1	1.02	1.02	1.02	1.02	1.02	1.15	1.13	1.05	1.11	1	1
2	1.02	1.02	1.02	1.02	1.02	1.15	1.13	1.06	1.11	1	1
3	1.02	1.02	1.02	1.02	1.02	1.15	1.12	1.06	1.11	1	1
4	1.08	1.02	1.02	1.02	1.02	1.15	1.12	1.10	1.12	1	1
5	1.02	1.02	1.02	1.02	1.02	1.14	1.17	1.10	1.12	1	1
6	1.02	1.02	1.03	1.03	1.02	1.14	1.18	1.12	1.13	1	1
7	1.02	1.02	1.03	1.03	1.02	1.14	1.18	1.11	1.13	1	1
8	1.02	1.02	1.03	1.03	1.02	1.14	1.18	1.12	1.14	1	1.09
9	1.03	1.03	1.03	1.03	1.03	1.14	1.17	1.11	1.14	1	1.09
10	1.03	1.03	1.03	1.03	1.03	1.14	1.16	1.14	1.14	1	1.09
11	1	1	1	1	1	1.04	1.06	1.06	1.05	1	1
12	1.01	1	1.01	1.01	1	1.07	1.06	1.06	1.05	1	1
13	1.01	1	1	1	1	1.03	1.06	1.05	1.06	1	1
14	1.02	1.01	1	1	1	1.03	1.05	1.05	1.06	1	1
15	1.01	1	1	1	1	1.03	1.05	1.05	1.06	1	1
16	1.01	1	1	1	1	1.02	1.05	1.04	1.06	1.01	1.05
17	1.01	1	1	1	1	1.01	1.04	1.04	1.06	1.01	1.04
18	1.03	1.03	1.02	1.02	1.02	1.03	1.06	1.05	1.09	1	1.06
19	1	1	1	1	1	1	1.02	1.02	1.07	1	1.03
20	1	1	1.01	1.01	1	1	1.03	1.02	1.07	1	1.09
21	1.04	1.02	1.02	1.02	1.02	1	1.02	1.01	1.07	1.02	1.09
22	1.04	1.03	1.03	1.03	1.02	1	1.01	1.01	1.08	1.02	1.10
23	1.04	1.03	1.03	1.03	1.02	1	1.02	1.01	1.08	1.02	1.10
24	1.04	1.03	1.03	1.03	1.03	1	1.02	1.01	1.09	1.01	1.01

not straightforward. In addition, as we explained, the linearized system was used to compute the initial cooling parameters after determining the workload distribution by these algorithms. So, they are also dependent on an efficient method to compute the cooling parameters.

Finally, to investigate the effect of considering two different upper bounds for the temperatures of idle and busy servers, we report the results for the case of continuous utilizations but with one upper bound for the server temperatures equal to T_{busy} (we have to choose the smallest upper bound, T_{busy}, because it should work for all utilizations). In Table 2.5, column OPT27 corresponds to this case by setting $T_{busy} = 27$ and solving the problem (using the *fmincon* function that is also slow due to the need to repeatedly evaluate the function M). As shown, the case of continuous utilizations

does not yield acceptable performance for small to medium values of D. The results also show that we can take advantage of the two different red-line temperatures to reduce the total power consumption compared to the solution calculated for the case of a single red-line temperature (by up to 14 percent for the middle range of D).

2.6 Conclusion

In this paper, a time-efficient holistic approach for solving thermal-aware workload distribution problems was proposed, including linearization of the problem and using the solution for the original system. We introduced a problem considering several cooling parameters, heat recirculation effects and two red-line temperatures for servers with a goal of minimizing the cooling power consumption for a data center. We proposed a framework including some assumptions, linearization of the problem and a heuristic for intelligent rounding of the fractional solution for the linearized model. We provided an analysis of the optimization problem for the linearized problem, by introducing simple rounding and genetic algorithms as the baseline algorithms and extracting two bad cases for simple rounding that are also used for evaluation of the proposed schemes. The proposed framework showed acceptable performance and running times for the cases of synthetic and real world systems. In some cases, the power consumption for the real system is reduced by 10 percent when compared with the solutions returned by the MATLAB function, which requires a much longer running time (due to the complexity of the thermal model). The results also confirm that considering two red-line temperatures is beneficial to reduce the cooling power consumption by up to 14 percent (which also has much greater running time as compared to the proposed heuristics). This was demonstrated by comparison with the case of

one red-line temperature and continuous utilizations. As for future work, modification of the proposed algorithms for similar problems such as set covering problems and facility location problems can be studied. The applicability of linearization for the case of server consolidation can also be investigated. This work may also be a good starting point for considering deeper analysis of the underlying optimization problems. For example, the cases of heterogeneous data centers and transient thermal models are more complicated problems that can be investigated. In contrast to our real system, the linearized model may not represent a real system well enough. Other representations such as piece-wise linear functions or general convex functions can be explored. Additional costs such as migration cost (moving workload between servers) can also be considered, when one considers a problem where the workload varies over time.

Bibliography

[1] W. Zhang, Y. Wen, Y. Wah Wong, K. Chuan Toh, and C. H. Chen, "Towards Joint Optimization Over ICT and Cooling Systems in Data Centre: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 3, pp. 1596-1616, 2016.

[2] A. Habibi Khalaj, S. K. Halgamuge, "A Review on Efficient Thermal Management of Air and Liquid-cooled Data Centers: From Chip to the Cooling System," *Applied Energy*, vol. 205, pp. 1165-1188, 2017.

[3] Y. Ran, H. Hu, X. Zhou, and Y. Wen, "DeepEE: Joint Optimization of Job Scheduling and Cooling Control for Data Center Energy Efficiency Using Deep Reinforcement Learning," *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 645-655, 2019.

[4] Y. Li, Y. Wen, D. Tao, and K. Guan, "Transforming Cooling Optimization for Green Data Center via Deep Reinforcement Learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 5, pp. 2002-2013, 2020.

[5] S. Akbar, S. U. R. Malik, K. -K. R. Choo, S. U. Khan, N. Ahmad and A. Anjum, "A Game-based Thermal-Aware Resource Allocation Strategy for Data Centers,"

in *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 845-853, July-Sept. 2021.

[6] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta. 2012. "TACOMA: Server and Workload Management in Internet Data Centers Considering Cooling-computing Power Trade-off and Energy Proportionality," *ACM Transactions on Architecture and Code Optimization*, vol. 9, no. 2, pp. 1-37, 2012.

[7] S. MirhoseiniNejad, G. Badawy, and D. G. Down, "EAWA: Energy-aware Workload Assignment in Data Centers," *2018 International Conference on High Performance Computing & Simulation (HPCS)*, pp. 260–267, 2018.

[8] M. Nakamura, "Learning and Optimization Models for Energy Efficient Cooling Control in Data Center," *Proceedings of the SICE Annual Conference*, 2016, Tsukuba, Japan.

[9] T. Van Damme, C. De Persis, and P. Tesi, "Optimized Thermal-Aware Job Scheduling and Control of Data Centers," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 760-771, 2019.

[10] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Data Centers: A Cyber-Physical Approach,"*IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1458-1472, 2008.

[11] L. Yang, Y. Deng, L. T. Yang. and R. Lin, "Reducing the Cooling Power of Data Centers by Intelligently Assigning Tasks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1667-1678, 2018.

[12] A. D. Carnerero, D. R. Ramirez, T. Alamo and D. Limon, "Probabilistically Certified Management of Data Centers Using Predictive Control," in *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2849-2861, Oct. 2022.

[13] Z. Wang, C. Bash, N. Tolia, N. Marwah, X. Zhu, and P. Ranganathan, "Optimal Fan Speed Control for Thermal Management of Servers," *ASME 2009 InterPACK Conference*, vol. 2, pp. 709-719, 2009, San Francisco, California, USA.

[14] J. Wan, X. Gui, R. Zhang, and L. Fu, "Joint Cooling and Server Control in Data Centers: A Cross-Layer Framework for Holistic Energy Minimization," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2461-2472, 2018.

[15] E. Pakbaznia, M. Pedram, "Minimizing Data Center Cooling and Server Power Costs," *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*, pp. 145–50, 2009.

[16] Q. Fang, J. Wang, Q. Gong, and M. Song, "Thermal-Aware Energy Management of an HPC Data Center via Two-Time-Scale Control," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2260-2269, 2017.

[17] Q. Fang, J. Wang and Q. Gong, "QoS-Driven Power Management of Data Centres via Model Predictive Control," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1557-1566, 2016.

[18] L. Parolini, B. Sinopoli, B. H. Krogh, and Z. Wang, "A Cyber–Physical Systems Approach to Data Center Modeling and Control for Energy Efficiency," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 254-268, 2012.

[19] J. Li, Y. Deng, Y. Zhou, Z. Zhang, G. Min and X. Qin, "Towards Thermal-Aware Workload Distribution in Cloud Data Centers Based on Failure Models," in *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 586-599, Feb. 2023.

[20] A. M. Al-Qawasmeh, S. Pasricha, A. A. Maciejewski, and H. J. Siegel, "Power and Thermal-Aware Workload Allocation in Heterogeneous Data Centers," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 477-491, 2015.

[21] S. Mirhoseininejad, G. Badawy, and D. G. Down, "A Data-driven, Multi-setpoint Model Predictive Thermal Control System for Data Centers," *Journal of Networks and Systems Management*, vol. 29, no. 7, 2021.

[22] R. Gupta, S. Asgari, H. Moazamigoodarzi, D. G. Down, and I. K. Puri, "Energy, Exergy and Computing Efficiency Based Data Center Workload and Cooling Management," *Applied Energy*, vol. 299, 117050, 2021.

[23] H. Moazamigoodarzi, R. Gupta, S. Pal, P. J. Tsai, S. Ghosh, and I. K. Puri, "Modeling temperature distribution and power consumption in IT server enclosures with row-based cooling architectures," *Applied Energy*, Vo. 261, 114355 2020.

[24] S. MirhoseiniNejad, H. Moazamigoodarzi, G. Badawy, and D. G. Down, "Joint Data Center Cooling and Workload Management: A Thermal-aware Approach," *Future Generation Computer Systems*, Vol. 104, 2020.

[25] P. C. Chu, and J. E. Beasley, "A Genetic Algorithm for the Multidimensional Knapsack Problem," *Journal of Heuristics*, vol. 4, pp. 63–86, 1998.

[26] J.E. Beasley, and P. C. Chu, "A Genetic Algorithm for the Set Covering Problem," *European Journal of Operational Research*, vol. 94, no. 2, pp. 392-404, 1996.

[27] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling "cool": Temperature-aware workload placement in data centers," in *Proc. USENIX Annual Technology Conference*, 2005, pp. 1-75.

Chapter 3

Thermal-aware Workload Distribution for Data Centers with Demand Variations

The content of this chapter is reproduced from: "Thermal-aware workload distribution for data centers with demand variations", S. Rostami, D. G. Down, and G. Karakostas. An abridged version has been submitted to the 14th International Green and Sustainable Computing Conference - IGSC 2023, Toronto, ON, Canada, October 2023. The author of this thesis is the first author and the main contributor of this manuscript.

Abstract

Thermal-aware workload distribution is a common approach in the literature for power consumption optimization in data centers. However, data centers also have other operational costs such as the cost of equipment maintenance and replacement. It has been shown that server reliability depends on frequency of their temperature variations, arising from workload transitions due to dynamic demands. In this work, we formulate a nonlinear optimization problem that considers the cost of workload transitions in addition to IT and cooling power consumption. To approximate the solution, we first linearize the problem; the result is a mixed integer programming problem. A modified heuristic is then proposed to approximate the solution of the linear problem. Finally, a Model Predictive Control (MPC) approach is integrated with the proposed heuristics for automatic workload reconfiguration when future demand is not known exactly, but predictions are available. Numerical results show that the proposed schemes are attractive in different settings.

Keywords: switching cost, model predictive control, integer programming, thermal-aware workload distribution, data center

3.1 Introduction

Energy consumption of data centers is increasing rapidly, due to growth in demand for internet services and cloud computing tasks. IT and cooling equipment are the main power consumers in a data center [1][2]. Due to heat recirculation effects, thermal-aware workload distribution is necessary to minimize the power consumption while

respecting operational temperature constraints [3].

Data centers also have operational costs such as the cost of equipment maintenance and replacement. The reliability of servers depends on several factors such as their inlet temperature and frequency of temperature variations. In thermal-aware workload distribution, the inlet temperatures are typically bounded by a red-line temperature [4]. However, when the workload distribution is changed due to dynamic demands, the cost of varying the workload on a server (which we will call switching costs) has not been addressed in the literature. The varying workload leads to temperature variations that can impact the reliability of servers [5]. To be more precise, most of the approaches solve the thermal-aware workload distribution problem for a fixed demand. Considering time varying demand is beneficial for reducing the switching costs [6]. In this case, it would be desirable to incorporate demand predictions into the problem. So, we are interested in a thermal-aware workload distribution problem where demand is time varying and the effects of varying the utilizations of servers are taken into account when (re)distributing workload.

While dynamic workload allocation may have costs associated with decreased server lifetimes, workload migration may also be required. This mainly affects the quality of service by imposing a delay on processing when virtual machines migrate between different physical machines [7][8]. In this work, we do not address the virtual machine migration problem directly, however, the problem we plan to solve can also consider the migration cost indirectly by adding a penalty for workload transitions. This penalty can lead to decrease the number of migrations.

There are a few works that consider switching costs in the workload distribution

policy. Most of the literature addresses thermal-aware workload distribution for a constant demand (steady- state) [9][10][11][12][13][14]. In [6], switching costs are considered but cooling power consumption is not considered. In [15], switching costs are considered in the thermal-aware workload distribution policy, where a transient thermal model is used. A particle-based optimization algorithm is then used to solve the problem. In this work, we formulate a thermal-aware workload distribution problem in discrete time that considers switching costs in addition to IT and cooling power consumption. The proposed problem is a generalized form of the problem introduced in [16][17].

The problem proposed in [17] is a general power optimization problem with nonlinear cooling power consumption and steady-state thermal model. As a power reduction scenario, they also consider two different red-line temperatures corresponding to idle and fully-utilized servers, respectively. However, the demand value is fixed. The approach proposed to solve nonlinear power optimization problems is to linearize the problem. Depending on the type of variables (continuous or integral) the resulting problem is a linear programming problem or an (mixed) integer linear programming problem. This approach is especially beneficial because the nonlinear problem is general enough to represent a range of similar problems in this area. The approach is also time efficient because the thermal models may be computationally expensive when they are used to calculate the temperatures, for example if they are based on solving differential equations. Using linear regression to construct a linear model helps reduce the time complexity. Finally, a heuristic is proposed to approximate the solution of the linear problem that is then applied to the original problem.

In this work, we also linearize the problem and generalize the heuristic to approximate the solution of the resulting mixed integer programming problem. The solution is then used for the original problem. When demand predictions are available, we integrate a Model Predictive Control (MPC) approach with the proposed heuristic. Using MPC is common in the literature in the presence of transient thermal models [15][16][18][19][20]. In this work, we show that an MPC approach is useful for reducing the size of the problem and for incorporating updates to the predicted demand. Our contributions can be listed as follows:

- Generalization of the constant demand problem to a discrete-time, time-varying problem which also considers switching costs
- Generalization of the heuristic proposed in [17] for the resulting mixed integer linear programming problem and proving its applicability for the proposed problem
- Integration of an MPC approach with demand predictions for the proposed heuristic
- Evaluation of the proposed schemes that suggest the potential for significant cost reductions, e.g. when compared to separating the problem into independent instances at each time step

In the remainder of the paper, we first describe the system model and introduce the optimization problem, in Section 3.2. We also linearize the problem. An approximation algorithm to solve the problem is proposed in Section 3.3. An MPC

approach is introduced in Section 3.4. Section 3.5 covers the evaluation of the proposed schemes for the introduced integer linear programming problem. Concluding remarks are provided in Section 3.6.

3.2 System Model

The problem introduced in this paper is a thermal-aware workload distribution problem that considers both power consumption and switching costs in the presence of demand variations. We consider a discrete time demand model in which there are K time slots and the demand at time slot k is denoted by D_k, the number of required servers at time slot k. The problem is a generalized form of the problem introduced in [16][17]. For the case of one time slot or fixed demand, presented in [17], a general nonlinear optimization problem is considered for minimizing the total power consumption in a data center. The system considered in this paper includes m cooling facilities and n servers. The decision variables are the cooling parameters and the server utilizations at time slot $k, k = 1, ..., K$, denoted by the vectors $v^{(k)}_{m\times 1}$ and $\rho^{(k)}_{n\times 1}$, respectively. As a power reduction scenario, two red-line temperatures are considered corresponding to idle or fully utilized servers, so the server utilizations are 0 or 1. This helps reduce the cooling effort because the lightly loaded servers have a higher red-line temperature. The servers are assumed to be identical. The power consumption and thermal models are generally nonlinear. The cost function is the summation of cooling and IT power consumption along with the cost of workload migration and switching the servers between idle or fully utilized (or on and off states in the case of server consolidation) in consecutive time slots. Addition of the switching cost controls the utilization variation of the servers (and the temperature

variations) which as discussed in Section 3.1, can lead to increased server lifetimes. So, the trade-off of power consumption and the impact on server reliability due to the frequency of varying server utilizations is considered. There are performance and temperature constraints for each time slot.

We assume that the initial workload distribution is denoted by $\rho^{(0)}$. Thus, the problem that we wish to solve is:

$$\begin{aligned}
\min \quad & \sum_{k=1}^{K} F(v^{(k)}) + \sum_{k=1}^{K} w_k \sum_{i=1}^{n} |\rho_i^{(k)} - \rho_i^{(k-1)}| + \sum_{k=1}^{K} \sum_{i=1}^{n} P(\rho_i^{(k)}, t_i^{(k)}) \\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i^{(k)} \geq D_k && \forall k = 1, ..., K \\
& M(v^{(k)}, \rho^{(k)}) \leq T_{idle} 1_{n\times 1} - (T_{idle} - T_{busy})\rho^{(k)} && \forall k = 1, ..., K \\
& v^{(k)} \geq V_{LB} && \forall k = 1, ..., K \\
& v^{(k)} \leq V_{UB} && \forall k = 1, ..., K \\
& \rho_i^{(k)} \in \{0, 1\} && \forall k = 1, ..., K \quad \forall i = 1, ..., n
\end{aligned} \tag{3.2.1}$$

where $F(v^{(k)})$ is the cooling power consumption corresponding to the cooling variable vector $v_{m\times 1}^{(k)}$ at time slot k, $\rho_{n\times 1}^{(k)}$ is the vector of workload distribution at time slot k, and $M(v^{(k)}, \rho^{(k)})$ is the function corresponding to the thermal model. Within each time slot, a steady-state thermal model is considered. In other words, we assume the time slots are long enough (in the range of minutes) so that a steady-state thermal model is appropriate. The first constraint is a performance constraint with the target demand D_k, and the second constraint limits the inlet temperatures to be less than the corresponding red-line temperatures, T_{idle} and T_{busy} (according to [4], $T_{idle} > T_{busy}$).

The cost of switching (and migration) per server for the kth time slot is denoted by w_k. The computing (IT) power consumption of server i in the kth time slot is denoted by $P(\rho_i^{(k)}, t_i^{(k)})$, where $t_{n\times 1}^{(k)} = M(v^{(k)}, \rho^{(k)})$ is the vector of server inlet temperatures at time slot k. The vectors of lower bounds and upper bounds for the cooling variables are V_{LB} and V_{UB}, respectively.

There are many possible models that could be used for IT power consumption, but we focus on one choice. We ignore the dependence of IT power consumption on the inlet server temperature. The model is $P(\rho_i^{(k)}) = c + d\rho_i^{(k)}$, where c and d are constants, but we assume that there is server consolidation, so that idle servers are turned off and $P(\rho_i^{(k)}) = 0$ when $\rho_i^{(k)} = 0$. In general, server consolidation may change the thermal model but we leave that as a topic for future work. Server consolidation requires an extra step of linearizing the IT power consumption.

The approach proposed in [17] to solve nonlinear power optimization problems is to linearize the problem and in NP-complete cases (when there are integral variables) propose heuristics developed for approximating the solution of the resulting integer programming problems. The linearized version of the single time slot problem extracted in [17] (with some modifications, assumptions and normalization) is problem (3.2.2), where $a = T_{idle} - T_{busy} > 0$, $b = T_{idle}$, $A_{n\times m}$, $B_{n\times n}$ and $E_{n\times 1}$ are the cooling matrix, the heat-recirculation matrix and the constant part, respectively and $A_{i,j}, B_{i,j} \geq 0$ (nonnegative entries). In [17], we showed that problem (3.2.2) is NP-complete and proposed a heuristic to construct an approximate solution.

Similarly, we first linearize problem (3.2.1) and then generalize the heuristic proposed in [17] to approximate the solution of the linear problem. Linearizing the

switching cost is straightforward and leads to introducing the new variables $s_{k,i}$. When the server utilizations are 0 or 1, linearizing the IT power consumption is also straightforward. In this case $P(\rho_i^{(k)}) = (c+d)\rho_i^{(k)}$. So, the integer linear programming problem is problem (3.2.3).

$$\begin{aligned}
\min \quad & \sum_{j=1}^{m} v_j \\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i \geq D \\
-\sum_{j=1}^{m} A_{l,j} v_j + & \sum_{i=1}^{n} B_{l,i}\rho_i + a\rho_l \leq b - E_l \quad \forall l = 1, ..., n \\
& v_j \geq V_{LB}^{(j)} \quad \forall j = 1, ..., m \\
& v_j \leq V_{UB}^{(j)} \quad \forall j = 1, ..., m \\
& \rho_i \in \{0, 1\} \quad \forall i = 1, ..., n
\end{aligned} \tag{3.2.2}$$

$$\begin{aligned}
\min \quad & \sum_{k=1}^{K}\sum_{j=1}^{m} v_j^{(k)} + \sum_{k=1}^{K} w_k \sum_{i=1}^{n} s_{k,i} + (c+d)\sum_{k=1}^{K}\sum_{i=1}^{n} \rho_i^{(k)} \\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i^{(k)} \geq D_k \qquad \forall k = 1, ..., K \\
-\sum_{j=1}^{m} A_{l,j} v_{k,j} + & \sum_{i=1}^{n} B_{l,i}\rho_i^{(k)} + a\rho_l^{(k)} \leq b - E_l \qquad \forall k = 1, ..., K \quad \forall l = 1, ..., n \\
& s_{k,i} - \rho_i^{(k)} + \rho_i^{(k-1)} \geq 0 \qquad \forall k = 1, ..., K \quad \forall i = 1, ..., n \\
& s_{k,i} + \rho_i^{(k)} - \rho_i^{(k-1)} \geq 0 \qquad \forall k = 1, ..., K \quad \forall i = 1, ..., n \\
& v_j^{(k)} \geq V_{LB}^{(j)} \qquad \forall k = 1, ..., K \quad \forall j = 1, ..., m \\
& v_j^{(k)} \leq V_{UB}^{(j)} \qquad \forall k = 1, ..., K \quad \forall j = 1, ..., m \\
& \rho_i^{(k)} \in \{0, 1\} \qquad \forall k = 1, ..., K \quad \forall i = 1, ..., n
\end{aligned} \tag{3.2.3}$$

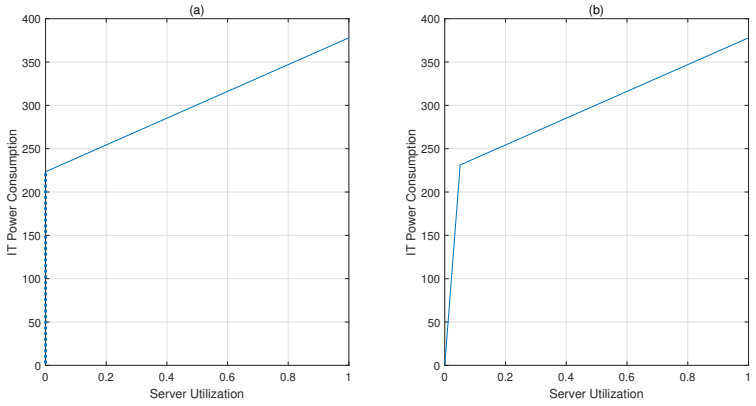

Figure 3.1: a) The actual IT power consumption (Watts) b) The piecewise linear approximation of IT power consumption

However, with the relaxation of server utilizations that is needed for the approximation algorithm, more work is needed to linearize the IT power consumption in problem (3.2.1). According to the IT power consumption model, in the case of consolidation there is a jump in $P(\rho_i^{(k)})$ when $\rho_i^{(k)} = 0$. We approximate $P(\rho_i^{(k)})$ with a piecewise linear function as is shown in Figure 3.1 For a small value ϵ, the IT power consumption is $P_\epsilon = c\epsilon + d$. If $\rho_i^{(k)} \leq \epsilon$, then the IT power consumption is approximated as $P(\rho_i^{(k)}) = \frac{P_\epsilon}{\epsilon}\rho_i^{(k)}$, and if $\rho_i^{(k)} > \epsilon$, then $P(\rho_i^{(k)}) = c\rho_i^{(k)} + d$. To linearize these conditions, we divide $\rho_i^{(k)}$ into two parts, $\rho_i^{(k)} = \rho_i^{-(k)} + \rho_i^{+(k)}$, where one of the following cases is true, depending on the value of the new 0-1 variable $y_i^{(k)}$. If $y_i^{(k)} = 1$, then $\rho_i^{-(k)} \leq \epsilon$ and $\rho_i^{+(k)} = 0$, otherwise $\rho_i^{-(k)} = 0$ and $\rho_i^{+(k)} \geq \epsilon$. This procedure leads to extra constraints being added to problem (3.2.3). Finally the relaxed form of the problem ($0 \leq y_i^{(k)} \leq 1$) is problem (3.2.4).

$$
\begin{aligned}
\min \quad & \sum_{k=1}^{K}\sum_{j=1}^{m} v_j^{(k)} + \sum_{k=1}^{K} w_k \sum_{i=1}^{n} s_{k,i} + \sum_{k=1}^{K}\sum_{i=1}^{n} (\frac{P_\epsilon}{\epsilon}\rho_i^{-(k)} + c\rho_i^{+(k)} + d(1-y_i^{(k)})) \\
\text{s.t.} \quad & \sum_{i=1}^{n} (\rho_i^{+(k)} + \rho_i^{-(k)}) \geq D_k && \forall k = 1,...,K \\
-\sum_{j=1}^{m} A_{l,j} v_{k,j} + \sum_{i=1}^{n} B_{l,i}(\rho_i^{+(k)} + \rho_i^{-(k)}) + a(\rho_l^{+(k)} + \rho_l^{-(k)}) & \leq b - E_l && \forall k = 1,...,K \quad \forall l = 1,...,n \\
s_{k,i} - (\rho_i^{+(k)} + \rho_i^{-(k)}) + (\rho_i^{+(k-1)} + \rho_i^{-(k-1)}) & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
s_{k,i} + (\rho_i^{+(k)} + \rho_i^{-(k)}) - (\rho_i^{+(k-1)} + \rho_i^{-(k-1)}) & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
\rho_i^{+(k)} + y_i^{(k)} & \leq 1 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
\epsilon y_i^{(k)} + \rho_i^{+(k)} & \geq \epsilon && \forall k = 1,...,K \quad \forall i = 1,...,n \\
\epsilon y_i^{(k)} - \rho_i^{-(k)} & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
v_j^{(k)} & \geq V_{LB}^{(j)} && \forall k = 1,...,K \quad \forall j = 1,...,m \\
v_j^{(k)} & \leq V_{UB}^{(j)} && \forall k = 1,...,K \quad \forall j = 1,...,m \\
\rho_i^{+(k)} & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
\rho_i^{-(k)} & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
y_i^{(k)} & \leq 1 && \forall k = 1,...,K \quad \forall i = 1,...,n \\
y_i^{(k)} & \geq 0 && \forall k = 1,...,K \quad \forall i = 1,...,n
\end{aligned}
\tag{3.2.4}
$$

3.3 Approximation Algorithm

Our aim is to approximate the solution of problem (3.2.3) and use it for the original problem (3.2.1). We generalize the H2 heuristic in [17] to approximate the solution of problem (3.2.3). The proposed heuristic is based on gradual rounding of the fractional solution of the relaxed linear problem (3.2.4). Let us denote the solution of problem (3.2.4) by $(v^{*(k)}, \rho^{*(k)}), \forall k = 1,...,K$. In H2, the main idea for approximating the

solution of problem (3.2.2) efficiently is to link the original problem to a problem that can be approximated more efficiently. The efficiency comes from reducing the number of constraints and decision variables. The main variables in problem (3.2.2) are the server utilizations, because by knowing them, the cooling parameters can be found by solving a standard linear programming problem. In the problem that H2 approximates, the decision variables are only the server utilizations and there is only one constraint, on the number of working servers. In the gradual rounding of the server utilizations, at each step, the decision of which server to be turned off (or keep idle) is made by redistribution of the load of each server to the other servers and calculation of the cost for a new optimization problem that aims at minimizing the total increase (as compared to the fractional cost) in the dominant cooling variables (the dominant cooling variable for server i is the variable with the largest entry in the ith row of cooling matrix A). That is because the total increase in the dominant cooling variables is assumed to be a good approximation for the total increase in the cooling variables for the original problem (3.2.2). For our problem, the proposed heuristic, called DCVS (Dominant Cooling Variable with Switching cost), is similarly based on gradual rounding of the fractional server utilizations. However, instead of one problem, K problems are approximated. The values of $\hat{\rho}^{(k)}$ are computed consecutively, as the greatest correlation between demands will typically be between consecutive time slots. The problem for time slot k is:

$$\begin{aligned}
\min \quad & \sum_{r=1}^{R} \max_{l \in S_r} \frac{[\sum_{i=1}^{n} B'_{l,i}\rho_i^{(k)} - (\sum_{j=1}^{m} A_{l,j}v_j^{*(k)} + b - E_l)]^+}{z_l} + w_k \sum_{i=1}^{n} |\rho_i^{(k)} - \hat{\rho}_i{}^{(k-1)}| + w_{k+1} \sum_{i=1}^{n} |\rho_i^{*(k+1)} - \rho_i^{(k)}| \\
\text{s.t.} \quad & \sum_{i=1}^{n} \rho_i^{(k)} = D_k^* \\
& \rho_i^{(k)} \in \{0,1\} \qquad \forall i = 1, ..., n
\end{aligned} \tag{3.3.1}$$

where $B' = B + I_{n\times n}$ (I is the identity matrix) and there are R dominant cooling variables (the variables with the largest corresponding coefficient for at least one row of A). S_r is the set of servers with corresponding dominant cooling variable r, z_l is the corresponding coefficient of the cooling variable r (in the lth row of A) for the server $l \in S_r$ and $D_k^* = \lfloor \sum_{i=1}^{n} \rho_i^{*(k)} \rfloor$ ($\lfloor\rfloor$ is the floor function). The cost function for (3.3.1) is an approximation of the component of the cost function of problem (3.2.3) that is affected by the value of $\hat{\rho}^{(k)}$. There is no IT power consumption term because it is a constant when the number of working servers is fixed (equal to D_k^*). The problem for $k = K$ does not include the last term in the cost function.

Similarly to H2, DCVS is greedy and includes three phases. The first (main) phase is modified to approximate the solution of problem (3.2.1) in terms of server utilizations, as described in Algorithm 2. At each step, the algorithm redistributes the load of each server to other servers (proportional to their current load) and chooses the server to be turned off, based on the cost for problem (3.3.1). The other two phases can be found in [17]. In phase 2, servers that are turned off in the fractional solution are also considered to be turned on in the integral solution. In phase 3, there are small perturbations of A, B and E, which lead to multiple fractional solutions where the best solution among them is chosen.

Algorithm 2 Calculation of $\hat{\rho}^{(k)}, \forall k = 1, ..., K$

Solve problem (3.2.4) and let the solution be $(v^{*(k)}, \rho^{*(k)}), \forall k = 1, ..., K$
$\hat{\rho}^{(0)} = \rho^{(0)}$
for $k = 1 : K$ **do**
    $S = \{i \in \rho^{*(k)} | 0 < \rho_i^{*(k)} \leq 1\}$
    $l = |S| - D_k^*$
    $\hat{\rho} = \rho^{*(k)}$
    while $l \neq 0$ **do**
        for $i \in S$ **do** ▷ distributing the load of server i to the other servers, proportional to their current load
            $\rho^{[i]} = \hat{\rho}$
            $S' = S - \{i\}$
            $r = \rho_i^{[i]}$
            $\rho_i^{[i]} = 0$
            for $j \in S'$ **do**
                $\rho_j^{[i]} = \rho_j^{[i]} + \frac{\rho_j^{[i]}}{\sum_{k \in S'} \rho_k^{[i]}} r$
            end for
            while $\exists s \in S', \rho_s^{[i]} > 1$ **do** ▷ fixing the loads that are greater than 1
                $r = \rho_s^{[i]} - 1$
                $\rho_s^{[i]} = 1$
                $S' = S' - \{s\}$
                for $j \in S'$ **do**
                    $\rho_j^{[i]} = \rho_j^{[i]} + \frac{\rho_j^{[i]}}{\sum_{l \in S'} \rho_l^{[i]}} r$
                end for
            end while
            $x_i =$ value of the cost function of problem (3.3.1) for $\rho^{[i]}$
        end for
        Remove i with the smallest x_i from S and $\hat{\rho} = \rho^{[i]}$
        $l = l - 1$
    end while
    $\hat{\rho}^{(k)} = \hat{\rho}$
end for
return $\hat{\rho}^{(k)}, \forall k = 1, ..., K$

3.4 MPC Approach

An MPC approach is useful in the presence of demand predictions, although it can also be used when demands are known exactly to reduce the size of the problem. We assume that instead of actual demands, we have a noisy version of demands coming from a prediction scheme. In the presence of demand predictions, the weights w_k should be chosen to depend on k as it is reasonable to assume that knowledge about the future demands is more accurate for closer time slots. In general, it is reasonable to assume that $w_k \leq w_{k'}$ when $k \geq k'$.

In problem (3.2.3), it may not be efficient or sufficiently precise to solve the problem for the whole time interval of size K. This is both due to the size of the problem and the fact that distant demand predictions may not be sufficiently accurate. One possibility to address these issues is using an MPC approach. The main idea of MPC is considering a window of size W and using the predictions for the next W time slots to compute the workload distribution in the next time slot. This reduces the greediness of the algorithm by using the information for several time slots. It also allows for updates to predicted demand values to be considered, each time the solution for the next time slot is calculated. We use the MPC scheme which is described in Algorithm 3. Each time, a problem of size W is solved and the solution for the first time slot is kept and used as the initial workload distribution for the next round.

Algorithm 3 Calculation of $\hat{\rho}^{(s)}, \hat{v}^{(s)}$ using MPC approach with window size W

1: update the (predicted) demand values

2: solve problem (3.2.3) for $k = s, ..., s+W-1$ and call the solution $(v'^{(k)}, \rho'^{(k)}), \forall k = s, ..., s+W-1$

3: $\hat{\rho}^{(s)} = \rho'^{(s)}, \hat{v}^{(s)} = v'^{(s)}$

4: **return** $\hat{\rho}^{(s)}$ and $\hat{v}^{(s)}$

3.5 Evaluation

The system we use for evaluation comes from an experimental data center at McMaster University that is modeled in [21]. The data center has 25 servers located in 5 racks and two cooling facilities. The cooling variables are the chilled water temperature and total air flow generated by two fans at either end of the racks. The top view of the data center is shown in Figure 3.2. The functions M and F in problem (3.2.1) are simulated based on the model in [21] for inlet temperatures and cooling power consumption. The platform we used was MATLAB R2021b running on a 64-bit system with an i7-1185G7 processor and 8-GB RAM. The function M is not explicitly given and the inlet temperatures are calculated based on solving a set of differential equations, an operation that is computationally intensive (each call takes around 1.4 seconds). The next step is regression on the functions F and M to linearize the problem. The function *regress* in MATLAB is used and the data points are uniformly at random chosen from the defined ranges for cooling variables and server utilizations (the server utilizations are continuous). We set $T_{idle} = 35$ and $T_{busy} = 27$ (degrees Celsius), $V_{LB} = [1300, 10]$ and $V_{UB} = [2300, 20]$. Additional details are provided

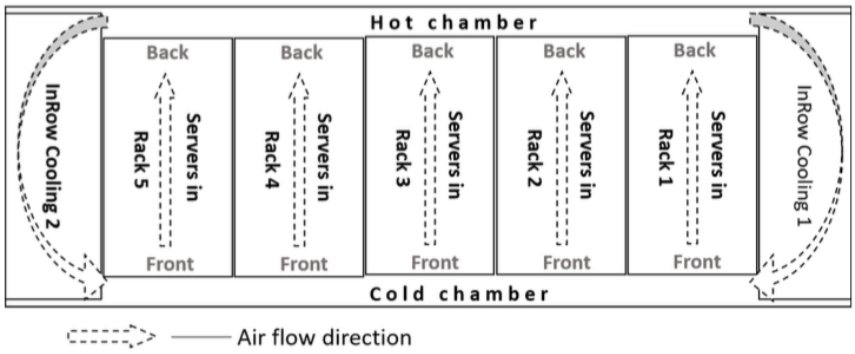

Figure 3.2: The data center's top view according to [21]

in [17]. So, the matrices A, B and E in problem (3.2.3) are known. We perform a (small) random perturbation of the matrices, each time that the algorithms are run. We assume the IT power consumption model is according to server consolidation with coefficients also coming from the model in [21]. In the simulation results presented in [17], we have shown that the solution of the linear system approximated by the proposed heuristic works well for the original nonlinear system. So, here we focus on the evaluation of the heuristics for the linearized system. We also assume $w_k = w, \forall k = 1, ..., K$.

We use simple rounding (SR) as the baseline algorithm. In simple rounding for each time slot k, the D_k^* largest values in $\rho^{*(k)}$ are rounded to one. We also solve the single time slot problem for each of the K time slots (without switching cost) using the *intlinprog* function in MATLAB and calculate the cost of the solution for the multiple time slot problem (with switching cost). This scheme is called Sep in the results. For this data center example, we show that although the performance of SR and H2 are very close for the one time slot problem according to [17], for the multiple time slot problem DCVS clearly works better.

The evaluation includes three parts. In the first part, the sensitivity of the approaches to the value of w is evaluated. In the second part, the performance is evaluated in the presence of demand fluctuations with different patterns. Finally, the performance of the integrated MPC approach is evaluated for the actual and noisy demand values. We present the average and the worst case ratios (avg and wrc columns in the results) when the solution is compared with the solution of the relaxed problem (3.2.4).

The first results correspond to sensitivity to w. As w increases, the switching cost becomes more dominant. Starting from $w = 1$, the value of w doubles. The number of intervals K is equal to 3. The pair of demands (D_1, D_3) covers all possible combinations, where the values for the demand are chosen from $D = \{1, 2, ..., 24\}$. For each combination, D_2 is randomly chosen from D. The results are reported in Table 3.1, with an extra column OPTi corresponding to solving the problem using the *intlinprog* function in MATLAB. Although OPTi has the best performance, in [17] we showed that the running time does not scale well for larger problem sizes. The results show that the performance of DCVS is more resilient to changes in w and for larger values of w, SR has poor performance with respect to the worst case ratio. The reason is that for the fractional optimal solution, the largest D_k^* utilizations in time slot k are not necessarily a subset of the largest D_{k+1}^* in time slot $k + 1$ or vice versa. So, when the switching cost is very large, SR leads to large costs due to the switching cost component. For example, we saw this effect for the input $[7, 9, 9]$ and $w = 1000$, when we used SR for several systems that are perturbed versions of the main system introduced in this section. This effect was seen for roughly one out of 10 of these systems. The utilizations for one of the servers were the same for the first

Table 3.1: Performance of the algorithms for different values of switching cost per server w

	OPTi		Sep		SR		DCVS	
w	avg	wrc	avg	wrc	avg	wrc	avg	wrc
1	1.01	1.02	1.01	1.10	1.02	1.05	1.01	1.05
2	1.01	1.02	1.01	1.06	1.02	1.06	1.01	1.06
4	1.01	1.02	1.01	1.10	1.02	1.04	1.01	1.07
8	1.01	1.02	1.02	1.06	1.02	1.05	1.01	1.07
16	1.01	1.02	1.01	1.08	1.02	1.05	1.01	1.06
32	1.01	1.02	1.01	1.07	1.01	1.05	1.01	1.08
64	1.01	1.02	1.02	1.13	1.01	1.04	1.01	1.05
128	1.01	1.01	1.02	1.13	1.01	1.05	1.01	1.05
512	1.01	1.01	1.04	1.24	1.01	1.04	1.01	1.04
1024	1.00	1.01	1.11	1.55	1.01	1.15	1.00	1.04
2048	1.00	1.01	1.21	1.87	1.01	1.16	1.01	1.02
4096	1.00	1.01	1.36	2.04	1.01	1.09	1.00	1.03
8192	1.00	1.00	1.49	2.63	1.01	1.16	1.00	1.02
16384	1.00	1.00	1.56	2.81	1.00	1.18	1.00	1.01
32768	1.00	1.00	1.63	2.96	1.01	1.23	1.00	1.00

and second time slots but SR rounded the first to 1 and the second to 0.

Another observation from Table 3.1 is that although the average ratio is better for DCVS, for smaller values of w, the worst case ratio is more for DCVS as compared to SR. When DCVS redistributes the load, it considers both the cooling power consumption and the switching cost, so in the process of rounding, some redistributions that generate lower switching cost may be chosen although they have greater cooling power consumption. This may be problematic because when the rounding is complete, the switching cost may be the same for other redistributions with lower cooling power consumption, although the switching cost was greater in the process. We saw this effect for the demand sequence $[1, 3, 1]$ and $w = 4$. So, the greediness of DCVS may be problematic in some cases. The results also show the performance of the Sep

Table 3.2: Performance of the algorithms in the presence of workload fluctuations with different rates

Sep		SR		DCVS	
avg	wcr	avg	wcr	avg	wrc
1.49	2.26	1.01	1.06	1.01	1.02
1.50	2.43	1.01	1.05	1.01	1.01
1.71	2.51	1.00	1.07	1.00	1.02
1.46	2.33	1.01	1.03	1.01	1.01
1.45	1.99	1.01	1.03	1.01	1.02
1.61	2.25	1.01	1.04	1.01	1.02
1.42	2.23	1.01	1.03	1.01	1.02
1.34	1.84	1.01	1.03	1.01	1.01
1.51	2.09	1.01	1.06	1.01	1.02
1.60	2.24	1.01	1.04	1.01	1.02
1.40	2.03	1.01	1.05	1.01	1.02
1.64	2.13	1.01	1.05	1.01	1.01
1.48	2.26	1.01	1.08	1.01	1.02
1.44	1.92	1.01	1.05	1.01	1.02
1.47	2.22	1.01	1.04	1.01	1.01
1.32	1.71	1.01	1.03	1.01	1.01
1.44	2.01	1.01	1.05	1.01	1.01
1.48	2.48	1.01	1.03	1.01	1.03
1.35	1.75	1.01	1.03	1.01	1.02
1.58	2.31	1.01	1.03	1.01	1.03

scheme is not as good as the others, specially for larger values of w, as a result of Sep ignoring correlations between consecutive time slots providing opportunities to reduce switching costs.

In the second part, we evaluate the performance of the algorithms in the presence of workload fluctuations with different rates. The number of time slots is $K = 9$. For each round of simulations, three values are chosen from D. So, there are three possibilities for the demand values. The rate of fluctuations is then controlled by a probability $p \in \{0.1, 0.5, 0.9\}$. D_1 is chosen randomly from the set of three demand

Table 3.3: Average running time of the algorithms (in seconds) in the presence of workload fluctuations with different rates

OPTi	SR	DCVS
17.53	0.56	0.71
3.08	0.68	0.78
8.7	0.68	0.87
10.55	0.61	0.79
6.58	0.70	0.87
10.14	0.65	0.81
15.42	0.73	0.98
10.68	0.61	0.80
8.83	0.60	0.77
3.10	0.61	0.80

values. For the next demand values D_2 to D_9, we pick the previous demand value with probability $1-p$ or pick a different value randomly with probability p. This procedure is repeated 100 times for each value of p in each round of simulations. We also set $w = 1000$. There are 20 round of simulations as reported in Table 3.2. According to the results in Table 3.2, DCVS has the best performance and is more resilient to the workload fluctuations with different rates. In addition, Table 3.3 reports the running times for $K = 20$, where at each round of simulations (each row in Table 3.3) the demand sequences are generated as explained for the previous results in Table 3.2. The results show that the running times for DCVS and SR are close and both are clearly faster than the *intlinprog* function in MATLAB.

The final results correspond to the integrated MPC approach. The number of time slots is $K = 50$ and the size of the planning window W is varied between 1 and 10. To calculate the solution over $K = 50$ time slots, the MPC approach uses a total of $K + W - 1$ demand values. So with $W_{max} = 10$, the length of the required demand sequence is $50 + 10 - 1 = 59$. We consider six scenarios for generation of demand

sequences. There are three cases for the range of demand values. Case 1 corresponds to choosing the demand values from $D = \{1, ..., 24\}$ uniformly at random. In Case 2, the next demand D_{k+1} is chosen from the range $[max(D_k - 5, 1), min(D_k + 5, 24)]$. In Case 3, the range for choosing D_{k+1} is $[max(D_k - 2, 1), min(D_k + 2, 24)]$. There is also a probability p that is the probability of changing the demand value for the next time slot. We chose two values of 0.2 and 0.8 for p. For example, in Case 3 and $p = 0.2$, with probability 0.2, the next demand value D_{k+1} is different from D_k and it is chosen from the range $[max(D_k - 2, 1), min(D_k + 2, 24)]/\{D_k\}$. We also set $w = 1000$. The total optimal cooling power consumption for each time slot ranges between 1500 and 2000. The IT power consumption for each working server is also around 375 Watts (223.4 + 154.5). The simulation is repeated 10 times for each scenario. For each round, the values in $\rho^{(0)}$ are chosen randomly from $\{0, 1\}$.

In addition to using the actual demands, we assume that we have a noisy version of demands coming from demand predictions. For a window of size W, starting from the time slot s, we assume that D_s is the actual value. However for $D_{s+1}, ..., D_{s+W-1}$, noise is added to the actual demand. The value of noise for the time slot $s+k-1, k = 2, ...W$, is randomly chosen from the interval $[-\eta \times k, \eta \times k]$, where η is the basic noise value. So, when the window shifts to the next time slot, the added noise is resampled with an updated distribution because k changes to $k - 1$ for the same time slot (s changes to $s + 1$). We apply the floor function to the noise value and add it to the actual demand. If the predicted demand is less than 1 or greater than 24, we choose the values 1 or 24, respectively. We consider three cases of $\eta = 0, \eta = 1, \eta = 3$, where $\eta = 0$ corresponds to the actual values without noise. For each value of $\eta > 0$ we repeat the procedure five times. The values reported in Table 3.4 are the ratios to

the cost calculated for the whole interval with actual demand values by using DCVS. The reference cost is very close to optimal.

When w is very small, the correlation between time slots is small, so using the MPC approach is not necessary. In this case, the problem can be solved by examining a single time slot problem. When w is larger, it becomes more important that the working servers in consecutive time slots are correlated. First, the load for each time slot is equal to the corresponding demand and in consecutive time slots, the set of servers with the smaller demand is a subset of the set of servers with larger demand. However, for larger values of w, to decrease the switching cost, the number of working servers may also be greater than the corresponding demand. Finally, as w becomes very large, the working servers for all time slots are the same. In this case, it is only necessary to focus on the time slot with the largest demand and use that solution for all other time slots. We expect that using the MPC approach is beneficial when there is an actual trade-off between the switching costs and the other components of the cost function.

The results for $w = 1000$ are shown in Table 3.4. We focused on $w = 1000$ because it clearly shows the trade-off between the costs and the performance of the MPC approach. The results for $\eta = 0$, show that for smaller window sizes (in particular $W = 1$), the performance is poor for all scenarios, with good performance achieved when $W = 4$. We can infer that in the short term the switching cost may be dominant.

However, when the size of W increases the IT (and cooling) power consumption does not allow extra servers to be working in several time slots. As an example, for demand $[15, 5, 5]$, to decrease the switching cost, it may be beneficial to increase the

Table 3.4: Performance of the integrated MPC approach with DCVS for different window sizes W, where there is noise with the parameter η and there are three cases for the next time slot demand range where the demand is changed with probability p for the next time slot.

Case 1 with $p = 0.2$						
	η=0		η=1		η=3	
W	avg	wrc	avg	wrc	avg	wrc
1	1.27	1.60	1.27	1.60	1.27	1.60
2	1.09	1.18	1.09	1.18	1.09	1.18
3	1.00	1.02	1.01	1.02	1.02	1.07
4	1.00	1.01	1.01	1.02	1.03	1.07
5	1.00	1.01	1.01	1.03	1.04	1.11
8	1.00	1.01	1.01	1.03	1.04	1.12
10	1.00	1.01	1.01	1.03	1.04	1.11
Case 1 with $p = 0.8$						
1	1.10	1.19	1.10	1.19	1.10	1.19
2	1.05	1.06	1.05	1.07	1.05	1.07
3	1.04	1.07	1.05	1.09	1.08	1.14
4	1.02	1.04	1.04	1.06	1.06	1.09
5	1.02	1.03	1.03	1.06	1.05	1.08
8	1.02	1.02	1.03	1.04	1.05	1.09
10	1.02	1.03	1.03	1.04	1.05	1.08
Case 2 with $p = 0.2$						
1	1.51	2.45	1.51	2.45	1.51	2.45
2	1.09	1.46	1.09	1.46	1.09	1.47
3	1.01	1.02	1.01	1.02	1.02	1.05
4	1.00	1.01	1.01	1.02	1.02	1.11
5	1.00	1.01	1.01	1.03	1.03	1.12
8	1.00	1.01	1.01	1.06	1.04	1.21
10	1.00	1.01	1.01	1.04	1.04	1.26
Case 2 with $p = 0.8$						
1	1.25	1.69	1.25	1.69	1.25	1.69
2	1.06	1.15	1.06	1.15	1.07	1.15
3	1.02	1.04	1.03	1.06	1.05	1.09
4	1.02	1.03	1.03	1.04	1.04	1.07
5	1.02	1.04	1.03	1.05	1.04	1.08
8	1.02	1.03	1.03	1.05	1.04	1.09
10	1.02	1.04	1.03	1.05	1.04	1.07
Case 3 with $p = 0.2$						
1	1.41	1.98	1.41	1.98	1.41	1.98
2	1.04	1.14	1.04	1.18	1.05	1.25
3	1.00	1.01	1.01	1.02	1.01	1.04
4	1.00	1.01	1.01	1.03	1.02	1.07
5	1.00	1.01	1.01	1.02	1.02	1.08
8	1.00	1.01	1.01	1.04	1.02	1.09
10	1.00	1.01	1.01	1.02	1.03	1.12
Case 3 with $p = 0.8$						
1	1.31	1.52	1.31	1.52	1.31	1.52
2	1.04	1.08	1.05	1.10	1.05	1.10
3	1.00	1.02	1.03	1.06	1.04	1.08
4	1.01	1.03	1.03	1.06	1.04	1.08
5	1.01	1.03	1.02	1.05	1.04	1.09
8	1.01	1.03	1.02	1.05	1.03	1.07
10	1.01	1.03	1.03	1.05	1.04	1.06

number of working servers at the second and third time slots above 5, however for the demand $[15, 5, 5, 5, 5, 5, 15]$, it might not be beneficial to increase the load in all time slots with demand equal to 5, because of the increase in IT (and cooling) power consumption. In general, the long term and short term solutions may be different. It can be inferred that as long as the window size is not too short, the MPC approach is beneficial as is shown for the case of $W = 3$ or $W = 4$. Using a window size of 4 (or 3) is also acceptable in the presence of noise. Using larger window sizes may not be helpful, in particular in the presence of noise when it might even degrade the performance, since the predicted demand is far from the actual demand. We also see how the performance degrades when the noise increases. So, updating information in the MPC approach is also helpful to improve the performance.

3.6 Conclusion

In this work, we formulated a nonlinear optimization problem for data centers that considers the switching costs in addition to cooling and IT power consumption. Workload transitions among the servers due to dynamic demands are not beneficial in terms of the switching costs, however they may decrease the power consumption. Next, we used a linearization approach proposed in [17] to approximate the solution. The steps were linearization of the problem and the development of a heuristic to approximate the solution of the linear problem. Finally, we proposed an integrated MPC approach with our proposed heuristics which is helpful to decrease the size of the problem and to incorporate demand predictions. The simulation results show that the proposed schemes are helpful to find a near-optimal solution efficiently. We showed that using an appropriate window size in the MPC approach is important and beneficial. As

future work, integrating transient thermal models with dynamic demands would be of interest. Modifying the proposed heuristic to address more problems, for example the case of heterogeneous data centers, is also a possibility. A procedure should also be defined for determining appropriate weights for switching costs. Finding a suitable window size for the MPC approach also needs more investigation.

Bibliography

[1] E. Masanet, N. Lei, “How Much Energy Do Data Centers Really Use?," *Aspen Global Change Institute*, Mar. 2020.

[2] W. Zhang, Y. Wen, Y. W. Wong, K. C. Toh, and C. H. Chen, “Towards Joint Optimization Over ICT and Cooling Systems in Data Centre: a Survey," in *IEEE Communications Surveys and Tutorials*, vol. 18, no. 3, pp. 1596-1616, thirdquarter 2016.

[3] Q. Tang, S. K. Gupta, and G. Varsamopoulos, “Thermal-aware Task Scheduling for Data Centers Through Minimizing Heat Recirculation," *IEEE International Conference on Cluster Computing*, Austin, TX, USA, 2007, pp. 129-138.

[4] S. MirhoseiniNejad, G. Badawy, and D. G. Down, “EAWA: Energy-aware Workload Assignment in Data Centers," *2018 International Conference on High Performance Computing & Simulation (HPCS)*, Orleans, France, 2018, pp. 260–267.

[5] N. EI-Sayed, I. A. Stefanovici, G. Amvrosiadis, and A. A. Hwang, “Temperature Management in Data Centers: Why Some (Might) Like It Hot," *SIGMETRICS*

'12: Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, pp. 163-174, Jun. 2012.

[6] Z. Xiong, M. Zhao, Z. Yuan, J. Xu, and L. Cai, "Energy-saving Optimization of Application Server Clusters Based on Mixed Integer Linear Programming," in *Journal of Parallel and Distributed Computing*, vol. 171, pp. 111-129, Jan. 2023.

[7] S. Shahryari, F. Tashtarian, and S. Hosseini-Seno, "CoPaM: Cost-aware VM Placement and Migration for Mobile Services in Multi-Cloudlet Environment: An SDN-based Approach," in *Computer Communications*, vol. 191, pp. 257-273, Jul. 2022.

[8] R. W. Ahmad, A. Gani, S. H. Ab. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A Survey on Virtual Machine Migration and Server Consolidation Frameworks for Cloud Data Centers," in *Journal of Network and Computer Applications*, vol. 52, pp. 11-25, Jun. 2015.

[9] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta, "TACOMA: Server and Workload Management in Internet Data Centers Considering Cooling-Computing Power Trade-off and Energy Proportionality," in *ACM Transactions on Architecture and Code Optimization*, vol. 9, no. 2, pp. 1-37, Jun. 2012.

[10] E. Pakbaznia, M. Pedram, "Minimizing Data Center Cooling and Server Power Costs," *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*, pp. 145–50, Aug. 2009.

[11] T. Van Damme, C. De Persis, and P. Tesi, "Optimized Thermal-Aware Job

Scheduling and Control of Data Centers," in*IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 760-771, Mar. 2019.

[12] M. Nakamura, "Learning and Optimization Models for Energy Efficient Cooling Control in Data Center," *Proceedings of the SICE Annual Conference*, 2016.

[13] A. M. Al-Qawasmeh, S. Pasricha, A. A. Maciejewski, and H. J. Siegel, "Power and Thermal-aware Workload Allocation in Heterogeneous Data Centers," in *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 477-491, Feb. 2015.

[14] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-efficient Thermal-aware Task Scheduling for Homogeneous High-performance Computing Data Centres: a Cyber-physical Approach," in *IEEE Transactions on Parallel and Distribution Systems*, vol. 19, no. 11, pp. 1458-1472, Nov. 2008.

[15] A. D. Carnerero, D. R. Ramirez, T. Alamo and D. Limon, "Probabilistically Certified Management of Data Centers Using Predictive Control," in *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2849-2861, Oct. 2022.

[16] S. Mirhoseininejad, G. Badawy, and D. G. Down, "A Data-driven, Multi-set Point Model Predictive Thermal Control System for Data Centers," in *Journal of Networks and Systems Management*, vol. 29, no. 7, 2021.

[17] S. Rostami, D. G. Down, and G. Karakostas, "Linearized Data Center Workload and Cooling Management," *arXiv:2304.04731 [eess.SY].*

[18] Q. Fang, J. Wang, Q. Gong, and M. Song, "Thermal-aware Energy Management

of an HPC Data Center via Two-time-scale Control," in *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2260-2269, Oct. 2017.

[19] Q. Fang, J. Wang, and Q. Gong, "QoS-driven Power Management of Data Centres via Model Predictive Control," in *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1557-1566, Oct. 2016.

[20] L. Parolini, B. Sinopoli, B. H. Krogh, and Z. Wang, "A cyber–physical Systems Approach to Data Center Modeling and Control for Energy Efficiency," in *Proceedings of the IEEE*, vol. 100, no. 1, pp. 254-268, Jan. 2012.

[21] R. Gupta, S. Asgari, H. Moazamigoodarzi, D. G. Down, and I. K. Puri, "Energy, Exergy and Computing Efficiency Based Data Center Workload and Cooling Management," in *Applied Energy*, vol. 299, 117050, Oct. 2021.

Chapter 4

Single-item Lot-sizing with Quantity Discount and Bounded Inventory

The content of this chapter is a publication under the following citation:

D. G. Down, G. Karakostas, S. G. Kolliopoulos, and S. Rostami, "Single-item lot-sizing with quantity discount and bounded inventory," *Operations Research Letters*, vol. 49, no. 6, pp. 877-882, 2021.

The author of this thesis is the main author and contributor of this publication. The author list is in alphabetical order.

Abstract

In this paper, an efficient $O(n^2)$ algorithm is proposed to solve a special case of single-item lot-sizing problems (SILSP) in which both the production and holding costs are piecewise linear, there is an all-unit discount with one breakpoint for the production cost, and the inventory is bounded. The algorithm is based on a key structural property that may be of more general interest, that of a just-in-time ordering policy. Finally, we show that when the problem is extended to two items, it is NP-complete.

Keywords: lot-sizing, quantity discount, bounded inventory

4.1 Introduction

Lot-sizing problems have seen continued research interest for the past six decades. Our focus is on single-item lot-sizing problems (SILSP), for which an extensive survey is provided in [1]. In the basic SILSP, there is a time-varying, but known in advance, demand for a single product over a time horizon of n periods. One wishes to determine the periods in which production (or purchase) of the product will take place, and the quantities that will be produced. The cost function is the summation of production and holding costs, where the production cost may include a fixed setup-cost component. The goal is to find a sequence of orders that satisfies the demands and minimizes the total cost. Several variations of the main problem can be defined, by including constraints such as production capacities, bounded inventory, stochastic demands, backlogging, and lost sales [1]. A typical assumption in the literature is that the cost function is piecewise concave, which provides the flexibility to model

scenarios such as discounts, minimum quantity requirements, and capacities [2]. A function with m breakpoints is piecewise concave if it is concave over each interval between two adjacent breakpoints. The case of capacitated production or bounded inventory can be modelled by adding a breakpoint to the production or holding cost function, and defining an infinite cost for the last interval [4].

In this paper, we study the very basic scenario where in every period one can buy the item at two price levels: expensively or cheaply depending on the quantity of units purchased. Demands have to be satisfied on time, without backlogging, and there are no order capacities, or set-up costs. In order to anticipate future demand, one may stock units bought cheaply, but this is curbed in two ways: hoarding incurs a holding cost *and* is limited by the hard constraint of a bound on the size of the inventory. In SILSP terminology, this is an all-unit discount pricing scheme with one breakpoint for the production cost and bounded inventory. *All-unit* means that the discount is applied to the entire volume of a purchase if this volume falls within a specified range. In our case, the range is any quantity at least equal to the single production breakpoint Q. To our knowledge this basic setting has not been considered separately in the literature. We show that an optimal solution can be found in polynomial time, and how exploiting a just-in-time ordering policy achieves significant running-time improvements. We assume that the production cost is a piecewise linear (and, hence, concave) function with two segments. The holding cost function is linear.

Solutions for SILSP problems are typically found using dynamic programming. There are several results in the literature, based on characterizations of the structure of an optimal solution that can be leveraged to develop efficient algorithms [3],[5],[6].

A seminal work of this nature (and for lot sizing in general) is [3]. They consider a problem with fixed unit production cost (the final inventory value is zero and only the setup cost changes) and linear holding cost, for which they derive an algorithm with time complexity $O(n^2)$. In [6], constraints on the production and inventory quantities are studied. For the case of bounded inventory and concave cost function, they propose an algorithm with complexity $O(n^3)$. In [7], an $O(mn^3)$ algorithm is presented for the problem with capacitated production and piecewise linear cost functions with an average of m breakpoints in each time period. In [8], an $O(n^{m+2}\log n)$ algorithm is constructed for the uncapacitated version of the problem in [7]. In [4], a model with an all-unit discount with one breakpoint and capacitated production is studied. The complexity of their algorithm is $O(n^4)$. In [9], a cost function with one breakpoint is considered and algorithms for the cases of all-unit and incremental discount with complexity of $O(n^2)$ and $O(n^3)$ respectively, are constructed. The production and holding costs considered in [10] are linear for the case of capacitated production. They study the NP-completeness of the problem based on five cases of general, nonincreasing, nondecreasing, constant and zero-value functions for the setup, production and holding costs, and production capacity. An $O(n^3)$ algorithm is constructed in [11] for the case of concave production cost and linear holding cost with constant capacities. An $O(n^2)$ algorithm is presented in [12] and fixed by [13] for the case of linear costs with a lower and upper-bounded inventory. A fully polynomial approximation scheme is proposed in [14] for an NP-hard case of SILSP. A more efficient approximation algorithm is proposed in [15]. In [16] a fixed cost and variable cost per unit for a change in the inventory size at each time period are also considered.

Our proposed problem is a special case of the general problem with piecewise

concave production and holding costs defined by Swoveland in [5]. Swoveland showed that there is an optimal solution which is a sequence of time intervals with starting and ending inventories either 0 or equal to the inventory bound, such that within each interval there is at most one order not equal to a production breakpoint level. In [5] the author used this to propose a pseudo-polynomial algorithm for capacitated production, bounded inventory and piecewise concave production and holding costs, which also works for our problem. In order to reduce the running time to polynomial, we essentially prove in Section 4.3 that there is an optimal solution which is a sequence of regeneration intervals with 0 starting and ending inventories, each comprised by at most two intervals of the type defined in [5]. In fact the structure of our regeneration interval is more specific than that, cf. Lemmas 3-5 for the exact description. We use this property to design in Section 4.4 an efficient $O(n^2)$ Dynamic Programming (DP) algorithm for the case of bounded inventory, piecewise linear production cost function with one breakpoint, and linear holding cost. We show that by adopting a just-in-time policy. Finally, we show in Section 4.5 that when the problem is extended to two items with separate budgets, it becomes NP-complete.

4.2 Problem Definition

A buyer has to order a number of units of a product, in order to satisfy n consecutive demand requests $d_1, d_2, \ldots, d_n$. At time period t, the buyer orders x_t units, so that, together with the remaining inventory I_{t-1} at the end of time $t-1$, demand d_t is satisfied, i.e., $x_t + I_{t-1} \geq d_t$. The new remaining inventory is $I_t = x_t + I_{t-1} - d_t$. Initially, the available inventory of the product is I_0. The buyer has an inventory capacity $B(t)$ at time t, which cannot be exceeded at any time, i.e., $I_t \leq B(t),\ t =$

$1, \ldots, n$.

The product pricing is as follows: Given threshold Q, if $x_t < Q$ then the buyer pays a price $p_1(t)$ per unit, otherwise the buyer pays a price $p_2(t)$ per unit, where $p_1(t) \geq p_2(t)$. We assume that unit prices are non-increasing as a function of time, i.e., $p_1(t) \geq p_1(t'),\ p_2(t) \geq p_2(t'),\ t < t'$. There is also a holding cost $h(t)$ per inventory unit for time t. Given Q, I_0, and $p_1(t),\ p_2(t),\ B(t),\ d_t,\ h(t),\ t = 1, 2, \ldots, n$, we would like to compute orders $x_t,\ t = 1, 2, \ldots, n$ that respect the inventory capacity constraint, and minimize the buyer's total cost.

In what follows, an *expensive* order at time t is an order priced at $p_1(t)$, and a *cheap* order is an order priced at $p_2(t)$. For $i \leq j$, $d_{i,j}$ denotes $\sum_{k=i}^{j} d_k$.

4.3 Structural Properties of an Optimal Solution

Let OPT be an optimal solution. In this section we show that there is an optimal solution of a special structure, which will allow its fast computation. Similarly to [2], we define a *regeneration interval* (i, j) of OPT (with $i \leq j$) to be a sequence of consecutive time periods $[i, i+1, \ldots, j]$ with inventories $I_k > 0$ for all $i \leq k \leq j-1$, and $I_{i-1} = 0$ or $i = 1$, and $I_j = 0$ or $j = n$. In other words, a regeneration interval is the time between two times i and j which starts and ends with 0 inventory (except for the interval with $i = 1$, or $j = n$), and maintains a non-zero inventory everywhere in between. Note that if $I_k > 0,\ \forall 1 \leq k < n$, then there is only one interval $(1, n)$.

Starting from OPT, we derive an optimal solution with the following structural properties:

Lemma 3 *There is an optimal solution where, for any regeneration interval (i, j), an expensive order can only occur at time j. If x_j is expensive, then $x_j = d_j - I_{j-1}$.*

Proof: Let (i, j) be an interval of OPT with an expensive order x_k at time $i \leq k < j$. Since there are leftover inventories $I_k > 0, I_{k+1} > 0, \ldots, I_{j-1} > 0$, we can reduce order x_k by 1, reduce all these inventories by 1, and increase order x_j by 1, without harming the cost or the feasibility of OPT (x_k was already an expensive order below Q, x_j is increased, prices are non-increasing, and the intermediate inventories are reduced, thus not increasing the holding costs). We continue this process of reducing x_k by 1, until either $x_k = 0$ or $I_l = 0$ for some $k \leq l < j$. In the second case, interval (i, j) is now split into two or more smaller intervals, and we can repeat the process with the new set of intervals, until we obtain the optimal solution claimed by the lemma.

If x_j is expensive, then if $j < n$, by the definition of an interval, we have $I_j = x_j + I_{j-1} - d_j = 0$; if $j = n$, then OPT uses the available inventory I_{n-1} and if $d_n > I_{n-1}$ orders an additional $x_n = d_n - I_{n-1}$ or Q units (whichever is cheaper). □

Lemma 4 *There is an optimal solution satisfying the property of Lemma 3, which satisfies the following: For any regeneration interval (i, j), let k be the total number of cheap orders occurring in it. Then the first $k - 1$ orders are for exactly Q units, and the k-th order is for at least Q units.*

Proof: Let OPT be an optimal solution satisfying the property of Lemma 3, and let (i, j) be any of its intervals.

Lemma 3 implies that the only expensive order can occur at time j, so the previous k orders are all cheap. Let $x_{l_1}, x_{l_2}, \ldots, x_{l_k}$ be these orders $l_k \leq j$, occurring at times $l_1, l_2, \ldots, l_k$, respectively. Let $l_m < l_k$ be the last of these times before the last, for which $x_{l_m} > Q$ (i.e., $x_{l_{m+1}} = \cdots = x_{l_{k-1}} = Q$). Then, since $I_{l_m} > 0, I_{l_m+1} > 0, \ldots, I_{l_k-1} > 0$, we can reduce x_{l_m} by 1 without changing its pricing, reduce inventories $I_{l_m}, I_{l_m+1}, \ldots, I_{l_k-1}$ by 1, and increase x_{l_k} by 1, without violating the feasibility of the solution or increasing its cost (both x_{l_m} and x_{l_k} remain cheap, the prices are non-increasing, and the holding cost is always non-increasing when postponing orders for later). We continue repeating this process, until either $x_{l_m} = Q$, or at least one of the inventories $I_{l_m}, I_{l_m+1}, \ldots, I_{l_k-1}$ becomes 0. In the first case, we repeat this process, but now concentrating on a cheap order bigger than Q that is earlier than l_m; in the second case, interval (i, j) is split into two or more (smaller) intervals, and we can repeat the process with the new set of intervals. □

In order to show that there is an optimal solution making cheap orders just-in-time, we will need the following definition.

Definition 1 *Let (i, j) be a regeneration interval of an optimal solution adhering to the structure of Lemma 4, with its last cheap order at time $i \leq t < j$. A* feasible time t' *is any time $t < t' \leq j$ where a cheap just-in-time order can be placed, without increasing I_j, and without violating inventory bounds, after setting $x_t := Q$. More specifically, given inventory I_{t-1} and $x_t := Q$, let $t < k \leq j$ be the time when both*

$$I_{t-1} + Q \geq d_{t,k-1} \text{ and } I_{t-1} + Q < d_{t,k}$$

hold. Then, if a cheap order can be placed at k without increasing I_j, and all inventory

bounds $B(l)$, $k \leq l \leq j$, are respected, we set $t' := k$.

Note that in Definition 1, it may be the case that the latest possible time k one can afford to wait until forced to place a cheap order just-in-time, may not be a feasible time, e.g., if $Q - d_k > B(k)$ and, therefore, there is not enough inventory space to accommodate a cheap order at k. If this is the case, then placing a cheap order for Q units earlier than k will still violate the inventory constraint at k, i.e., if k cannot be a feasible time, no time $t < t' < k$ can be a feasible time.

Lemma 5 *There is an optimal solution satisfying the properties of Lemmas 3 and 4, which, for any regeneration interval (i, j), also satisfies the following: If there are cheap orders, then every cheap order occurs at the furthest possible time from the previous one (or from time i, for the first cheap order); also, for $j < n$, there is no feasible time for a new cheap order between the last cheap order and j.*

Proof: Let OPT be an optimal solution complying with Lemmas 3 and 4.

First we prove the lemma for intervals (i, j) with $i > 1$. For simplicity, we assume that $d_i > 0$. Since $I_{i-1} = 0$, $x_i > 0$ in OPT. If this order is the expensive one, or the only cheap one, then there is nothing more to prove. Otherwise, since there is more than one cheap order, so $x_i = Q$ (Lemma 4). Let $k > i$ be the latest time when there must be a cheap order, otherwise some demands will not be satisfiable, i.e., k is the latest possible time in the sense of Definition 1 (since there is more than one cheap order, k is well-defined). If there are no other cheap orders by OPT in the times between i and k, then notice that OPT has to order cheaply at k, and we repeat our arguments here with k playing the role of i. Otherwise, let $x_{l_1} = x_{l_2} = \ldots = x_{l_m} \geq Q$ be these cheap orders with $i < l_1 < \ldots < l_m < k$. Then we can place all these orders

(together with any preexisting order $x_k > 0$ in OPT) cheaply at time k, since the inventory I_{k-1} is now reduced by $\sum_{g=1}^{m} x_{l_g}$, and the other intermediate inventories can only decrease. By the definition of k, no demand is left uncovered by this transfer of orders. If after the transfer inventory I_t drops to zero for some $i < t \leq k-1$ we split (i, j) into smaller regeneration intervals, like in Lemma 3. Moreover, similarly to the previous lemma the cost does not increase. By this process, we get another optimal solution OPT', with its first two cheap orders at times i, k, with $x_i = Q$ and $x_k \geq Q$, and which can be brought into the format of Lemma 4. Then we repeat the argument above (with k now playing the role of i) repeatedly, until we obtain an optimal solution satisfying the property of the lemma.

For the first interval $(1, j)$, we can define time k exactly as before, while it may be the case $x_1 = 0$ in OPT (OPT uses the initial inventory I_0 to satisfy the initial demands). Then the argument proceeds exactly as before.

For the second property, assume that (i, j) (with $j < n$) is a regeneration interval of an optimal solution that satisfies the first property of the lemma. Note that if there is a feasible time t' after the last cheap order x_t at time $t < t'$, then we can set $x'_t := Q$ and $x'_{t'} := x_t - Q$, without increasing the cost of the solution; this is because $x'_{t'}$ is cheap (by the feasibility of t'), and the inventories $I_t, I_{t+1}, \ldots, I_{t'-1}$ can only decrease, ensuring that holding costs do not increase. If there is more than one feasible time, let t' be the feasible time furthest from t. Then the new optimal solution also satisfies the first property of the lemma. As before, if the new inventory $I'_l = 0$ for some $t \leq l \leq t' - 1$, then we split (i, j) into two new intervals $(i, l), (l + 1, j)$ and repeat the process with the new set of intervals. Otherwise, we can repeat this process until

there is no feasible time after the last cheap order in our final optimal solution. □

4.4 A Dynamic Programming Algorithm

We use the properties of Lemmas 3-5 to compute an optimal solution, using Dynamic Programming (DP). Let $OPT(i)$ be the optimal cost for interval (i, n). Then

$$OPT(i) = \begin{cases} \min_{i \le k \le n}\{cost(i,k) + OPT(k+1)\}, & i \le n \\ 0, & i = n+1 \end{cases} \tag{4.4.1}$$

where $cost(i, j)$ is the minimum feasible cost in a regeneration interval (i, j).

The crucial idea for the algorithm is to notice that the cheap orders at level exactly Q made to achieve $cost(i, j)$ remain valid for the sequence of orders that achieve $cost(i, j + 1)$, due to Lemma 5. This means that the addition of d_j affects only the last cheap order of $cost(i, j)$ in two possible ways: (i) either it also remains the last cheap order for $cost(i, j)$, but with a different number of units (and with a potential corresponding expensive order x_{j+1}), or (ii) there are new feasible time(s) created, so it breaks into one or more Q just-in-time cheap orders, according again to Lemma 5.

In what follows, we denote by $x^{(i,j)}(t)$ the order placed at time t for the solution that achieves $cost(i, j)$. A sequence of orders is *feasible for* (i, j) if it respects the inventory bounds, $I_j = 0$ and $I_k > 0,\ i \le k \le j - 1$, and moreover complies with Lemma 5.

Preprocessing: The algorithm starts by calculating the following quantities:

- $d_{i,j} = \sum_{k=i}^{j} d_k$, $h_{i,j} = \sum_{k=i}^{j} h(k)$, and $H(i,j) = \sum_{k=i}^{j} h(k)d_{i,k}$ for all $1 \leq i \leq j \leq n$. These quantities are needed in calculations performed by the algorithm, and they are done in $O(n^2)$ time.

- For all $1 \leq i \leq j \leq n$, let $A(i,j)$ be the maximum order size at time i, that does not violate any inventory bounds $B(l),\ i \leq l \leq j$ when we assume that $I_{i-1} = 0$ and no other orders happen between $i+1$ and j (inclusive). Then

$$A(i,i) = B(i) + d_i$$

$$A(i,j+1) = \begin{cases} A(i,j), \text{ if } A(i,j) - d_{i,j+1} \leq B(j+1) \\ d_{i,j+1} + B(j+1), \text{ otherwise.} \end{cases}$$

  Note that if there is inventory $I_{i-1} > 0$ coming into i, the value $A(i,j) - I_{i-1}$ gives an upper bound for order x_i, which ensures that x_i will not violate any inventory bounds in time interval (i,j). All values $A(i,j)$ can be calculated in total $O(n^2)$ time.

- For every i, the algorithm calculates a sequence $\mathcal{T}^i = \{t_1, t_2, \ldots, t_l\}$ of potential just-in-time order times, as well as inventory values $\hat{I}_{t_1-1}, \hat{I}_{t_2-1}, \ldots, \hat{I}_{t_l-1}$ as follows: The initial condition is that $t_1 = i$ and $\hat{I}_{t_1-1} = 0$. If $x_i = Q$ is infeasible, the sequence $\mathcal{T}^i$ is empty. To calculate t_s, $s \geq 2$ given inventory $\hat{I}_{t_{s-1}-1}$, we set $x_{t_{s-1}} := Q$ and scan times $t_{s-1}+1, t_{s-1}+2, \ldots$ until we reach the furthest time t_s when a cheap order can be placed with no demand being unsatisfied, and no inventory bound violated. The former can be checked by checking that $Q + \hat{I}_{t_{s-1}-1} \geq d_{t_{s-1},f}$, and the latter can be checked by checking that $Q \leq A(t_{s-1}, f) - \hat{I}_{t_{s-1}-1}$, for all times $f = t_{s-1}+1, t_{s-1}+2, \ldots$ we scan. If

there can be no cheap order at t_s, then t_{s-1} is the last element of sequence $\mathcal{T}^i$. Otherwise, we also calculate inventory value $\hat{I}_{t_s-1} = \hat{I}_{t_{s-1}-1} + Q - d_{t_{s-1},t_s-1}$, and continue to discover the next sequence element t_{s+1}. Clearly sequence $\mathcal{T}^i$ and values $\hat{I}$ can be calculated in $O(n-i)$ time, or $O(n^2)$ for all i. Inventory values $\hat{I}$ will be used in the calculation of functions f_1, f_2 below. In what follows i is fixed so we use $\mathcal{T}$ in place of $\mathcal{T}^i$.

Remark: The sequence $\mathcal{T}^i$ is the sequence of possible cheap order times that is promised by Lemma 5, i.e., for period $(i, j+1)$ all cheap orders happen at times that are a prefix of $\mathcal{T}^i$, all of them except possibly the last are exactly Q, and there is potentially one more order at $j+1$.

- For each s'th element of sequence $\mathcal{T}^i$, we calculate the following cost

  $C^i(t_s) = \sum_{w=1}^{s-1} \Big(Qp_2(t_w) + \sum_{k=t_w}^{t_{w+1}-1} h(k)(\hat{I}_{t_w-1} + Q - d_{t_w,k}) \Big)$. This is the ordering and holding cost incurred in time period $(t_1, t_s - 1)$ when all orders at times $t_1, t_2, \ldots, t_{s-1} \in \mathcal{T}^i$ are exactly Q.

In the special case of $i = j$ with no capability of placing a cheap order at time i ($d_i < Q$), we set $x_i^{(i,i)} = d_i$, $cost(i,i) = p_1(i) \cdot d_i$, and $cost(i, i+k) = \infty$, $\forall 1 \leq k \leq n-i$; otherwise we set $x_i^{(i,i)} = d_i$, $cost(i,i) = p_2(i) \cdot d_i$.

Assuming that $cost(i,j) \neq \infty$ has been calculated, $cost(i, j+1)$ is calculated as follows: Let $t_{s-1}, t_s \in \mathcal{T}$ be the last two times in $\mathcal{T}$ before $j+1$ in $(i, j+1)$ (since $t_1 = i$, there is always at least one such time; if there is only t_1, then what follows is adjusted accordingly). There are orders of exactly Q units up to time t_{s-2}, and there are two possibilities for ordering after t_{s-2}: (i) Set $x_{t_{s-1}} := Q$ and order at t_s and at $j+1$, or (ii) order at t_{s-1} and at $j+1$. We can calculate (in constant time)

the optimal orders $x_{t_s}^{(i,j+1)}, x_{j+1}^{(i,j+1)}$ and cost for (i) as follows (the calculation for (ii) is exactly the same):

We define the following functions

$$\begin{aligned} f_1(x_{t_s}, x_{j+1}) = \\ &= p_2(t_s)x_{t_s} + p_1(j+1)x_{j+1} + \sum_{k=t_s}^{j} h(k)(\hat{I}_{t_s-1} + x_{t_s} - d_{t_s,k}) \\ &= (p_2(t_s) + h_{t_s,j})x_{t_s} + p_1(j+1)x_{j+1} + \hat{I}_{t_s-1}h_{t_s,j} - H(t_s, j) \quad (4.4.2) \end{aligned}$$

and

$$\begin{aligned} f_2(x_{t_s}, x_{j+1}) = \\ &= p_2(t_s)x_{t_s} + p_2(j+1)x_{j+1} + \sum_{k=t_s}^{j} h(k)(\hat{I}_{t_s-1} + x_{t_s} - d_{t_s,k}) \\ &= (p_2(t_s) + h_{t_s,j})x_{t_s} + p_2(j+1)x_{j+1} + \hat{I}_{t_s-1}h_{t_s,j} - H(t_s, j). \quad (4.4.3) \end{aligned}$$

Given orders x_{t_s} and x_{j+1}, the functions f_1, f_2 give the cost incurred when x_{t_s} is cheap and x_{j+1} is cheap or expensive, respectively. Note that these functions can be computed in $O(1)$ time, since values $h_{t_s,j}, H(t_s, j)$, and $\hat{I}_{t_s-1}$ have been precomputed.

We solve the following two linear integer programs:

$$
\begin{aligned}
c_1 = \min f_1(x_{t_s}, x_{j+1}) + h(j+1)I_{j+1} + C^i(t_s) \text{ s.t.} \\
x_{t_s} + \hat{I}_{t_s-1} - d_{t_s,j} \geq 1 \\
x_{t_s} + \hat{I}_{t_s-1} + x_{j+1} - I_{j+1} = d_{t_s,j+1} \\
I_{j+1} \begin{cases} = 0, & \text{if } j \leq n-2 \\ \leq B(j+1), & \text{if } j = n-1 \end{cases} \\
x_{t_s} \leq A(t_s, j) - \hat{I}_{t_s-1} \\
x_{t_s} \geq Q \\
0 \leq x_{j+1} \leq Q - 1
\end{aligned}
$$

and

$$
\begin{aligned}
c_2 = \min f_2(x_{t_s}, x_{j+1}) + h(j+1)I_{j+1} + C^i(t_s) \text{ s.t.} \\
x_{t_s} + \hat{I}_{t_s-1} - d_{t_s,j} \geq 1 \\
x_{t_s} + \hat{I}_{t_s-1} + x_{j+1} - I_{j+1} = d_{t_s,j+1} \\
I_{j+1} \begin{cases} = 0, & \text{if } j \leq n-2 \\ \leq B(j+1), & \text{if } j = n-1 \end{cases} \\
x_{t_s} \leq A(t_s, j) - \hat{I}_{t_s-1} \\
x_{t_s} = Q \\
x_{j+1} \geq Q
\end{aligned}
$$

The objective is the cost f_1 or f_2, plus the holding cost for time $j+1$ and the costs

incurred before t_s. The first constraint ensures that x_{t_s} satisfies all demands until time $j+1$ while always leaving an inventory (otherwise this cannot be a feasible regeneration period), the second that the order values as well as inventory I_{j+1} are consistent and satisfy all demands, and the remaining constraints ensure that the final inventory and the order values satisfy their inventory restrictions, as well as the cheap/expensive definitions. Note that both integer programs have variables $x_{t_s}, x_{j+1}, I_{j+1}$, and can be solved in $O(1)$ time. If c_1 is infeasible, then we set $c_1 = \infty$, and the same for c_2. Let $cost_1 = \min\{c_1, c_2\}$.

We repeat the same process for $x_{t_{s-1}}^{(i,j+1)}$ and $x_{j+1}^{(i,j+1)}$, and let $cost_2 = \min\{c_1', c_2'\}$, where c_1', c_2' are the analogues of c_1 and c_2. If $cost_1 < cost_2$, then we return the calculated $x_{t_s}^{(i,j+1)}, x_{j+1}^{(i,j+1)}$, and $cost(i, j+1) = cost_1$. Otherwise, if $cost_2 = \infty$ return $cost(i, j+1) = \infty$, else return the calculated $x_{t_{s-1}}^{(i,j+1)}, x_{j+1}^{(i,j+1)}$, and $cost(i, j+1) = cost_2$.

Correctness: We show the following

Theorem 1 *The DP algorithm above computes an optimal solution.*

Proof: To prove that the algorithm is correct, it is enough to prove that it produces an optimal solution of the structure guaranteed by Lemma 5, i.e., it is enough to show that the $cost(i, j+1)$ calculated for regeneration period $(i, j+1)$ (and has the structure of Lemma 5) is optimal.

Lemma 5 implies that all cheap orders (except possibly x_{j+1}) in $(i, j+1)$ are done on the times of sequence $\mathcal{T}^i$, by the definition of the latter as the just-in-time sequence of potential cheap order times. Let t_{s-2}, t_{s-1}, t_s be the last members of $\mathcal{T}^i$ before $j+1$. We observe that, in $(i, j+1)$, the cheap orders up to time $t_{s-2} \in \mathcal{T}^i$ (inclusive)

are orders of exactly Q units. This is due to the fact that if t_{s-2} were the last cheap order time for $(i, j+1)$, then $x_{t_{s-2}} \geq 2Q$ to cover d_{t_{s-2},t_s}, since orders of size Q had to be placed at t_{s-2}, t_{s-1} to cover these demands by just-in-time sequence $\mathcal{T}^i$. As a result, we can set $x_{t_{s-2}} := Q$ and move the rest of the units to a cheap order at t_{s-1}, without increasing the cost. Hence, it is enough for the algorithm to check the pairs of times $t_{s-1}, j+1$ and $t_s, j+1$ for the calculation of the best last two orders, which will also result in the optimal $cost(i, j+1)$. □

Complexity: First observe that the calculation of sequence $\mathcal{T}^i$ can be done in time $O(n-i)$, since all its orders can all be calculated by scanning interval (i, n) once. In every interval (i, j) we need to calculate orders for 3 times, namely j and the last 2 times of $\mathcal{T}^i$ contained in (i, j), and this can be done in $O(1)$ time. All other times of $\mathcal{T}^i$ before j get an order of exactly Q. Also note that there is no need to record separately all orders placed in all intervals $(i, j), i \leq j \leq n$, since all we need to record for each one is the last time of $\mathcal{T}^i$ utilized by (i, j), the last cheap order time and size, and $x^{(i,j)}(j)$. With this information, we can reconstruct the orders that incur $cost(i, j)$. Hence, the overall running time and space needed for all i is $O(n^2)$.

Collecting all running times and space needed by the different algorithm components we get

Theorem 2 *The DP algorithm takes $O(n^2)$ time and space.*

4.5 The Case of Two Items

The two-items decision problem we examine is defined as follows: We have two items, *black* and *red,* which can be stored in an inventory of total capacity I. The pricing scheme for black is given by fixed unit prices $b_1 > b_2$, and threshold Q_B above which price b_2 applies. The pricing scheme for red is given by fixed unit prices $r_1 > r_2$, and threshold Q_R. Given budget targets C_B, C_R for black and red, as well as a sequence of n black and red demands, the question is whether they can be fulfilled, within the corresponding budgets, and without ever exceeding the inventory capacity I. There can also be initial inventories of the two items, but here we prove the NP-completeness of the problem, even in the case where these initial inventories are 0. More formally, the problem is defined as follows:

2-Bulk Ordering

Input: Sets $F = \{f_1, \ldots, f_n\} \subseteq \mathbb{Z}^+$ of black demand values and $G = \{g_1, \ldots, g_n\} \subseteq \mathbb{Z}^+$ of red demand values. Pricing schemes (b_1, b_2, Q_B) and (r_1, r_2, Q_R) for the black and red item respectively. Inventory size I. Nonnegative target costs C_B and C_R.

Question: Are there two n-vectors of orders for the black and the red item so that all demands are satisfied, the inventory capacity is never exceeded, the cost of black orders is at most C_B and the cost of red orders is at most C_R?

A pair of n-vectors of black and red orders is an *admissible sequence* for a 2-Bulk Ordering instance if the units purchased can be consumed at the time they were bought or stored so that all the demands at each time are covered and the inventory

capacity I is never exceeded. We reduce from the EQUIPARTITION problem.

EQUIPARTITION

Input: Set $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{Z}^+$ with $\sum_{i=1}^{n} a_i = 2B$.

Question: Is there an index set $A' \subseteq [n]$, $|A'| = \frac{n}{2}$, such that $\sum_{i \in A'} a_i = B$?

It is well-known that EQUIPARTITION is NP-complete [17]. Given an EQUIPARTITION instance A, we construct an instance $\phi(A)$ of the 2-BULK ORDERING problem, that has an affirmative answer iff the EQUIPARTITION instance does. We now proceed to define $\phi(A)$. We set $b_1 = r_1 = p_1$ and $b_2 = r_2 = p_2$ for two positive values satisfying $p_1 > p_2$, i.e., both items have the same prices $p_1 > p_2$ (whose exact values will not matter in the reduction). We set

$$I = 5 \max_{i \in [n]} a_i \tag{4.5.1}$$

$$Q_B = Q_R = Q = 2I + 3B + 1 \tag{4.5.2}$$

$$C_B = C_R = p_1(\frac{n}{2}Q + 2B) + p_2(\frac{n}{2}Q + 2B). \tag{4.5.3}$$

The time horizon of $\phi(A)$ consists of n consecutive intervals $S_1, \ldots, S_n$, with each interval consisting of four time periods with respective demands $Q - (I - 2a_i), 0, I, 0$ for black, and $0, Q - (I - 2a_i), 0, I$ for red (cf. Figure 4.1). When a specific interval is implied from the context its time periods are numbered from 1 to 4.

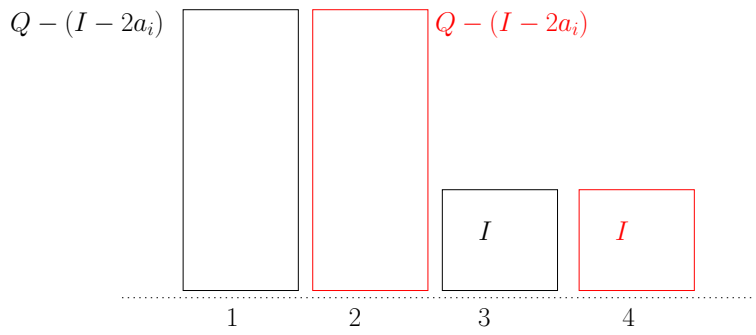

Figure 4.1: Demands for the four times of interval S_i, $i \in [n]$.

Lemma 6 *In any admissible sequence for the instance $\phi(A)$ at most one of black, red makes a cheap purchase in interval S_i, $i \in [n]$. This purchase happens at time 1 for black or at time 2 for red.*

Proof: Because of (4.5.2) a cheap purchase is possible only at time 1 or 2. In interval S_i, $i \in [n]$, each of the items requires $I - 2a_i$ space in the inventory for a cheap purchase at time 1 (for black) or 2 (for red). By (4.5.1) $2a_i < I - 2a_i$, hence there is not enough free inventory space for cheap purchases in both times 1 and 2. □

Lemma 7 *In any admissible sequence for $\phi(A)$ where the cost of black orders is at most C_B at least $n/2$ cheap orders for the black item are placed.*

Proof: Assume that in the admissible sequence $n/2-x$ cheap orders have been placed for black for some $x \geq 1$. Let N_0 be the set of indices of the intervals in which there are no cheap black purchases, and $N_1 = [n] \setminus N_0$. Lemma 6 implies that $|N_1| = \frac{n}{2} - x$, and, therefore, $|N_0| = \frac{n}{2} + x$. Let ℓ_i be the total net amount of black item that can be fetched from the inventory to cover demand during interval S_i. This amount must have been purchased at some previous time, and at a unit price of at least p_2. Hence,

the total cost $cost_B$ incurred for the black item is lower-bounded as follows:

$$\begin{aligned} cost_B &\geq p_1[(\frac{n}{2}+x)Q + \sum_{i\in N_0} 2a_i] - p_1 \sum_{i\in N_0} \ell_i + \\ &\quad + p_2[(\frac{n}{2}-x)Q + \sum_{i\in N_1} 2a_i] + p_2 \sum_{i\in N_0} \ell_i \\ &\geq p_1(\frac{n}{2}+x)Q - p_1 \sum_{i\in N_0} \ell_i + p_2[(\frac{n}{2}-x)Q] + \\ &\quad + 4Bp_2 + p_2 \sum_{i\in N_0} \ell_i \\ &= C_B + (p_1-p_2)(xQ - 2B - \sum_{i\in N_0} \ell_i). \end{aligned} \tag{4.5.4}$$

Note that, for any admissible sequence, we can assume that items are never purchased expensively, in order to be stored in the inventory. Let the *excess* of a cheap purchase $x_t \geq d_t$ at some time t be the amount $x_t - d_t$ which will be stored in the inventory. Given an admissible sequence the associated red and black costs are uniquely determined by the order vectors and do not depend on when a stored unit is actually consumed. For the purposes of cost accounting we may allocate the excess units to satisfy demand at any time of our choice as long as all demand has been satisfied by the end of the time horizon and all the excess has been consumed. The *maximum* amount of a cheap purchase at time 1 of interval S_i, $i \in N_1$, is $Q - (I - 2a_i) + I = Q + 2a_i$, i.e., the excess is at most $2a_i$. The calculation in (4.5.4) allocated $Q + 2a_i$ units to cover the black demand in interval $i \in N_1$. This way all potential excess has been consumed and it follows that $\sum_{i\in N_0} \ell_i = 0$. But in that case, (4.5.2) and (4.5.4) imply that $cost_B > C_B$, a contradiction to the assumption that the black cost is at most C_B. Therefore $x = 0$. □

Similarly we can prove:

Lemma 8 *In any admissible sequence for $\phi(A)$ where the cost of red orders is at most C_R at least $n/2$ cheap orders for the red item are placed.*

Lemma 9 *The* Equipartition *instance A is a* `Yes`*-instance iff the* 2-Bulk Ordering *instance $\phi(A)$ is a* `Yes`*-instance.*

Proof: Let the Equipartition instance have a solution given by $A' \subseteq [n]$, $|A'| = n/2$ such that $\sum_{i \in A'} a_i = B$. There is a corresponding admissible sequence for the 2-Bulk Ordering instance, in which a cheap red purchase of $Q + 2a_i$ units takes place at every interval S_i, $i \in A'$, and a cheap black purchase at the amount of $Q + 2a_i$ takes place for every S_i, $i \in [n] \setminus A'$. The sequence meets the cost targets for both items.

Conversely, assume that the 2-Bulk Ordering instance is a `Yes`-instance. Let T_B (T_R) be the indices of the intervals in which a black (red) cheap purchase occurs. By Lemmas 6, 7 and 8 $|T_B| = |T_R| = n/2$. The excess of a cheap purchase (either black at time 1 or red at time 2) is at most I. We can lower bound the cost by assuming that it is always I.

We allocate all potential excess of a cheap purchase at interval $S_i \in T_B$ ($S_i \in T_R$) to cover the demand at time 3 (resp. 4). Therefore the total black item amount bought expensively is at least $\frac{n}{2}Q + \sum_{i \in T_R} 2a_i$, and the total red item amount bought expensively is at least $\frac{n}{2}Q + \sum_{i \in T_B} 2a_i$. It holds that $\sum_{i \in T_R \cup T_B} 2a_i = \sum_{i \in [n]} 2a_i = 4B$. If $\sum_{i \in T_R} 2a_i = 2B + y$ for some positive y, the cost paid for the black item is $C_B + (p_1 - p_2)y > C_B$, a contradiction, and similarly for the red. Therefore $\sum_{i \in T_R} 2a_i \leq 2B$ and $\sum_{i \in T_B} 2a_i \leq 2B$, which imply that $\sum_{i \in T_R} 2a_i = \sum_{i \in T_B} 2a_i = 2B$,

i.e., the EQUIPARTITION instance has a feasible solution. □

Lemma 9, together with the easy fact that 2-BULK ORDERING is in NP, imply the following

Theorem 3 *The* 2-BULK ORDERING *problem is* NP*-complete even when both items have the same pricing scheme* (p_1, p_2, Q).

A pair of values (x_1, x_2) *dominates* another pair (y_1, y_2) if $x_1 \leq y_1$ and $x_2 \leq y_2$. The *cost pair* of an admissible sequence σ for a 2-BULK ORDERING instance is the pair of black and red costs incurred by σ. The *trade-off* or *Pareto curve* for 2-BULK ORDERING is the set of all admissible sequences for which the corresponding cost pair is not dominated by the cost pair of any other (admissible) sequence. Observe that in the proof of Lemma 9 we actually established that the EQUIPARTITION instance is a `Yes`-instance iff there is an admissible sequence for the corresponding 2-BULK ORDERING instance where the cost for black is exactly C_B and for red is exactly C_R. The following corollary is immediate and suggests the intractability of identifying points on the Pareto curve.

Corollary 1 *Given an instance of* 2-BULK ORDERING *and a pair of cost targets* (C_1, C_2) *it is* NP*-complete to determine whether (i) there is an admissible sequence whose cost pair dominates* (C_1, C_2) *and (ii) there is an admissible sequence on the Pareto curve with cost pair* (C_1, C_2).

Acknowledgement The authors thank an anonymous reviewer for comments that helped to substantially improve the paper.

Bibliography

[1] N. Brahimi, N. Absi, S. Dauzére-Pérés, A. Nordli, Single-item dynamic lot-sizing problems: An updated survey, European Journal of Operational Research, 263 (3) (2017) pp. 838-863.

[2] E. Koca, H. Yaman, M. Selim Aktürk, Lot sizing with piecewise concave production costs, INFORMS Journal on Computing 26 (4) (2014) pp. 767-779.

[3] H. M. Wagner, T. M. Whitin, Dynamic version of the economic lot size model, Management Science, 5 (1) (1958) pp. 1007-1013.

[4] Y. Malekian, S. Hamid Mirmohammadi, M. Bijari, Polynomial-time algorithms to solve the single-item capacitated lot sizing problem with a 1-breakpoint all-units quantity discount, Computers & Operations Research, 134 (2021).

[5] C. Swoveland, A deterministic multi-period production planning model with piecewise concave production and holding-backorder costs, Management Science, 21 (9) (1975) pp. 89-96.

[6] S. F. Love, Bounded production and inventory models with piecewise concave costs, Management Science, 20 (3) (1973) pp. 313-318.

[7] J. Ou, A polynomial time algorithm to the economic lot sizing problem with constant capacity and piecewise linear concave costs, Operations Research Letters, 45 (5) (2017) pp. 493-497.

[8] J. Ou, Improved exact algorithms to economic lot-sizing with piecewise linear production costs, European Journal of Operational Research, 256 (3) (2017) pp. 777-784.

[9] A. Federgruen, C. Lee, The dynamic lot size model with quantity discount, Naval Research Logistics (NRL), 37 (5) (1990) pp. 707-713.

[10] G. R. Bitran, H. H. Yanasse, Computational complexity of the capacitated lot size problem, Management Science, 28 (10) (1982) pp. 1174-1186.

[11] C. P. M. van Hoesel, A. P. M. Wagelmans, An $O(T^3)$ algorithm for the economic lot-sizing problem with constant capacities, Management Science, 42 (1) (1996) pp. 142-150.

[12] T. Liu, Economic lot sizing problem with inventory bounds, European Journal of Operational Research, 185 (1) (2008) pp. 204-215.

[13] M. Önal, W. van den Heuvel, T. Liu, A note on "The economic lot sizing problem with inventory bounds," European Journal of Operational Research, 223 (1) (2012) pp. 290-294.

[14] C. P. M. van Hoesel, A. P. M. Wagelmans, Fully polynomial approximation schemes for single-item capacitated economic lot-sizing problems, Mathematics of Operations Research, 26 (2) (2001) pp. 339-357.

[15] C. T. Ng, Mikhail Y. Kovalyov, T. C .E. Cheng, A simple FPTAS for a single-item capacitated economic lot-sizing problem with a monotone cost structure, European Journal of Operational Research, 200 (2) (2010) pp. 621-624.

[16] J. Fan, G. Wang, Joint optimization of dynamic lot and warehouse sizing problems, European Journal of Operational Research, 267 (3) (2018) pp. 849-854.

[17] M. R. Garey, D. S. Johnson, Computers and Intractability: A guide to the theory of NP-completeness, W. H. Freeman and Company, (1979).

Chapter 5

Conclusion

In this thesis, techniques from combinatorial optimization were leveraged to minimize different operational costs for data centers. In Chapter 2, a linearization technique was used to define and study a baseline thermal-aware workload distribution problem, in the form of a mixed integer linear programming problem, which considers different cooling parameters and heat recirculation effects while two different redline temperatures are defined for idle and fully-utilized servers to reduce the cooling effort. Linearization is a time-efficient approach that also helps develop more accurate heuristics by gaining insights about the structure of the problem. The results confirmed that linearization and the proposed heuristic to solve the resulting problem form an effective approach compared to baseline algorithms (simple rounding and a genetic algorithm) and commercial solvers (MATLAB). In the big picture, the proposed approach can be used to build a theoretical framework for thermal-aware workload distribution in data centers. In Chapter 3, a more practical scenario was

defined where additional operational costs, the cost of workload fluctuations that impact the reliability of servers (due to temperature changes) and QoS (due to imposing delay), were also considered. We generalized the proposed heuristic for the previous problem to solve this new problem. An MPC approach was also integrated to help reduce the size of the problem and use more precise demand predictions. The results confirmed that integrating the proposed heuristic with the MPC approach while using an appropriate window size efficiently solves the problem. The proposed problem in Chapter 4 is completely different but can also be related to operational costs in data centers. It is a single-item multi-period lot sizing problem with quantity discounts and bounded inventory. Three structural properties for an optimal solution were extracted and used to develop an efficient $O(T^2)$ dynamic programming algorithm to solve the problem, where T is the number of periods. The case of two items was proved to be NP-hard.

As explained, the thermal-aware workload distribution problem for data centers is not a trivial problem to solve. The thermal models are one of the most important challenges in solving such problems because they are not explicitly given and for models that use physical equations, they are also computationally expensive. When the trade-off between IT and cooling power consumption is considered, the decision variables include integral variables corresponding to the selection of thermal-efficient servers (computing nodes). The problem becomes more complicated for the case of heterogeneous data centers in which the servers have different characteristics in terms of power consumption and computational capacity. There are two possible holistic approaches to tackling such problems. Simplification of the problem and defining a model-independent problem through standard techniques such as linearization has

been discussed in this thesis. The other possibility is the definition of a general problem considering all the complexities and then finding a holistic approach to solve the problem. For data-driven thermal models, a potential holistic approach is the combination of machine learning with modern optimization approaches such as meta-heuristics. Because the optimization should be performed for online applications, the simplification approach may be more practical in terms of implementation and computational complexity. However, linearization may not be precise enough to represent more complicated models.

With this introduction we list some points, challenges and possible future works as follows:

- The proposed mixed integer linear programming problem defined in Chapter 2 is an interesting problem from both practical and theoretical views. There are interactions between cooling power consumption and workload distribution that should be investigated. In our efforts to approximate the solution, we tried to link the problem to well-known problems such as multi-dimensional knapsack problems and partial set-covering problems, and modify proposed algorithms in the literature that use a variety of methods such as classic methods and modern meta-heuristic methods (we used a related genetic algorithm in Chapter 2). However, there were some challenges. Specifically we tried to apply the method presented in [82] to our problem, which guarantees an approximation factor. Their problem is a partial set covering problem, where in contrast to our problem, all the variables are integral and the linear matrices are fixed (not

parametric). The approach has two parts. For the first part, a related Lagrangian relaxation problem should be solved. We planned to use a randomized rounding algorithm developed in [83] for this part. In the second part, there is a ranking procedure that turned out to be challenging for our problem. Another possible approach to approximate the solution is a primal-dual scheme. We defined the complementary slackness conditions to help decide which servers to choose. This was also challenging due to several complicated conditions. Designing an approximation algorithm for the defined mixed integer programming problem is still open to more efforts, in particular, we suggest the investigation of the Lagrangian relaxation method.

- Another approach to find an efficient algorithm is to define specific structures (that correspond to practice) for the cooling and/or heat-recirculation matrices which may make solving the problem easier. We did this in Chapter 2 by defining dominant cooling variables for each server. Additional helpful structures may be found.

- The basic problem defined in Chapter 2 (or Chapter 3) can be generalized to consider scenarios such as heterogeneous data centers and transient thermal models. Both scenarios may make the problem more complicated. For transient thermal models, the current temperature typically depends on the temperature in the previous state. These models are commonly linear, so they appear to be consistent with the linearization approach.

- We defined an inlet temperature for each server in the proposed problem. However in the literature the temperatures are commonly defined for abstract zones

or at the rack level. In this case the workload distribution problem is not a 0-1 optimization problem because each thermal node consists of several servers. The defined problems and proposed heuristics can be modified correspondingly.

- As we explained, the proposed problems in the literature are similar but are defined and solved differently. There is a lack of a holistic model-independent approach that can be used for different scenarios. We proposed a bottom-up approach which helps build a theoretical framework for thermal-aware workload distribution problems. We think holistic approaches should be considered as future perspectives. They can integrate the efforts in the literature to solve thermal-aware workload distribution problems.

- As explained, linearization may not represent the complicated models precisely (this was not the case in Chapter 2). More general structures such as convex functions or piecewise-linear functions may be valuable to study. For these cases, the relaxation of the problem can still be solved in polynomial time but for the rounding of the solution efficient algorithms should be developed.

- In the proposed problem the servers can be kept idle but server consolidation is also consistent with the proposed approach if the thermal model can still be represented by linear models. The challenge is that when some of the servers are turned off, the heat recirculation pattern changes accordingly. One possibility to minimize the changes is to keep the server fans on. Addressing this challenge is one of the more important future directions.

- For the lot-sizing problem, one possibility for future work is to consider other structures for the costs, for example a non-decreasing pattern in the prices

(which may correspond well with practice) instead of a non-increasing pattern. Another important possibility is to develop approximation algorithms for the case of multiple items, two items for example.

Bibliography (Chapters 1 and 5)

[1] Y. Cui, C. Ingalz, T. Gao, and A. Heydari, "Total cost of ownership model for data center technology evaluation," *16th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, Orlando, FL, USA, 2017, pp. 936-942.

[2] W. Zhang, Y. Wen, Y. Wah Wong, K. Chuan Toh, and C. Chen, "Towards joint optimization over ICT and cooling systems in data center: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1596-1616, third quarter 2016.

[3] C. Nadjahi, H. Louahlia, and S. Lemasson, "A review of thermal management and innovative cooling strategies for data center," *Sustainable Computing: Informatics and Systems*, vol. 19, pp. 14-28, 2018.

[4] E. Masanet, and N. Lei, "How much energy do data centers really use?," *Aspen global change Institute*, Mar. 2020.

[5] N. EI-Sayed, I. A. Stefanovici, G. Amvrosiadis, and A. A. Hwang, "Temperature management in data centers: Why some (might) like it hot," *SIGMETRICS*

'12: Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, Jun. 2012, pp. 163-174.

[6] Q. Fang, J. Wang, and Q. Gong, "QoS-driven power management of data centers via model predictive control," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1557-1566, Oct. 2016.

[7] A. H. Khalaj, and S. K. Halgamuge, "A Review on efficient thermal management of air- and liquid-cooled data centers: From chip to the cooling system," *Applied Energy*, vol. 205, pp. 1165-1188, 2017.

[8] D. Han, and T. Shu, "Thermal-aware energy-efficient task scheduling for DVFS-enabled data centers," *International Conference on Computing, Networking and Communications (ICNC)*, Garden Grove, CA, 2015, pp. 536-540.

[9] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed DVS scheduling for scientific applications on power-aware clusters," *SC '05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, Seattle, WA, USA, 2005, pp. 34-34.

[10] Q. Fang, J. Wang, Q. Gong, and M. Song, "Thermal-aware energy management of an HPC data center via two-time-scale control," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2260-2269, Oct. 2017.

[11] A. M. Al-Qawasmeh, S. Pasricha, A. A. Maciejewski, and H. J. Siegel, "Power and thermal-aware workload allocation in heterogeneous data centers," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 477-491, Feb. 2015.

[12] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732-794, First quarter 2016.

[13] L. Parolini, B. Sinopoli, and B. H. Krogh, "Reducing data center energy consumption via coordinated cooling and load management," *Proceedings of the 5th USENIX Symposium on Networked Systems Design & Implementations*, San Francisco, CA, April 2008, pp. 14-14.

[14] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta, "TACOMA: Server and workload management in internet data centers considering cooling-computing power trade-off and energy proportionality," *ACM Transactions on Architecture and Code Optimization*, vol. 9, no. 2, pp. 1-37, Jun. 2012.

[15] J. Wan, X. Gui, R. Zhang, and L. Fu, "Joint cooling and server control in data denters: a cross-layer framework for holistic energy minimization," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2461-2472, Sep. 2018.

[16] Z. Wang, C. Bash, N. Tolia, N. Marwah, X. Zhu, and P. Ranganathan, "Optimal fan speed control for thermal management of servers," *Proceedings of the ASME 2009 InterPACK Conference*, San Francisco, California, USA, Jul. 2009, vol. 2, pp. 709-719.

[17] S. Ilager, K. Ramamohanarao, and R. Buyya, "Thermal prediction for efficient energy management of clouds using machine learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1044-1056, May 2021.

[18] S. MirhoseiniNejad, F. M. García, G. Badawy, and D. G. Down, "ALTM: Adaptive learning-based thermal model for temperature predictions in data centers," *IEEE Sustainability through ICT Summit (StICT)*, Montréal, Canada, 2019, pp. 1-6.

[19] S. Asgari, H. Moazamigoodarzi, P. J. Tsai, S. Pal, R. Zheng, G. Badawy, and I. K. Puri, "Hybrid surrogate model for online temperature and pressure predictions in data centers," *Future Generation Computer Systems*, vol. 114, pp. 531-547, 2021.

[20] R. Gupta, S. Asgari, H. Moazamigoodarzi, D. G. Down, and I. K. Puri, "Energy, exergy and computing efficiency based data center workload and cooling management,"*Applied Energy*, vol. 299, 117050, Oct. 2021.

[21] S. Akbar, R. Li, M. Waqas, and A. Jan, "Server temperature prediction using deep neural networks to assist thermal-aware scheduling," *Sustainable Computing: Informatics and Systems*, vol. 36, 100809, 2022.

[22] Q. Tang, S. K. Gupta, and G. Varsamopoulos, "Thermal-aware task scheduling for data centers through minimizing heat recirculation," *IEEE International Conference on Cluster Computing*, Austin, TX, USA, 2007, pp. 129–138.

[23] Q. Tang, S. K. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1458–1472, 2008.

[24] T. Khan, W. Tian, G. Zhou, S. Ilager, M. Gong, and R. Buyya, "Machine learning (ML)-centric resource management in cloud computing: A review and future

directions," *Journal of Network and Computer Applications*, vol. 204, 103405, 2022.

[25] Z. Liu et al., "Renewable and cooling aware workload management for sustainable data centers," *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, 2012, pp. 175–186.

[26] E. Pakbaznia, M. Ghasemazar, and M. Pedram, "Temperature-aware dynamic resource provisioning in a power-optimized datacenter," *Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)* Dresden, Germany, 2010, pp. 124–129.

[27] A. Khan, X. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," *IEEE Network Operations and Management Symposium*, 2012, pp. 1287–1294.

[28] P. Bodık, R. Griffith, C. Sutton, A. Fox, M. Jordan, and D. Patterson, "Statistical machine learning makes automatic control practical for internet data centers," *HotCloud '09: Workshop on Hot Topics in Cloud Computing*, 2009, pp. 12–12.

[29] R. J. Hyndman, and G. Athanasopoulos, "Forecasting: Principles and practice (2nd ed)," *OTexts*, 2018.

[30] H. Sun, P. Stolf, J. Pierson, and G. D. Costa, "Energy-efficient and thermal-aware resource management for heterogeneous data centers," *Sustainable Computing: Informatics and Systems*, Vol. 4, no. 4, 2014, pp. 292-306.

[31] R. Azimi, X. Zhan, and S. Reda, "Thermal-aware layout planning for heterogeneous data centers," *Proceedings of the 2014 International Symposium on Low Power Electronics and Design*, 2014, pp. 245–250.

[32] X. Li, P. Garraghan, X. Jiang, Z. Wu, and J. Xu, "Holistic virtual machine scheduling in cloud data centers towards minimizing total energy," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 6, pp. 1317-1331, Jun. 2018.

[33] S. Ilager, K. Ramamohanarao, and R. Buyya, "Thermal prediction for efficient energy management of clouds using machine learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1044-1056, May 2021.

[34] E. K. Lee, H. Viswanathan, and D. Pompili, "Proactive thermal-aware resource management in virtualized HPC cloud data centers," *IEEE Transactions on Cloud Computing*, vol. 5, no. 2, pp. 234-248, Jun. 2017.

[35] R. Chen, B. Liu, W. Lin, J. Lin, H. Cheng, and K. Li, "Power and thermal-aware virtual machine scheduling optimization in cloud data center," *Future Generation Computer Systems*, Vol. 145, pp. 578-589, 2023.

[36] C. Guo, K. Xu, G. Shen, and M. Zukerman, "Temperature-aware virtual data center embedding to avoid hot spots in data centers," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 1, pp. 497-511, 2021.

[37] H. Feng, Y. Deng, and Y. Zhou, "A heat-recirculation-aware VM placement strategy for data centers," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, France, 2020, pp. 626-629.

[38] H. Liu, W. K. Wong, S. Ye, and C. Yu Tak Ma, "Joint energy optimization of cooling systems and virtual machine consolidation in data centers," *29th International Conference on Computer Communications and Networks (ICCCN)*, Honolulu, HI, USA, 2020, pp. 1-8.

[39] S. MirhoseiniNejad, G. Badawy, and D. G. Down, "EAWA: Energy-aware workload assignment in data centers," *2018 International Conference on High Performance Computing & Simulation (HPCS)*, pp. 260–267, 2018.

[40] A. D. Carnerero, D. R. Ramirez, T. Alamo, and D. Limon, "Probabilistically certified management of data centers using predictive control," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2849-2861, Oct. 2022.

[41] Y. Cho, and N. Chang, "Energy-aware clock-frequency assignment in microprocessors and memory devices for dynamic voltage scaling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 6, pp. 1030– 1040, Jun. 2007.

[42] T. Mukherjee, A. Banerjee, G. Varsamopoulos, S. K. Gupta, and S. Rungta, "Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers," *Computer Networks*, vol. 53, no. 17, pp. 2888–2904, 2009.

[43] B. Shi, and A. Srivastava, "Thermal and power-aware task scheduling for hadoop based storage centric data centers," *International Conference on Green Computing*, Chicago, IL, 2010, pp. 73–83.

[44] M. Nakamura, "Learning and optimization models for energy efficient cooling

control in data center," *Proceedings of the SICE Annual Conference*, Tsukuba, Japan, Sep. 2016.

[45] L. Parolini, B. Sinopoli, B. H. Krogh, and Z. Wang, "A Cyber–physical systems approach to data center modeling and control for energy efficiency," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 254-268, Jan. 2012.

[46] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1458-1472, Nov. 2008.

[47] L. Yang, Y. Deng, L. T. Yang, and R. Lin, "Reducing the cooling power of data centers by intelligently assigning tasks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1667-1678, June 2018.

[48] E. Pakbaznia, and M. Pedram, "Minimizing data center cooling and server power costs," *In Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design (ISLPED '09). Association for Computing Machinery*, New York, USA, 2009, pp. 145–150.

[49] T. Van Damme, C. De Persis, and P. Tesi, "Optimized thermal-aware job scheduling and control of data centers," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 760-771, Mar. 2019.

[50] W. Zhang, R. Yadav, Y. C. Tian, S. K. S. Tyagi, I. A. Elgendy, and O. Kaiwartya, "Two-phase industrial manufacturing service management for energy efficiency of

data centers," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 11, pp. 7525-7536, Nov. 2022.

[51] S. Mirhoseininejad, G. Badawy, and D. G. Down, "A data-driven, multi-setpoint model predictive thermal control system for data centers," *Journal of Network and Systems Management*, vol. 29, no. 7, 2021.

[52] Z. Xiong, M Zhao, Z. Yuan, J. Xu, and L. Cai, "Energy-saving optimization of application server clusters based on mixed integer linear programming," *Journal of Parallel and Distributed Computing*, vol. 171, pp. 111-129, 2023.

[53] I. Castiñeiras, D. S. Chisca, D. Mehta, and B. O'Sullivan, "Trichotomic search for thermal-aware data centre workload optimisation," *IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, Limassol, Cyprus, 2015, pp. 528-533.

[54] Q. Wang, Y. Yu, B. Li, and Y. Zhu, "Tensor-based optimal temperature control of CRACs in multi-data centers," *IEEE Access*, vol. 7, pp. 41445-41453, 2019.

[55] J. Li, Y. Deng, Y. Zhou, Z. Zhang, G. Min, and X. Qin, "Towards thermal-aware workload distribution in cloud data centers based on failure models," *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 586-599, Feb. 2023.

[56] S. Akbar, S. U. R. Malik, K. K. R. Choo, S. U. Khan, N. Ahmad, and A. Anjum, "A Game-based thermal-aware resource allocation strategy for data centers," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 845-853, Sep. 2021.

[57] A. Sansottera, and P. Cremonesi, "Cooling-aware workload placement with performance constraints," *Performance Evaluation*, vol. 68, no. 11, pp. 1232-1246, 2011.

[58] B. Shi, and A. Srivastava, "Unified data center power management considering on-chip and air temperature constraints," *Sustainable Computing: Informatics and Systems*, vol. 1, no. 2, pp. 91–98, 2011.

[59] L. Parolini, B. Sinopoli, and B.H. Krogh, "Reducing data center energy consumption via coordinated cooling and load management," *Proceedings of the 2008 conference on power aware computing and systems, HotPower*, vol. 8, pp. 14, 2008.

[60] A.H. Khalaj, T. Scherer, J. Siriwardana, and S.K. Halgamuge, "Multi-objective efficiency enhancement using workload spreading in an operational data center," *Applied Energy*, vol. 138, pp. 432–44, 2015.

[61] Y. Ran, H. Hu, X. Zhou, and Y. Wen, "DeepEE: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning," *IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, 2019, pp. 645-655.

[62] S. MirhoseiniNejad, G. Badawy, and D. G. Down, "Holistic thermal-aware workload management and infrastructure control for heterogeneous data centers using machine learning," *Future Generation Computer Systems*, vol. 118, pp. 208-218, 2021.

[63] N. Brahimi, N. Absi, S. Dauzère-Pérès, and A. Nordli, "Single-item dynamic lot-sizing problems: An updated survey," *European Journal of Operational Research*, vol. 263, no. 3, pp. 838-863, 2017.

[64] H. M. Wagner, and T. M. Whitin, "Dynamic version of the economic lot size model," *Management Science*, vol. 5, no. 1, pp. 89-96, 1958.

[65] N. Brahimi, N. Absi, S. Dauzère-Pérès, and A. Nordli, "Single item lot sizing problems," *European Journal of Operational Research*, vol. 168, no. 1, pp. 1–16, 2006.

[66] Nusrat T. Chowdhury, M.F. Baki, and A. Azab, "Dynamic economic lot-sizing problem: A new O(T) algorithm for the Wagner-Whitin model," *Computers & Industrial Engineering*, vol. 117, pp. 6-18, 2018.

[67] G. R. Bitran, and H. H. Yanasse, "Computational complexity of the capacitated lot size problem," *Management Science* , vol. 28, no. 10, pp. 1174-1186, 1982.

[68] S. F. Love, "Bounded production and inventory models with piecewise concave costs," *Management Science*, vol. 20, no. 3, pp. 313-318, 1973.

[69] M. Önal, W. Van den Heuvel, and T. Liu, "A note on "The economic lot sizing problem with inventory bounds"," *European Journal of Operational Research*, vol. 223, no. 1, pp. 290–294, 2012.

[70] A. Aggarwal, and J. K. Park, "Improved algorithms for economic lot size problems," *Operations Research*, vol. 41, no. 3, pp. 549-571, 1993.

[71] W. I. Zangwill, "The piecewise concave function," *Management Science*, vol. 13, no. 11, pp. 900-912, 1967.

[72] C. Swoveland, "A deterministic multi-period production planning model with piecewise concave production and holding-backorder costs," *Management Science*, vol. 21, no. 9, pp. 1007-1013, 1975.

[73] M. Florian, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Deterministic production planning: algorithms and complexity," *Management Science*, vol. 26, no. 7, pp. 669-679, 1980.

[74] A. Federgruen, and C. Y. Lee, "The dynamic lot size model with quantity discount," *Naval Research Logistics*, vol. 37, no. 5, pp. 707-713, 1980.

[75] C. P. M. van Hoesel, and A. P. M. Wagelmans, "Fully polynomial approximation schemes for single-item capacitated economic lot-sizing problems," *Mathematics of Operations Research*, vol. 26, no. 2, pp. 339-357, 2001.

[76] E. Koca, H. Yaman, and M. S. Aktürk, "Lot sizing with piecewise concave production costs," *INFORMS Journal on Computing*, vol. 26, no. 4, pp. 767-779, 2014.

[77] J. Ou, "Improved exact algorithms to economic lot-sizing with piecewise linear production costs," *European Journal of Operational Research*, vol. 256, no. 3, pp. 777-784, 2017.

[78] J. Ou, "A polynomial time algorithm to the economic lot sizing problem with constant capacity and piecewise linear concave costs," *Operations Research Letters*, vol. 45, no. 5, pp. 493-497, 2017.

[79] Y. Malekian, S. H. Mirmohammadi, and M. Bijari, "Polynomial-time algorithms

to solve the single-item capacitated lot sizing problem with a 1-breakpoint all-units quantity discount", *Computers & Operations Research*, vol. 134, 105373, 2021.

[80] K. A. Bunn, and J. A. Ventura, "A dynamic programming approach for the two-product capacitated lot-sizing problem with concave costs," *European Journal of Operational Research*, vol. 307, no. 1, pp. 116-129, 2023.

[81] H. B. Ammar, O. Ayadi, and F. Masmoudi, "An effective multi-objective particle swarm optimization for the multi-item capacitated lot-sizing problem with set-up times and backlogging," *Engineering Optimization*, vol. 52, no. 7, pp. 1198-1224, 2020.

[82] J. Könemann, O. Parekh, and D. Segev, "A unified approach to approximating partial covering problems," *Algorithmica*, vol. 59, pp. 489–509, 2011.

[83] S. G. Kolliopoulos, and N. E. Young, "Tight approximation results for general covering integer programs," *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, Newport Beach, CA, USA, 2001, pp. 522-528.