AUTOMATED STRUCTURAL OPTIMIZATION USING THE FINITE ELEMENT METHOD AND AN EXPERT SYSTEM

by

ZHANG WU, B.Eng., M.Eng

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

August 1987

AUTOMATED STRUCTURAL OPTIMIZATION USING THE FINITE ELEMENT METHOD AND AN EXPERT SYSTEM

DOCTOR OF PHILOSOPHY (1987) McMASTER UNIVERSITY
(Mechanical Engineering) Hamilton, Ontario

TITLE: Automated Structural Optimization Using the Finite Element Method and an Expert System

AUTHOR: Zhang Wu, B.Eng. (Huazhong University of Science and Technology)

M.Eng. (McMaster University)

SUPERVISOR: Professor J. N. Siddall

NUMBER OF PAGES: xx, 354

ABSTRACT

This thesis describes the development of a software system, which integrates the optimization, finite element, expert system and computer aided design techniques to achieve a highly automatic structural design.

New knowledge has been developed for the theory of expert systems, including an knowledge acquisition approach by linear programming and machine learning.

Several algorithms have been developed to increase the efficiency of the finite element based optimization.

ACKNOWLEDGEMENTS

The author wishes to express his sincere gratitude and appreciation to his supervisor Professor J. N. Siddall for his guidance and encouragement throughout the course of this study. He also thanks Professor W. R. Newcombe, Dr. R. L. Judd and Dr. F. Murza, members of his supervisory committee for their continuing interest.

The University and the Department of Mechanical Engineering financial support through the scholarship, bursary and Teaching Assistantship is gratefully acknowledged.

TABLE OF CONTENTS

NOTATION

A	area of the structure
A_i	the value of design variable on a safe or fail line
a_{ij}	parameters of the quadratic polynominal in quadratic method
B	1) strain-displacement matrix in finite element calculation 2) initial value of Q
BL	lower bound of design variable
BU	upper bound of design variable
C	elasticity matrix
C_j	jth order moment of the desirability values or importance values
CM_i	curve merit value of the ith candidate
D	displacement vector of a node
d	stress difference of FEM node
D^i	desirability value of group i
D_{ij}	desirability value of the ith candidate to the jth performance characteristic
d_i	difference of the curve characteristics between the ith candidate and the input
E	modulus of elasticity
e	a predetermined small positive number
g_i	inequality constraint function
h_i	equality constraint function
I	moment of inertia
K	1) stiffness matrix 2) number of performance characteristics

ℓ	length of a column
M_c	the merit value of the most hopeful candidate
M_d	merit value of the desired candidate
M_i	merit value of the ith candidate
m	number of parameters in quadratic method
N	1) number of design variables 2) number of candidates
n	1) factor of safety 2) index of end conditions of column 3) number of critical performance characteristics
NBS	numbers of bisections in global direction method
NC	number of inequality constraint functions in linear programming
ND	number of candidates in the desired group
NE	number of FEM elements in a substructure
NM	number of multiplications in full search method
NP	number of FEM nodes in a substructure
NSA	number of examples in the adapt sample
NV	total number of variables in linear programming
P	number of inequality constraints
PHI	array of constraint functions
P_j	importance value of the jth performance characteristic
Q	importance value reducing factor
q	displacement vector of an FEM element
R	1) load vector 2) function of the higher order moments of the desirability value or importance value
r	aspect ratio of FEM mesh

S	1) strength 2) scale factor for functions
SUM_i	sum of the desirability values of the ith candidate
t	1) thickness of a column 2) consistence value
U	1) optimization function 2) displacement vector
u,v	normalized coordinates of a region
V_i	different of the complete merit values between the most hopeful and the ith candidates
w	width of a column
X_i	design variable
$\overline{X}_o$	constraint optimum in global direction method
$\overline{X}_s$	original starting point in global direction method
$\overline{X}_t$	bisecting point in global direction method
$\overline{X}_u$	unconstraint optimum in global direction method
XO	original design variable
XS	scaled design variable
ϵ	strain
μ	means of the desirability values or importance values
ν	Poisson's ratio
σ	1) stress 2) standard deviation of the desirability value or importance value

LISTING OF TABLES

LISTING OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 THE OBJECTIVE OF THE RESEARCH

The primary objective of this research program is to develop new techniques for applying structural optimization to design, in which the structural analysis technique is the finite element method. It is important from the point of view of the designer's productivity and effectiveness that the design process be automated as much as possible, using the computer. This means that the interface between a complex tool like structural optimization and the designer is an important candidate for this automation. Using current technology, the designer must be both an expert in the art of optimization, and an expert in the finite element method. And even if this rare combination exists, it is a complex and time consuming procedure to properly formulate and execute a problem. However, not utilizing these sophisticated software systems will be an enormous waste of our intellectual resources. A promising technique for automating this interface is the use of expert systems, one of the new methods used in applied artificial intelligence. The goal of the program will be to develop an automatic system that will make it feasible for a practicing designer with limited experience and expertise in both optimization

and structural analysis to routinely do applications in structural optimization. It is also our goal to develop new knowledge in the area of applying artificial intelligence methods to design automation in general.

The project will thus have four main thrusts.

1. To develop a super-expert system (expert system shell) in order to establish expert systems for the design of different kinds of structures.

2. To utilize any of such established expert systems to design a structure.

3. To explore new knowledge in the area of interactive, graphics oriented and user friendly programming for structural optimization in the sense of the overall value.

4. To develop new algorithms for saving computer time in finite element based optimization.

The present system is limited to the preliminary designs and to the 2-dimensional problems only, however the basic ideas about the expert system can be applied to 3-dimensional problems directly.

There is an important additional long term goal for this work. A major world problem is that countries with limited broad technological development are tending to fall further behind those countries with a very highly developed technological infrastructure. This infrastructure enables these countries to develop new products, new technologies,

and greater productivity at an accelerating rate. One of the important components of this infrastructure is the extensive engineering educational system and the large pool of experienced experts in the area of design and associated fields. Methods must be found that bypass this constraint on the progress of less developed countries, and applied artificial intelligence in design automation is one promising possible approach, in which the computer would provide the mass expertise required. Structural optimization is just one example of this expertise; and it is hoped that the knowledge developed in this research would be generally applicable to other areas of expertise in design.

1.2 LITERATURE SURVEY

1.2.1 Expert System

Over the past 20 or 30 years, a lot of effort has been devoted to research in AI (artificial intelligence). Scientists doing fundamental research in AI attempt to set up algorithmic principles for simulating thinking processes of the brain, to develop some sort of general problem solving or human-like intellectual behaviour.

However it seems to be generally conceded that not too much progress has been made in fundamental research into AI. On the other hand, rather strong claims are now being made for the progress and utility of applied research in AI,

particularly in the area of expert systems.

Barr and Feigenbaum (1982) described a valid expert system as follows.

> "Most of the applications systems described ... can be viewed as consultants that formulate opinions and give advice to their users. The tasks these consultants are designed to perform require the application of facts and relationships known only by a specialist. The current systems emphasize the cognitive abilities that support interaction with the user during problem solving, such as the ability to explain lines of reasoning or to acquire new domain knowledge interactively.
>
> Typically, such a system will be considered 'intelligent' if it meets the following criteria: (a) The system gives correct answers or useful advice, and (b) the concept and reasoning processes it uses to solve the problem resemble those that the user might employ."

An expert system consists of a knowledge database and an inference engine. The knowledge database mainly contains a lot of logic rules extracted from the expertise of the human expert. The inference engine is the control algorithm used to decide the search path.

A distinctive feature of the expert systems is that it includes not only knowledge that can be found in books, but also privileged knowledge known only to highly trained and experienced experts in a field. Such knowledge is

primarily intuitive. In a given situation an expert makes an intuitive decision based on judgement and experience and the context of the problem.

Another feature of expert system is that the scope of the problem domain, usually small and specific, must be clearly defined. In case the need is very broad in nature, the system should be decomposed into subsystems (Naylor, 1983).

Different approaches have been developed to deal with the problem of uncertainty, based on Bayes' theorem, fuzzy set theory, or a kind of pseudo probability.

As in the other areas in AI, expert systems adopt the conventional search algorithms, e.g. forward chaining or backward chaining, breadth first search or depth first search, combined with intuitive decision making. When the data base is large, searching takes a lot of computer time. Some algorithms have been studied to accelerate the search by discarding some weaker candidates earlier (Shirai 1985).

An expert system generally works on one of the following types of tasks: interpretation, diagnosis, prediction, instruction, monitoring, planning, and design.

Siddall (1984) predicted that several factors would be good applications of expert systems in engineering design.

1. The expert systems will in some sense automate the design process, make it more efficient, by reducing

engineering cost or engineering leading time. The expert system can assist the designer in setting up an engineering model, synthesis, aesthetics and risk judgement.

2. The expert systems can help to increase the overall value of the engineering design to society. It is meant to include all of the basic, and some what abstract, desires of mankind into the design by combining judgement with the immediate criteria.

3. Expert systems can be incorporated with sophisticated software packages, and thus provide the necessary expertise for use by an engineer or technologist who is not specifically familiar with the theory used, nor has a strong experience for judgemental aspects of the software.

The same author suggested possible examples of applying expert systems in mechanical design, including material selection and configuration selection. He gave a course project (1982) of a simplified version of selecting configuration. Hanley (1980) described a computer program for selecting material based on numerical evaluation.

The expert design system that is most frequently mentioned and most successfully in active use is the program used by Digital Equipment Corporation to select an appropriate configuration for their VAX computer, in order to meet a specific customer's requirements (McDermott, 1981). This would be considered applying expertise to

design synthesis. Also in the area of synthesis, a paper by Lenat, et al (1982) described a method of finding new microcircuit structures. Closely related to this is a paper by Greenberg (1980) on a knowledge-based system for digital electronics, and a paper by Mitchell, et al (1983) on circuit redesign. Roswnmcm (1986) built an expert system in CAD with the help of a shell. Maher (1984) in his paper discussed the tools and techniques for a knowledge based expert system for engineering design.

There are very few concrete examples of expert design systems in mechanical engineering. Bennet and Engelmore (1979) have described an expert system for structural analysis. And Bundy et al (1979) wrote an expert system program for solving mechanics problems. A research group led by Dixon and Simmon (1985) has been developing expert system project in mechanical design. Their published papers examine the concept of expert systems and how it might be applied to mechanical design. They are working on a system that performs a design "...through evaluation and redesign". It appears to be a synthesis process for selecting candidate configurations. A major research issue has been the development of a configuration data base composed of many feature-primitives. They mentioned as examples, V-belts, extrusions, casting design, plastic material selection. Recently, Jones (1986) reported an expert system for selecting welding materials.

Some other authors have tried to apply expert system in optimization. Azarm and Papalambros (1984) have suggested a way of using symbolic reasoning to assist the numerical search strategy and using the knowledge base to determine active constraints. In another paper by Li and Papalambros (1985), a production system was described for use of global optimization knowledge instead of exploring the search only by local topography. Arora et al (1986) presented some arguments suggesting that AI can be used to determine the optimization parameters, in order to assure a successful run. As the authors conceded, it is just "the first step toward development of an expert optimization system..."

Up to date, the building of an expert system for a particular application is still a challenging, difficult and time consuming task. It requires the close cooperation of a knowledge engineer and a human expert in the field. Exploring the expertise, finding out the rules, converting the expertise to a data base, adjusting a great number of parameters, all take a great deal of time, half to several years.

The primary computer languages of AI are LISP and PROLOG, which are designed particularly to work with word symbols rather than numerical calculations and analytical algorithms, and with which most of the engineers are not familiar and find difficult to use for programming.

Some effort has been made to break these tight bonds in order to encourage the more widespread use of AI applications in engineering.

Some researchers have developed the tools called expert system shells, which support one knowledge representation formalism and one inference mechanism (Goodall, 1985). Boose (1986) described a Expert Transfer System (ETS) which is a framework for knowledge elicitation, analysis and testing. Thompson et al (1986) made a program which reduces redundancy in the knowledge base, organizes the data to recognize pattern, and produces a set of rules that can then be manipulated by an expert system shell. However, the utility of the shells have never been well established and none of them can replace the knowledge engineer completely.

Recently, more and more expert systems have been developed in other high level languages, such as PASCAL, FORTRAN and even BASIC (Thompson 1986, Frey 1986, Schrodt 1986). It is reported that all of them perform very well.

Considerable research work is currently being done on expert systems incorporating machine learning (Michalski 1983 and 1986, Naylor 1983, Forsyth 1984, Crawford 1987), in which the computer systems learn to perform a task through examples or by analogy to a similar, previously solved task, or they can improve significantly on the basis of past

mistakes or acquire new abilities by observing and imitating experts.

Much investigation has been done on learning from examples (Naylor 1983, Terheyden 1987). Given a set of examples of a concept, a general concept description can be induced that describes all of the examples. Naylor described in detail the building of an expert system by learning from examples. Simple formulas are used to adapt the rules (parameters) from examples. However for larger problem, there is no guarantee of success. In Naylor's demonstration expert system for predicting weather, where there are four items of evidence and two outputs (dry or rain), the percentage of correct prediction is 75%.

Another widely used learning strategy is learning by memorization (Schrodt 1986), in which the programs record and process the information obtained during the procedure, and use them to update the system data base or guide the subsequent procedure dynamically.

1.2.2 Integration of CAD, Optimization and FEM

Over the last several decades, great achievements in three areas, CAD (computer-aided design), optimization and FEM (finite element method), have been made. It is now time to integrate them in order to increase design efficiency and improve design quality as a whole. There are several key problems that must be solved before this comes to reality.

1. How to develop an overall database to manage the data uniformly through the whole procedure.

2. How to determine or create the design parameters such as material and configuration in CAD and, more important, how to adapt them to the model appropriate for optimization design and FEM analysis?

3. How to reduce the computer time when each optimization iteration includes a time-consuming FEM analysis?

4. How to generate the FEM mesh automatically with minimum interface with the user, and, more important, how to adjust or regenerate the mesh automatically during the optimization search?

5. How to monitor the interface between such a large and complicated software package and the user, and assist him in making a lot of design decisions? Here an expert system should be a competent candidate.

Recently more and more authors have discussed the different aspects in integrating CAD, optimization and FEM. However no program has been found that combines these three techniques in a unified system. Dixon and Simmons (1985) are developing a system that integrates geometry model building (by solid model), manufacturable evaluation (only for casting currently) and analysis (by FEM). Lambourne (1986) discussed the necessity and possibility of integrating CAD, CAM and production management.

Many authors (Sabourin 1986, Fulton 1987, Gloudeman

1987) have reported the development of database management for integrated CAD. A large project called IPAD is under development in the USA.

Most of the CAD systems use solid models to constitute structural configurations (Requicha 1983, Tan 1986, Tang 1987). However for 2-dimensional configurations, interactive creation of the graphic primitives (like AutoCAD) may be more convenient and flexible. The problem remaining is how to transfer the data created in this stage to the optimization and FEM programs. Revelli (1985) reported his approach in getting CAD and FEM to cooperate, in which he linked an existing computer-aided drafting program with a mesh generator for FEM.

In the sixties and early seventies, a lot of research efforts were put into the study of structural reanalysis (Fox 1971, Kavlie 1971, Kirsch 1972). The common algorithms can be divided into following types.

1. Reduced basis method including reducing the degrees of freedom, number of design variables or number of constraint functions.

2. Iteration method including some algorithms to improve the convergence.

3. Taylor series expansion including first and second order approximations.

4. Substructure method.

Other authors (Arora 1976, Lee 1987) evaluated the

efficiency and accuracy of different algorithms. Structural reanalysis closely relates to FEM based optimization. However the above authors studied the problem basically on the matrix algebra level; it was not program-oriented, and they seldom implemented the algorithms for this purpose.

Morris (1982) in his text book discussed in detail the theory of structural optimization, where the FEM is used for the analysis process. Hornbuckle (1975), Cook (1977), Bennet (1979), Queau (1980), Silva (1985), Thevendran (1986) gave examples in plane, beam and truss and axisymmetric structural designs. However most authors seem to be unaware of the previous work in structure reanalysis and pay little attention to increasing the efficiency of programs.

Later, Grandhi et al (1985) discussed a program using approximate formulation (Taylor expansion) to present complicated constraints or objective functions. Siddall and Wu Zhang (1986) suggested several algorithms to save the computer time in FEM based optimization.

Shephard et al (1986) and Kela et al (1986) discussed the indispensable prerequisites of automated FEM modeling for the unification of engineering design and analysis, i.e, fully automated mesh generation and adaptive error estimation. Since the error analyses processes proposed by other authors are very difficult to implement, Shephard suggested using expert systems.

Nowadays a lot of mesh generators are available in the market (Boubez 1986). A few of them (Rusnock et al 1985) appear to approach being fully automated. The majority require that the structure to be analysed be decomposed into appropriate regions first. However all of the mesh generators have to interface with the user more or less.

Zienkiewicz (1987) and Jiang (1987) described some error estimators and mesh refinement algorithms for FEM analysis. An alternate algorithm suggested by Schiemeier (1987) for increasing the accuracy of analysis is to increase the degrees of shape function instead of increasing the number of elements.

Cuthill (1969), Akhra (1976) and other authors have developed algorithms for minimizing the bandwidth of the stiffness matrix in FEM analysis.

1.3 THE LAYOUT OF THE PROJECT

Figure 1.1 is the layout of the project. It actually consists of two programs: **SES** (Super-Expert System) for building the knowledge databases of the expert systems, and **ESSF** (Expert System of a Structural Family) for utilizing the expert systems.

Although symbolically oriented LISP and PROLOG are the dominant computer languages in artificial intelligence, the computer language used in this project is FORTRAN. It

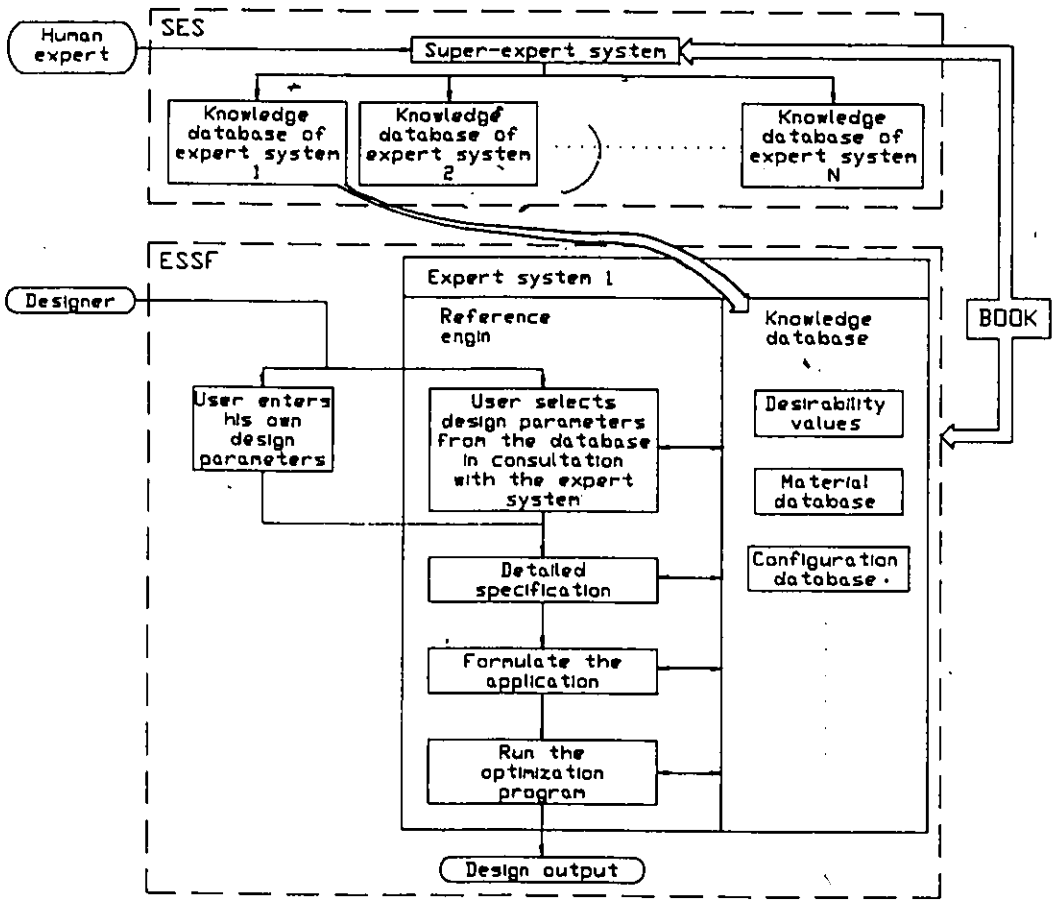


Figure 1.1 Layout of the project

is because this project involves a lot of numerical calculations; even the main inference engine of the expert system is numerically oriented. The coding by FORTRAN may not be as elegant, or as compact, or as efficient. This does not seem to have been demonstrated as necessarily true, but in any event these may not be primary concerns. The major considerations are ease of reading, ease of maintaining or updating, portability, familiarity of the programmer with the language, and time to code. We believe the more widespread use of artificial intelligence applications in engineering may in fact be encouraged if a more familiar language is used.

1.3.1 Building the Knowledge Databases of the Expert Systems

SES is similar to an expert system shell, i.e, a tool for building expert systems with similar knowledge representation formalisms and inference mechanisms. The user of **SES** is the human expert in the field of designing a certain kind of structure. The human expert interacts with **SES** through a cathode ray tube (CRT in short) terminal to build the knowledge database of his expert system for the design of that kind of structures. **SES** itself is an expert system at the same time, since it contains a lot of knowledge for the design of general structures. Such general knowledge can assist the human expert's building the knowledge database of his expert system in a more specific

domain.

As a convention in this thesis, the term expert system always refer to that one which is built by the human expert and which, in a more specific domain, aids the design of certain kind of structures.

An expert system must contain an inference engine and a knowledge database. Unlike many existing expert systems, our main inference engine is to evaluate the merit values of the candidates as follows.

$$M_i = \sum_{j=1}^{k} (P_j * D_{i,j}) \qquad i=1,2,...,N \qquad (1.1)$$

where M_i ---- merit value of the ith candidate

P_j ---- importance value of the jth performance characteristic

$D_{i,j}$ -- desirability value of the ith candidate to the jth performance characteristic

N ----- number of candidates

K ----- number of performance characteristics

The candidate (e.g. material, configuration, etc.) with the highest merit value is considered to be the best. The importance values designate the relative importance of the performance characteristics (e.g. cost, weight, accuracy) in the design. Desirability values are used to evaluate the goodness of the design candidates with regard to a performance characteristic. For example, plain carbon steel is a cheaper material candidate, so its desirability

value with regard to cost is high.

Desirability values represent the most important expertise in the knowledge databases of our expert systems. A new algorithm has been developed for acquiring these desirability values systematically using machine learning based on a sample of examples and linear optimization programming. Previous design examples can be used in this procedure. In case no existing examples are available, a facility in the super-expert system can help the human expert or a group of experts to create the sample of examples.

Besides the desirability values, the knowledge database also contains the material database, configuration database, knowledge about the selection of the failure mode, the factor of safety, the optimization methods, the analysis algorithms based on the finite element method, and so on. The material database is a permanent database residing in the super-expert system. The configuration database has to be created by the human expert, because the configuration databases are quite different for different kinds of structures. The design variables, optimization function and constraint functions should be defined at the same time relating to the configurations.

Chapter 2 describes the building of the expert system in detail.

1.3.2 Utilizing the Expert Systems

When an expert system has been built, the designer can utilize it to do optimization design by executing the program **ESSF** (Expert System of a Structural Family) at a CRT terminal.

First of all, the designer enters the requirements of the design, i.e. the importance values of the performance characteristics. The expert system will search the database, evaluate the goodness of the material candidates or configuration candidates there, and return them to the designer. Three search methods have been developed: the Full Search method, the Discard Search method and the Quick Search method. All of them are based on formula (1.1), but have different accuracy and speed in search.

The designer also has to determine some other design parameters, such as the failure mode and factor of safety, in consultation with the expert system.

In an alternative mode, if the designer wants to create a completely new design, he can bypass the expert system and enter all of the design parameters directly.

When all of the design parameters have been determined, the designer is asked to give detailed specifications for the design, such as concrete dimensions of the configuration, and upper and lower bounds of the design variables.

In each stage mentioned above, the program fetches

data and knowledge from the knowledge database of the expert system. On return, it pushes some new information back to update the knowledge database. Then available data will be converted by the expert system to constitute the user written subroutines and data files for the existing optimization and finite element method (FEM in short) packages.

The optimization search begins! The configuration of the structure at the current design point and the topography of the optimization search will be displayed on the screen alternately. The designer has full freedom to detect or adjust the design point at any time. The expert system continually adjusts or regenerates the FEM mesh in order to avoid severe distortion.

Since the FEM based optimization is very time consuming, six algorithms have been adopted to improve it, i.e. the Substructure method, the Skipping method, the Global Direction method, the Safe-Fail Line method, the Quadratic method and the Differential method. The latter four are new algorithms developed in this project.

When the design is finally completed, a set of hardcopies of the solution are plotted at the designer's request.

Chapters 4 and 5 describe how to utilize the expert system in detail.

1.3.3 Book

Book is a manual for explanation which is available to both the super-expert system and the expert systems.

CHAPTER 2

THE EXPERT SYSTEM APPROACH

2.1 THE TWO-LEVEL EXPERT SYSTEM

2.1.1 The Top Level Expert System --- Super-Expert System

There are an enormous number of different structures in this world; some are quite different; some are similar. We can catalogue them into many classes (kinds). Our objectives in this project is to develop an expert system which is able to design all kinds of structures.

Unfortunately, it is well known that any realistic expert system must be limited to a specific domain of expertise. Should we develop a lot of expert systems independently and in isolation, each for the design of a certain kind of structures? Our answer is no. Because this would cost too much in intellectual resources and engineering lead time.

A challenging design always requires three kinds of knowledge.

1. Common knowledge: it can be applied to the design of almost all kinds of structures without modification.

2. General knowledge: it can be applied to the design of most kinds of structures. Some modification may, or may not be required when applying the general knowledge

to a specific structure.

3. Specific knowledge: it is the core of an expert system, but only suitable for a certain kind of structure.

An example of common knowledge is the material database. An example of specific knowledge is the configuration database.

We first create a super-expert system whose knowledge database contains the common knowledge and general knowledge for structural optimization of a design. Then following the three steps below, the human expert can build the knowledge database for the expert system for the design of a specific kind of structure.

1. The common knowledge is copied from the knowledge database of the super-expert system to the knowledge database of the expert system. This frees the human expert from dealing with the common knowledge.

2. The general knowledge is fetched from the knowledge database of the super-expert system, modifications are made if necessary, and then it is pushed into the knowledge database of the expert system.

3. The specific knowledge is created, and added to the knowledge database of the expert system.

The human expert has the full freedom to accept, ignore or revise the common knowledge and general knowledge, and to add the specific knowledge.

So the super-expert system provides a useful

foundation to accelerate the creation of the knowledge database of each individual expert system, and at the same time retains the distinct features of the expert system, i.e. privileged expert knowledge for a specific problem to be solved.

The basic function of the super-expert system is similar to that of an expert system shell. Both are tools for assisting in building expert systems. With the aid of the super-expert system, the human expert need not cooperate closely with a knowledge engineer and need not know anything about programming, such as manipulating databases, use of data structures, or even the computer language. In the building of an expert system, he has to enter his expertise by interacting with the super-expert system at a CRT terminal. The super-expert system manipulates and converts his expertise and pushes it into the knowledge database of the expert system to be built.

The super-expert system is different from an expert system shell in two points.

1. The super-expert system itself is an expert system. It has its own knowledge database containing the common knowledge and the general knowledge. And also, during the interface with the human expert, the super-expert system can provide a lot of advice, guidance and suggestion.

2. An expert system shell aids in the building of the complete expert system, including the inference

engine and knowledge database. The super-expert system only helps build the knowledge databases of an expert system.

The super-expert system is an expert system that builds the knowledge databases of the low level expert systems.

The detailed structure of the super-expert system will be described in sections 2.2 and 2.3.

2.1.2 The Low Level Expert System

The low level expert system is the expert system that we generally refer to in this thesis. It assists in the designing of certain kind of structures.

While the knowledge databases of two expert systems are rather different, their inference engines are very similar. The main mechanism is the evaluation of the merit values of the design candidates by formula (1.1). So we only need to create one inference engine, and combine it with any knowledge database to obtain a complete expert system.

Now let us look at an example. If a company wants to design an expert system which is able to design ten kinds of structures, i.e. a gas chamber family, liquid tank family,.... What should we do?

1. Develop the super-expert system.

2. Develop the inference engine that is common to all of the expert systems.

3. Find ten human experts, each in the area of the design of a certain kind of structure.

4. Each human expert builds the knowledge database of an expert system by interacting with the super-expert system.

Thus we obtain the knowledge databases of ten expert systems. Then, if a non-expert designer in the company wants to design a gas chamber, what should he do?

5. Load the inference engine of the expert systems in the computer.

6. Load the knowledge database of the gas chamber family.

7. Run the expert system for the design of the gas chamber family, built in 5, 6.

Next, if another non-expert designer wants to design a liquid tank, what should he do?

8. Unload the knowledge database of the gas chamber family.

9. Load the knowledge database of the liquid tank family.

10. Run the expert system for the design of the liquid tank family. Its inference engine was loaded in step five and its knowledge database in step nine.

2.1.3 Adopting Same Routines in Both Levels of the Expert Systems

Another advantage of developing a two-level expert system is that the same program routines can be called by both levels of the expert systems. It makes the system developer's job much easy. It is based on the fact that many procedures, and the knowledge to be manipulated in both levels, are the same.

At the top level, the super-expert system works as the consultant and the human expert as user. When a decision has to be made, the super-expert system gives advice, the human expert performs his judgement and makes the decision. In the expert system, the procedure of the decision making is almost the same, except that the consultant is now the expert system and the user is the designer. The knowledge manipulated in both levels is very similar, except that it is more general in the top level and more specific in the bottom level.

Let us give an example. In the top level expert system, the human expert has to determine a set of standard importance values. First, the super-expert system gives the recommended values by means of a bar chart. The human expert can modify the values by moving a crosshair at the CRT terminal to change the heights of the bars. Later these modified importance values will be recommended to the designer in the expert system, still by means of the bar chart, and the designer can modify them in the same way as

the human expert did in the top level. So a program module dealing with the bar chart of the importance values can be called by both levels of the expert systems.

Some other examples of this type are as follows.

1. The routine concerned with configuration graphics, which can be called by the super-expert system to create configuration database, or by the expert system to modify a configuration in the database or enter a completely new one.

2. The routine to determine the failure mode and the factor of safety.

3. The routine to determine the optimization method and FEM algorithm, and their parameters.

4. The routine to show all of the design candidates in the database.

2.1.4 Summary

The two-level expert system has the following advantages,

1. It helps to solve the contradiction between the desire that an expert system be able to solve a wide range of applications, and the requirement to limit the domain of an expert system strictly.

2. It frees the human expert from relying on a knowledge engineer to build an expert system.

3. It lightens the burden on the human expert to

be concerned about both the common knowledge and the general knowledge.

This algorithm is realistic and valid as an expert system approach due to the following two points.

1. The human expert can accept, ignore or modify the common knowledge and general knowledge from the super-expert system.

2. The human expert can freely add his own specific knowledge.

It is believed that the two-level expert systems can be applied to other fields, such as medical diagnosis, mineral exploration, and so on. Even a multi-level expert system could be adopted. Figure 2.1 is a hypothetic example of a three-level expert system for fixing vehicles.

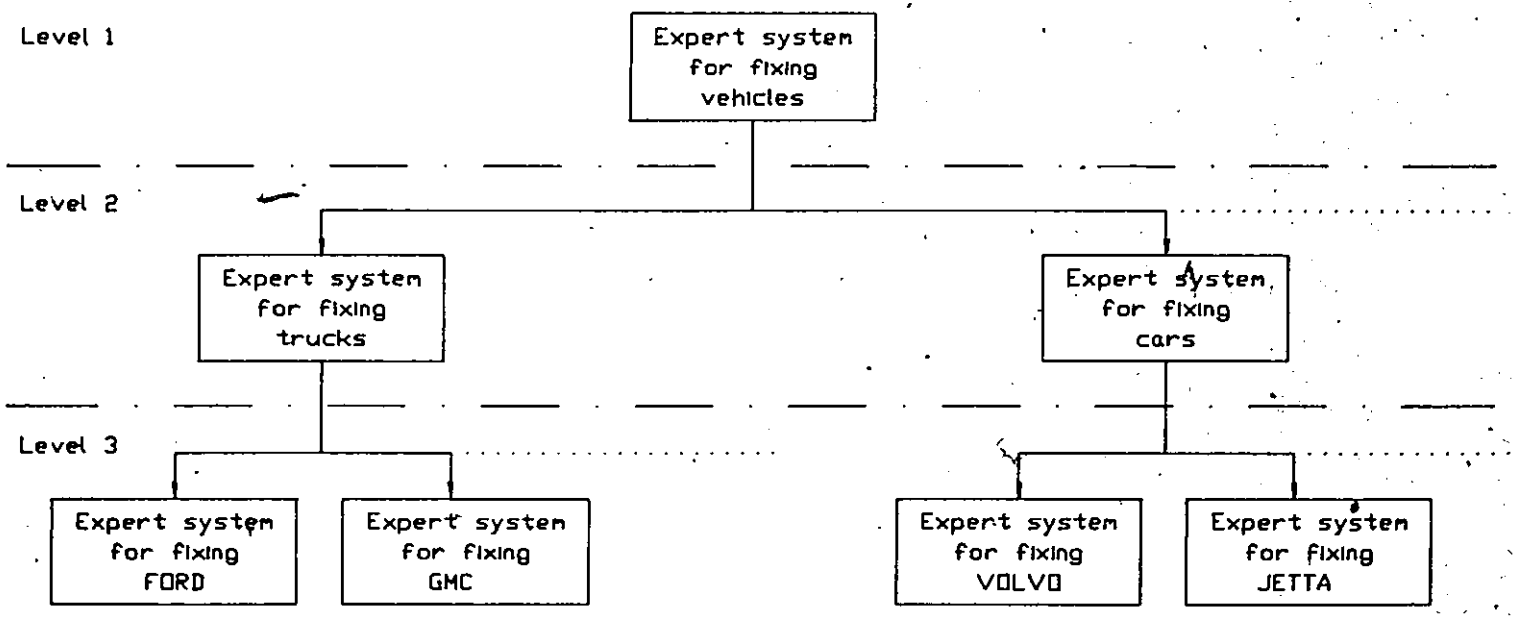


Figure 2.1 Multi-level expert system for fixing vehicles

2.2 SPECIAL FEATURES OF THE EXPERT SYSTEM APPROACH

2.2.1 The Overall Value of the Optimized Design

In our expert systems, formula (1.1) is the main inference engine. It is called the merit method, and it is rewriten below, with a detailed explanation using a hypothetical example.

$$M_i = \sum_{j=1}^{k} (P_j * D_{i,j}) \qquad i=1,2,\ldots,N \qquad (1.1)$$

where M_i ---- merit value of the ith candidate

P_j ---- importance value of the jth performance characteristic

$D_{i,j}$ -- desirability value of the ith candidate to the jth performance characteristic

N ----- number of candidates

K ----- number of performance characteristics

Suppose there are three performance characteristics, i.e. cost, weight and volume. If the designer feels that cost is a very important consideration in the design, he should specify the importance value of cost as 10. On the contrary, if the volume is thought less important in the design, an importance value of 2.0 might be assigned to it (Table 2.1). The importance value ranges from 0 to 10. The larger the importance value, the more important the performance characteristic is in the design.

Now let us consider the selection of material, and

Table 2.1 Importance values and desirability values for the hypothetical example

Performance characteristic	Importance value	Desirability value		
		Steel	Copper	Aluminium
Cost	10	0.5	0.1	0.2
Weight	8	0.05	0.2	0.7
Volume	2	0.2	0.3	0.2

assume for illustration that there are only three materials in the database: steel, copper and aluminum.

For a certain performance characteristic, e.g. cost, the steel has a desirability value of 0.5, copper 0.1, aluminum 0.2 (Table 2.1). The larger the desirability value, the more suitable the material is based on cost consideration. So in this case, steel is the best one if only cost is concerned, aluminum is the second, and copper is the worst.

For weight, another performance characteristic, steel, copper and aluminum have desirability values of 0.05, 0.2, 0.7 respectively. This means if we only consider weight, aluminum is the best material while steel is the worst one.

All of the information of the hypothetical example is shown in Table 2.1. We shall explain how to obtain the desirability values shortly. Now let us calculate the merit values of each material, using formula (1.1).

$$M_{steel} = 10*0.5 + 8*0.05 + 2*0.2 = 5.8$$
$$M_{copper} = 10*0.1 + 8*0.2 + 2*0.3 = 3.2$$
$$M_{aluminum} = 10*0.2 + 8*0.7 + 2*0.2 = 8.0 \qquad (2.1)$$

The larger the merit value, the more desirable the material is for the design from an overall viewpoint. In this example, obviously we shall select aluminum, whose greatest advantage is light weight, which the designer has specified as important, and the desirability values of

aluminum for other performance characteristics are not too low. Although steel is good for cost, it is not desirable for weight, having a very small desirability value of 0.05, so the merit value of steel is less than that of aluminum.

The merit method can also be considered as a multi-objective optimization procedure from the viewpoint of value theory (Siddall 1982), where the objective of the optimization is the overall value of the engineering device or system to society, as a whole or as individuals. It is meant to include all of the basic desires of mankind --- pleasure from food, drink, warmth, comfort, safety, excitement, convenience, status, playing games, love of technological devices, to name a few. Formulating such objective functions includes a lot of intuitive judgement and synthesis. Because of the difficulty in implementing it, conventional optimization methods simply take a single performance characteristic (e.g. cost, weight) as the objective.

Recalling the hypothetical example, we find that the search for a desirable material using the merit method is similar to the practice of multi-objective optimization using the value concept. Each performance characteristic is actually an optimization objective and its importance value is the weighting factor for the total value.

Referring to Table 2.1 and Figure 2.2, each material is similar to a design point, each desirability value is the

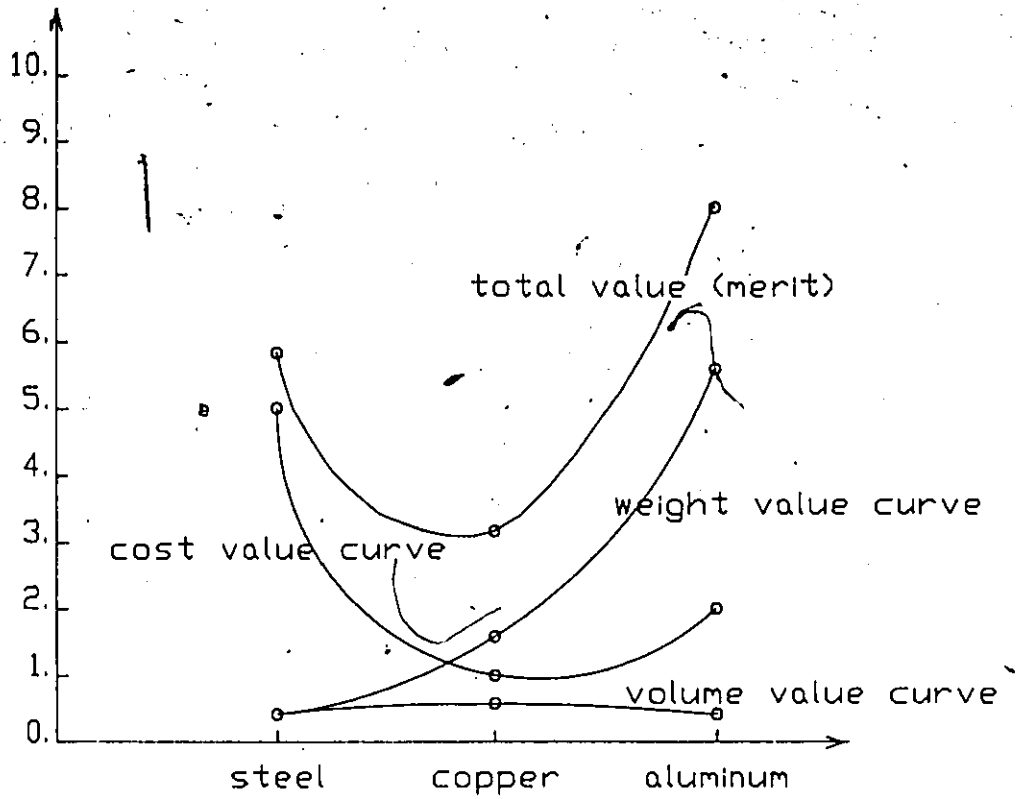


Figure 2.2 Value curves

value for a performance characteristic. For example, the desirability value of steel relative to cost is the value of steel on the value curve of cost. The sum of the values of a material on each value curve is the total value of this material, i.e. the merit value of the material. The importance values of the performance characteristics scale the different value curves.

The design procedure in this project actually includes two steps.

1. Using the expert system to determine an optimum model in the sense of maximum overall value, based on the intuitive expertise of the human expert.

2. Optimizing this model based on a single performance characteristic, using the numerical algorithm.

The result should lead to a better design.

2.2.2 Using Linear Programming to Fit the Desirability Values

2.2.2.1 Introduction to the Algorithm

In this section, we discuss a systematic method, based on a linear optimization program and machine learning from a sample, to adapt the desirability values with the help of the computer.

In expert system literature, it is reported that it takes several months for the human expert and the knowledge engineer to work together to adjust many parameters disseminated in the expert system, e.g. the possibility

values in medical diagnosis, by testing some examples. If the expert system so developed is expected to work well in a realistic field, a great number of examples should be tested. However the larger the number of the examples, the more difficult for the parameters to be adjusted.

The desirability values in formula (1.1) are similar parameters. We have tried to derive the desirability values of material candidates by some formulas (Farag, 1979) based on the material properties. But these formulas seem not very practical or reliable. For configuration candidates, there is no formula at all. The best way would seem to be to resort to the heuristic knowledge of the human expert. Usually a human expert is good at making a conclusion directly, given an input, and not good at answering why. This is because the conclusion is derived by some theoretically based rules, combined with a lot of experience and heuristic knowledge, which is difficult to express exactly, particularly in a numerical form like the parameters.

In this project, the expert is only required to give a desired candidate for the input of each example in a sample. The desirability values are adapted to fit all of the examples by the computer. In this thesis, a sample consists of a lot of examples, each of which includes an input and a desired candidate. The input is a set of importance values each for a performance characteristic.

The desired candidate is appointed by the human expert and reflects his heuristic expertise. We shall postpone a discussion of how to acquire the sample, and assume for the time being that it is available already.

We have tested three methods to adapt the desirability values from the sample

1. Direct Adapt Method

All of the examples are fitted one by one, by adjusting the desirability values based on some theoretical or heuristic rule, to make,

$$M_d > M_i \qquad i=1,2,\ldots,N \quad i\neq d \tag{2.2}$$

where M_d is the merit value of the desired candidate, N is the number of the candidates.

Each time the last example is fitted, we go back to fit the first example again until all of the examples have been fitted at the same time.

2. Nonlinear Program

For each example,

$$T_r = \sum_{\substack{i=1 \\ i\neq d}}^{N} < M_d - M_i > \qquad r=1,2,\ldots,NSA \tag{2.3}$$

where NSA is number of examples,

$$< M_d - M_i > = \begin{cases} 0, & M_d > M_i \\ -(M_d - M_i), & M_d \leq M_i \end{cases} \tag{2.4}$$

The objective function is to minimize the sum of T_r,

i.e.

$$U = \sum_{r=1}^{NSA} T_r \qquad (2.5)$$

The design variables are the desirability values D_{ij}, subject to

$$D_{ij} \geq 0 \qquad (2.6)$$

Above two methods have two disadvantages.

(1) Both of them require the starting desirability values which is still a burden on the human expert.

(2) No solution can be guaranteed in a definite number of iterations, especially when the number of examples is large.

3. Linear Program

Linear programming can eliminate the above two disadvantages, and is very robust. It becomes the preferable method in this project.

For each example, there are (N-1) inequality constraints,

$$M_d - 1.01*M_i > 0.01 \qquad i=1,2,\ldots,N \quad i\neq d \qquad (2.7)$$

Referring to formula(1.1),

$$\sum_{j=1}^{K} P_j*D_{d,j} - 1.01 \sum_{j=1}^{K} P_j*D_{i,j} > 0.01 \qquad i=1,2,\ldots,N \quad i\neq d \qquad (2.8)$$

So the total number of constraint equations is,

$$NC = NSA * (N - 1) \qquad (2.9)$$

The design variables are (K*N) desirability values and NC slack variables. So the total number is,

$$NV = K*N + NC = K*N + NSA*(N-1) \qquad (2.10)$$

In linear programming, all of the design variables are greater or equal to zero automatically. The parameter 1.01 at the left hand side of formula (2.7) guarantees that the merit value of the desired candidate will be larger than all others. The number 0.01 at the right hand side of the same formula avoids the trivial solution for the inequality equation. A peculiarity in our problem is that it needs no objective function. Any feasible solution of linear programming means all of the examples are fitted; and this is all that is required. So as soon as all of the artificial variables have been dropped, i.e. when the first phase optimization in the simplex method is finished, the linear program stops, and a solution is returned.

Originally, we tried to make use of an objective function to improve the distribution of the desirability values. Numerical trials show that it is unnecessary. The desirability values restrict each other through the constraint equations and their distribution is reasonable.

For a trial of 25 performance characteristics, nine candidates and 50 examples in the adapt sample, the number of inequality equations is,

$$NC = 50 * (9 - 1) = 400 \qquad (2.11)$$

The number of the variables (including the slack variables) is

$$NV = 9 * 25 + 400 = 625 \qquad (2.12)$$

It took about 4.5 hours of CPU (Central Process Unit) time on a VAX 730 to get a feasible solution.

From the above discussion, it can be found that the desirability values adapted by the linear programming in the computer should behave no worse than the parameters adjusted by a human expert and a knowledge engineer manually in general expert systems. While the best result that can be expected in the latter is that all of the examples used to test the expert system have been satisfied, the desirability values adapted by linear programming can satisfy all of the examples in the sample by 100 %.

The ultimate purpose in adapting the desirability values is not to fit the sample; it is rather to use such adapted desirability values to produce correct conclusions in a real design problem, where the design input is usually more or less different from the input of any example in the sample.

2.2.2.2 Testing the Algorithm

A method has been developed to test whether the desirability values adapted by linear programming from the sample will work well. This test is only done here in order to confirm that the linear programming procedure

adequately fits the adaption of the desirability values, and need not be called in later applications. First let us introduce some terminology.

Standard Importance Values -- These are a set of importance values for the performance characteristics. They are specified by the human expert and are assumed to be the most typical input in the real field.

Pseudo Desirability Values -- They are fabricated just to serve the test purpose and assumed to be "exact". However they do not really exist and are unnecessary in real applications.

Adapt Sample -- This is a group of examples used to adapt the desirability values. We assume that the importance values of a performance characteristic in different example inputs have a normal distribution. The mean is the standard importance value and the standard deviation is four. Then we use Monte Carlo simulation with the rejection method in order to create the example inputs. These example inputs have two characteristics.

1. Since the standard deviation is four, the example inputs cover a large percentage of the whole range of the importance value, i.e. different possible inputs in a real application will be simulated.

2. The adapted desirability values will be more suitable for those inputs which are closer to the standard importance value, which is the mean of the normal distribution.

For each example input, we use the pseudo desirability values to calculate the merit value of each candidate, using formula (1.1), the candidate with the largest merit value is the desired candidate for that example input. In a real sample, the desired candidate would be appointed directly by the human expert.

Adapted Desirability Values -- The adapted desirability values are the result of the linear programming operation on the adapt sample, and they will be used in the real design applications.

The algorithm of the test can be interpreted as follows.

1. The standard importance values are used to create the inputs of the examples of the adapt sample by Monte Carlo simulation.

2. The pseudo desirability values are used to mimic the human expert's finding a desired candidate for the input of each example in the adapt sample. Thus the adapt sample is completed.

3. The adapt sample is used to adapt the adapted

desirability values by linear programming.

If the method is good enough, the adapted desirability values should give results similar to the pseudo desirability values. To evaluate this analogy, let us introduce two additional terms.

Test Sample -- The test sample is created in the same way as the adapt sample, but the number of examples in the test sample is much larger and the seeds to generate the random numbers are different.

Consistency Value -- We shall explain the consistency value through an example. First we assume there are only three candidates to be selected, i.e. A, B, C.

Taking the input of an example from the test sample, if the merit values for each of the candidates, calculated by using pseudo desirability values, rank as,

B A C

that means B is the best for this test example, A is the second and C is the worst.

We now use the adapted desirability values to evaluate the merit values of the same test example. If the candidates rank as B, C, A, i.e. the best candidate determined by the pseudo desirability values is also the best if determined by the adapted desirability values, then the consistency value is

$$t = INT(N/2)$$

where N is the number of the candidates. In our example, N is three and thus t is one.

If they rank as A, B, C, i.e. the best candidate for pseudo desirability values now ranks second. for the adapted desirability value, we let t equal zero, or smaller by one than the first case.

Using the same argument, if the merit values from adapted desirability values rank C, A, B, then t equals -1, which is the smallest possible consistency value when there are only three candidates.

The total consistency value is the sum of the consistency values for each test example.

Suppose that there are 1000 examples in the test sample. The total consistency value ranges from -1000 to 1000. The larger this value, the better our linear programming method is found to be. If the total consistency value is 1000, that means the adapted desirability value is consistent with its father, the pseudo desirability value, completely for all of the 1000 test examples.

Now let us first test a small example, where there are three candidates, five performance characteristics, and 1000 test examples; so the perfect total consistency value is 1000.

Table 2.2 gives the total consistency values corresponding to different sizes of the adapted sample (number of examples). They are also shown in solid line in Figure 2.3.

We can make the following observations on these results,

1. As the size of the adapt sample increases, the quality of the adapted desirability values tend to be improved and the total consistency value approaches its perfect value.

2. The total consistency value fluctuates a lot, it is obviously due to the random characteristic of the adapt sample. When the adapt sample size is small, such fluctuation is inevitable.

Increasing the adapt sample size is prohibited by the CPU time and computer central memory required. The number of constraint equations in linear programming is directly proportional to the number of the examples in the adapt sample. According to the literature (Reklaitis, 1983), if the number of constraint equations increases K times, the CPU time will increase K^3 times and the central memory K^2 times.

Suppose fitting ten adapt examples requires CPU time of T, and central memory of C. Now if we have 20 adapt examples and try to fit them simultaneously, we have to use 8*T CPU time and 4*C central memory.

Table 2.2 The consistency values of the first example

Adapt sample size	Total consistency value
3	404
6	648
12	843
24	704
25	912
50	883
51	865
100	924

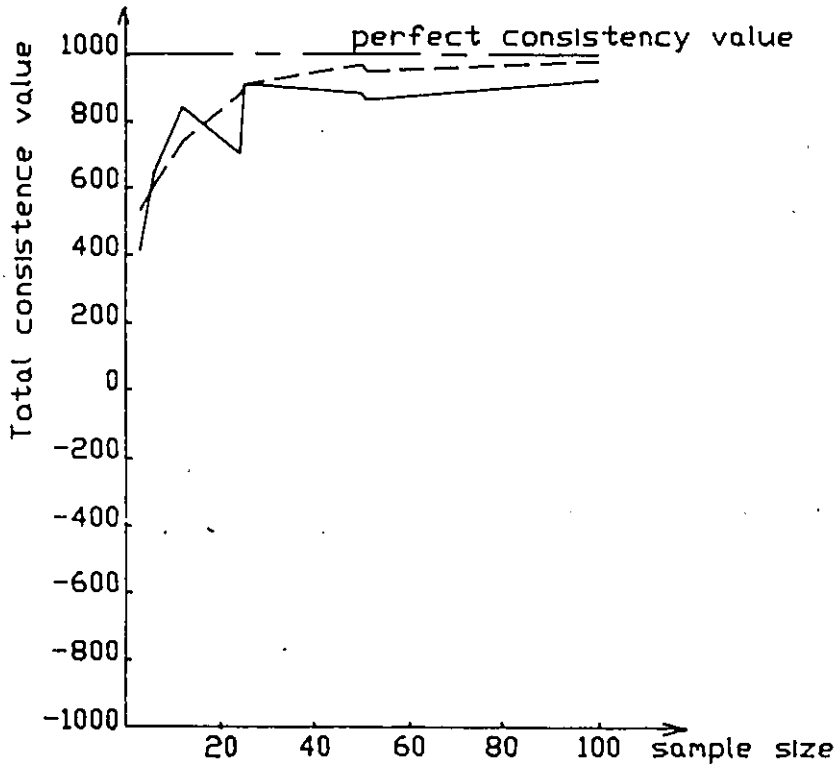


Figure 2.3 The consistency values of the first example

We could divide these 20 adapt examples into two groups each with 10 adapt examples, and fit each group independently. Then we could combine the adapted desirability values from two independent groups somehow in order to improve the total consistency value. Now the CPU time required is only 2*T and no more central memory is required at all.

Let a pair of corresponding desirability values be D^1 (in group one) and D^2 (in group two). If both of them are large (or small), for fitting the total 20 adapt examples, the overall D should be large (or small). If one of D^1 and D^2 is large and another is small, the overall D should be a compromise, so that it is the mean.

So the simple expression

$$D = 0.5 * (D^1 + D^2) \qquad (2.13)$$

should represent the above arguments. Since desirability values have only relative meaning, the parameter 0.5 can be dropped.

However due to the random characteristic of the adapt examples, the value level of the desirability values in each group are very different. To make the desirability values of each group contribute equally to the overall ones, the desirability values within each group are unified as follows

$$S = \sum_{i=1}^{N} \sum_{j=1}^{K} D_{ij} \qquad (2.14)$$

Then,

$$D_{ij} = D_{ij} / S \quad ((j=1,2,\ldots,K),i=1,2,\ldots,N) \quad (2.15)$$

where N and K have the same meaning as in formula (2.10). In this way, the sum of the desirability values in each group is always unity.

Table 2.3 lists the total consistency value versus a triple-group adapt sample, where the sample size is NSA, three groups of examples are used, and each group has NSA adapt examples. Table 2.3 is represented by the dashed line in Figure 2.3.

We can conclude the following from these results.

1. The multi-group adapt sample does increase the total consistency value from an overall point of view. When the sample size is 100, the total consistency value is almost perfect.

2. What is more significant is that the curve produced by the multi-group adapt sample is very stable, almost continually increasing along with the sample size.

3. Although now the adapted desirability values cannot fit all of the examples in the adapt sample exactly, they are less sensitive to a few abnormal examples.

The multi-group adapt sample is useful for increasing the total consistency value without requiring much more CPU time.

Finally a large example was tested. There are 9 candidates and 25 performance characteristics. The 900 adapt examples were divided in three ways.

Table 2.3 Consistency values of the triple-group sample

Sample size	Total consistency value
3	534
6	611
12	741
24	882
25	909
50	969
51	946
100	978

Case 1: the 900 examples were divided into 90 groups, each having 10 adapt examples, so the sample size was 10, and group number was 90.

Case 2: the sample size was 20, and the group number was 45.

Case 3: the sample size was 30, and the group number was 30.

For all of the cases, the total sample size was

(sample size) * (group number) = 900

The results are shown in Table 2.4 and Figure 2.4.

From these results, we can conclude the following,

1. This example had a total of 225 desirability values to be adapted. So the size of the adapt sample was much larger than that of the first example. For the total of 900 examples, all of the three cases gave reasonable results, i.e. the total consistency values were between 2000 to 3000 (note the range of the total consistency value was from -4000 to 4000 in this example). It is equivalent to the case where, if a candidate is considered to be the best by the pseudo desirability values, it will always be considered to be the second or third best by the adapted desirability values among the nine candidates.

2. The three curves fluctuate a lot at the beginning. As the number of the examples to be fitted increases, they become stable and tend to increase little by little.

Table 2.4 The consistency values for the second example

	Case 1	Case 2	Case 3
Number of groups	90	45	30
Number of examples in each group	10	20	30
Total sample size	900	900	900
Max. total consistency value	2133	2650	2765
CPU time (sec.)	3918	27949	74689
Central memory	6800	26400	58800

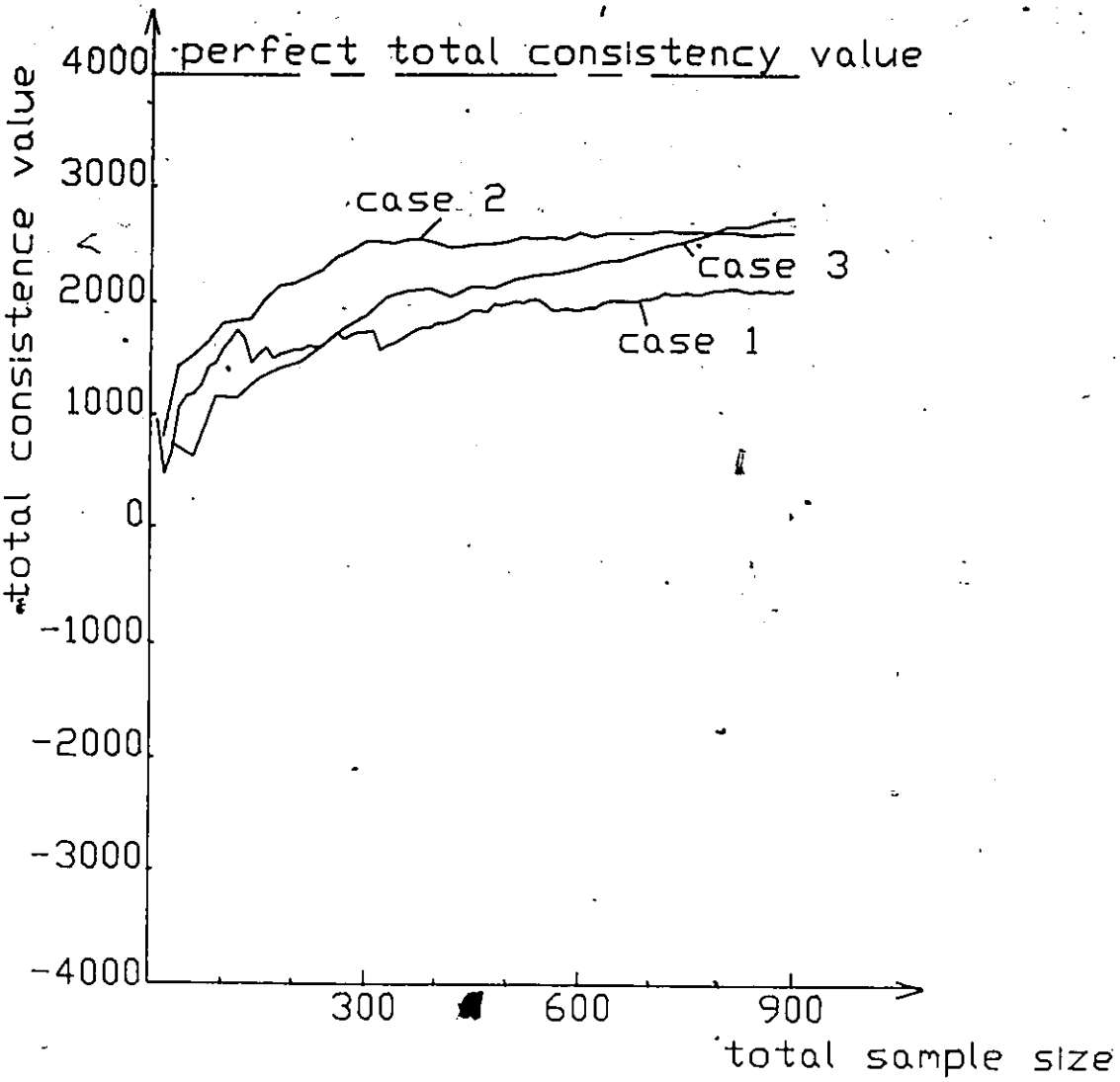


Figure 2.4 The consistency values of the second example

3. The role of the multi-group sample is significant. For example, in case 3, it increases the total consistency value from 763 to 2765. Furthermore the latter is much more stable than the former. Without the multi-group method, we have no way to deal with the total 900 examples with currently available computer resources.

4. In Figure 2.4, we can see that, after the total sample size exceeds 600, the curve for case 1 and 2 are almost horizontal. The curve for case 3 appears to be still rising after a total sample size of 900. It suggests that we cannot increase the total consistency value infinitely by just increasing the number of groups. When the total sample size is increased, the number of examples in each group should be increased correspondingly. However it will increase the required CPU time and central memory dramatically.

5. The best total consistency value obtained so far was 2765, which represents an array

499 196 122 78 39 24 18 15 9

It means, within 1000 test examples, the candidates which had been considered best by pseudo desirability values, were the best 499 times by the adapted desirability values, 196 times the second best by the adapted desirability values, and so forth.

If we have a total of 1200 adapt examples which are divided into 30 groups, each of which has 40 examples, we

should be able to obtain desirability values with a total consistency value higher than 3000, which is equivalent to the result that the best candidate from pseudo desirability values is always the best or second best candidate by the adapted desirability values. However it would take more than 70 CPU hours in VAX 730.

The testing of the algorithm described above suggests the following conclusions about the method of adapting desirability values by linear programming and machine learning from a sample.

1. This method provides a systematic and uniform formulation to manipulate the expertise, which is the most difficult task in building expert systems.

2. This method frees the human expert from the bothersome job of giving his reasoning for a lot of intermediate steps in the derivation of a conclusion which can now be given directly, and which includes intuitively all of the potential factors, and even the uncertainty factors, affecting the conclusion. It is exactly consistent with the basic idea of intuitive expertise in expert systems.

3. The strenuous job of adjusting the large number of parameters (desirability values in this method) in an expert system is taken over by the computer from the human expert and the knowledge engineer. The development time of an expert system drops from several months to several hours.

4. Linear programming is a very successful and mature numerical algorithm. It is much easier to call the linear programming than to code a lot of production rules which vary quite a lot from application to application.

5. When new information is available (i.e. a new adapt example), it is very easy to update the knowledge database by calling once again the linear programming.

6. From the above two examples, it was found that this method can give reasonably accurate results. Furthermore the accuracy can be continually improved if more examples are adapted. Of course, it depends on the available computer resources.

7. This method is not very sensitive to abnormal inputs of the adapt sample.

8. By adopting a multi-group sample, this method can adapt a rather large sample.

2.2.2.3 Applying the Algorithm to Other Fields

In the expert system world, people tend to use a lot of IF ... THEN statements, or even more complicated structures to codify the derivation. It is not only a matter of making the programming complex, it also results in different applications having quite different programming structures. Eventually the development of expert systems will become a privileged field for a few specially trained people.

It is anticipated that the merit method (formula (1.1)) will unify a large area of expert systems, so that expert systems become an easy and routine application in science and engineering.

Now let us study a hypothetical example in medical diagnosis. Suppose the diagnosis depends on three symptoms -- temperature T, blood pressure B and speed of heart beat S.

$$34 < T < 46$$

$$120 < B < 200$$

$$40 < S < 100$$

There are ten hypothetical diseases which correspond to the candidates in structural design. The symptoms correspond to the performance characteristics, both are the input features. Figure 2.5 is its logic tree with obvious meaning. For example, if T = 35, B = 162 and S = 80, then it can be found from the logic tree that the diagnosis is disease nine.

To use linear programming to adapt the desirability values of the diseases (a better name in this example may be the likelihood values of the diseases), a set of adapt examples have to be acquired first. For example, T = 35, B = 162, S = 80 and disease nine constitute an adapt example.

However, since the values of the symptoms have quite different ranges, it is better to normalize them into a

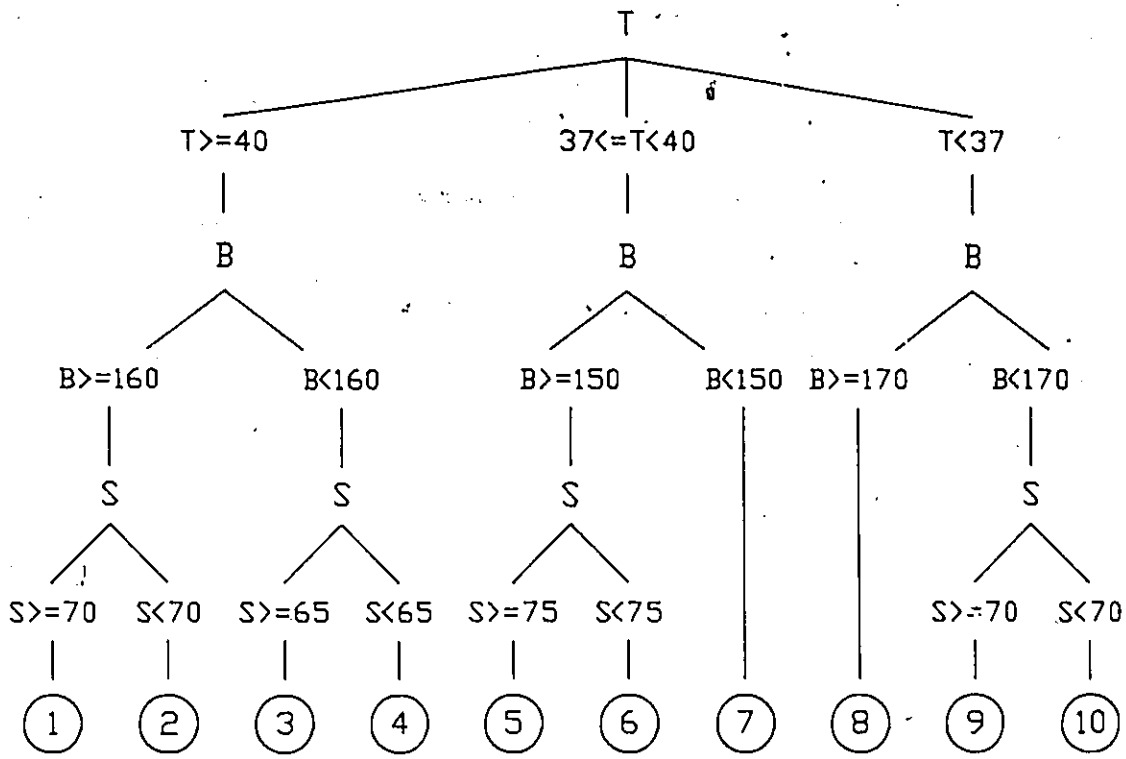


Figure 2.5 Logic tree for medical diagnosis

value R between zero and one, using the following formula.

$$R = \frac{X - 0.5 * (U + L)}{U - L} + 0.5 \qquad (2.16)$$

where X is the value of a symptom, and U and L are its upper and lower bounds. For T = 35, B = 162, S = 80, their equivalent importance values t, b and s can be calculated using formula (2.16).

$$t = \frac{35 - 0.5 * (46 + 34)}{46 - 34} + 0.5 = 0.083$$

$$b = \frac{162 - 0.5 * (200 + 120)}{200 - 120} + 0.5 = 0.525$$

$$s = \frac{80 - 0.5 * (100 + 40)}{100 - 40} + 0.5 = 0.667 \qquad (2.17)$$

So it is actually the equivalent importance values t = 0.083, b = 0.525, s = 0.667 and disease nine that are entered into the linear programming as an adapt example.

Using Monte Carlo simulation, a total of 90 adapt examples have been created, and divided into three groups. Linear programming is called to adapt them, and the desirability (likelihood) values corresponding to each of the symptoms for all of the ten diseases are obtained.

Now another 1000 test examples are created. For each of them, we can derive the diagnosis by two different methods: (1) the logic tree method, using the values T, B and S of the symptoms directly; (2) transferring T, B and S into their equivalent importance values t, b and s, and then

using the merit method (formula (1.1)). Table 2.5 shows the results. There are 516 times when the disease diagnosis by the two methods are the same, and 233 times when the disease diagnosis by the logic tree method ranks second to the merit method, and so forth. The probability that the diagnoses of the first two diseases from the merit method are the same as the ones from logic tree is fairly high.

$$P = (516 + 233) / 1000 = 75\% \qquad (2.18)$$

Unfortunately, the possibility that these diagnoses from the logic tree will rank as the last two in the merit method is rather high too.

$$P = (41 + 65) / 1000 = 10.6\% \qquad (2.19)$$

Let us look for some way to improve it. It is very difficult to determine a complete logic tree, even for the human expert. However if we just try to determine the logic relationship about a key symptom (say body temperature T), it is relatively much easier. Figure 2.6 is a simplified logic tree.

The simplified logic tree is combined with the merit method as follows.

1. For the input body temperature T, the simplified logic tree is used to find the three (or four) possible diseases.

2. The merit method is used to evaluate the merit values of these three possible diseases, ranking them from one to three according to their merit values.

Table 2.5 Comparison of the results from two different methods for the medical diagnosis

Position of the accurate diagnosis in the rank obtained from the merit method	The number of occurrences in 1000 test examples
1	516
2	233
3	57
4	44
5	26
6	9
7	5
8	4
9	41
10	65

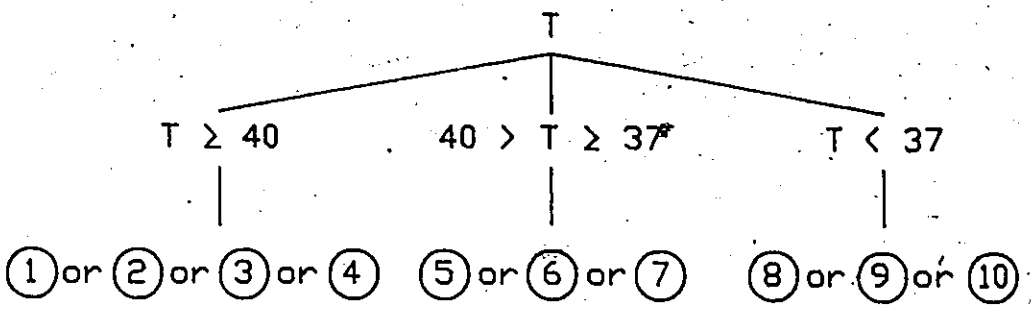


Figure 2.6 Simplified logic tree of the medical diagnosis

Table 2.6 Simplified logic tree of the medical diagnosis

Position of the accurate diagnosis in the rank obtained from the combined method	The number of occurrences in 1000 test examples
1	720
2	139
3	41
4	100
5	0
6	0
7	0
8	0
9	0
10	0

3. The merit values of other diseases are evaluated, ranking them from four to ten.

Table 2.6 shows the results, which appear quite good.

From this example, we found it is indeed possible to apply the merit method to many other expert systems (maybe combined in some way with another inference engine, e.g. a simplified logic tree).

Actually, the logic tree is analogous to the "hard specification", while the merit method is analogous to the "soft specification" (Siddall, 1982). The latter must reflect the real world better than the former.

2.2.3 Setting Up the Adapt Sample

This section is concerned with how to create a large enough number of examples, which will be then adapted to generate desirability values and other information for the candidates in the database.

It is ideal if there are detailed design records in a company or institute. Each design record corresponds to a real example, and the desired candidate is known and design requirements can be converted into the form of importance values.

When there are no such design records available, we can generate an example input, i.e. a set of importance values each for a performance characteristic, by Monte

Carlo method. Then the human expert determines a desired candidate according to this example input. Note that an example consists of an example input and a relevant desired candidate.

Since the concept of the importance values of the performance characteristics is abstract and intuitive, the importance values can be determined only in a subjective sense. The human expert should be very clear about what kind of structure the expert system he developed is going to help design, since the meaning of the importance values is more clear in a well specified domain. Furthermore, the human expert and the designer, who will use the expert system, should be as consistent as possible with the measures of the importance values. This can be accommodated by adequate explanations to the designer about the definitions of the performance characteristics from the expert system.

As described in section (2.2.2), the human expert must first determine a set of standard importance values for each performance characteristic, which is considered as the typical design requirements for practical designs. Such standard importance values will be taken as the mean values of the normal distribution of the importance values.

Now the human expert has to determine the standard deviation of the normal distribution. If the expert system will be used for widely varying design applications, the standard deviation should be larger, otherwise it should be

smaller.

When an example input is created, it will be shown on the screen as a bar chart (Figure 2.7). The human expert has three options.

1. If he feels the example input is reasonable, he can accept it.

2. If he feels the example input is too abnormal to be realistic for any design application, he can reject it.

3. If he thinks the example input is basically reasonable but some modification is needed, he can change the height of the bars interactively, corresponding importance value in the example input will be modified automatically. Then this modified example input is accepted.

Since the human expert can modify the example inputs, the inputs must be more realistic than those totally resorting to the Monte Carlo simulation as in section 2.2.2. Therefore the adapted desirability values should be better as well.

Finally the human expert provides a desired candidate for the accepted example input by his knowledge and experience, thus creating a complete example. Usually, several human experts could sit down together before the CRT screen, each would give his opinion independently, and the opinion of the majority would be accepted.

The super-expert system provides two facilities to

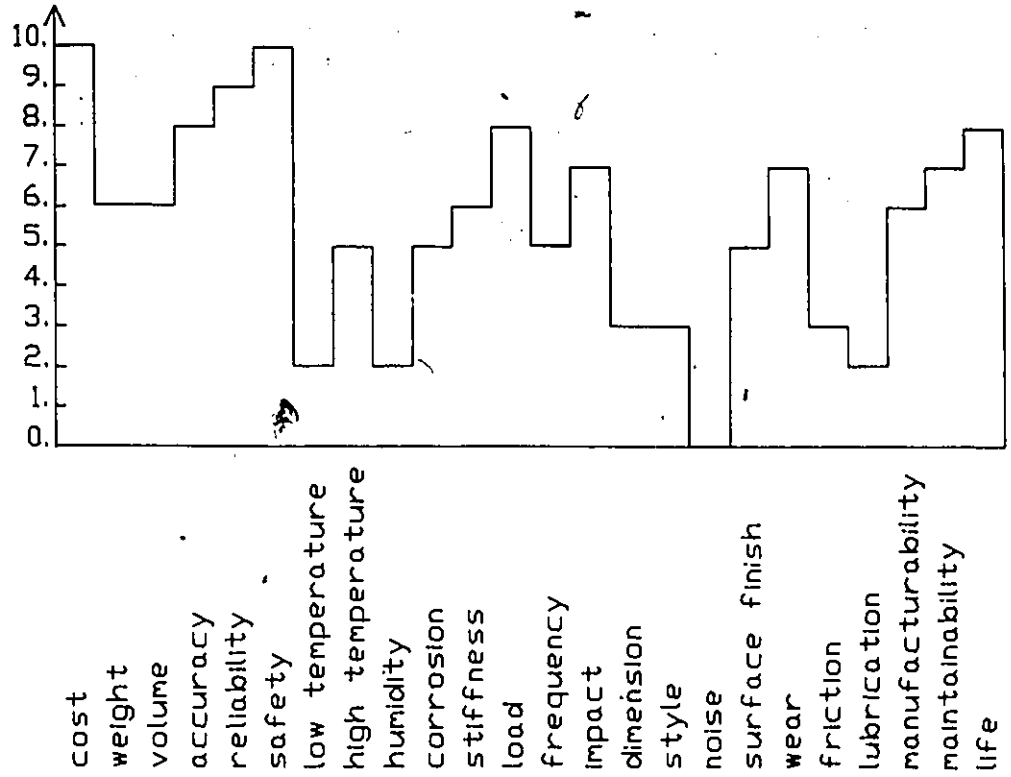


Figure 2.7 Bar chart of the importance values

help the expert or experts to determine a desired candidate for an example input.

1. Information acquisition

If at any time, the human expert types the H key on the keyboard, the information about the candidates will be displayed on the screen. If the candidates are materials, the information is their properties, e.g. cost, strength, modulus of elasticity. If the candidates are configurations, all of them will be displayed on the screen. Since there may be a lot of candidates, each configuration in the display may be very small. The human expert can move the crosshair to point to a certain configuration and press the D key, all of the configurations disappear, except the pointed one being amplified to the full size of the screen, in order to let the human expert have a close look.

2. Reference

Suppose we have set up some examples already, and the input of a new example is available. Let its importance values be

$$P_1, P_2, \cdots, P_k$$

where k is the number of the performance characteristics. Let the importance values of the ith existing example be

$$P_{i1}, P_{i2}, \cdots, P_{ik}$$

The characteristic difference of the ith existing example

with the new example is defined as

$$D_i = \sqrt{\sum_{j=1}^{k} (P_j - p_{ij})^2} \qquad (2.20)$$

Since D_i only has relative meaning, we simply let it be

$$D_i = \sum_{j=1}^{k} (P_j - p_{ij})^2 \qquad (2.21)$$

If D_i is very small, it means the input of the new example is quite similar to that of the ith existing example. So it is reasonable to recommend the desired candidate of the ith existing example as the solution for the new example. It is actually a "self-studying" procedure; the super-expert system studies the accumulated information to induce new knowledge. However, it is just a reference or recommendation, the human expert should take the responsibility to make the final decision.

Suppose there are three candidates A, B, C and six existing examples as in Table 2.7. It can be seen that candidate B is most likely to be recommended as the desired candidate for the new example, due to the third existing example. Candidate A is the second due to the fourth existing example. Candidate C is the last due to existing example two. Now all of the three candidates are listed on the screen in the sequence as follows.

1 B 2 A 3 C

This serves the human expert as a useful reference to decide on the desired candidate.

Table 2.7 Existing examples and the characteristic difference

Example number	Desired candidate	Characteristic difference
1	B	7.0
2	C	2.0
3	B	0.1
4	A	0.5
5	A	4.0
6	C	3.0

Since the number of the performance characteristics and the number of existing examples are large, the evaluation of formula (2.21) costs significant CPU time. The actual algorithm simplifies the calculation as follows.

(1) For a new example, select n (say one fifth of the total number of performance characteristics) performance characteristics with the highest importance values as the critical performance characteristics. A pointer array, K(n), points to them. The importance values of the critical performance characteristics are

$$P_{k(1)}, P_{k(2)}, \cdots, P_{k(n)}$$

with $$P_{k(1)} > P_{k(j)} \qquad j=2,3,\ldots,n \qquad (2.22)$$

For example, if a new example contains the importance values of six performance characteristics,

2.3 5.2 7.4 1.6 9.2 3.3

from which three are taken as the critical performance characteristics (i.e. n = 3). So

$$k(1) = 5$$

$$k(2) = 2$$

$$k(3) = 3$$

$$P_{k(1)} = P_5 = 9.2$$

$$P_{k(2)} = P_2 = 5.2$$

$$P_{k(3)} = P_3 = 7.4$$

(2) Loop for each existing example

$$\text{If } (\; | P_{k(1)} - P_{i,k(1)} | > a \;)$$

neglect the ith existing example,

otherwise,

$$D_i = \sum_{j=1}^{n} \left(P_{k(j)} - P_{i,k(j)} \right)^2 \qquad (2.23)$$

Here, a is a predetermined positive number, say 2. Note that $P_{i,k(j)}$ the importance value of the k(j)th performance characteristic of the ith existing example.

Since the human expert has to set up a great number of examples, he may lose consistency in his judgement inadvertently. Some examples may conflict with each other, creating a situation where the linear programming intrinsically has no feasible solution. Let D_{rt} be the desirability value of the rth candidate to the tth performance characteristic, P_{it} be the importance value of the tth performance characteristic of the ith example. If the desired candidate for the ith example is the dth candidate, then

$$\sum_{t=1}^{k} P_{it}*D_{dt} > \sum_{t=1}^{k} P_{it}*D_{st} \qquad s \neq d \qquad (2.24)$$

where K is still the number of performance characteristics. If the desired candidate for the jth example is the sth candidate,

$$\sum_{t=1}^{k} P_{jt}*D_{dt} < \sum_{t=1}^{k} P_{jt}*D_{st} \qquad d \neq s \qquad (2.25)$$

Both sides of (2.25) are subtracted from (2.24),

$$\sum_{t=1}^{k} (P_{it} - P_{jt})*D_{dt} > \sum_{t=1}^{k} (P_{it} - P_{jt})*D_{st} \qquad (2.26)$$

If $P_{it} = P_{jt}$, $t=1,2,\ldots,K$, i.e. all of the corresponding importance values of the ith example and the jth example are the same, then equation (2.26) will never be satisfied. Conflict occurs. If $P_{it} = P_{jt}$, $t=2,3,\ldots,K$, but $P_{il} \neq P_{jl}$, i.e. only one performance characteristic has different importance values in the two examples. Equation (2.26) becomes,

$$(P_{il} - P_{jl})*D_{dl} > (P_{il} - P_{jl})*D_{sl}$$

i.e. $D_{dl} > D_{sl}$ if $P_{il} > P_{jl}$

or $D_{dl} < D_{sl}$ if $P_{il} < P_{jl}$ (2.27)

Thus it is mandatory for D_{dl} be greater (or less) than D_{sl} if equation (2.26) is to be satisfied. However between some other two examples, it may be mandatory that D_{dl} be less (or greater) than D_{sl}. Conflict occurs again.

From the above study, it has been found that, the greater number of performance characteristics that have the same importance values, the greater possibility there is that conflict will occur. To avoid the conflict, the super-expert system makes a check before a new example input is accepted. It requires that the new input have at least five importance values different from their counterparts in each existing example.

Now that the sample is available, linear programming can be called to adapt the desirability values.

2.2.4 Search Methods

In the expert system literature, the word search usually means finding a solution in the database. In this project, search is the evaluation of the merit values of the candidates in the database when their desirability values are available. Then all of the candidates can be ranked according to their merit values, i.e. the goodness for a design.

Three search methods have been developed. All of them are based on formula (1.1).

1. The Full Search method, in which formula (1.1) is used directly.

2. The Discard Search method, in which some hopeless candidates are discarded at the beginning of, or during, the search procedure.

3. The Quick Search method, which compares some characteristics of the importance value curve and the desirability value curve, and discards up to 80% of the weaker candidates quickly. Formula (1.1) is then applied to the remaining 20% candidates only.

Since the full search method is quite straightforward, we shall only describe the latter two methods here. The motivation for developing them was to reduce search time.

2.2.4.1 The Discard Search Method

Before the search, the list of the performance characteristics is ranked so that the performance characteristic with the highest importance value is first in the list, while the performance characteristic with the lowest importance value is last. The performance characteristics with zero importance values are rejected from the list. In this way, the more important performance characteristics are considered first. For the convenience of later manipulation, all of the desirability values are normalized into the range from 0 to 10 when they are worked out from the sample.

Table 2.8 is the importance values of ten performance characteristics and the desirability values of nine material candidates of an example, which will help to explain the discard search method. Note that the performance characteristics in this Table have been ordered, and the desirability values have been already normalized.

We shall first consider the performance characteristic, that is cost, which has the highest importance value. It was found that stainless steel and bronze have very low desirability values for cost, 2 and 3 respectively. A design engineer must discard a material immediately if it cannot satisfy a very important specification, even it may satisfy other less important specifications very well. This is done by the following

Table 2.8 Importance values and desirability values of an example

Performance characteristic	Importance value	Desirability value								
		plain carbon ST	alloy ST	stainless ST	gray iron	ductile iron	wrought Al alloy	cost Al alloy	brass	bronze
cost	10	10	5	2	7	8	5	5	4	3
safety	10	5	5	5	5	5	8	7	5	5
environment	9	2	7	10	3	7	5	5	7	8
weight	8	5	9	6	3	5	10	10	2	2
impact	8	6	8	8	8	5	5	4	5	6
load	7	6	10	7	3	6	3	3	3	4
accuracy	6	5	5	5	3	5	5	5	5	5
wear	6	7	10	8	6	7	4	4	7	8
manufacture	3	8	4	5	7	7	7	7	9	9
style	2	5	5	8	2	5	6	6	8	8

rules.

If the importance value of a performance characteristic has the value

$$P > 8 \tag{2.28}$$

and the desirability value of a candidate for this performance characteristic has the value

$$D < P - 7 \tag{2.29}$$

Then, get rid of this candidate from the search immediately.

Stainless steel and bronze were rejected due to cost, and plain carbon steel due to environment. Since the performance characteristics are ordered from high to low, the bad materials were rejected in the first few stages. This left only six materials. After searching the first five performance characteristics, all of the most important specifications have already been considered. In this example, they are cost, safety, environment, weight and impact. The importance values of all the rest of the performance characteristics are equal to or less than 7. So the goodness of each material is basically determined at this point. The merit values accumulated so far, of all remaining materials in this example, are as follows.

$$M_{\text{alloy steel}} = 10*5 + 10*5 + 9*7 + 8*9 + 8*8 = 299$$

$$M_{\text{gray iron}} = 10*7 + 10*5 + 9*3 + 8*3 + 8*8 = 235$$

$$M_{\text{ductile iron}} = 10*8 + 10*5 + 9*7 + 8*5 + 8*5 = 273$$

$$M_{\text{wrought Al}} = 10*5 + 10*8 + 9*5 + 8*10 + 8*5 = 295$$
$$M_{\text{cost Al}} = 10*5 + 10*7 + 9*5 + 8*10 + 8*4 = 277$$
$$M_{\text{brass}} = 10*4 + 10*5 + 9*7 + 8*2 + 8*5 = 209 \qquad (2.30)$$

Alloy steel has the largest merit value, and brass the smallest, they are called the most and the least hopeful candidates respectively.

From the sixth performance characteristic on, after each five stages (i.e. the 6th stage, the 11th stage, ...), the difference is checked of the accumulated merit values between the most and least hopeful candidates. If it is less than a predetermined value, nothing will happen, however if it is greater than this value, the discard procedure will be triggered.

If it is the first time that the discard procedure has been triggered, the complete merit value Mc of the most hopeful candidate is calculated, i.e. a depth search is performed. In this example,

$$Mc = M_{\text{alloy steel}}$$
$$= 299 + 7*10 + 6*5 + 6*10 + 3*4 + 2*5$$
$$= 481 \qquad (2.31)$$

Later if the accumulated merit value of any other material exceeds 481, the most hopeful material is replaced and Mc is updated. Then a value V_i for each material is calculated,

$$V_i = Mc - (Q*P*SUM_i + M_i) \qquad (2.32)$$

The term in the bracket is the estimated complete merit

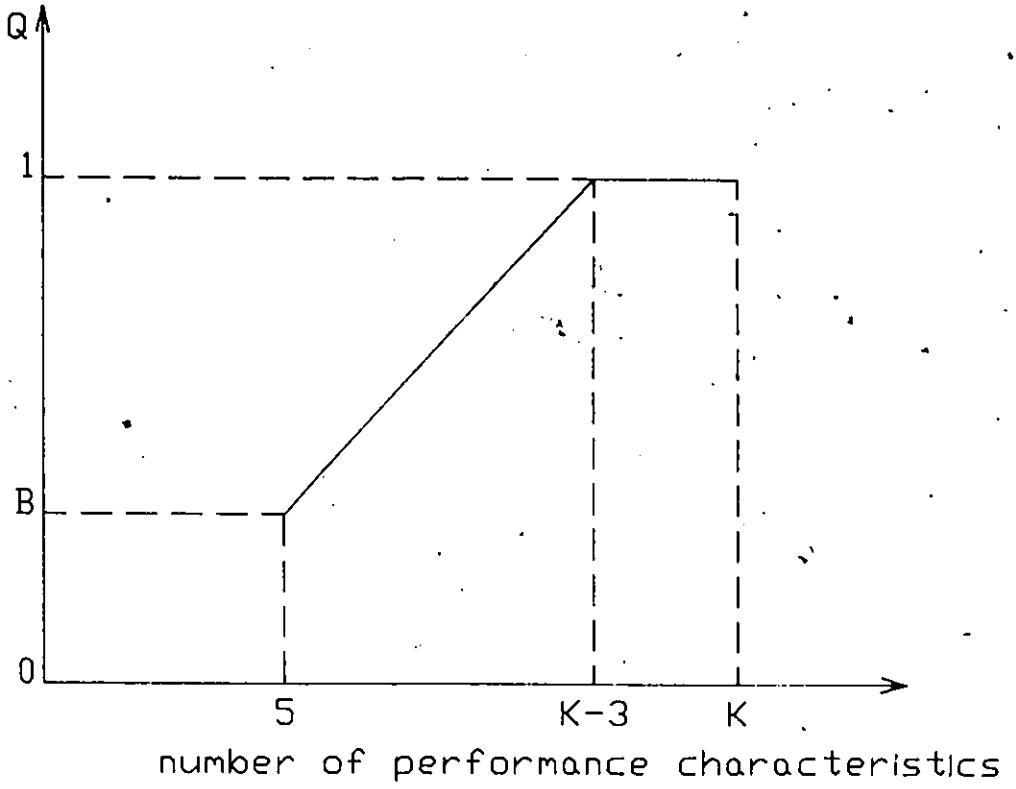
value for the ith material. So V_i is the difference between the complete merit values of the most hopeful and the ith materials. P is the importance value of the next performance characteristic. In this example, it is the importance value 7 of load. SUM_i is the sum of the remaining desirability values of the ith material. For example, if the ith material is gray iron, SUM_i is the sum of the desirability values of load, accuracy, wear, manufacture and style of gray iron, i.e.

$$SUM_i = 3 + 3 + 6 + 7 + 2 = 21 \qquad (2.33)$$

SUM_i is actually evaluated immediately after the desirability values are worked out from the sample. Whenever a performance characteristic has been considered,

$$SUM_i = SUM_i - (\text{the desirability value for the just considered performance characteristic}) \qquad (2.34)$$

M_i is the merit value accumulated so far for the ith material. For gray iron, M_i is 235, as in formula (2.30). Q is a factor taking care of the fact that the importance value of some performance characteristics to be considered later would be less than P. For example, the importance values of accuracy, wear, manufacture and style are less than P (i.e. seven). Q is smaller or equal to one, and should become larger towards the end of the search procedure. We use a linear interpolation to determine the value of Q as in Figure 2.8. B is a predetermined number.



Note: K is the total number of the performance characteristics

Figure 2.8 Interpolation of factor Q

If Q is smaller, more weaker candidates may be discarded earlier and the search tends to be quicker, but there is more risk of losing a good candidate. At present, 0.7 is used as the value of B.

Returning to formula (2.32), if V_i is greater than zero, the ith material will be discarded. However if the surviving materials number is less than three, only the appropriate number of the least hopeful materials are discarded, in order to retain at least three candidates at the end of the search. And in this situation, the trigger and discard procedures are no longer used.

If the search is about to finish (say there are only five performance characteristics left to be dealt with), the trigger and discard mechanism are shut down also, because at this time, the computer time saved by discarding a hopeless candidate hardly pays off for the time spent in running the trigger and discard program.

For a large database, the discard search method will accelerate the search procedure significantly.

2.2.4.2 The Quick Search Method

The number of multiplications in formula (1.1) is

$$NM = N * K \qquad (2.35)$$

where N --- the number of the design candidates

K --- the number of the performance characteristics

In the quick search method, the concept is to try to extract

a fewer number of abstract and essential characteristics from the original performance characteristics, in order to reduce NM in formula (2.35).

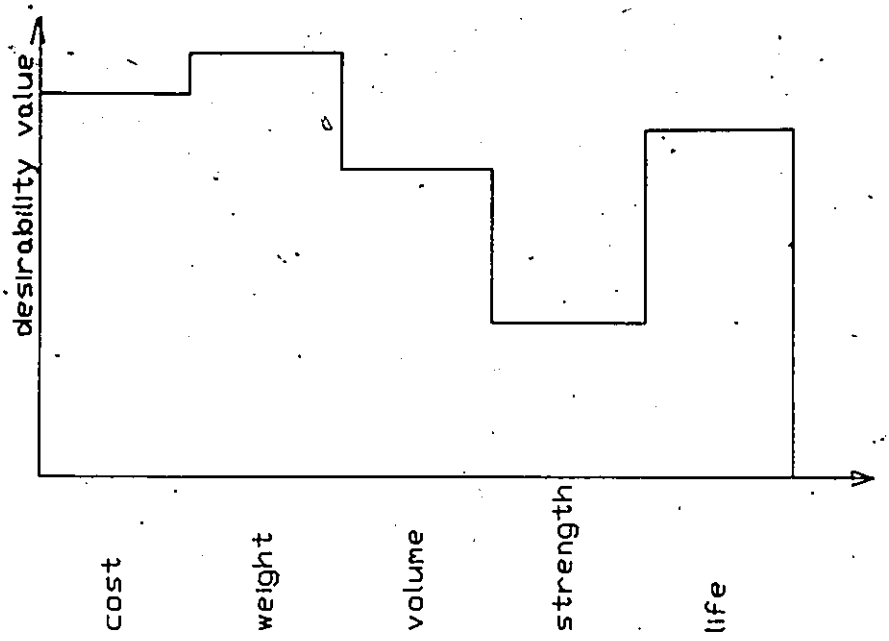
Figure 2.9(a) is a bar chart of the desirability values of a candidate. It can be converted roughly to a smooth curve in Figure 2.9(b), where the abscissa is the performance characteristics, and the ordinate is the corresponding desirability values. We call such a curve a desirability value curve.

Similarly, an input can be presented as an importance value curve, where the abscissa is still the performance characteristics and the ordinate is now the corresponding importance values.

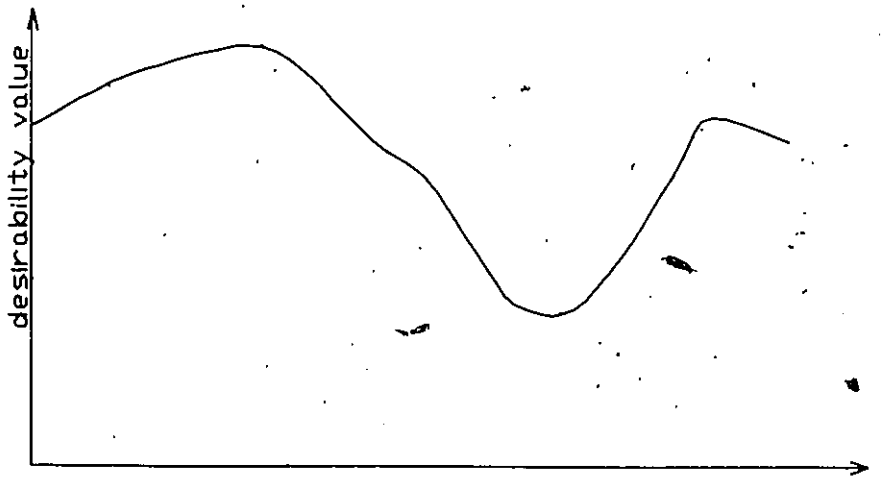
If we neglect the physical meaning of the desirability value curve and the importance value curve, the only characteristics left are the area under the curve, the mean, the standard deviation and higher order moments. These are designated curve characteristics, which are determined in fact by the original performance characteristics. The following example demonstrates that we can reach the same conclusion by just dealing with the curve characteristics instead of the original performance characteristics.

Suppose now there are two candidates as in Figure 2.10 (a), (b), and an input as in (c).

Sum of Desirability Values



(a) Bar chart



(b) Converted curve

Figure 2.9 Disirability value curve

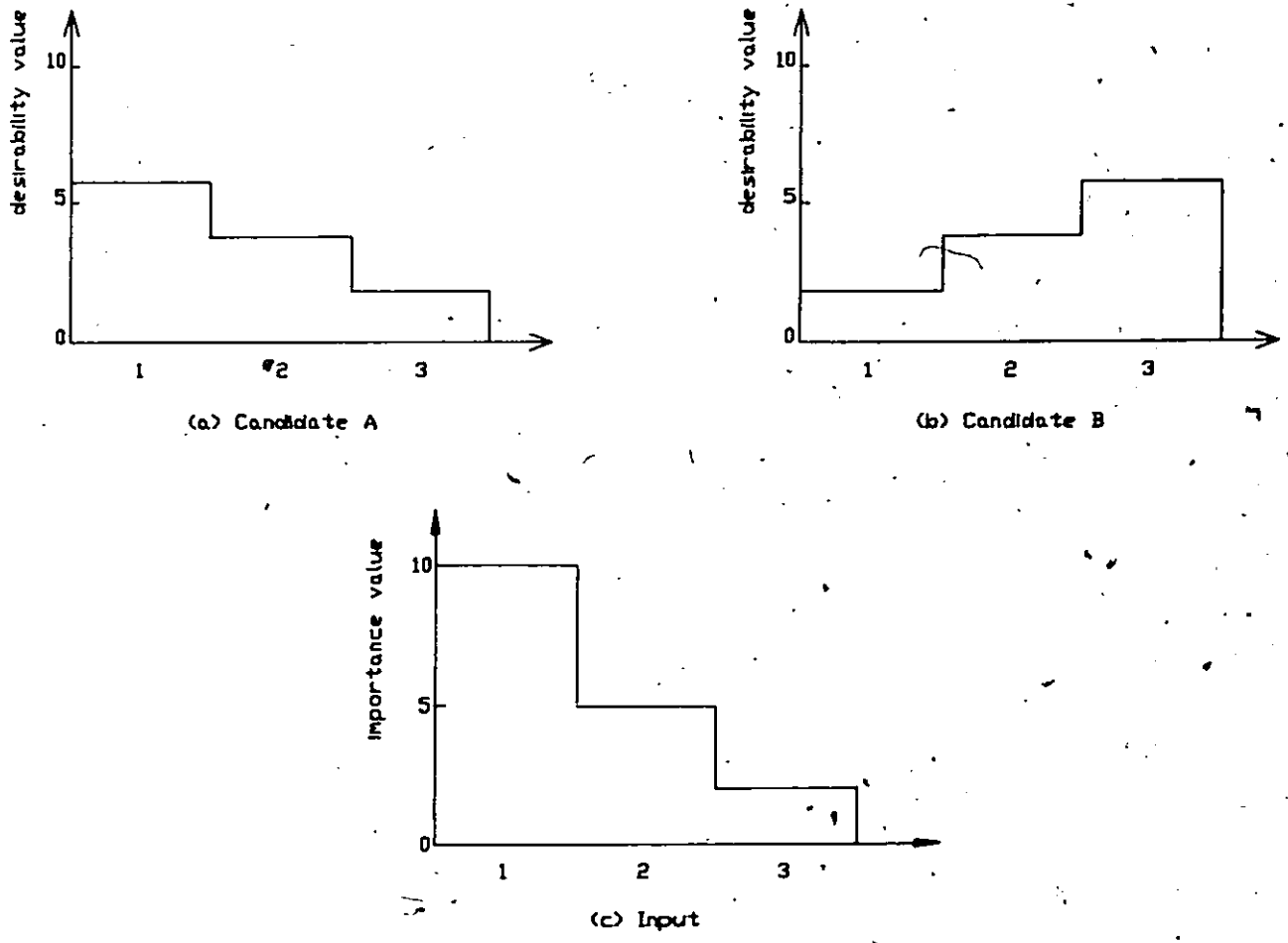


Figure 2.10 Candidates and Input

Candidate A: $6 + 4 + 2 = 12$

Candidate B: $2 + 4 + 6 = 12$ (2.36)

Merit Values

Candidate A: $6*10 + 4*5 + 2*2 = 84$

Candidate B: $2*10 + 4*5 + 6*2 = 52$ (2.37)

Mean of the Curves

Candidate A: $(1*6 + 2*4 + 3*2)/12 = 1.67$

Candidate B: $(1*2 + 2*4 + 3*6)/12 = 2.33$

Input: $(1*10 + 2*5 + 3*2)/17 = 1.53$ (2.38)

Variance of the Curves

Candidate A:

$[(1-1.67)^2*6 + (2-1.67)^2*4 + (3-1.67)^2*2]/12 = 0.56$

Candidate B:

$[(1-2.33)^2*2 + (2-2.33)^2*4 + (3-2.33)^2*6]/12 = 0.56$

Input:

$[(1-1.53)^2*10 + (2-1.53)^2*5 + (3-1.53)^2*2]/17 = 0.48$

(2.39)

Third Order Moment of the Curves

Candidate A:

$[(1-1.67)^3*6 + (2-1.67)^3*4 + (3-1.67)^3*2]/12 = 0.25$

Candidate B:

$[(1-2.33)^3*2 + (2-2.33)^3*4 + (3-2.33)^3*6]/12 = -0.25$

Input:

$[(1-1.53)^3*10 + (2-1.53)^3*5 + (3-1.53)^3*2]/17 = 0.32$

(2.40)

The following remarks can be made.

1. Although the sums of the desirability values of candidates A and B are the same, their distributions are different. So their merit values for an input are different.

2. Candidate A has higher merit value. At the same time, its curve characteristics are generally closer to those of the input, i.e. the shape of the desirability value curve of candidate A is more similar to the shape of the importance value curve of the input.

3. Now instead of using the performance characteristics to calculate the merit value, we may compare the differences of the curve characteristics between the candidates and the input. Since the number of the curve characteristics to be used (3 to 5) is much less than the number of the performance characteristics (25 to 30 in this project), the search procedure can be accelerated.

However the merit value of a candidate is also determined by the sum of the desirability values. Looking at Figure 2.11, we see that, although the shape of the desirability value curve of candidate A is more similar to the shape of the importance value curve of the input, the merit value of candidate A is obviously less than that of candidate B, because the area under the desirability value curve of candidate A is much less than that of candidate B.

Now we define a curve merit value to distinguish candidates using curve characteristics.

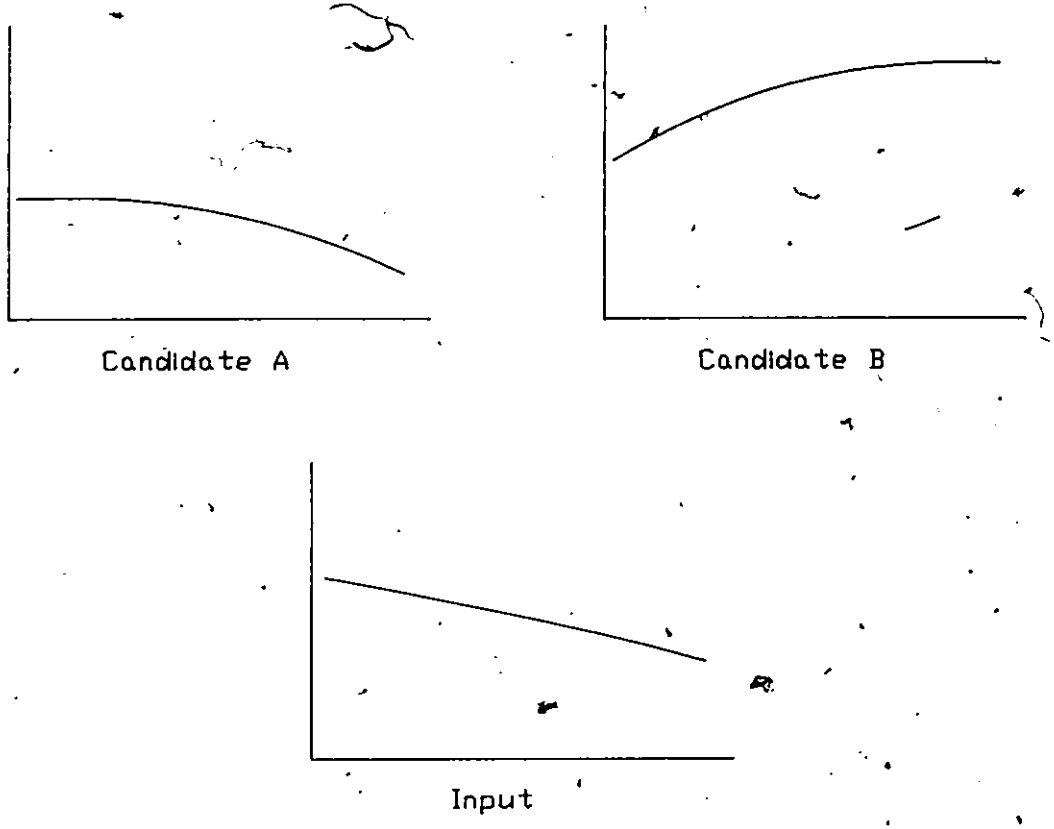


Figure '2.11' Influence of the sum of the desirability values

$$CM_i = SUM_i / d_i \qquad i=1,2,\ldots,N \qquad (2.41)$$

where SUM_i -- sum of the desirability values of the ith candidate, which is the same as SUM_i in formula (2.32) of the discard search method.

d_i ---- difference of the curve characteristics between the ith candidate and the input.

The candidate with the highest value of CM_i will be taken as the best.

$$d_i = \sqrt{(\mu_i-\mu)^2 + (\sigma_i-\sigma)^2 + \sum_{j=1}^{k} (R_{ij}-R_j)^2} \qquad (2.42)$$

where μ = mean

σ = variance

R_{ij} = function of the jth order moment of the ith candidate

K = highest order of moment to be involved

Two methods were tried to calculate R_{ij}

1. $R_{ij} = C_j / \sigma^j$ (2.43)

where C_j is the jth order moment. Here R_{i3} is the coefficient of skewness, R_{i4} is the coefficient of kurtosis.

2. $R_{ij} = \sqrt[j]{C_j}$ (2.44)

Numerical trials have shown formula (2.44) is better than (2.43), and also indicate that the highest order of moment to be calculated should be the third, fourth or fifth.

The bar chart of the desirability values or

importance values is usually random, as in Figure 2.7. This will degrade the validity of the curve characteristics. It was noticed that usually only the largest desirability values or importance values play a great part in determining the curve characteristics. So the E largest desirability values (for a candidate), or importance values (for an input), are selected and called the effective desirability values or effective importance values. It is equivalent to filtering the noise from the useful signal. Using only the effective desirability values and effective importance values also reduces the CPU time for evaluating curve characteristics.

Since this algorithm includes an uncertainty factor, we cannot expect the results from formula (2.41) to be exactly the same as those from formula (1.1). A method has been developed to check the validity of this algorithm.

When a test input is entered, the curve merit values are calculated, using formula (2.41) for all of the candidates. Then ND candidates are selected from the total N candidates with the highest curve merit values,

ND = max (N/5, 3)

These ND candidates are called a desired group. An array C(N) is set up and initialized to zero. Then the merit values for all of the N candidates are calculated, using formula (1.1). If the candidate with the highest merit value is in the desired group, then

C(1) = C(1) + 1

If the best candidate is not in the desired group, but the candidate with the second highest merit value is in the desired group, then

C(2) = C(2) + 1

..... and so forth. After NSA test examples have been tested, we let

C(I) = (C(I) / NSA) * 100% I=1,2,...,N (2.45)

Now by inspecting the value of array C, we have some idea about the validity of the quick search method. Following are two examples.

Example 1

Number of candidates = 10

Number of candidates in the desired group = 3

Number of performance characteristics = 25

Number of effective desirability values = 6

Highest order of moment to be involved = 4

Results

C(1) = 51.1%

C(2) = 36.1%

C(3) = 9.5 %

C(4) = 2.4 %

C(5) = 0.8 %

C(6) = 0.1 %

C(I) = 0 I=7,8,9,10 (2.46)

It means that the probability that the best candidate is

involved in the desired group is 51.1%. The probability that the best candidate is not involved but the second best candidate is involved in the desired group is 36.1%, and so forth. It is not too bad, since the probability that the desired group includes at least the third best candidate is

$$C(1) + C(2) + C(3) = 96.7\% \qquad (2.47)$$

Example 2

Number of candidates = 50

Number of candidates in the desired group = 10

Number of performance characteristics = 25

Number of effective desirability values = 8

Highest order of moment to be involved = 3

Results

C(1) = 34.1%

C(2) = 29.4%

C(3) = 18.9%

C(4) = 9.8 %

C(5) = 3.9 %

C(6) = 2.2 %

C(7) = 0.7 %

C(8) = 0.5 %

C(9) = 0.3 %

C(10) = 0

C(11) = 0.1 %

C(12) = 0.1 %

$$C(I) = 0 \qquad I=13,14,\ldots,50 \qquad (2.46)$$

The probability that the desired group includes at least the third best candidate is

$$C(1) + C(2) + C(3) = 82.4\% \qquad (2.47)$$

Figure 2.12 is the flowchart of the quick search method. The curve characteristics of the candidates were evaluated and stored in the knowledge database of the expert system at the same time as the desirability values of the candidates were being evaluated.

Since formula (1.1) will be applied to all of the candidates in the desired group, they are finally ranked according to the true merit values, not the curve merit values. So if the candidate with the highest merit value has been included in the desired group (whether or not it is the first one or last one in the desired group), resulting from the quick search method, it will be finally pushed to the first by the merit formula (1.1).

We can summarize as follows.

1. The quick search method discards 80% of the weaker candidates quickly by evaluating their curve merit values, (since they deal with only a few curve characteristics). It evaluates merit values for only 20% of the candidates in the desired group.

2. The probability that particularly good candidates are retained is high.

3. The method can be used for any search problems, disregarding its physical meaning.

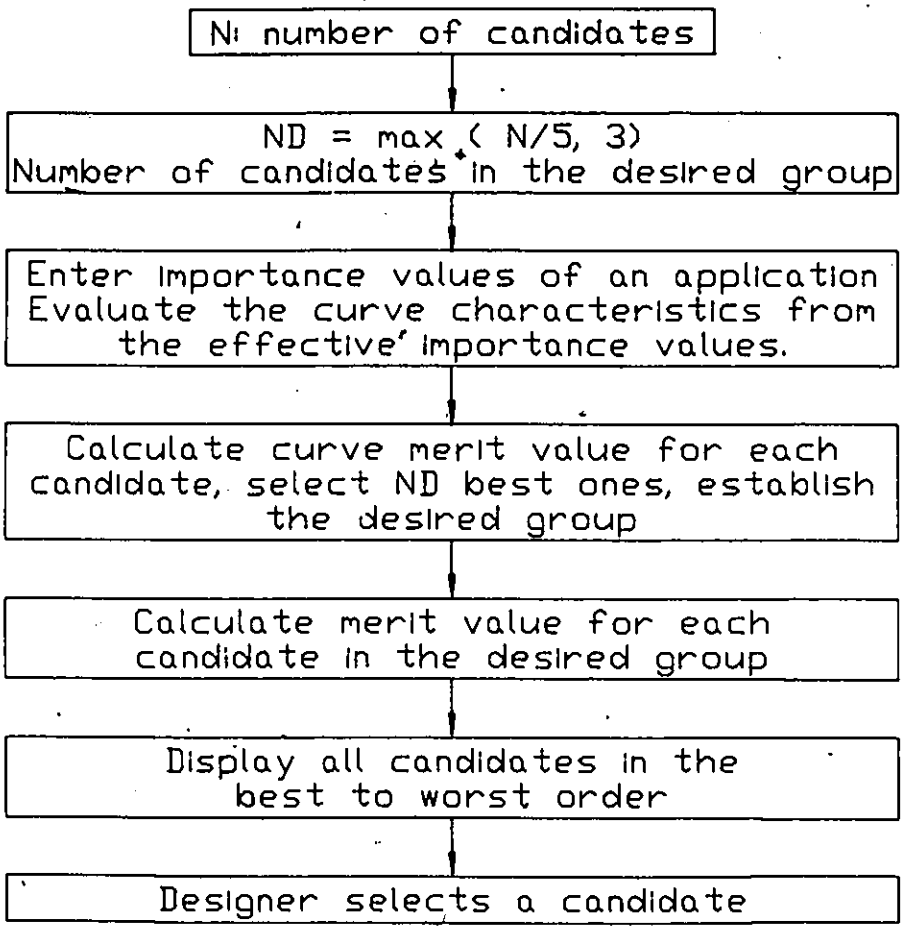


Figure 2.12 Flowchart of the quick search method

In this project, the CPU time spent on searching a candidate is only a minor part of the CPU time required for a design. However in the future, if the number of candidates increases a lot, the search problem is critical. As a research project in expert system, it is worthwhile to develop new knowledge to accelerate the search procedure.

A comparison of accuracy and search speed of the three methods will be shown in the case study in chapter 6.

2.2.5 Class and Subclass Design Candidates

There is a technique in industry called "Group Technology", in which the basic idea is to classify the general structures of parts into groups and subgroups in order to increase the efficiency of design and manufacture. The concept of class and subclass is analogous to this idea. Taking material as example, we first divide all of the materials in the database into broad classes, e.g. plain carbon steel, gray iron, and so on.

It is the common way for a human expert to first determine a class of material, and then go further down to select one of the many subclasses within this determined class. For example, to design a structure, an engineer may first decide to use plain carbon steel, and then select AISI-SAE 1020 within plain carbon steel. It is equivalent to first reaching an intermediate conclusion and then proceeding to the next stage.

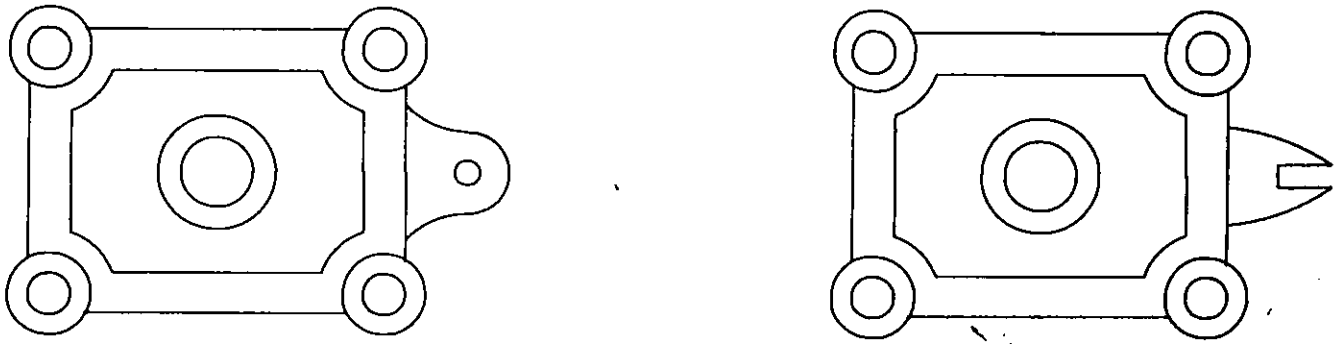
One purpose of dealing with class and subclass candidates is to save search time. Suppose there are ten classes of materials, and each of them has ten subclasses. If we search directly in the subclass level, we shall face 100 candidates at the same time. The alternative method is first to search in the class level, where we face only ten possible objects. After the class has been determined, we go on searching through the ten subclasses within the determined class. Obviously, the latter method saves a great deal of search time.

Another purpose is to save the effort of the human expert in developing the knowledge database of the expert system and the computer memory for the database. Let us look at the example for a configuration database.

The two structures in Figure 2.13 (a) and (b) are very similar except for a small difference in the right side. So they would be included in a configuration class. The human expert creates the configuration database by drawing the configurations directly at the CRT screen, which is supported by an interactive graphics program in the super-expert system. Instead of developing and storing the two configuration subclasses independently, the human expert accomplishes it in a different way.

1. He first draws all of the features of subclass 1 of this configuration class as in Figure 2.14 (a).

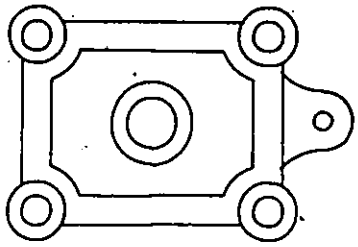
2. He hits the S key to indicate the completion of



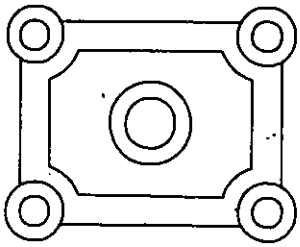
(a) Subclass 1

(b) Subclass 2

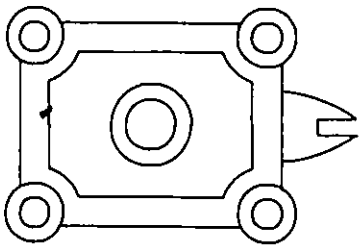
Figure 2.13 A configuration class



(a) Subclass 1



(b) Trunk



(c) Subclass 2

(d) Two branches

Figure 2.14 The decomposition of a configuration class

subclass one.

3. Then he erases the right element of subclass 1 (Figure 2.14 (b)).

4. He adds a right element to Figure 2.14 (b) to complete subclass 2 as in Figure 2.14 (c).

5. He hits the S key again to indicate the completion of subclass two.

6. Finally the super-expert system processes the two subclasses automatically, decomposing them into one trunk (Figure 2.14 (b)) and two branches (Figure 2.14 (d)). Only the trunk and the two branches are stored in the database.

Later a configuration can be retrieved from the trunk and its branch.

In summary, the class and subclass organization has three advantages.

1. It reduces search time.
2. It accelerates the building of the database.
3. It saves computer memory for the database.

2.3 BUILDING THE KNOWLEDGE DATABASE OF AN EXPERT SYSTEM

This section describes step-by-step how a human expert builds the knowledge database of an expert system for the design of a certain kind of structures, by executing the super-expert system. Besides being very knowledgeable in the design of that kind of structures, it is assumed that the human expert knows a little about optimization and the finite element method (FEM).

Figure 2.15 is the layout of the super-expert system. Its knowledge database contains a lot of information about the design of general structures. This information is written to the knowledge database of the expert system to be built, and becomes its initial or default knowledge. Later during the execution of each module of the super-expert system, the knowledge database of the expert system is continually updated and becomes finally complete. In the following description, the knowledge database usually refers to that one in the lower level expert system. It mainly consists of a lot of data files. The super-expert system actually fetches or updates the database by reading from or writing to the data files. Some of these data files are direct-access, e.g. those containing the desirability values, because the desirability values of an individual candidate should be easily accessible by the super-expert system at any time. Others are sequential-access data files, where data are usually read

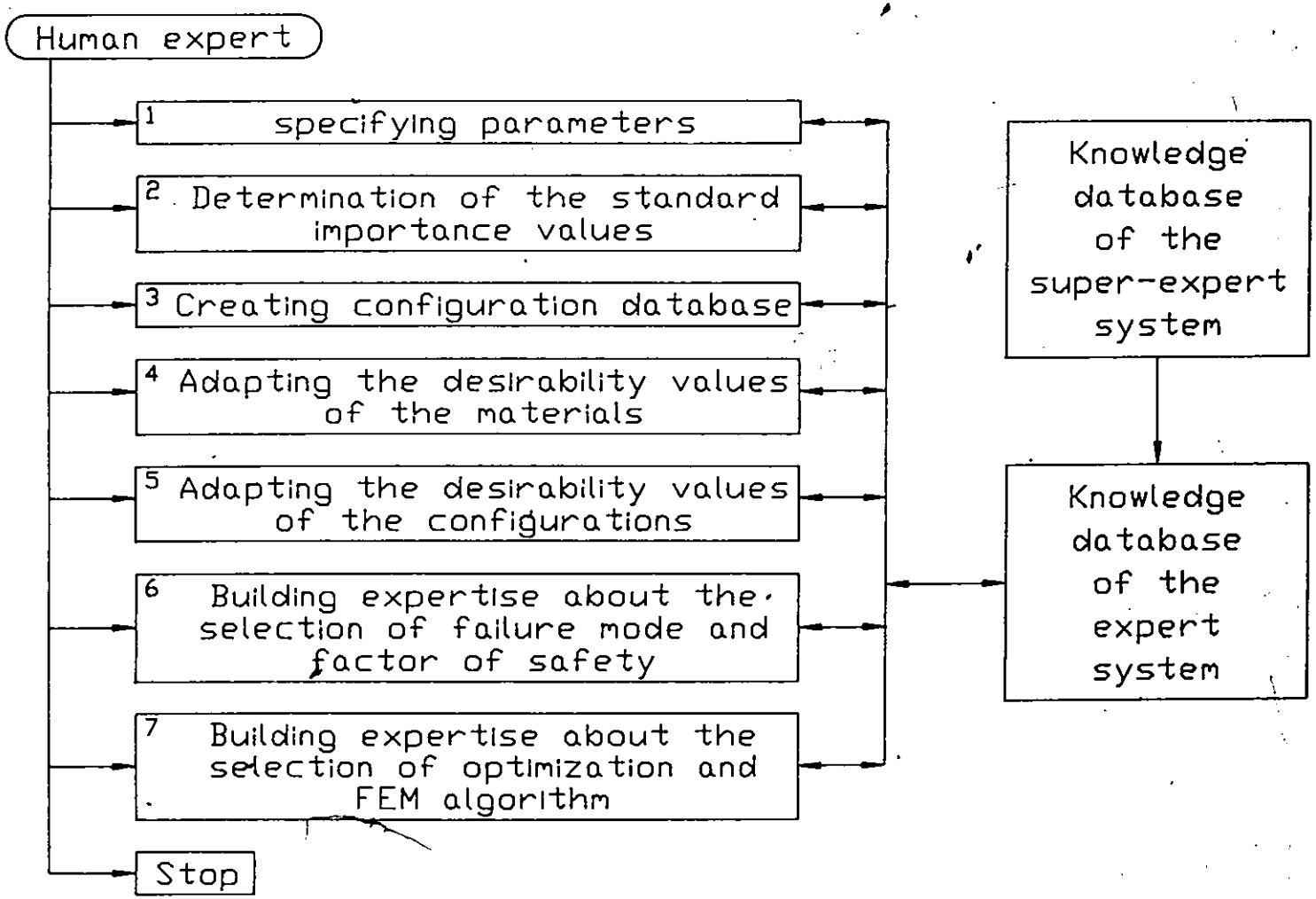


Figure 2.15 Layout of the super expert system

or written totally in one call.

Each one of the seven modules in the super-expert system takes part in building a certain part of the knowledge database. Since the whole job will take a relatively long time, the modules are designed to be quite independent of each other. Each interacts with the knowledge database only. So the human expert can execute the super-expert system in different runs, each time setting up one or more parts of the knowledge database. For example, he or she can specify the fundamental parameters and determine the standard importance values in the morning, and create the configuration database in the evening, leaving other jobs for later.

2.3.1 Specification of Fundamental Parameters

In this module, the human expert specifies some fundamental parameters, which include the type of the element in the FEM (e.g. plain stress element, axisymmetric element), the number of performance characteristics, the number of material or configuration candidates, and so on.

2.3.2 Determination of the Standard Importance Values

This module will determine the standard importance values for the expert system. They represent, as mentioned in an early section, the most typical input for the design applications of a certain kind of structures. Also, some

rules about the relationships of the performance characteristics will be set up.

2.3.2.1 Determination of the Performance Characteristics

The super-expert system first shows its 24 permanent performance characteristics to the human expert (Table 2.9). They represent almost all of the factors that should be considered for general structural designs. However there may be some special factors that should be considered as well for the design of certain kind of structures that have not been included in the permanent performance characteristics. They are called additional performance characteristics, which can be supplemented by the human expert, who types in their names, default importance values and a short text of interpretation for each.

Since this project concerns the preliminary design only, some problems such as manufacturing process or heat treatment will not be considered.

2.3.3.2 Determination of the Interpolation Values of Some Performance Characteristics

For some performance characteristics, it is easier for the designer to enter their direct values rather than the importance values. For example, a designer may know clearly that the highest working temperature is 2000^{o} F, but he is not quite sure what the equivalent importance value should be for this temperature. There are a total of six

Table 2.9 Permanent performance characteristics and their default importance values in the super-expert system

Performance characteristic	Default importance value
cost	10.0
weight	6.0
volume	6.0
accuracy	8.0
reliability	9.0
safety	10.0
low temperature	2.0
high temperature	5.0
humidity	2.0
corrosion	5.0
stiffness	6.0
load	8.0
frequency	5.0
impact	7.0
dimension	3.0
style	3.0
noise	0.0
surface finish	5.0
Wear	7.0
friction	3.0
lubrication	2.0
manufacturability	6.0
maintainability	7.0
service life	8.0

such performance characteristics,

lowest temperature

highest temperature

maximum load

maximum load frequency

overall dimension

service life

The direct values for the maximum load and the overall dimension at this point are just some approximate values which are used for the selection of material, configuration, and so on. The exact values of loads and dimensions for an application will be specified later.

The expert system will convert the direct values entered by the designer into the equivalent importance values by linear interpolation. As shown in Table 2.10, if the highest temperature is 210° F, the equivalent importance value is 4, if the highest temperature is 295° F, the equivalent importance value is 5.5.

The super-expert system provides the default interpolation values; however they should be adapted by the human expert. For example, a load of 100 pound for a clock would be too large; however the same load would seem very small for a huge mining machine.

When the human expert decides to revise the interpolation values for a performance characteristic, a table similar to Table 2.10 is displayed on the soreen. He

Table 2.10 Interpolation values of high temperature

Direct value (Fahrenheit)	Equivalent importance value
77	0
95	1
150	2
185	3
210	4
270	5
320	6
400	7
570	8
750	9
900	10

moves the crosshair to the interpolation values he wants to revise and types in the new values, and the old ones are erased automatically.

2.3.2.3 Determination of the Rules

There are two kinds of rules in the expert system: interactive rules and number rules. Interactive rules deal with the influences between performance characteristics. Number rules deal with the influences of the number of products on other performance characteristics.

When the condition of a rule is satisfied, instead of enforcing an action, the expert system will display a short message to the designer, and let him to decide if an action is accepted or rejected.

All of the rules take care of the risk that the designer may neglect or is unaware of some interactions between the performance characteristics. However the knowledge level of designers may be very different. A very skillful designer may have already considered all of these interactions when he or she specified the importance values. In this situation, an enforcing revision of the existing importance values is absolutely redundant. So just a suggestion is better and more reasonable.

Fourteen interactive rules are defined in the super-expert system.

1. If $P_{safe} > 7.$ then $P_{reliability} > 7.$

2. If $P_{safe} > 8.$ then $P_{reliability} > 8.$
3. If $P_{safe} > 9.$ then $P_{reliability} > 9.$
4. If $P_{frequency} > 6.$ then $P_{sur.\ finish} > 7.$
5. If $P_{frequency} > 8.$ then $P_{sur.\ finish} > 9.$
6. If $P_{frequency} > 6.$ then $P_{corrosion} > 7.$
7. If $P_{frequency} > 8.$ then $P_{corrosion} > 9.$
8. If $P_{corrosion} > 8.$ then $P_{style} < 4.$
9. If $P_{corrosion} > 5.$ then $P_{maintainability} > 6.$
10. If $P_{corrosion} > 7.$ then $P_{maintainability} > 8.$
11. If $P_{wear} > 7.$ then $P_{lubrication} > 8.$
12. If $P_{friction} > 7.$ then $P_{lubrication} > 8.$
13. If $P_{service\ life} > 6.$ then $P_{reliability} > 7.$
14. If $P_{service\ life} > 8.$ then $P_{reliability} > 9.$

(Note: P_A means the importance value of A)

The rules are displayed on the screen as in Figure 2.16. The human expert can delete, accept, modify them or create his own new rules, all by moving the crosshair to the appropriate position and typing.

Sometimes the human expert may make some mistakes in modifying the default interactive rules or creating new ones. The super-expert system can check such errors and warn the human expert. For example, suppose there are three new interactive rules defined by the human expert.

1. $P_1 > 3.$ then $P_2 > 4.$
2. $P_2 > 4.$ then $P_3 < 6.$

Psafety	LARGER	7.0	Preliability	LARGER SMALLER	7.0
	LARGER	8.0		LARGER SMALLER	8.0
	LARGER	9.0		LARGER SMALLER	9.0

(a) Example 1: Rules 1, 2 and 3

Pcorrsion	LARGER	8.0	Pstyle	LARGER SMALLER	4.0
	LARGER			LARGER SMALLER	
	LARGER			LARGER SMALLER	

(b) Example 2: Rule 8

Figure 2.16 The rules on the screen

3. $P_3 > 6.$ then $P_1 < 3.$

Here P_1 modifies P_2, P_2 modifies P_3, and finally P_3 modifies P_1. The expert system does not accept such recursive rules since they may cause trouble and instability. The code in the super-expert system to check such recursive rules is recursive programming itself, and is implemented easily in Fortran.

Finally the super-expert system rearranges the sequences of the rules to avoid an undesired interaction. For example, let us look at following two rules.

1. $P_1 > 5.$ then $P_4 < 6.$
2. $P_3 > 7.$ then $P_1 > 6.$

The expert system will executes rule 1 first, and then rule 2. They are not recursive rules; however when executing rule 2, P_1 may be changed, therefore rule 1 has to be executed again. The super-expert system will shift the sequences of these two rules in order to avoid this.

The super-expert system provides no default number rules. They are specific to each expert system and have to be defined by the human expert. Following is a typical number rule shown on the screen.

MIN	MAX	PC	FACTOR
200	1000	cost	1.2
1000	-	cost	1.5

This means that if the number of products is larger than 200 and less than 1000, it is suggested that the importance

value of cost be multiplied by a factor of 1.2; if larger than 1000, it is multiplied by a factor of 1.5. The number rules can be created in a similar way as the interactive rules at the CRT screen.

2.3.2.4 Modification of the Standard Importance Values

The super-expert system provides the default standard importance values which are considered to represent the most typical specifications for the design of general structures. The human expert converts these to the standard importance values of his expert system by making necessary modifications. The detailed procedure will be described in section (4.3), where the designer calls the same program subroutine to adapt the standard importance values to his design application. It is a typical application of the two-level expert system; also see Table 2.11.

2.3.3 Establishment of the Configuration Database

In this module, the human expert establishes the configuration database of his expert system, and the design variables and functions which closely relate to the configurations. In the present state of the art of computer science, we do not expect that a computer can imagine and create a configuration by itself. It is the duty of the human expert to create and store the desired configurations in the knowledge database of his expert system. It is, of

Table 2.11 Bar chart subroutine in the two-level expert system

	Super-expert system	Expert system
User	Human expert	Designer
Input	Default importance values	Standard importance values
Output	Standard importance values	Importance values for a design application

course, a big job. However if the developed expert system is to be applied in a large society and for a long period, this effort will pay off. The human expert creates a configuration by "drawing" it on the screen. The supporting program subroutine will be described in section (4.2), where the designer enters his own configuration by calling the same subroutine.

The configurations are created, organized and stored in classes and subclasses as discussed early in section (2.2.5). Each configuration class can be created independently in separate computer runs.

The human expert can pick the possible material candidates from the permanent material database of the super-expert system. In the present project, the permanent material database contains nine classes of metal materials: plain carbon steel, alloy steel, stainless steel, gray iron, ductile iron, wrought aluminum alloy, cast aluminum alloy, brass and bronze.

2.3.4 Adapting the Desirability Values of the Materials

At the start, all of the material classes in the material database of the expert system are displayed on the screen. The human expert decides whether to adapt the desirability values in a class level or in a subclass level within a material class. Then he has to decide whether to use an existing sample (available design records or a sample

created before) or create a new sample. If he selects the latter, he will do that by following the algorithm described in section (2.2.3).

When the sample is available, the human expert divides them into several groups and lets the computer work out the desirability values using linear programming.

Finally in this module, the super-expert system will

1. Normalize all of the desirability values to the range from 1.0 to 10.0.

2. Calculate the array SUM, where SUM(I) is the sum of the desirability values of the Ith candidate.

3. Calculate the curve characteristics.

These data are prepared for the discard search method or the quick search method as discussed in section (2.2.4).

2.3.5 Adapting the Desirability Values of the Configurations

This module is quite similar to the above one for the material candidates, except that when the human expert is required to make a choice, all of the configuration candidates, instead of the material properties, will be displayed on the screen. If the human expert wants a closer view of any of them, he can point to it, and the super-expert system will enlarge this configuration to the full size of the screen.

2.3.6 Determination of the Failure Mode and Factor of Safety

The failure mode here refers to the theory of failure. In the super-expert system, there are two failure modes for ductile materials,

1. the distortion energy theory
2. the maximum shear stress theory

and two for brittle materials,

1. the modified Mohr theory
2. the Colulomb-Mohr theory

Some other failure modes, such as buckling, excessive displacement, and so on, are included in the constraint functions.

The accurate value of the factor of safety (FS) can be determined only by extensive and expensive testing on the prototype made of the same material and same size as the structure to be designed, under the same environment and loading condition. This is particularly true for the fatigue problem. However our system is developed to design a lot of different kinds of structures, selecting different material and configuration from the expert system database, or even adopting a material and configuration entered by the designer. Furthermore the configurations in the database are just conceptual ones; the concrete dimensions are determined later by the designer for different applications. Therefore, it is impossible to test an infinite number of prototypes in order to get an accurate FS.

In industrial practice, if a production batch number is small, intensive and expensive testing is not worthwhile. The FS is generally determined by reference to a similar, successful design. The FS thus determined can serve at least as a starting value for testing.

In our basic algorithm for determining the FS, the human expert provides a reference factor of safety; and the designer can accept this value or do some modification by his own judgement. The reference FS tends to be a little conservative, and in the case of a novice designer with little experience, it would be accepted without any modification. If the human expert (or the designer) defines a FS less than 1.25 or greater than 10., the super-expert system (or the expert system) will display a warning message.

Following are the five steps for the human expert to determine a reference FS.

1. Determination of the standard material and configuration

The human expert determines a failure mode and the reference value of FS based on the standard material and the standard configuration which are considered to be the most desirable ones by the human expert among all of the materials and configurations in the database. They are found by the full search method according to the standard importance values (determined in section (2.3.2)) and the desirability values (determined in section (2.3.4) and

(2.3.5)).

2. Determination of the failure mode and the basic FS

The name and properties of the standard material is displayed on the left of the screen, and the standard configuration on the right. Knowing it, the human expert selects a failure mode for ductile materials and another for brittle materials, since the designer can select any material for his design. Then the human expert provides a so called basic factor of safety for the standard material and configuration based on prototype testing or confident expertise. This basic FS might be modified little by little and finally becomes the reference factor of safety provided to the designer by the human expert. At the same time, he can type in several lines of texts to explain the determination of the basic FS. It can be shown to the designer later if he asks for it.

3. Modification of the FS due to design specifications

The determination of FS is very complex, even an expert may sometimes forget some factors relating to it. The super-expert system reminds the human expert of such factors one by one. If he had taken a particular factor into account already, he can just ignore the reminder. Otherwise he can adjust the basic FS in one of two ways.

(1) By setting FS to a certain value

(2) By multiplying the current FS by a factor

In the present project, we consider the following factors: reliability, safety, lowest-temperature, corrosion, impact, uncertainty of load, material properties and environment. If the load cycle is larger than 1000, fatigue may occur. More factors should be considered, i.e. surface finishing, humidity, highest-temperature and stress concentration. Some of the above factors, e.g. reliability, are performance characteristics. So whether a reminder about such a factor should be evoked depends on its importance value. For example, if the importance value of reliability is 9.5, the system will remind the human expert,

> "The requirement on reliability is very high, would you consider increasing the factor of safety?"

On the other hand, if the importance value of reliability is 2, no reminder is evoked.

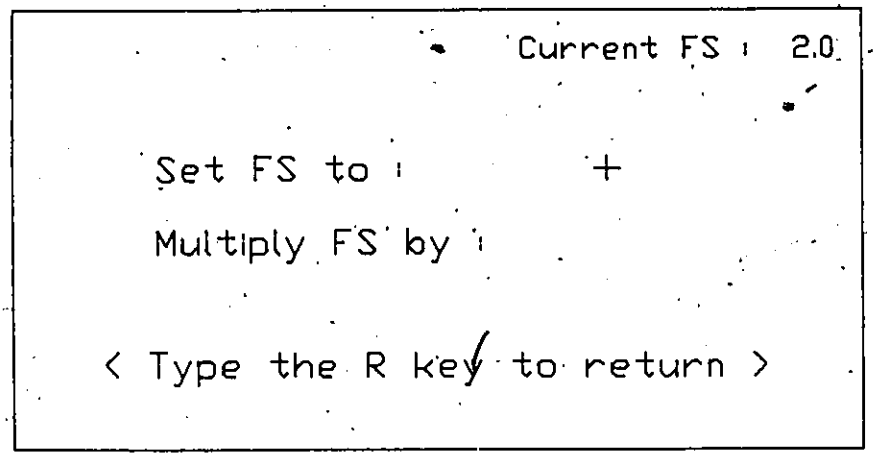
4. Modification of FS due to a specific material or configuration

It must be remembered that the reference FS is based on the standard material and configuration. The designer can select any material and configuration in the database of the expert system. When a specific material or configuration is selected, some modification of the FS may be necessary. Material selection provides an example. The super-expert system displays all of the material classes and

their typical properties on the screen. Suppose plain carbon steel AISI-SAE 1020 is the standard material and the current FS is 2. Now the human expert finds that some modification is necessary if the second material class, i.e. alloy steel, is adopted by the designer. He types 2, and the screen appears as in Figure 2.17. He moves the crosshair to the first line and types 2.5. That means if alloy steel is adopted, the reference FS should be set to 2.5. Then all of the subclass materials in the alloy steel class will be displayed on the screen. The human expert can do a similar modification on the FS as was done in the material class level. For example he can specify that, if alloy steel AISI 4140 is adopted, the reference should be multiplied by a factor of 2. So the reference FS is now 2.5 * 2 = 5. Modification of the reference FS due to specific configurations can be done similarly. For each modification, the human expert can type in a one line explanation.

5. Final check

Finally the human expert can go back to check the whole process, to ensure that the reference FS has been created and modified. The explanation provided by the super-expert system or by himself can serve as the memo.



Note: + is the crosshair

Figure 2.17 Modification of the factor of safety

2.3.7 Determination of Optimization and Finite Element Method Algorithm

In this final module, the human expert determines the optimization method, the penalty function, the optimization parameters, and the algorithm that combines the finite element method with optimization, in consultation with the super-expert system.

In the present project, there are five basic optimization methods.

(1) Hooke and Jeeve's direct search method

(2) Reduced gradient method

(3) Powell's direct search method

(4) Jacobson and Oksman Method

(5) Adaptive random method

These five methods represent many of the currently available optimization methods. Other methods can easily be added.

Six algorithms are adopted to save computer time for finite element based optimization.

(1) Substructure method

(2) Global Direction method

(3) Skipping method

(4) Safe-Fail Lines method

(5) Quadratic method.

(6) Differential method

For each specification, the super-expert system will

display all of the options to the human expert and recommend the best. The human expert can either accept the recommendation or select andther option by his own judgement. There is a text for each specification to describe the advantage and disadvantage of each option. These texts can be reached at any time by typing a key H(elp).

The interface is very user friendly. For example, when dealing with an optimization method, all of the options are displayed. The crosshair is located initially on the option recommended as the best by the super-expert system. If the human expert accepts it, he just type key G. Otherwise he moves the crosshair to his desired option and then types key G. If the specification involves some numbers (e.g. parameters of optimization subroutines), all of the recommended (default) values are displayed. The human expert can either accept them or type in new values that he feels are more suitable for his expert system.

Now the knowledge database of the expert system is complete. Before describing how a designer applies the expert system to a design application, we shall first discuss the algorithms for combining the optimization with the FEM in the next chapter.

CHAPTER 3

COMBINATION OF THE OPTIMIZATION AND THE FINITE ELEMENT METHOD

3.1 INTRODUCTION

Both the optimization and the finite element methods (FEM) are advanced techniques in computer aided design.

Optimization provides the optimum design of a structure, which will satisfy all of the specifications. Usually the most important specifications are related to the maximum stress and the maximum displacement.

The finite element method, on the other hand, provides the means for calculating the stresses and displacements of a structure accurately no matter how complex its configuration is.

By combining the optimization with FEM, it is possible to get the optimum design of a complex structure. However when this is done, the computer time required increases by a considerable amount, and tends to become prohibitively expensive. This is because the time used for the FEM calculation has to be performed as many times as the number of the optimization iterations. In this project, we have implemented six algorithms for saving computer time (we shall call them FEM algorithms), they are

1. Substructure method

2. Global Direction method
3. Skipping method
4. Safe-fail Line method
5. Quadratic method
6. Differential method

The last four are new algorithms developed in this project.

In the first two sections of this chapter, we shall give a brief introduction to the optimization and the finite element methods.

3.2 THE OPTIMIZATION SUB-SYSTEM

In the optimization, we look for the maximum or minimum value of a performance characteristic, examples of which are cost, weight, size, and so on of a structure. We may generalize and summarize the optimization or objective function

$$U = U(X_1, X_2, \ldots, X_n) = \text{maximum or minimum} \quad (3.1)$$

where U is the desired performance characteristic, which is defined as the function of the design variables X_1, X_2, ..., X_n. Usually we only cope with the minimum problem, the maximum problem can be dealt with by minimizing its negative value.

We shall also formulate the feasibility expressions in the form of equality and inequality constraints having the general form

$$h_i = h_i(X_1, X_2, \ldots, X_n) = 0 \qquad i=1,m \qquad (3.2)$$

$$g_j = g_j(X_1, X_2, \ldots, X_n) > 0 \qquad j=1,p \qquad (3.3)$$

In nonlinear optimization, the equality constraints are usually converted to inequality constraints.

In this project, we have adopted five optimization algorithms and four penalty functions directly from the optimization package OPTIVAR (Siddall, 1982).

The five optimization algorithms are,

1. Hooke and Jeeve direct search method. It is a typical direct search method, combining the local exploratory searches along the coordinate directions with some accelerated fashion of pattern moves.

2. Reduced gradient method. It is a technique for handling constraints with any of the methods which use successive one-dimensional minimizations.

3. Powell's direct search method. It is a direct search method along the conjugate directions with quadratic convergence.

4. Jacobson and Oksman Method. It is a quadratic method based on homogeneous functions.

5. Adaptive random method. It is a random method of line searches with both the search directions and step sizes selected randomly.

These five methods represent many of the currently available optimization methods.

The four penalty functions are,

1. One pass exterior penalty function.
2. Penalty function based on the Fiacco-Mccormick method.
3. Penalty function based on Powell's method.
4. Penalty function based on Schuldt's method.

An optimum value with good convergence will be obtained by adopting the best combination of the optimization method and penalty function. However the present state of art gives no general guidance in the choice, which is usually determined by experience and experiments.

3.3 THE FINITE ELEMENT SUB-SYSTEM

The finite element method is essentially a process through which a continuum with infinite degrees of freedom can be approximated to by an assemblage of elements, each with a specified but now finite number of unknown displacements.

The displacements of any point within an element are expressed as a vector [D], and the displacements of the n nodes of the element as a vector [q]. For a plain stress element,

$$[D] = [\ u\ v\ ]^t \tag{3.4}$$

$$[q] = [\ u_1\ v_1\ u_2\ v_2\ \ldots\ldots\ u_n\ v_n\ ]^t \tag{3.5}$$

where u is the displacement in x direction, v in the y

direction. Also,

$$[D] = [N]\ [q] \tag{3.6}$$

where [N] is the matrix of shape function.

A strain-displacement matrix [B] is defined as,

$$[B] = [\ B_1\ B_2\ \ldots\ldots\ B_n\ ] \tag{3.7}$$

For a plain stress element,

$$B_i = \begin{bmatrix} \partial N_i/\partial x & 0 \\ 0 & \partial N_i/\partial y \\ \partial N_i/\partial y & \partial N_i/\partial x \end{bmatrix} \tag{3.8}$$

The strains at a point within the element can be evaluated as a vector $[\epsilon]$,

$$[\epsilon] = \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \partial u/\partial x \\ \partial v/\partial y \\ \partial u/\partial y + \partial v/\partial x \end{bmatrix} = [B]\ [q] \tag{3.9}$$

The vector of stress at a point, $[\sigma]$, can be related to the vector of strains $[\epsilon]$ by the application of Hook's law,

$$[\sigma] = \begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{bmatrix} = [C]\ [\epsilon] = [C]\ [B]\ [q] \tag{3.10}$$

where [C] is an elasticity matrix which is symmetric. For a plain stress problem,

$$[C] = E/(1-\nu^2) \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix} \tag{3.11}$$

where E is modulus of elasticity and ν is Poisson's ratio.

The equilibrium equations can be derived following

Castigliano's first theorem or the variational principle of the potential energy.

$$[K]\ [q] = [R] \tag{3.12}$$

where [K] is the stiffness matrix and [R] is the load vector.

$$[K] = \int_V [B]^t\ [C]\ [B]\ dv \tag{3.13}$$

The load vector can be acquired from concentrated loading or distributed loading.

Now by assembling the global stiffness matrix and load vector from those of the elements, we can solve the equilibrium equations of the structure, usually using the Gauss elimination method.

In this project, we have adopted four types of two-dimensional elements: plain stress, plain strain (Figure 3.1), plate bending (Figure 3.2) and axisymmetric (Figure 3.3) elements. They are all isoparametric elements with eight nodes per element. Table 3.1 gives the numbers of the basic parameters of these elements.

3.4 BRIEF REVIEW OF THE SUBSTRUCTURE METHOD AND THE SKIPPING METHOD

3.4.1 The Substructure Method

This method was developed in the author's master project (Wu, 1986). It divides the whole structure into several substructures which are defined as absolutely

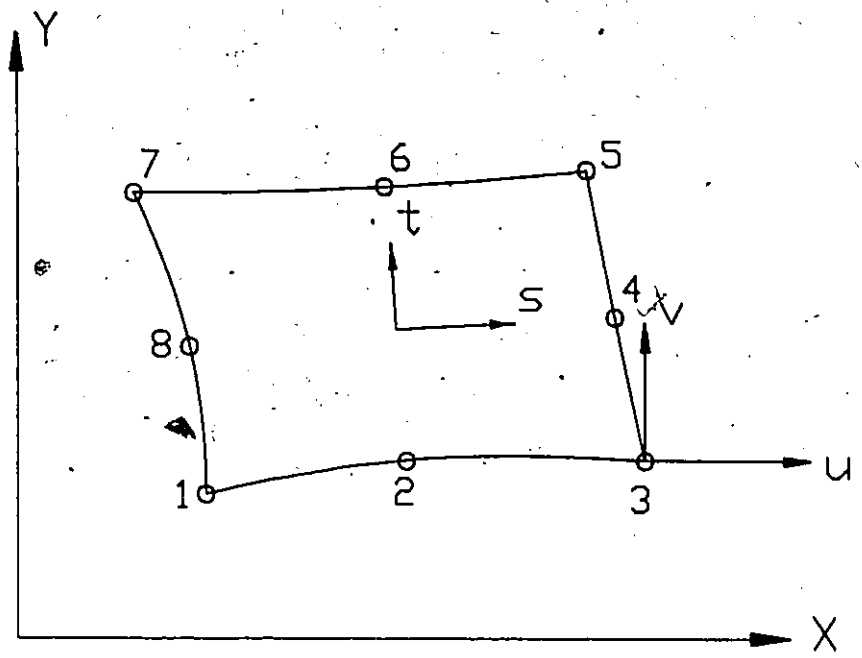


Figure 3.1 Plain stress and plain strain elements

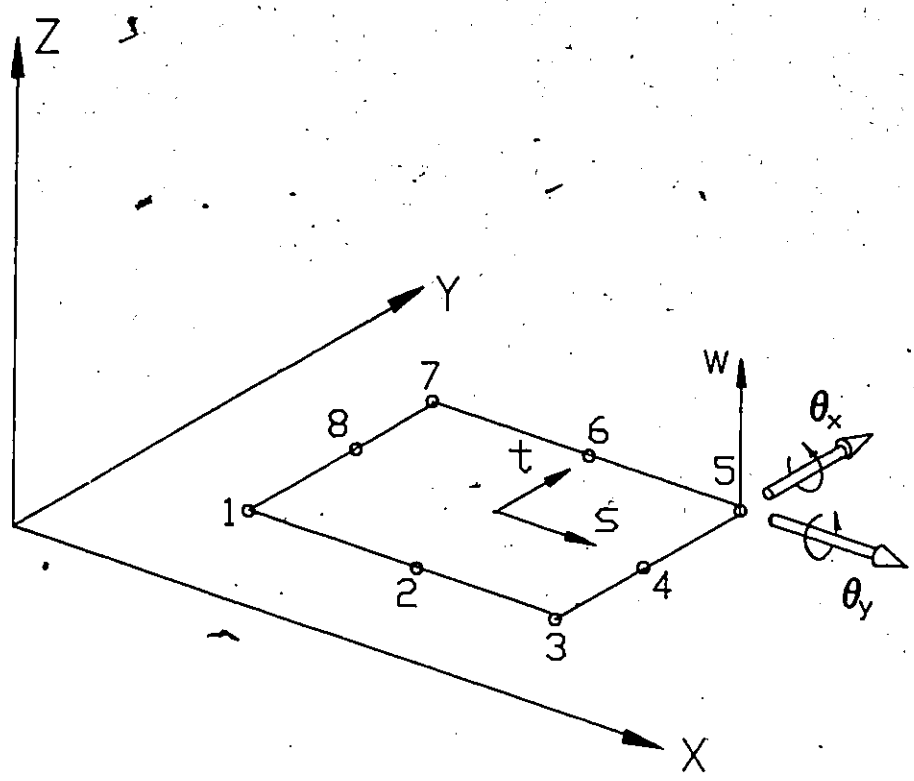


Figure 3.2 Plate bending element

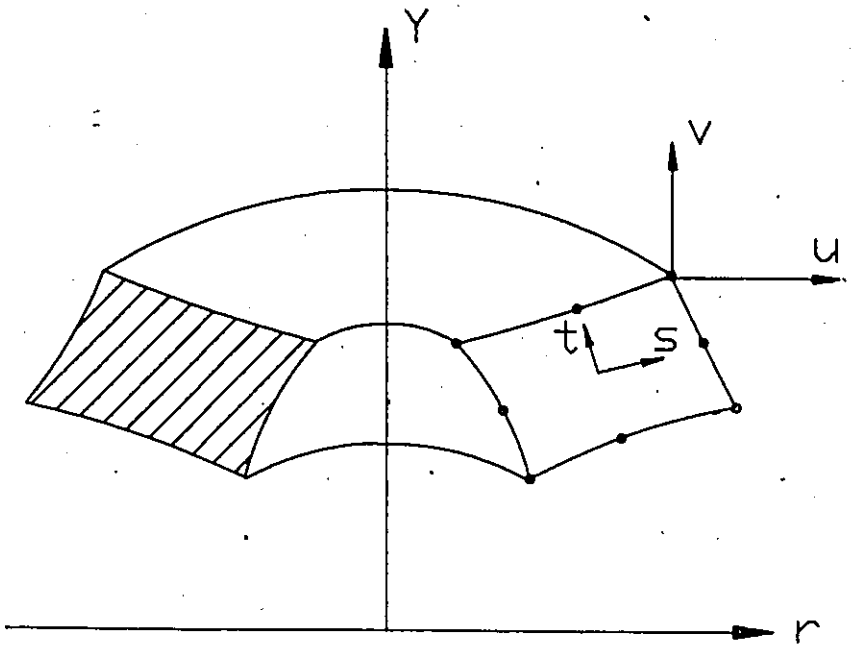


Figure 3.3 Axisymmetric element

Table 3.1 Basic parameters of elements

Name	Dimension	Degrees of freedom per node	No. of stress components
Plain stress	2	2	3
Plain strain	2	2	3
Plate bending	2	3	5
Axisymmetric	2	2	4

constant, conditionally constant, thick-type or non-constant substructures respectively. Different algorithms apply to them in order to save CPU time.

Only the concept of the absolutely constant structure (constant substructure in short) is adopted in this project, since it is quite easy for it to be defined in an automatic program and it makes a large contribution to saving CPU time.

During the optimization search, there is no change occurring in the equilibrium equations of the constant substructure. Therefore we need to construct them only in the first iteration of optimization, then condensing them immediately, and storing the condensed stiffness matrix and load vector of the external nodes. Later in every subsequent iteration, we only add these condensed stiffnesses and loads to the global stiffness matrix and load vector, without dealing with all other tedious computations within the constant substructure.

3.4.2 The Skipping Method

This method is the implementation of the following quote (Siddall, 1982).

> "One possible trick to save computer time, if only some constraints require a great deal of computation, is to ignore them if any other constraints are violated at any stage of the search (Suggested by W. Michael)."

In our program, the constrains requiring a great deal of computation are those whose evaluations are based on the FEM calculation.

The skipping method is very simple, but very effective sometimes.

3.5 THE GLOBAL DIRECTION METHOD

In this method, an attemp is made to conduct the optimization search not only by using the information about the local topography, but also by using some global information. Suppose a climber is walking in a mountain area towards to the highest peak. If it is a sunny day, he can always have the highest peak in sight, the probability of his success must be higher than if he were groping his way with the help of a flashlight in a dark night. Similarly in an optimization search, if the unconstrained global optimum is known, it should be easier to reach the constrained global optimum.

In Figure 3.4, the search begins at point S. If only local topography is made use of, the search will lead to point B, the local optimum, since the rising slope is towards B rather than A at point S. However if we know the unconstrained global optimum A, the search will lead to point C, and then point D, which is the real constrained optimum.

A distinct characteristic of FEM based optimization

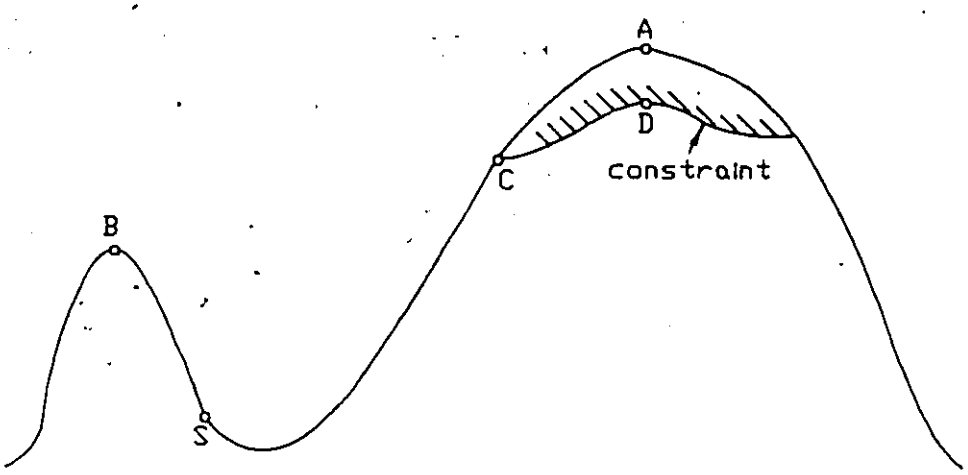


Figure 3.4 Climbing the mountain

is that most of the CPU time is spent in the FEM calculations. An unconstrained optimization run almost takes no time compared to the constrained optimization run. So it is worthwhile to search for the unconstrained optimum first and use it as a guide for the constrained optimization run. Following is the basic algorithm.

1. Randomly determine a starting point within the upper and lower bounds of the design variables.
2. Run an unconstrained optimization (it means excluding the FEM based constraints, but still including all other simple constraints).
3. IF (result is a feasible optimum solution) THEN

   GO TO 4

   ELSE

   GO TO 1

   END IF
4. Store this optimum solution and the corresponding design point.
5. IF (there are three optimum solutions stored already) THEN

   GO TO 6

   ELSE

   GO TO 1

   END IF
6. Determine the minimum optimum solution among the three. It is taken as the global unconstrained optimum. Its

corresponding design point is $\bar{X}_u$.

7. Use bisection to approximate the constrained optimum. Suppose the starting point of the application (the original constrained problem) is $\bar{X}_s$

```
DO 10 I = 1, NBS            ; NBS is the number of
                            ; bisections
  X̄t = 0.5 * ( X̄u + X̄s )
  Evaluate all of the constraints, including the FEM
  based constraints at X̄t
  IF ( X̄t is feasible ) THEN
    X̄s = X̄t
  ELSE
    X̄u = X̄t
  END IF
10 CONTINUE
```

8. Using the last $\bar{X}_s$ as the new starting point, run the original constrained optimization problem.

Figure 3.5 illustrates the concept, showing that after three unconstrained optimization runs, the unconstrained global optimization $\bar{X}_u$ has been found. Now suppose NBS (number of bisections) is three, then from the original starting point $\bar{X}_s$, we obtain $\bar{X}_{t1}$, $\bar{X}_{t2}$ and $\bar{X}_{t3}$ by subsequently bisecting the interval between $\bar{X}_s$ and $\bar{X}_u$, where $\bar{X}_{t1}$ and $\bar{X}_{t3}$ are feasible, while $\bar{X}_{t2}$ is infeasible. Finally we start the original constrained optimization from the new starting point $\bar{X}_{t3}$, and reach the constrained optimum $\bar{X}_0$.

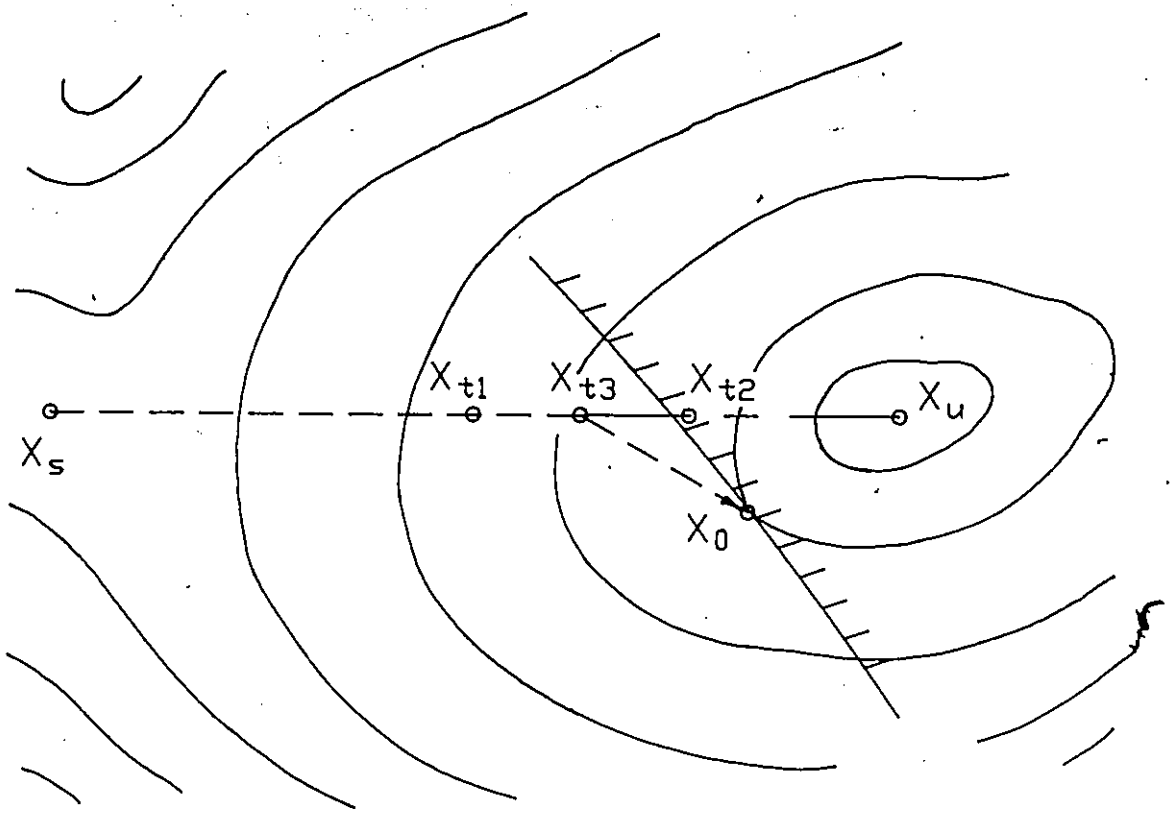


Figure 3.5 Global direction method

The value of NBS should not be too large, since a large NBS makes the new starting point too close to the constraint line, causing difficulty for many currently available optimization programs in the search along the constraint line. Numerical trials have shown that four is an appropriate value for NBS.

We first try the global direction method in a simple example (Figure 3.6). The objective function is

$$U = -(X_1 + X_2) = \min.$$

The only constraint is

$$g = 2.0*\cos X_2 - X_1 \geq 0.$$

$$0 \leq X_1 \leq 2, \quad 0 \leq X_2 \leq 1.57$$

The results are in Table 3.2.

Another example, in Figure 3.7, is a reinforcement adopted from the author's master project (Wu 1984). Table 3.3 displays the results.

From the above two examples, it can be concluded that

1. As discussed above, the numbers of bisections should not be too large; four is usually appropriate.

2. Generally speaking, when the starting point is far away from the constraint line, a lot of CPU time can be saved. When the starting point is close to the constraint line, a little more CPU time may be required. In structural design, the starting point is usually conservative in order to guarantee a feasible starting point, so we can expect to

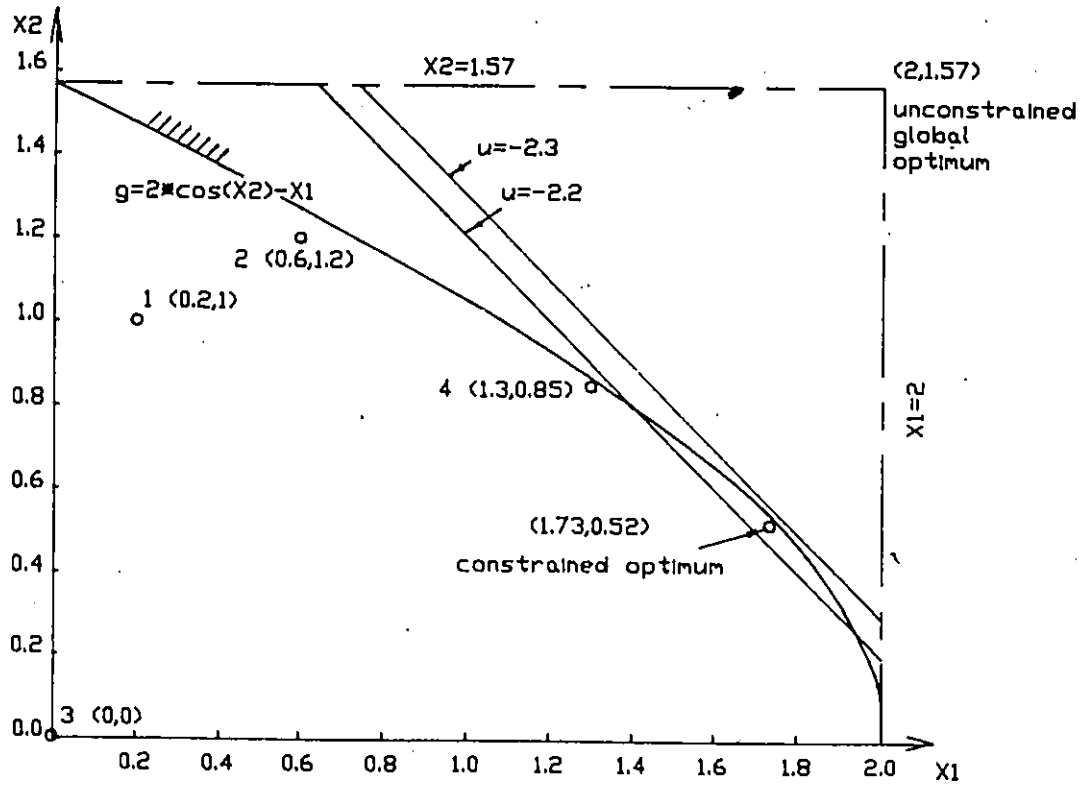


Figure 3.6 Example one of the global direction method

Table 3.2 Results of example 1 for the global direction method

	Start point		Number of bisections	Number of iterations	Note
	X(1)	X(2)			
1	0.20	1.00	0	140	
			7	111	
			5	109	
			4	108	
2	0.60	1.20	0	194	
			7	354	No convergence
			5	195	
			4	198	
3	0.00	0.00	0	132	
			7	111	
			5	109	
			4	116	
4	1.30	0.85	0	69	
			7	274	No convergence
			5	74	
			4	70	

Note: (1) Number of bisections equal to zero means that the global direction method is not adopted.

(2) All of the optimum results converge to the same point, so they have not been presented on the table.

(3) Each iteration takes the same computer time, so the number of iterations is proportional to the CPU time.

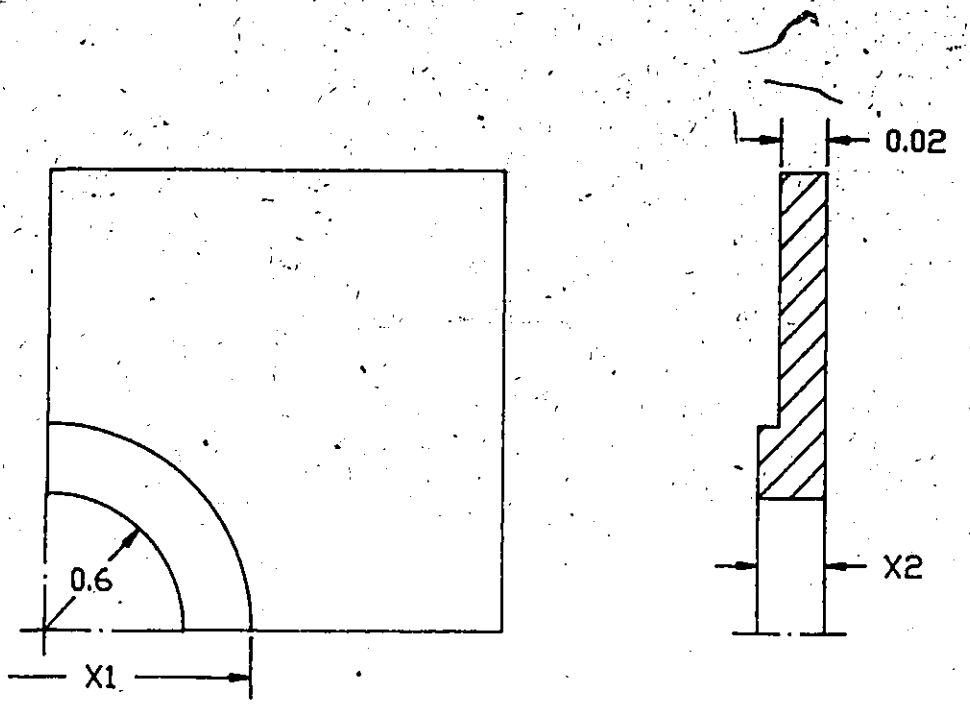


Figure 3.7 Reinforcement

Table 3.3 Results of the reinforcement for the global direction method

	Not adopting the global direction method	Adopting the global direction method
Number of iteration	72	59
X1 at optimum	69.4	72.9
X2 at optimum	3.3	3.1
U at optimum	92383	92585
Critical constraint at optimum	0.522E-4	0.295E-3
CPU time (second)	436.45	302.14

save CPU time in many cases.

3. Actually, whether CPU time is saved or lost depends to a great degree on the topography of the application. It is similar to most optimization search algorithms. For example, if the topography is as in Figure 3.8, more CPU time can be saved than in Figure 3.5. From Figure 3.8, we can also find that this method is most effective when the optimization function is a quadratic. This is characteristic of many methods, which use mathematical rigour, and have provable quadratic convergence. Other search techniques than bisection could be used to locate the vicinity of the constraint lines. This method should be particularly good combined with the reduced gradient method, which supposedly works particularly well along an inequality constraint.

In summary, the global direction method has two advantages.

1. It generally can save some computer time due to the fact that by just a few bisections, the search can be pushed to a new starting point closer to the optimum. Each bisection includes only one call of the FEM subroutine, no matter how many design variables there are in the application.

2. The second advantage of this method is more significant. It makes the global optimum more likely to be obtained, rather than a local optimum as indicated in Figure

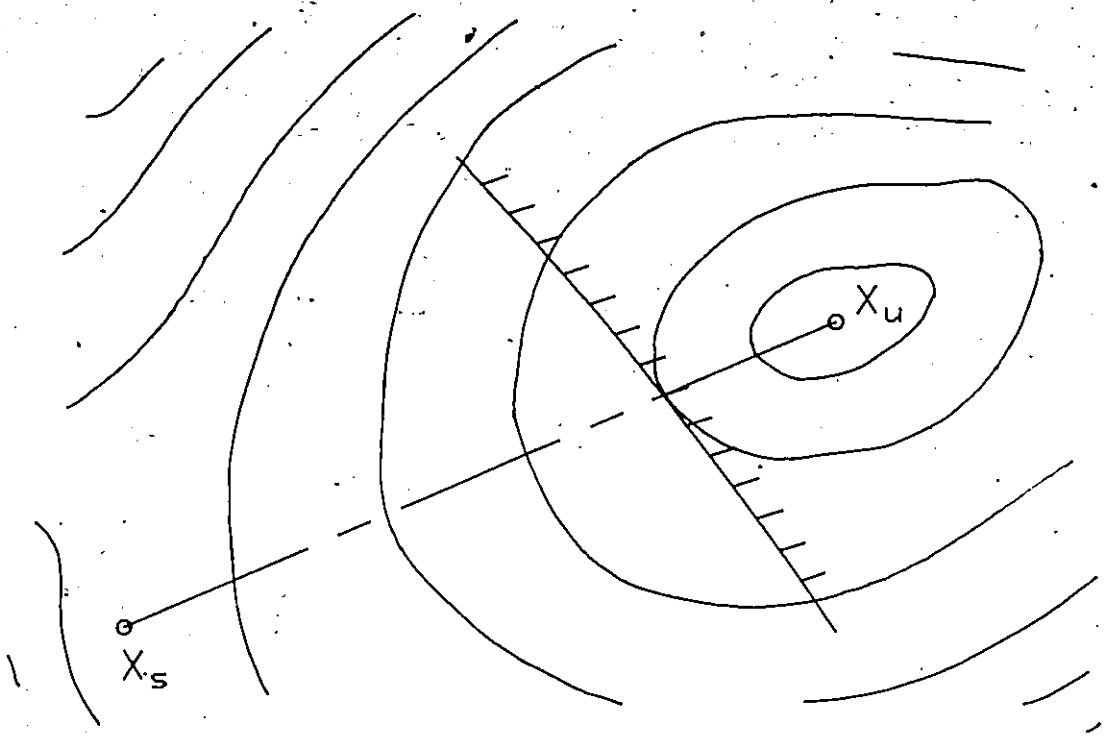


Figure 3.8 Ideal topography for the global direction method

3.4.

The global direction method requires a feasible starting point, which is guaranteed by our expert system.

3.6 THE SAFE-FAIL LINE METHOD

This method tries to set up a direct relationship between the design variables and the FEM based constraints. In many optimization iterations, just by checking the variations of the design variables, we can predict whether the constraints are satisfied or not.

3.6.1 Algorithm

3.6.1.1 Four Kinds of Design Variables

A constraint can be expressed as follows.

$$g = C - c > 0 \qquad (3.14)$$

where C is the specification (e.g. allowable stress) and c is the calculated value (e.g. maximum stress) at a design point.

Positive gradient variable: the value of g increases (c decreases) along with the increment of the positive gradient variable.

Negative gradient variable: the value of g decreases along with the increment of the negative gradient variable.

Inactive variable: the variation of the inactive variable has no effect on the value of g.

Mutable variable: the value of g sometimes increases

and sometimes decreases along with the increment of the mutable variable.

In the cantilever beam of Figure 3.9, h and t are positive gradient variables while L is a negative gradient variable.

In some published finite element based structural optimization systems (Mounir, 1982, Fungtai, 1985), all of the design variables are the sectional areas of beams or trusses. When these design variables increase, the stresses in the frame must certainly decrease, so they are all positive gradient variables.

In Figure 3.10, there is a local turnover, and X is a mutable variable. However from an overall viewpoint, the value of g generally increases when X increases. Numerical trials show that, in most circumstances such as this, we can treat the mutable variable as a positive gradient variable. In this way, good convergence can still be guaranteed and more benefit can be obtained from this algorithm.

3.6.1.2 Safe Line and Fail Line

Suppose there are three design variables, A1, A2 and A3. A1 and A2 are positive gradient variables, while A3 is a negative gradient variable. We also assume that,

$$A_i > 0 \qquad i=1,2,3 \tag{3.15}$$

We introduce transformed variables X1, X2 and X3, where X1=A1, X2=A2, X3=1/A3. The transformation is used to ensure

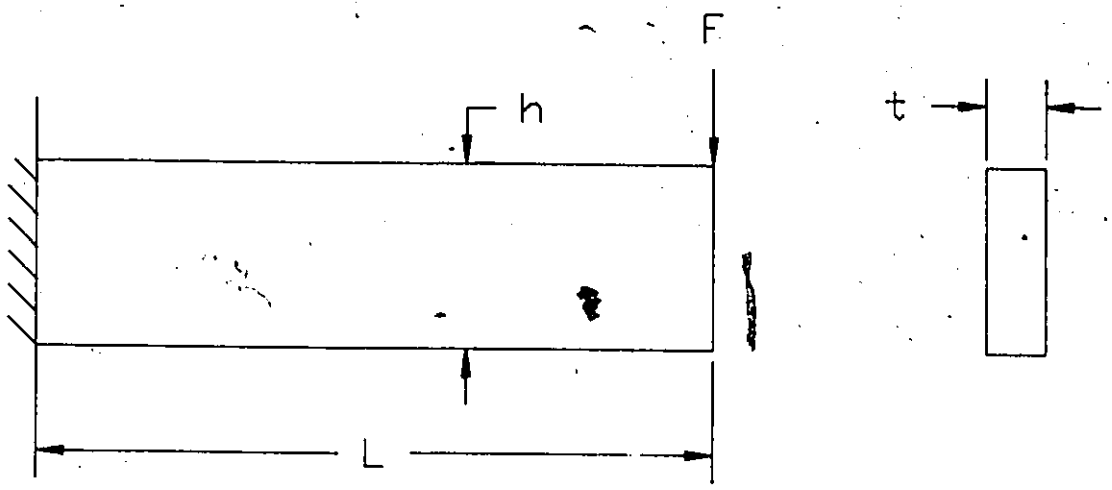


Figure 3.9 A cantilever beam .

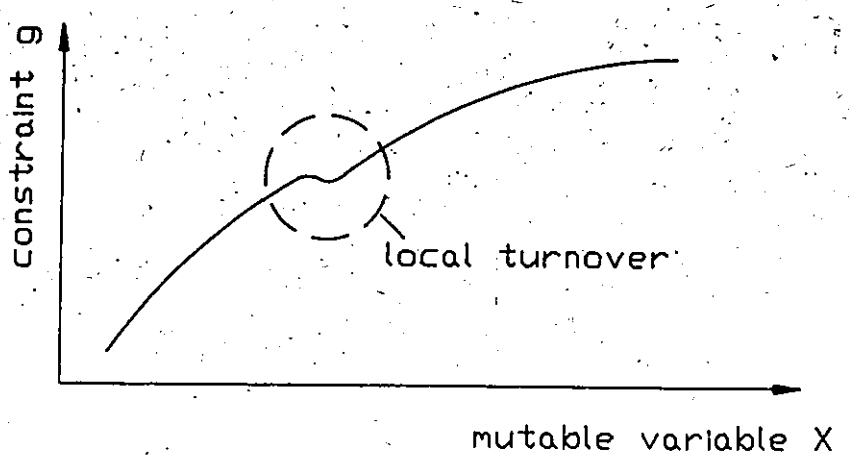


Figure 3.10 Local turnover

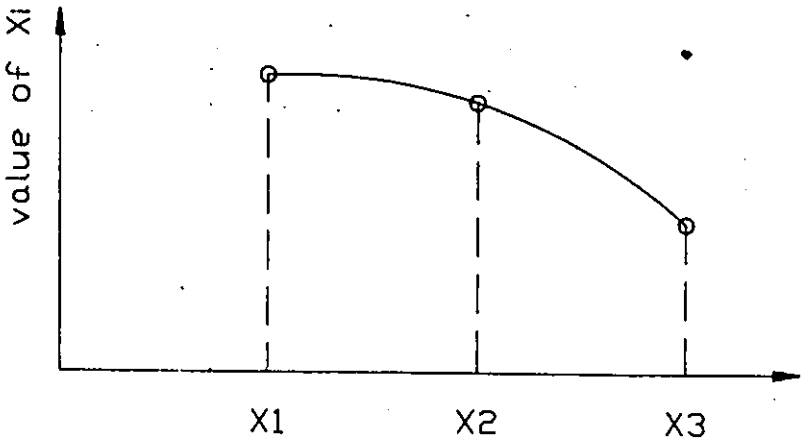


Figure 3.11 A line representing a design point

that they all are positive gradient variables.

In an optimization search stage, X1, X2 and X3 have certain values and form a line as in Figure 3.11. A line corresponds to a search stage or a design point.

In the first search stage, if the design point is feasible, we store the line, and call it as a safe line. Now suppose line 1 is a safe line, then line 2 (second stage) has three possible relationships with this safe line, as in (a), (b), (c) of Figure 3.12.

(a) Line 2 is above the safe line, and thus passes the safe check. Stage 2 must be feasible and the current FEM calculation is not needed.

(b) Line 2 is below the safe line. An FEM calculation is required to determine whether it is feasible or not. If it is feasible, line 2 becomes a new safe line, and the former safe line (line 1) should be discarded. The new safe line is closer to the constraint line than the old one, and therefore functions better. For example, in Figure 3.13, the old safe line cannot indicate whether line 3 is feasible or not, whereas the new safe line can.

(c) Line 2 intersects with the safe line (Figure 3.12 (c)). An FEM calculation is also needed here. If line 2 is feasible, it becomes a new safe line, while the old safe line is also retained. So there exist two safe lines at the same time.

Now assume that we already have three safe lines.

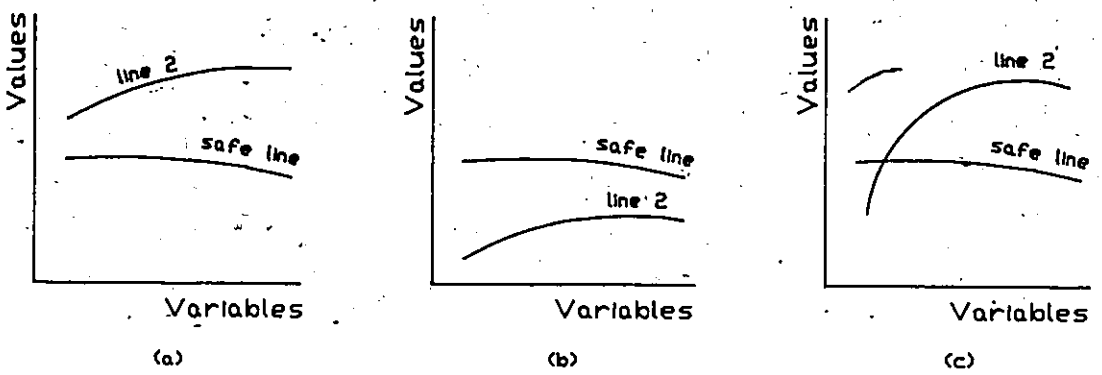


Figure 3.12 A safe line and a new line

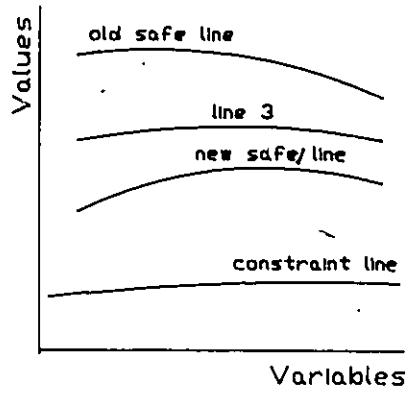


Figure 3.13 New and old safe lines

The next line, say line 4, will have two kinds of positions relative to them.

(a) In Figure 3.14, line 4 is above one, or more than one, safe line, so the safe check is passed, and line 4 must be feasible. The FEM calculation is not needed.

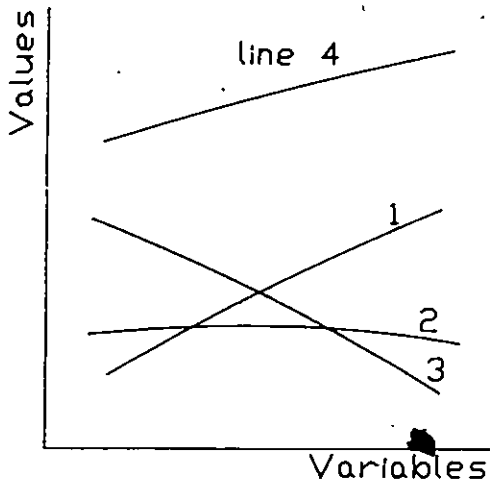
(b) In Figure 3.15, line 4 is not above any safe line, so the FEM calculation is needed. If line 4 is feasible, it becomes a new safe line. All of the old safe lines which are above the new safe line are discarded.

Similarly, we set up a series of fail lines during the optimization search. A fail line corresponds to an infeasible design point. If a subsequent line 2 is below a fail line (Figure 3.16(a)), the fail check is passed, this line must be infeasible and the FEM calculation is not needed. Otherwise, a calculation is needed to determine whether this line is feasible. If it is infeasible, it becomes a new fail line. In Figure 3.16(b), the old fail line should be discarded. In Figure 3.16(c), both the new and old fail lines remain.

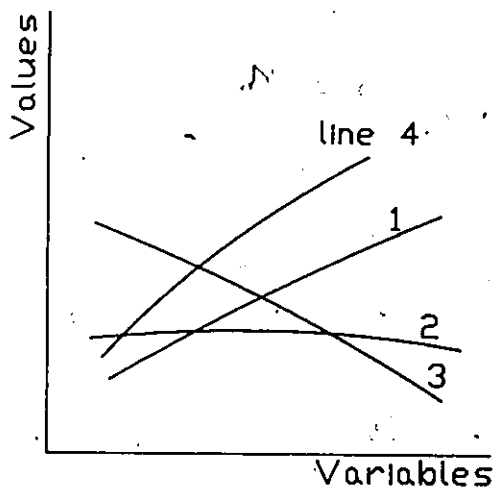
3.6.1.3 Check Criteria

1. An inactive variable need not be checked at all, because it has nothing to do with the FEM based constraints.

2. If a positive gradient variable has its current value equal to or greater than the corresponding value of a safe line, the safe check is passed. If its current value

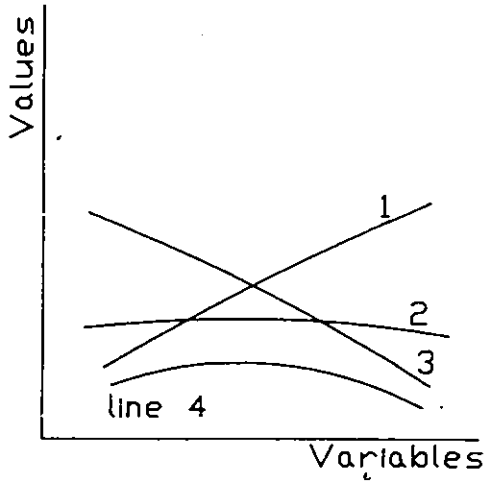


(a) Line 4 is above all of the safe lines

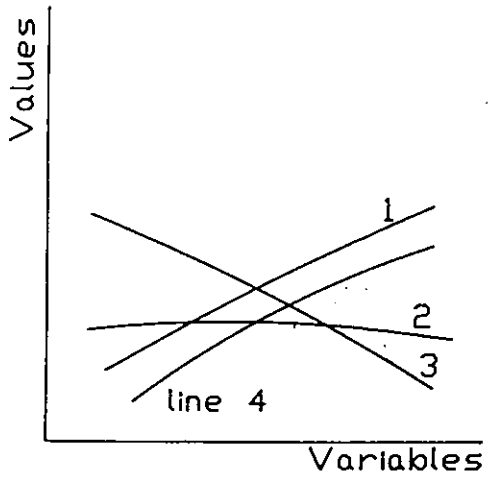


(b) Line 4 is above safe line 1

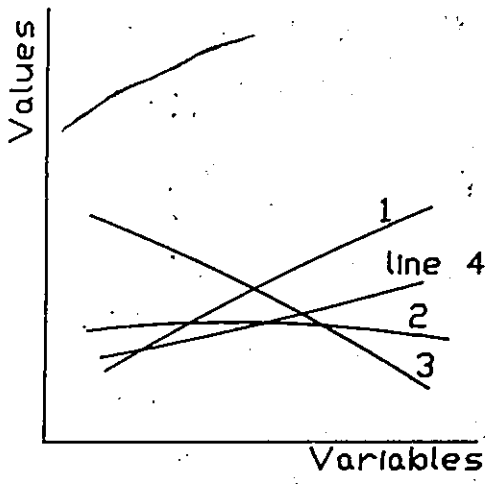
Figure 3.14 A new line above one or more existing safe lines



(a) If line 4 is a new safe line, all old safe lines are discarded

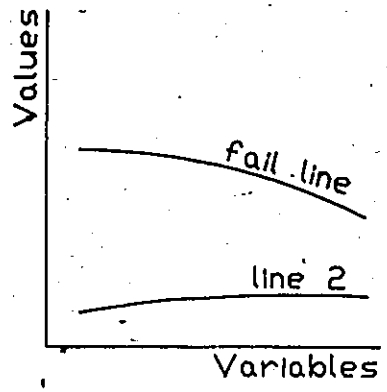


(b) If line 4 is a new safe line, safe line 1 is discarded

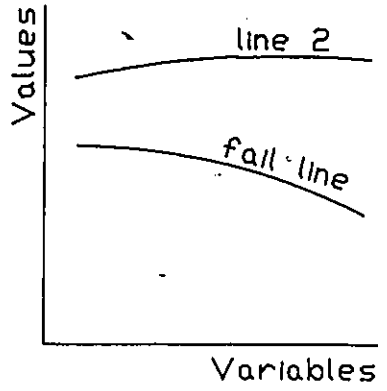


(c) If line 4 is a new safe line, none of the old safe lines are discarded

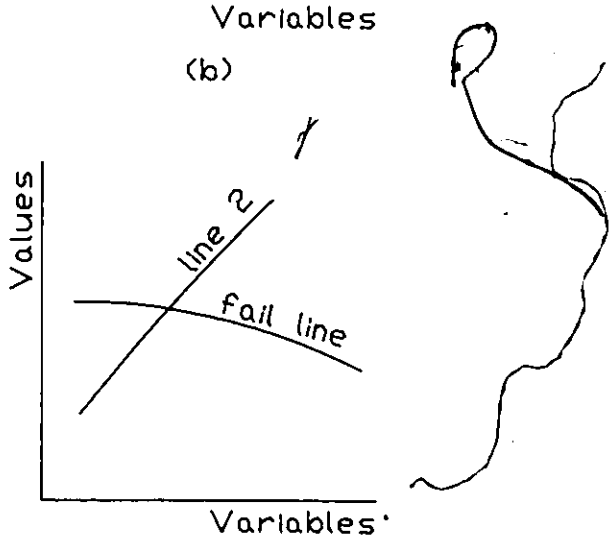
Figure 3.15 A new line not above any existing safe lines



(a)



(b)



(c)

Figure 3.16 A fail line and a new line

is equal to or less than the corresponding value of a fail line, then the fail check is passed. Similar criteria can be applied to negative gradient variables.

3. A mutable variable passed the safe or fail check when the following equation is satisfied.

$$\left| \frac{Xi - Ai}{Ai} \right| < e \qquad (3.16)$$

where Xi is the current value of the mutable variable, Ai is its corresponding value in a safe line or fail line, and e is a predetermined small positive number or zero. The criterion is that the mutable variable does not change or is within a small region.

At a design point, if and only if all of the design variables have passed the check, does the design point itself pass the check.

3.6.1.4 Some Practical Manipulations

1. During the optimization search, a lot of safe lines and fail lines will be produced. From many numerical trials, it was found that only a few of the newest safe lines and fail lines are active for the algorithm. In some applications, the number of the FEM calculations saved by using only the newest safe line is almost the same as the result of storing all of the safe lines. The optimal number of safe lines and fail lines that should be stored depends on the application. From our experience, five safe lines

and five fail lines are always enough. Limiting the number of safe lines and fail lines helps to save a small amount of computer memory and running time. However the running time spent on manipulating the safe and fail lines is trivial compared with that spent on calling an FEM calculation. Each time a new safe line or fail line arises, we discard the oldest safe line or fail line without checking whether it is above or below the new line. Therefore we always maintain the same number of safe lines and fail lines, and they are generally the most active ones. This manipulation is very simple and effective.

2. If by an FEM calculation, a design point is infeasible and has been rejected, it is called an unacceptable failure point, and it is set as a fail line. Any design point that has passed the fail check must go further into the infeasible region than the unacceptable failure point and must be definitely rejected. So we can simply assign a large value to c (formula (3.14)), say 1.5*C, to guarantee the rejection. On the other hand, if by an FEM calculation, a design point is infeasible, but has been accepted, it is called an acceptable failure point. Such a situation occurs, for example, when a search stage decreases the volume of a structure (objective function) a lot, and makes the maximum stress exceed the allowable value only a little. Now if a subsequent design point has been checked for failure after an acceptable failure point, it

may or may not be rejected. So as a regulation, we do not set an acceptable failure point of this type as a fail line, in order to avoid misleading the optimization search.

3.6.1.5 Pseudo Code

In an optimization search stage,

```
IF (the design point passes a safe check) THEN
  The design is certainly feasible. The stress at
  this point is the minimum of the stresses of the
  safe lines.
ELSE IF (the design point passes fail check) THEN
      The design is certainly infeasible. A large
      value is assigned to the stress to guarantee
      the rejection.
    ELSE
      FEM calculation is called
      IF (the design is feasible) THEN
        The current design point becomes a new safe
        line. The oldest safe line is discarded.
      ELSE IF (the search stage has been rejected)
                                                THEN
          The current design point becomes a new
          fail line. The oldest fail line is
          discarded.
        END IF
      END IF
```

```
        END IF
    END IF
```

As the optimization approaches the optimum solution, the families of safe lines and fail lines approach the constraint line from opposite directions, and form a narrow channel. Only a design point that falls in this channel will call an FEM calculation, which in turn makes the channel more narrow. Now our expert system is able to learn from its own experience and becomes more and more clever.

3.6.1.6 Optimization Route

Sometimes, by adopting the safe-fail line method, the optimization search route and optimum solution are different from the original ones, especially when the number of safe lines and fail lines is large. However no matter how many safe lines and fail lines are used, the optimization process in our examples has consistently followed the general path of the original optimization algorithm. So the convergence is always no worse than the original one.

If we want the optimization process to always follow exactly the original route (although this is really unnecessary), two measures should be avoided,

1. Do not treat the mutable variables as positive gradient or negative gradient variables.

2. Do not use fail lines.

3.6.2 Examples

Example 1: Cantilever Beam

A cantilever beam has six design variables (Figure 3.17). The optimization function is the minimum volume of the beam. The critical constraint is the stress at point A or B. Formulas in solid mechanics are used to calculate the stress. The weight of the beam contributes to the moment.

In this application, X1 and X4 are negative gradient variables, X2 and X3 are positive gradient variables. X5 and X6 are actually mutable variables, but we can treat them as positive gradient variables, as discussed above.

The optimization method tried first was the Hooke and Jeeve direct search method combined with the Schuldt's penalty function (Siddall, 1982). Table 3.4 shows the results from a few different starting points. They have converged very well.

In this application, if we define X5 and X6 as mutable variables, the optimization solutions are exactly the same, but the value of GAIN decreases, as in Table 3.5.

When more safe lines and fail lines are stored, the value of GAIN will increase. However it saturates quickly. Table 3.6 shows this effect, where X5 and X6 are defined as mutable variables.

Next the following seven optimization methods with different penalty functions (Siddall, 1982) were tried: (1)

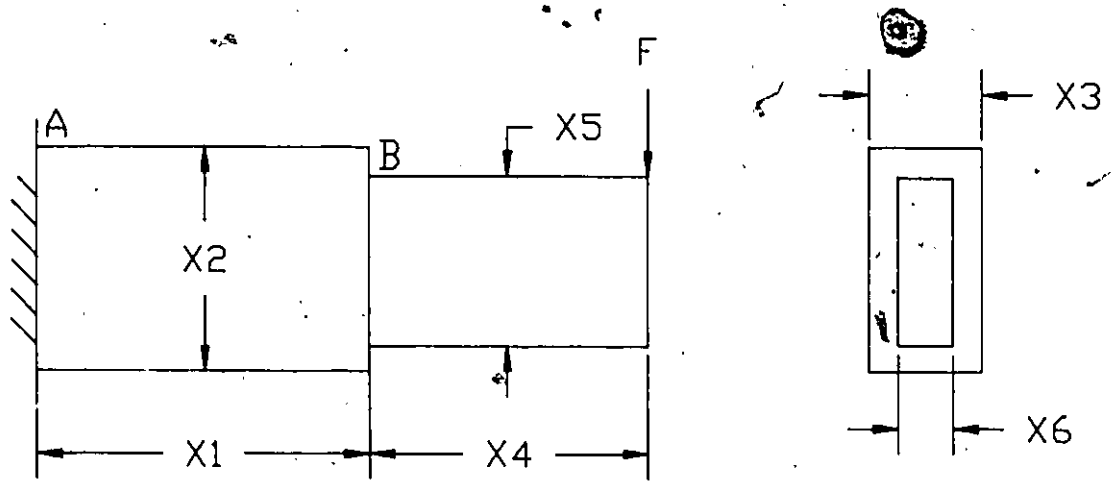


Figure 3.17 Example of the cantilever beam

Table 3.4 Results for the cantilever beam

Starting point						Optimum	Stress		GAIN
X1	X2	X3	X4	X5	X6	solution	A	B	%
6	4	2	4	3	1	33.808	599.947	599.729	50.2
6	3	1	4	2	0.5	31.714	599.414	599.772	45.0
4	4	2	2	5	1	25.760	599.944	599.451	49.1
5.5	7	1	4	7	0.1	22.082	599.976	599.682	48.0

Note: (1) The first starting point is feasible, others are all infeasible.

(2) The assumed allowable stress is 600.

(3) GAIN is the quotient of the number of saved FEM calculations divided by the total number of optimization iterations.

Table 3.5 The effect of the mutable variables

Starting point						GAIN			
						X5 & X6 as positive gradient variables	X5 & X6 as mutable variables		
X1	X2	X3	X4	X5	X6		e=0.00	e=0.01	e=0.10
6	4	2	4	3	1	50.2	25.9	28.4	32.3
6	3	1	4	2	0.5	45.0	20.3	20.3	28.2
4	4	2	2	5	1	49.1	22.9	28.3	29.7
5.5	7	1	4	7	0.1	48.0	25.3	29.9	31.0

Note: e is defined in formula (3.16).

Table 3.6 The effect of the number of safe lines on the value of GAIN

Starting point						Number of safe lines					
X1	X2	X3	X4	X5	X6	1	2	3	4	9	25
6	4	2	4	3	1	25.9	29.7	31.0	32.9	32.9	32.9
6	3	1	4	2	0.5	20.3	23.5	25.9	26.2	28.5	28.5
4	4	2	2	5	1	22.9	25.6	29.7	31.1	31.1	31.1
5.5	7	1	4	7	0.1	25.3	28.7	31.0	31.0	31.0	31.0

adaptive random search, (2) Davidon-Fletcher-Powell method, (3) Fletcher's 1972 method, (4) Jacobson and Oksman method, (5) Powell's direct search, (6) Hooke and Jeeve direct search, (7) Simplex method. Except for the Powell's direct search method with certain penalty functions, all of the above methods worked very well when combined with the safe-fail line method. Furthermore, by properly selecting the number of safe and fail lines and defining the design variables, the original optimization route can be followed exactly.

Among the above optimization methods, the Jacobson-and-Oksman method and Hooke Jeeve's method produced largest values of GAIN. This algorithm may be most suitable for use with those optimization methods in which each design variable changes one at a time during the search. Thus the safe checks or fail checks have more opportunities to be passed. However this conclusion must be confirmed by additional testing.

The following three examples are adopted from the author's master project (Wu, 1984). Stresses were evaluated by the FEM calculation. The results were compared with the original solution.

Example 2: Reinforcement (Figure 3.7)

The results are in Table 3.7.

Example 3: Bolt (Figure 3.18)

The results are in Table 3.8.

Table 3.7 Results for the reinforcement using the safe-fail line method

No. of safe lines	No. of fail lines	Type of variable		Opti. route	Opti. solution	Final stress	GAIN %
		X1	X2				
Original solution							
					0.01844	39984	0.0
1	0	P	P	same	0.01844	39984	9.9
3	0	P	P	same	0.01844	39984	30.9
5	0	P	P	same	0.01844	39984	31.5
1	1	P	P	same	0.01844	39984	13.8
3	1	P	P	same	0.01844	39984	34.8
5	1	P	P	same	0.01844	39984	35.4
3	3	P	P	dif.	0.01844	39984	37.9
4	4	P	P	dif.	0.01844	39984	39.3
10	10	P	P	dif.	0.01844	39984	40.2

Note: (1) The assumed allowable stress is 40000.

(2) Optimization route:
"same" means same as the original route;
"dif." means different from the original route.

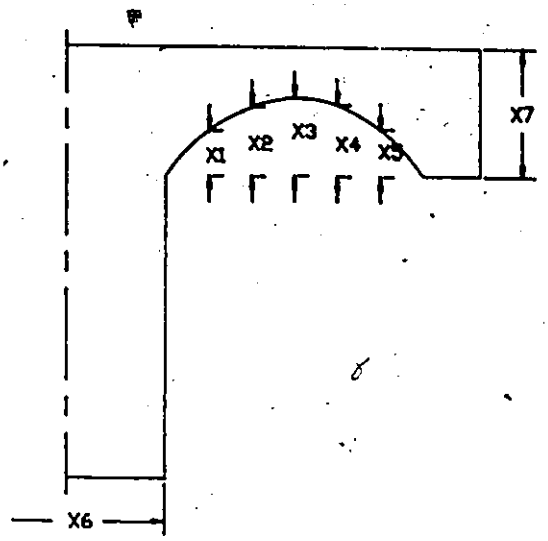


Figure 3.18 Bolt

Table 3.8 Results for the bolt using the safe-fail line method

No. of safe lines	No. of fail lines	Type of variable							Opti. route	Opti. solu.	Final stress	Gain %
		X1	X2	X3	X4	X5	X6	X7				
Original solution												
										1.132	39985	0.0
5	0	m	m	m	m	m	p	p	same	1.132	39985	8.3
1	0	n	n	n	n	n	p	p	dif	1.127	39993	23.0
5	0	n	n	n	n	n	p	p	dif	1.127	39993	30.6
3	3	n	n	n	n	n	p	p	dif	1.132	39986	44.2
4	4	n	n	n	n	n	p	p	dif	1.132	39986	44.8
5	10	n	n	n	n	n	p	p	dif	1.132	39986	46.8
10	10	n	n	n	n	n	p	p	dif	1.132	39986	46.8

Example 4: Plate (Figure 3.19)

The results are in Table 3.9.

3.6.3 Summary

Numerical trials show that the safe-fail line method has the following advantages.

1. It can save a significant percentage of CPU time used for calculating a very time consuming constraint function (e.g. calculating stresses and displacements by FEM).

2. It works reliably in the examples tested.

3. It requires very little additional computer memory and computer time for itself.

3.7 THE QUADRATIC METHOD

This method is similar to the second order Taylor expansion approximation mentioned by some authors (Grandhi, 1985). In this research, we used direct parameter fitting to avoid calculating the second order derivatives and have solved some difficulties in the implementation.

The constraint function related to the maximum stress in a structure is usually very complex, including the calling of an FEM subroutine. However in a small region, it can be expressed approximately as a quadratic polynomial as follows.

$$g = a_0 + \sum_{i=1}^{N} a_i X_i + \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} X_i X_j \qquad (3.17)$$

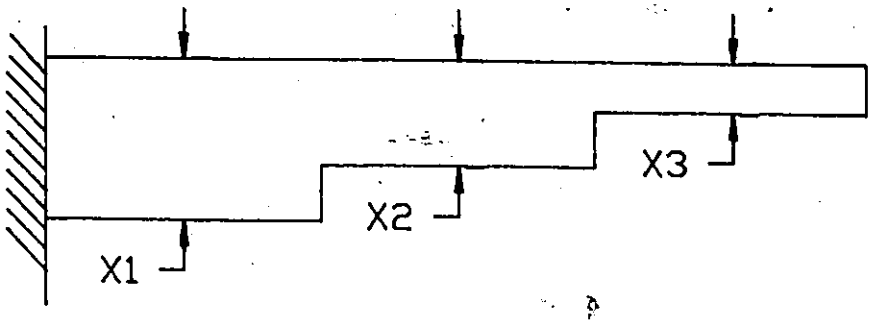


Figure 3.19 Plate

Table 3.9 Results for the plate using the safe-fail line method

No. of safe lines	No. of fail lines	Type of variable			Opti. route	Opti. solu.	Final stress	Gain %
		X1	X2	X3				
Original solution								
						7.25	39901	0.0
1	0	p	p	p	same	7.25	39901	44.4
1	1	p	p	p	same	7.25	39901	48.1
1	5	p	p	p	same	7.25	39901	57.4
10	10	p	p	p	same	7.25	39901	44.4

where n is the number of design variables.

m sample points are required to adapt the m parameters a_0, a_i and a_{ij}, in order to define g. It can be evaluated easily,

$$m = 1 + 1.5*n + 0.5*n^2 \qquad (3.18)$$

Using formula (3.17) to evaluate g is much simpler than calling the FEM subroutine. And it can be applied to any complex constraint as well as the maximum stress.

To guarantee adequate accuracy, the sample points should be confined within a small region, and the design point in which the constraint function g is evaluated using formula (3.17) must be inside this region. The algorithm is as follows.

1. Set up an array TAB to store the sample points.
2. IF (a quadratic polynomial as formula (3.17) available) THEN

   GO TO 12

   ELSE

   GO TO 3

   END IF
3. Calculate g by calling the FEM subroutine.
4. IF (the current design point is the same as one of the existing sample points in array TAB) THEN

   GO TO 14

   ELSE

GO TO 5

END IF

5. Store the current design point as a new sample point in array TAB (If there are m sample points in TAB already, discard the oldest one first).

6. IF (m sample points in TAB now) THEN

GO TO 7

ELSE

GO TO 14

END IF

7. Are all of the m sample points within a small region? i.e.

$$| X_{i,k} - X_{j,k} | < e \quad (((k=1,n),j=1,m-1),i=j+1,m) \tag{3.19}$$

where $X_{i,k}$ is the value of kth design variable for the ith sample point, $X_{j,k}$ is the value of the same design variable for the jth sample point. n is the number of design variables.

IF (it is so) THEN

GO TO 8

ELSE

GO TO 14

END IF

8. Get the parameters from the m sample points, thus establishing the quadratic polynomial (3.17). This quadratic polynomial can be made use of until a new one is established to replace the old one.

9. Evaluate the centroid X_0 of the sample points.
10. Discard all of the old sample points in TAB. New sample points will be stored.
11. GO TO 14
12. IF (the distance between the current design point and the centroid X_0 equal to or less than (0.5*e)) THEN
    GO TO 13
    ELSE
    GO TO 3
    END IF
13. Evaluate g, using formula (3.17).
14. Continue.

Adapting the parameters from the sample points means solving simultaneous equations which are sometimes ill-conditioned, i.e. give rise to a division with very small divider. In our programming, each time such a numerical difficulty was encountered, all of the existing sample points in TAB were discarded. New sample points were then collected.

The algorithm can be interpreted as in Figure 3.20. The search is assumed to start at point B, where g is approximated by the quadratic polynomial q_1. When the search approaches point C, an updated quadratic polynomial q_2 replaces q_1. Finally q_3, approximating g, will lead the search to the optimum A.

Table 3.10 shows some results corresponding to

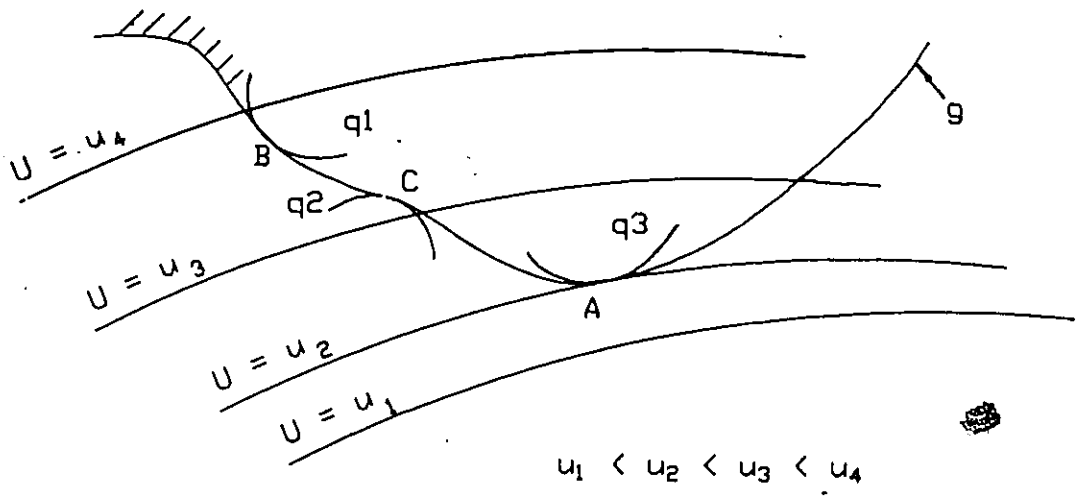


Figure 3.20 Topography of the quadratic method

Table 3.10 Results of the quadratic method

e	KCOUNT	GAIN	At optimum				CPU (sec.)	Average error (%)
			X(1)	X(2)	U	g		
0.20	102	60	69.4	3.30	92383	4.38E-5	267.82	160.20
0.11	119	41	70.3	3.24	92459	3.23E-3	481.45	48.64
0.10	127	58	69.4	3.30	92383	4.97E-5	431.88	3.18
0.08	127	53	69.4	3.30	92383	4.97E-5	461.67	3.47
0.05	127	37	69.4	3.30	92383	5.25E-5	554.12	4.41
0.00	102	0	69.4	3.30	92383	4.48E-5	611.16	0.00

Note: (1) There are two design variables for this example.

(2) KCOUNT is the number of iterations of optimization search.

(3) GAIN is the number of iterations when the constraint g is evaluated using quadratic polynomials.

(4)
$$\text{error} = \left| \frac{g_{FEM} - g_q}{g_{FEM}} \right|$$

where g_{FEM} is the value of g evaluated by calling FEM; g_q is the value of g evaluated by quadratic polynomial at the same design point.

different values of e in formula (3.19). The example is the reinforcement (Figure 3.7). It was found that the solutions have no significant difference from the original one. However, as expected, the larger the e, the more CPU time can be saved, but with less accuracy. In our programming, e was set as 0.08, which is independent of different applications, since the design variables in all of the applications are scaled to the range from 10 to 20 by the expert system.

The quadratic method is particularly useful when the step size in the optimization search is small.

3.8 THE DIFFERENTIAL METHOD

3.8.1 Introduction

This method tries to replace the increments of the displacements by their differentials, in order to reduce the computer time required. It is equivalent to the first order Taylor expansion approximation discussed by some authors (Lee, 1987). However in this research, the nodal displacements are defined as variables instead of the independent design variable, thus the accuracy of the approximation can be increased greatly. Also the evaluation of the gradient vector is avoided by using increments only.

3.8.2 Algorithm

The overall effectiveness of an FEM analysis depends

to a large degree on the numerical procedures used for the solution of the system equilibrium equations introduced in equation (3.12), i.e.

$$K\,U = R \tag{3.20}$$

where K is the stiffness matrix, U is the displacement vector, and R is the load vector.

At a design point X_0 (X_0 is the design variable vector), we have the equilibrium equations,

$$K_0\,U_0 = R_0 \tag{3.21}$$

During an iteration, the design variables move to a new position X, where $dX = X - X_0$. The step size is usually small, and as the search approaches the optimum point, it becomes smaller and smaller. If dX is small enough, we can use the differentials of the displacements to replace their increments, and at the same time maintain adequate accuracy.

Using the linear terms of a Taylor's series expansion gives,

$$y = y_0 + \frac{\partial y}{\partial x}\,dx \tag{3.22}$$

which in our problem is,

$$R = R_0 + dK\,\frac{\partial R}{\partial K}\Big|_{x=x_o} + \frac{\partial R}{\partial U}\Big|_{x=x_o} dU$$

$$= R_0 + dK\,U_0 + K_0\,dU \tag{3.23}$$

$$dR = R - R_0 = dK\,U_0 + K_0\,dU \tag{3.24}$$

i.e.

$$K_0\,dU = -dK\,U_0 + dR \tag{3.25}$$

Solving this equation, we can obtain dU.

$$U = U_0 + dU \tag{3.26}$$

which is the result we want.

Now let us look at equation (3.21) again. The Gauss Elimination method (Bathe, 1982), which is considered very efficient in the FEM calculation, has been adopted to solve the equation. First of all, the stiffness matrix is decomposed as

$$K_0 = L\ D\ L^t \tag{3.27}$$

where L is a lower triangle matrix, D is a diagonal matrix. Now equation (3.21) becomes,

$$L\ D\ L^t\ U_0 = R_0 \tag{3.28}$$

This expression is very easy to solve.

The decomposition of K_0 uses most of the computer time required to solve the equation and is independent of the load term in the right hand side of the equation. So the decomposed stiffness matrix can be used many times in equation (3.25), which becomes,

$$L\ D\ L^t\ dU = -dK\ U_0 + dR \tag{3.29}$$

The number of multiplications for decomposing K_0 is about $(0.5*NN*NB^2)$, where NN is the number of equations and NB is the bandwidth. However the evaluation of $(dK*U_0)$ in (3.29) requires fewer than (2*NN*NB) multiplications. It can

therefore be concluded that it will save computer time when NB>4, which is always the case, and the larger NB is, the more computer time can be saved.

To guarantee the required accuracy, dK and dR in (3.29) must be small. We can achieve this by checking each element in the matrix dK and dR. However a quicker method is to check $dX=X-X_0$, the increments of the design variables, which control dK and dR directly. Thus, when using the differential method, the following conditions should be satisfied,

$$dX_i = \left| \frac{X_i - X_{io}}{X_{io}} \right| < e \qquad i=1,2,\ldots,n \qquad (3.30)$$

where n is the number of the design variables and e is a small predetermined positive number. When e is larger, more computer time may be saved, but the accuracy will be poorer.

Following is the pseudo code for the differential method.

1. Define e.
2. In each optimization iteration, evaluate K and R.

IF $\left| \frac{X_i - X_{io}}{X_{io}} \right| < e, \quad i=1,2,\ldots,n$ THEN

$L\ D\ L^t\ dU = -(\ K - K_o\ )\ U_o + (\ R - R_o\ )$

$U = U_o + dU$

ELSE

Decompose K to LDL^t

Solve equation: $L\ D\ L^t\ U = R$

Store LDL^t, $K_o=K$, $U_o=U$, $R_o=R$, $X_{io}=X_i$

END IF

3. Evaluate stresses from displacements U.

4. Return stresses and displacements to the main program.

3.8.3 Examples

The differential method was tried in two examples, the reinforcement (Figure 3.7) and the plate (Figure 3.19). In the first example, the increments of the design variables are 1%, and the corresponding discrepancies of the calculated stresses are less than 0.1%. In the plate example, the increments of the design variables are 10%, and the discrepancies of the stresses are less than 2%. These discrepancies may be smaller than the error produced by the FEM itself. Table 3.11 shows the results.

3.8.4 Summary

The differential method applies the differential technique to the FEM based structural optimization in order to reduce the computer time. It achieves this purpose to some degree, especially for those applications where the stiffness matrix has a large bandwidth.

By assigning a reasonable value for e, the accuracy of the result is adequate.

One important advantage of this method is that it

Table 3.11 Results of the differential method

Method	No. of search steps	Optimum solution	Final stress	Total CPU (sec.)	Average CPU per step	Required computer memory
Reinforcement						
Incremental	181	0.0184	39984	860.7	4.76	10707
Differential	167	0.0184	39984	588.1	3.52	17659
Plate						
Incremental	108	7.25	39901	1031.5	9.55	11795
Differential	122	7.20	39938	814.2	6.67	22111

Note: The assumed allowable stress is 40000.

can be applied to any optimization method. The additional implementation of coding is separate from the existing optimization subroutines.

Its obvious disadvantage is that it requires considerably more computer memory. Because we have to store the stiffness matrix of the base point, K_o, and the decomposed stiffness matrix LDL^t.

3.9 INTEGRATION OF THE SIX ALGORITHMS

Let us first make some comments and comparisons on the algorithms.

Substructure Method

It is well developed and reliable. In many applications, it can save a lot of computer time.

Skipping Method

It is very simple and can save a lot of computer time in some applications where a non-FEM based constraint is frequently violated during the optimization search.

Global Direction Method

It helps to drive the design to the global optimum. In many applications, it can save computer time. However in some applications, it may lose computer time.

Safe-Fail Line Method

It is very reliable and powerful. In almost all of

the applications, it can save a large percentage of computer time. Furthermore it usually leads to the same solution as the original optimization algorithm.

Quadratic Method

When the quadratic polynomial replaces an FEM call, a lot of computer time can be saved. However the times that this method takes effect during the optimization run is usually less than those of the safe-fail line method or the differential method. But when the step size of the search, determined by some optimization parameters, is small (i.e. a finer convergence is required), this method is activated more times. The accuracy of the results obtained by this method is a little poorer than that of the differential method.

Differential Method

This method takes effect frequently during the optimization run. However each time it is brought into effect, the computer time saved is less than that saved by the skipping method, the safe-fail line method and the quadratic method. And it requires more computer memory.

In this project, any of the six algorithms can be adopted independently or combined with any others. The human expert will check the circumstance and give advice for an application concerning which algorithms should be adopted.

Generally, all of the six methods can be adopted all together in order to achieve the greatest benefit. In this case, the precedence in adopting them is as follows:

Substructure method -- Global Direction method -- Skipping method -- Safe-fail Line method -- Quadratic method -- Differential method.

The procedure is as follows.

1. For each application, the expert system checks if a constant substructure can be determined. If so, the substructure method is adopted automatically when establishing the FEM model.

2. The global direction method is called to obtain a new starting point closer to the global optimum.

Then the optimization search begins. In each stage,

3. If any non-FEM based constraint has been violated, the FEM subroutine is skipped.

4. Check is made to determine if the safe-fail line method can take effect.

5. Check is made to determine if the quadratic method can take effect.

6. Check is made to determine if the differential method can take effect.

7. If all of the methods fail to take effect at a design point, a call to the FEM subroutine is necessary.

The FEM algorithms discussed in this chapter combine to provide a significant saving of computer time in the FEM

based optimization. Usually about 70 - 80% of the computer time can be saved. However the actual value is different from problem to problem.

CHAPTER 4

BUILDING THE DESIGN APPLICATION

4.1 INTRODUCTION

It was shown in chapter 2 how the human expert establishes the knowledge database of an expert system. Integrating it with the reference engine creates the expert system **ESSF** (Expert System of a Structural Family). Now a designer can do routine design applications in structural optimization by executing **ESSF**.

The designer is assumed to be an experienced general designer of machine systems of a specific type; he or she could be either a design draftsman or an engineer. Different designers have different knowledge levels; the basic requirements are

1. The designer must understand in general the concepts of optimization and constraint functions.

2. He or she must be able to divide a structure into several regions for the finite element mesh generation.

The designer's basic function is to enter the requirements of the design and perform necessary judgements for decision making.

Figure 4.1 is the layout of **ESSF**. **ESSF** is actually a versatile CAD (computer aided design) tool, rather than just an expert system. The designer can run **ESSF** in

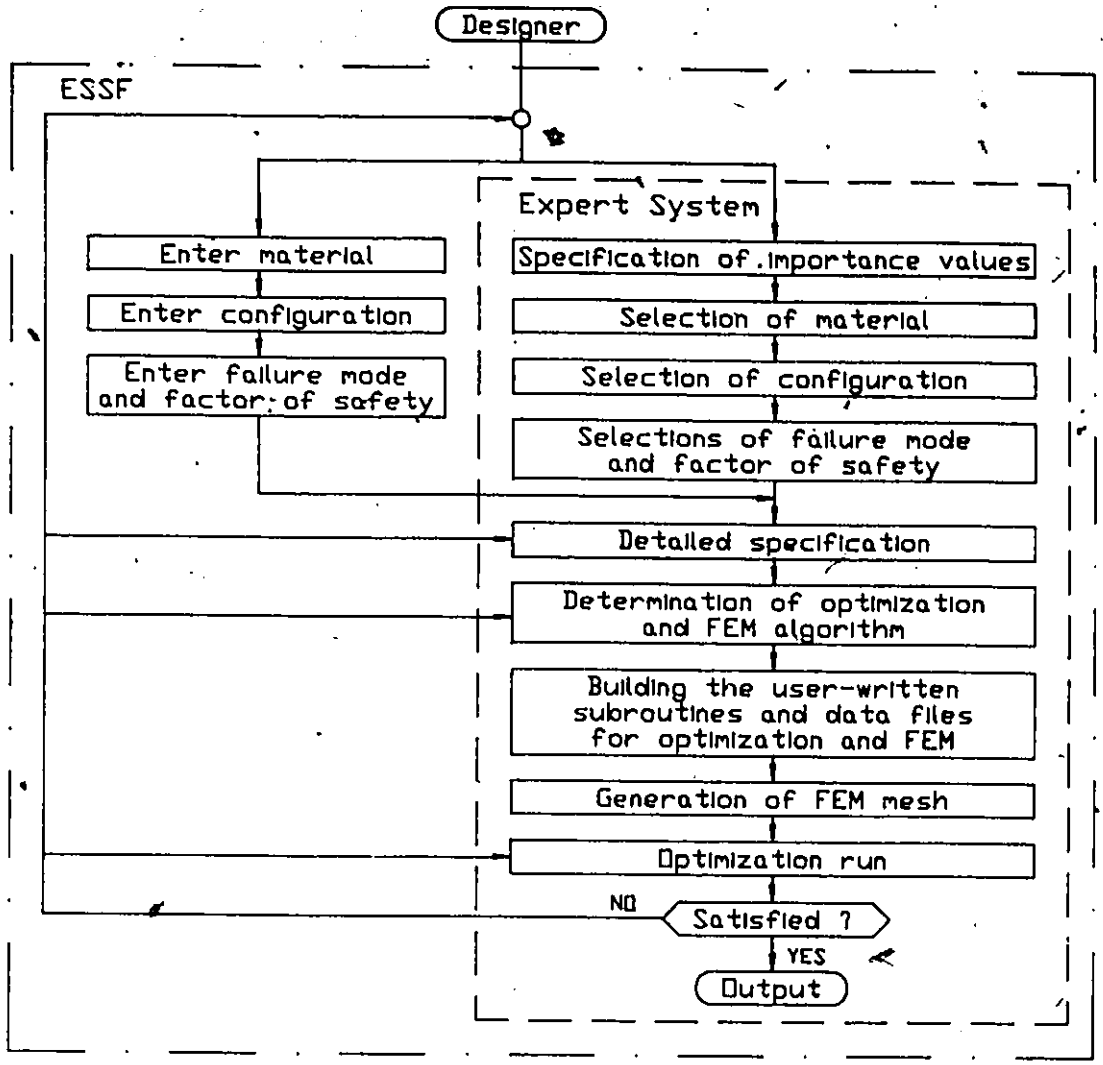


Figure 4.1 Layout of the program ESSF

different modes.

1. Independent Mode. The designer can bypass part of the expert system and enter all of his design parameters, i.e. material, configuration, failure mode and factor of safety. He need not enter the design specifications, i.e. the importance values of the performance characteristics. In return, the expert system provides only very little advice for his determination of the design parameters. This mode is most suitable for a very skillful designer in the creation of a complete new design, making use of ESSF as an optimization and analysis tool.

2. Dependent Mode. The designer selects all of the design parameters from the expert system database. Then he or she may or may not do some modifications of them. The importance values of the performance characteristics must be specified. In return, the expert system provides a lot of suggestions and advice. These include displaying the goodness of all of the candidates according to the design requirements. This mode is most suitable for a novice designer who tends to more depend on the expert system.

3. Semi-dependent Mode. This mode is intermediate between the above two modes. The designer enters the importance values of the performance characteristics. He or she will receive as many suggestions and as much advice from the expert system as in the dependent mode. He or she can select the design parameters in the expert system database,

and at the same time be free to enter one or more of his or her own design parameters.

In any mode, the expert system just plays the part of a consultant. The designer has full freedom to perform his or her judgements and make the final decisions.

4.2 INDEPENDENT MODE: DESIGNER ENTERING ALL HIS OWN DESIGN PARAMETERS

In this mode, the designer enters his four design parameters, i.e. material, configuration, failure mode and factor of safety one by one.

4.2.1 Input Material

4.2.1.1 Determination of Material Class

First of all, the **ESSF** system will display all of the material classes in the database of the expert system. The designer is asked to tell which class his material would belong to, i.e. whether his material belongs to plain carbon steel, or alloy steel, and so on. The usage of this information will be made clear shortly in section (4.2.1.3).

If the material entered by the designer does not belong to any existing material class in the database, or the designer has no idea about it, this part of the program can be bypassed.

4.2.1.2 Input Material Properties

The designer next enters the values for eleven major

properties of the new material: (1) modulus of elasticity, (2) Poisson's ratio, (3) unit weight, (4) thermal expansion coefficient, (5) ultimate strength, (6) yield strength, (7) compressive strength, (8) elongation, (9) Brinell hardness, (10) cost of unit weight, (11) endurance limit. Not all of these have to be specified; the designer need only specify those about which he is sure. In section (4.2.1.3), the values of the missing properties will be supplemented.

But if, in section (4.2.1.1), the designer did not specify a class for the new material, the supplement does not work. In this case, the values of the modulus of elasticity and strength are mandatory. Because they are required by the FEM calculation. Poisson's ratio is required by the FEM calculation too. But, since it is about 0.3 for most of the metals, the ESSF system sets it as 0.3 if the designer did not specify it.

4.2.1.3 Supplement

If the new material was designated as belonging to a material class by the designer, the expert system can find a material with the closest similar properties in this class in the database, and then assign the values of the corresponding properties of the closest material to the missing properties of the new material.

For example, suppose the new material belongs to ductile iron in the database and there are two subclasses of

ductile iron as in Table 4.1.

The relative difference of the property values between the new material and an existing material is calculated as follows.

$$D = \left| \frac{(\text{value of new mat.}) - (\text{value of existing mat.})}{\text{value of new mat.}} \right| \qquad (4.1)$$

Suppose the designer enters the values for only three properties.

modulus of elasticity = 26 mpsi

yield strength = 60 ksi

elongation = 17 %

All other properties are missing. The relative difference between the new material and ASTM A339 is,

$$\left| \frac{26-25}{26} \right| + \left| \frac{60-53}{60} \right| + \left| \frac{17-18}{17} \right| = 0.214 \qquad (4.2)$$

The relative difference between the new material and ASTM A395 is,

$$\left| \frac{26-24}{26} \right| + \left| \frac{60-108}{60} \right| + \left| \frac{17-5}{17} \right| = 1.583 \qquad (4.3)$$

So ASTM A339 is the closest material, and we assign its values to the missing properties of the new material. Thus the new material is now assumed to possess the following properties.

modulus of elasticity	26.00 mpsi	(input)
Poisson's ratio	0.29	(supplement)

Table 4.1 Material class of ductile iron

	modulus of elasticity	Poisson's ratio	unit weight	therm expan.	ultimate strength	yield strength	compressive strength	elogation	Brinell hardness	cost of unit weight	endurance limit
	mpsi		lb/in^3	$10^{-6}/F^0$	ksi	ksi	ksi	%	HB	$/lb	ksi
ASTH A339	25	0.29	0.26	7.5	70	53	55	18	170	0.75	24
ASTH A395	24	0.28	0.26	7.5	135	108	130	5	310	0.80	59

unit weight	0.26	lb/in^3	(supplement)
thermal exp. coef.	7.50	$10^{-6}/F$	(supplement)
ultimate strength	70.00	ksi	(supplement)
yield strength	60.00	ksi	(input)
compressive strength	55.00	ksi	(supplement)
elongation	17.00	%	(input)
Brinell hardness	170.00	HB	(supplement)
cost per unit weight	0.75	$/lb	(supplement)
endurance limit	24.00	ksi	(supplement)

If the class of the new material was not specified by the designer, then the supplement does not work, since for two materials in different classes, even though some of their properties are similar, others may differ a lot. The supplement procedure is only valid for a preliminary design.

The supplement is an algorithm for dealing with incomplete information.

4.2.2 Input Configuration

4.2.2.1 Introduction

The program supporting the configuration entered by the designer is an interactive graphics module, which can be called by the human expert to also build the expert system configuration database as well. The major difference of this module from a general graphics package is that it does not only create an image, but also defines the design variables, functions and other data at the same time. These

data can be used to set up the subroutines and data files for the optimization and FEM procedures.

4.2.2.2 Data

The data produced are contained in eight arrays ANODE, LINE, KARC, KCURV, KDIV, KVAR, TERM, FUN.

1. ANODE stores the information about a node.
   (1) Identification of the node type.
   (2) The load and boundary conditions.
   (3) If it is a dummy node, which nodes it refers to. Dummy node will be explained shortly.
   (4) The coordinates of the node.
2. LINE stores the information about a straight line.
   (1) The node numbers of the two ends of the line.
   (2) Edge load on the line.
3. KARC stores the information about a circle or circular arc.
   (1) The node numbers of the center, starting and end nodes.
   (2) The edge load on it.
4. KCURV stores the information about a curve.
   (1) The nodes on the curve.
   (2) The edge load on it.

   There are two kinds of curves: the Cartesian curve and the polar curve. The nodes on the Cartesian curve have Cartesian coordinates, whereas the

nodes on the polar curve can have either Cartesian or polar coordinates.

5. KDIV stores the information about a region.
   (1) The corner nodes of the region.
   (2) Body load on it.
   Up to the present time, most of the automatic mesh generators are not able to set up the mesh model for an arbitrarily complex configuration. It requires that the user divide the configuration into several appropriate regions first. A region is comprised of four edges which can be straight lines, curves or arcs.
6. KVAR stores the information about the dimensional design variables.
7. TERM stores the information about the non-dimensional variables and constants.
8. FUN stores the information about the optimization and constraint functions.

Detailed explanation about variables, constants and functions will be given in section (4.2.5).

4.2.2.3 Menu

This graphics module has twelve functions which are listed in a menu.

1. LINE creates straight lines.
2. ARC creates circles or circular arcs.

3. CURVE creates Cartesian or polar curves.
4. MARK As mentioned above, for generating the FEM mesh, we have to divide a complex configuration into appropriate regions. For example in Figure 4.2, the configuration is to be divided into four regions. The nodes at points 6, 7, 8, 9 are dummy nodes located by the user in order to give the computer guidance in establishing regions. Since it is a conceptual configuration, the concrete coordinates of the dummy nodes cannot be determined until the actual dimensions of the configuration are specified later. So the dummy nodes must refer to existing nodes. For example, dummy node 6 refers to the two ends of line 1-2, dummy node 7 refers to the two ends of arc 3-2, dummy node 9 refers to origin 0 and a benchmark, i.e. MARK. For example, we can set node 5 as MARK.
5. DUMMY creates dummy nodes.
6. REGION picks up the corner nodes and therefore defines a region. A small circle will appear at the centroid of the region. Later, the user can pick up a region by locating the crosshair at this centroid point. In this project, each region consists of four edges which may be a line, arc or curve, and such information will be identified and stored automatically.
7. BOUNDARY defines the boundary conditions of fixed nodes.
8. LOAD defines concentrated loads for nodes, edge loads

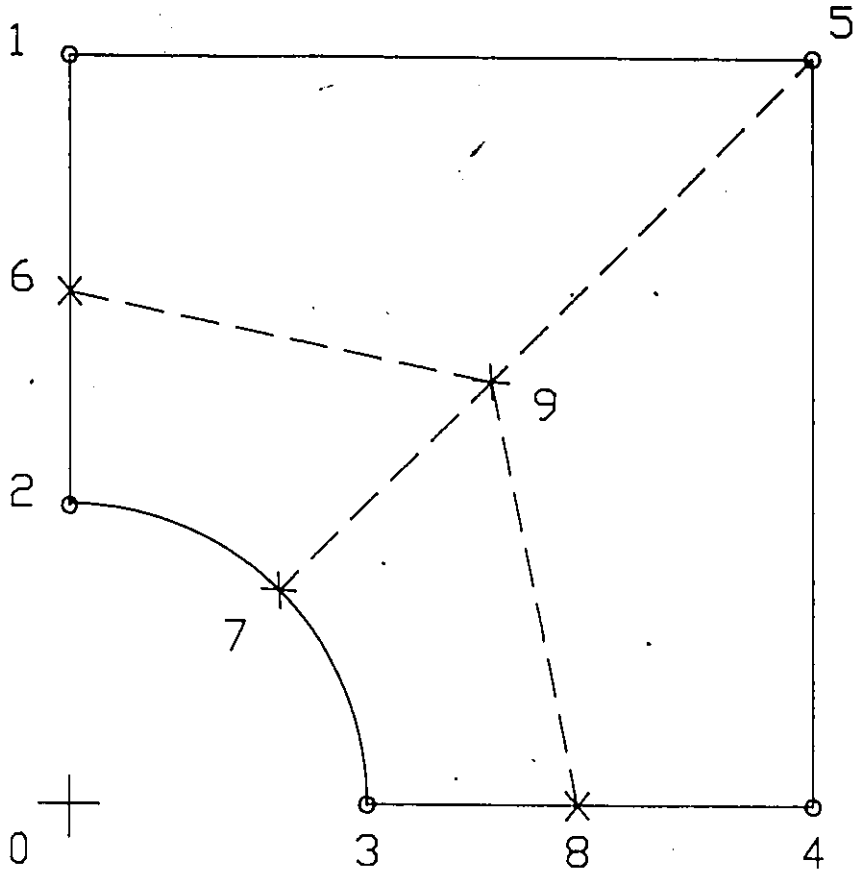


Figure 4.2 Division of regions

for lines, arcs and curves, and body loads for regions.

9. VARIABLE defines the variables and constants. More detail are given in section (4.2.5).
10. FUNCTION defines the optimization and constraint functions. More details are given in section (4.2.5).
11. SUB is used by the human expert to designate the completion of a subconfiguration.
12. EXIT on exiting from this module, the configuration on the screen is erased. All of the useful data will be written into the data files for future use.

4.2.2.4 Procedure

The program first asks the user (human expert or designer) to determine the position of the origin, and then draws the axes and a grid on the screen. A grid coordinate system is used, so that when a node is created, it will be shifted to the nearest intersection point of the grid. In this stage, the conceptual configuration is created, and the concrete dimensions are not important. What is important is to specify whether a line is horizontal, vertical or inclined, and whether a node is on the axes. The grid coordinates make it easy.

Then the menu will be shown on the right of the screen. When the user moves the crosshair into a menu item, typing the Z key will enter that item; typing the P key causes information about that menu item to be shown. The

user can enter or exit from the menu items randomly with a few exceptions. For example, before entering FUNCTION, he should enter VARIABLE first, because during the execution of LINE, ARC, REGION, and so on, the user may delete some nodes or regions. The variables related to these nodes or regions must be redefined. Similarly, before EXIT, the user should enter FUNCTION first, because if some variables have been changed, the functions have to be revised. The user need not remember these exceptions, the system will remind him at the appropriate time.

Some menu items have submenus; for example, LOAD has submenus CONCENTRATED LOAD, EDGE LOAD and BODY LOAD. FUNCTION has submenus OPTIMIZATION and CONSTRAINT. A submenu appears only after the master main menu item is selected.

The user can revise the configuration by first deleting some old primitives (i.e. node, line, and so on), and then creating new ones.

There are some chain effects as in Figure 4.3. When the user deletes a node, the lines, arcs and curves connecting to this node are deleted along with it. When a line is deleted, the dummy nodes on this line and the regions using this line as an edge are deleted too. When a dummy node is deleted, the regions taking this dummy node as a corner is deleted, and so forth.

In this module, the action the computer will execute

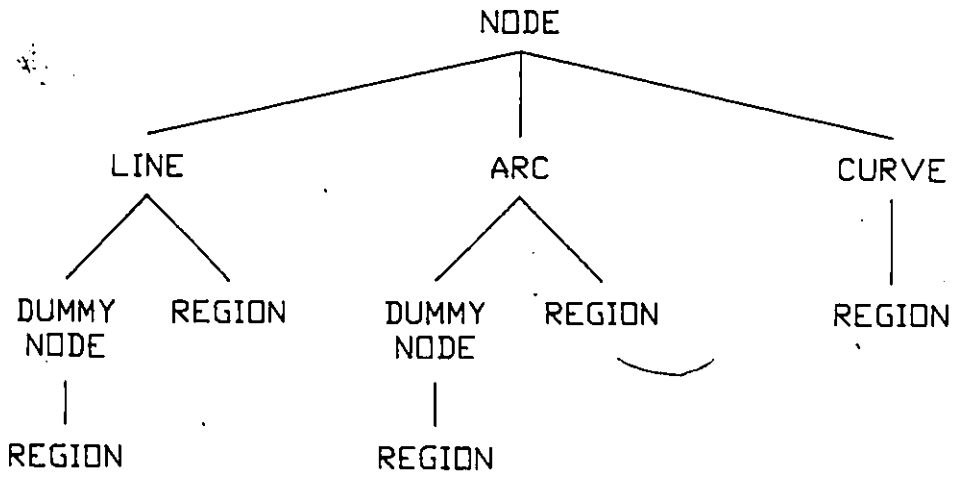


Figure 4.3 Chain effect of deletion

is determined by the combination of the position of the crosshair and the key the user has typed. On the top and the bottom of the screen, there are always some messages telling the user which keys can be options under a certain circumstance and what action will be provoked by typing a particular key. However the user need not worry about typing a wrong key, no damage will occur to the configuration. The system will give some appropriate message and suggest that the user type a correct key again, or relocate the crosshair to the correct position.

4.2.3 Input Failure Mode

If the elongation of the selected material is equal to or larger than 5%, the material is ductile. The ESSF system asks the designer to select a failure mode based on either the distortion energy theory or the maximum shear stress theory.

If the elongation of the selected material is less than 5%, it is a brittle material. The ESSF system asks the designer to make a decision between the Modified Mohr theory and the Colulomb-Mohr theory.

4.2.4 Input Factor of Safety

The ESSF system asks the designer to specify a factor of safety directly, which must be larger than one.

4.2.5 Definition of the Variables and Functions

The program in this section is actually contained in the graphics module of section (4.2.2). Since the definition of the variables and functions requires more description than other menu items, it is discussed in this separate section.

4.2.5.1 Definition of Variables and Constants

A typical function is as follows (also see Figure 4.4), it constrains that the width of the structure should be larger than the sum of the diameters of the holes.

$$g = X1 - X2 * X3 \qquad (4.4)$$

X1 (WIDTH) --- variable, width of a structure

X2 (NUMB) ---- variable, number of holes in the structure

X3 (DIA) ----- constant, diameter of the hole

Here X1 is called a dimensional variable; X2 a non-dimensional variable, and X3 a constant. The reason X3 is set as a symbol instead of a concrete number is that its value is different for different applications. However in a certain application, it is constant, not changing during the optimization search.

Five kinds of dimensional variables (Figure 4.5) have been defined. It is very easy to define a dimensional variable on the screen. For example, if the user moves the crosshair to node 1 and types the X key, the x coordinate of

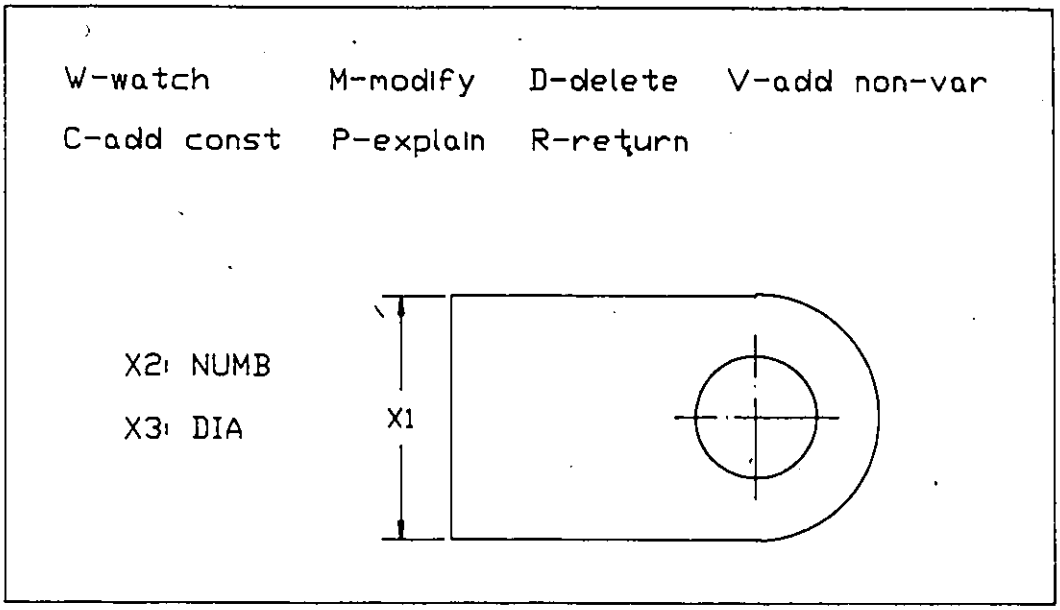
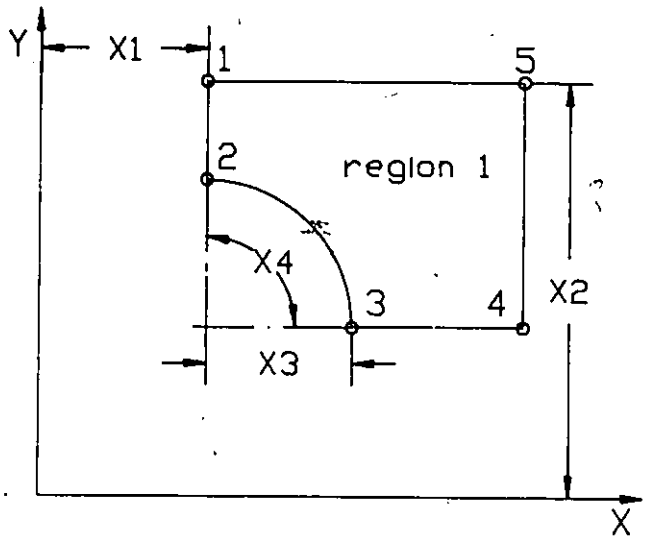


Figure 4.4 Screen display for defining variables



X1 --- x coordinate variable of node 1
X2 --- y coordinate variable of node 5
X3 --- radius variable of node 3
X4 --- angle variable of node 2
X5 --- thickness variable of region 1

Figure 4.5 Dimensional variables

node 1 has been set as a dimensional variable.

For a non-dimensional variable and a constant, the module prompts the user to type in their names (i.e. NUMB for X2), units, along with a brief explanation.

This module can be run by the human expert to build the expert system database, and a brief explanation is, of course, very helpful for the future designer. The module can also be run by the designer when he is entering his own configuration. In this case, the brief explanation serves as his own memo.

Now the screen appears as Figure 4.4. All of the dimensional variables, non-dimensional variables and constants are displayed. One can watch, modify, delete, or add any one of them by moving the crosshair to it and typing an appropriate key. For example, if the user moves the crosshair to X2: NUMB, and types W(atch), then the name, unit and brief explanation of X2 will be displayed on the top of the screen.

Finally the user has to specify the types of the variable, i.e. positive or negative gradient variables and so on. This information is required by the safe-fail line method as described in section (3.6).

4.2.5.2 Definition of Functions

This programing also can be used by the human expert to set up an expert system database, or by the designer to

specify his particular design.

The human expert can set up more than one optimization function, and rank them according to their importance. Later, the designer can select one of them.

We are interested in this work in developing an interactive-style optimization package, requiring a minimum of user knowledge and effort. It would be very difficult and time consuming for most designers to develop optimization software, and to set up the expressions for some optimization and constraint functions. For example, the expression of the buckling of a column in a complicated structure is not an easy one. It involves cumbersome analysis of loads and geometry.

It was worthwhile to put in some research effort to develop some subprograms (system functions) for the expert system in advance. They take care of the cumbersome formulation. Later, in setting up an optimization or constraint function, the user only has to move the crosshair to locate a graphic primitive, and then type a predetermined key on the keyboard.

In the present project, five system functions have been developed.

1. $V : It will calculate the volume of the structure automatically, no matter how complicated it is. However the user is encouraged to type in the volume expression if it is very simple. In this

way, some computer time can be saved.

2. $C : Some nodes on a polar curve have to be convex. By moving the crosshair to point to these nodes and typing C, the constraints are set up.

3. $S : Maximum stress in a region. By moving the crosshair to a region and typing the S key, a constraint is set up, i.e. the maximum stress in this region should not exceed an allowable value which will be specified later.

4. $D : Displacement. By moving the crosshair to a node and typing the X, or Y, or Z key, a constraint function is set up, i.e. the displacement in x, or y, or z direction of this node should not exceed an allowable value which will be specified later.

5. $B : Buckling failure mode. The user can move the crosshair to the inside of the column and type the B key. The two wireframes (line or curve) which constitute the column will become red in colour. After the user confirms that it is the column he wants to deal with, a diagram of all of the kinds of end conditions for columns is displayed. Again by moving the crosshair, the user can select a suitable end condition. Thus the constraint function is determined.

The constraint functions concerning the lower and upper bounds of the design variables need not be specified

at all; the expert system will take care of them.

All of the functions other than the above system functions should be typed in standard Fortran language, together with a brief interpretation. When defining the functions, all of the dimensional variables, non-dimensional variables and constants will be displayed as in Figure 4.4. The user can see more detailed information about a variable or constant by pointing to it and typing the P key.

A type-in function can be a calling statement as,

CALL WEIGHT (X1, X2, WE1)

The last argument (e.g. WE1) must contain the returned value of the constraint function, i.e.

g = WE1 > 0

An auxiliary file, AUXI.FOR, must be prepared to contain the called subroutines, such as WEIGT.

The system can check some errors in the functions. If an error is found, an error message will be displayed and the function is rejected. For example, if the total number of variables and constants are ten, and a symbol X13 is found in a function, it is obviously an error. Another example is that if a user wants to set up a system function $C (curvature of the nodes on polar curve), but there is no polar curve in the configuration, an error message is given and the function is rejected.

Since the user can go back to modify the configuration or change variables after he has set up the

functions, some programming code has been developed to trace the revisions in configuration and variables, so that corresponding modifications in functions can be made. For example it is assumed that the original configuration, variables and functions are as in Figure 4.6(a), and node 1 and X3 (NUMB) are deleted. Since node 1 is deleted, X1 disappears too. So g_1 and g_2 should be deleted. Besides, the original X2 is now X1, the original X4 (CONST) is now X2 (CONST), and g_3 becomes correspondingly

$$g_1 = X1 + X2$$

(See Figure 4.6(b)). When a function is deleted, notice should be given. The user may have to set up a new function to replace the deleted one.

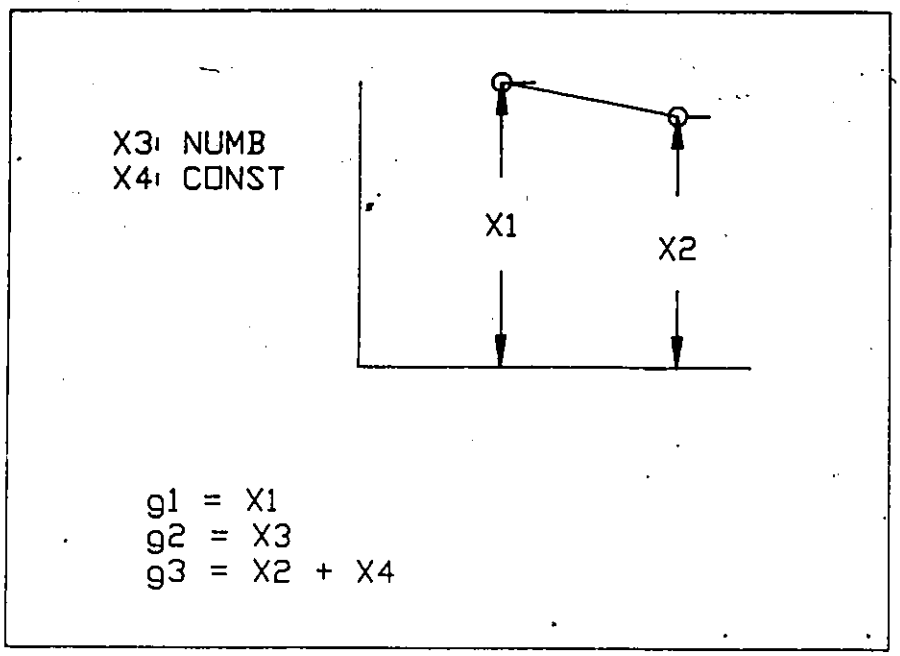
Some other influences on the system functions are as follows.

1. If one or two of the wireframes of a column are deleted, the constraint function of buckling for this column is erased.

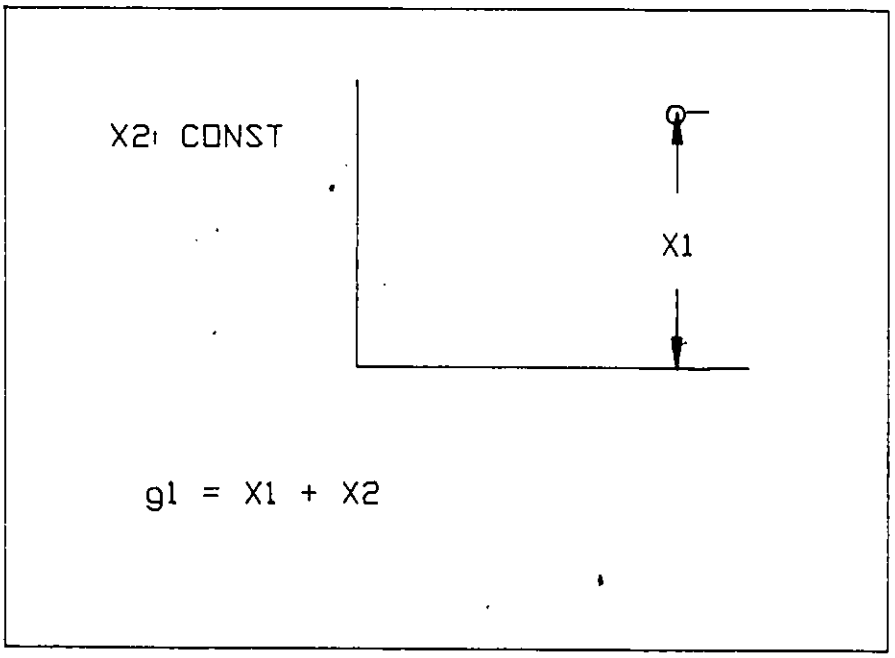
2. If a polar curve is deleted, the curvature constraint function for it is erased.

3. If a node is deleted, the displacement constraint function for it is erased.

4. If a region is deleted, the stress constraint function for it is erased.



(a) Original situation



(b) Final situation

Figure 4.6 Modification of functions

4.2.5.3 Formulation of the System Functions

1. $V (Volume of the structure)

To evaluate the volume of the structure, we divide it into several groups. Within a group, all of the regions have the same thickness. We first calculate the volumes of each group, then their sum is the volume of the whole structure.

Let us look at the annulus in Figure 4.7. The area is given by the following integration along the route denoted by the two arrows in Figure 4.7.

$$A = \int_{\ell_1} X \, dy + \int_{\ell_2} X \, dy \qquad (4.5)$$

Suppose this annulus is one of the groups of a structure and contains four regions, as in Figure 4.8(a). The dashed arrows denote the directions of regions. In region 1, there are four edges ab, bc, cd, da. Edges ab and cd are adjacent to other regions in the group, they are called common edges. Edge bc and da are called external edges. If now we cancel all of the common edges in the group, Figure 4.8(a) changes to Figure 4.8(b), which is exactly the same as Figure 4.7.

So our algorithm for evaluating the structural volume is as follows,

(1) Divide the regions of a structure into several groups in which all of the regions have the same thickness.

(2) Within each group, cancel all of the common edges.

(3) Integrate formula (4.5) for each external edge.

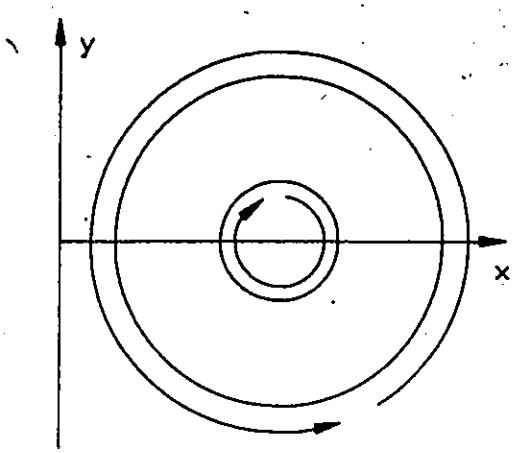


Figure 4.7 An annulus

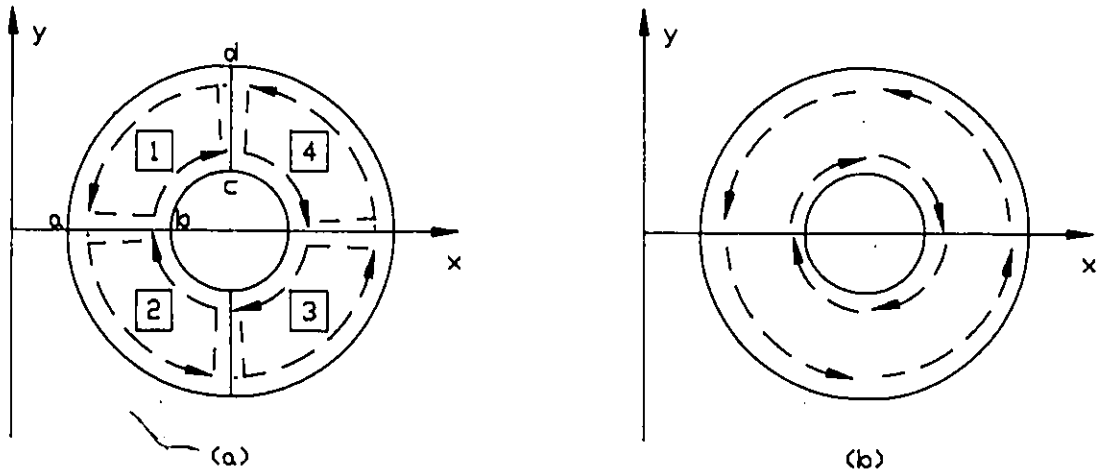


Figure 4.8 Manipulation of the regions in a group made up of an annulus

The sum is the area of the group.

(4) Multiply the area by the thickness of the group, to obtain its volume.

(5) The sum of the volumes of all of the groups is the volume of the structure.

If an external edge is a straight line, then formula (4.5)' is,

$$A = 0.5*(x1+x2)*(y2-y1) \qquad (4.6)$$

where (x1,y1) and (x2,y2) are respectively the start and end nodes of this external edge.

If the external edge is an arc,

$$A = \int_{y_1}^{y_2} x \, dy \qquad (4.7)$$

Since

$$x=x_o+R*\cos(t), \quad y=y_o+R*\sin(t), \quad dy=R*\cos(t)dt$$

then

$$A = R\int_{t_1}^{t_2}(x_o+R*\cos(t))*\cos(t)dt$$

$$= R*\{ x_o*\sin(t) + R*[0.5*t + 0.25*\sin(2t)] \}_{t_1}^{t_2} \qquad (4.8)$$

where x_o ----- x coordinate of the center of the arc

R ------ radius of the arc

t_1, t_2 -- angles of the starting and ending nodes of the external edge

For the external edges of curves, we resort to numerical integration.

If it is an axisymmetric element, formulas (4.5),

(4.6), (4.8) will become formulas (4.9), (4.10), (4.11),

$$A = \int_{\ell_1} (\pi * x^2)\, dy + \int_{\ell_2} (\pi * x^2)\, dy \qquad (4.9)$$

$$A = (1/3) * \pi * (y_2 - y_1) * (x_1^2 + x_2^2 + x_1 * x_2) \qquad (4.10)$$

$$A = \pi * R * \{ x_o^2 * \sin(t) + 2 * x_o * R * [0.5 * t + 0.25 * \sin(2t)] + R^2 * [(1/3) * \cos^2(t) \sin(t) + (2/3) * \sin(t)] \}_{T_1}^{T_2} \qquad (4.11)$$

2. $SC (Curvature of the nodes on a polar curve)

If node i (Figure 4.9) is to be convex,

$$g = (r_i - a_i) / a_i \qquad (4.12)$$

is the required constraint function. When g is negative, a concave curve occurs.

3. $SS (Maximum stress in the region)

If an FEM module is called, the principal stresses for each FEM node are returned, i.e. σ_1, σ_2 and σ_3.

(1) Distortion energy theory

$$\sigma = \{ 0.5 * [(\sigma_1 - \sigma_2)^2 + (\sigma_2 - \sigma_3)^2 + (\sigma_3 - \sigma_1)^2] \}^{0.5} \qquad (4.13)$$

The factor of safety is

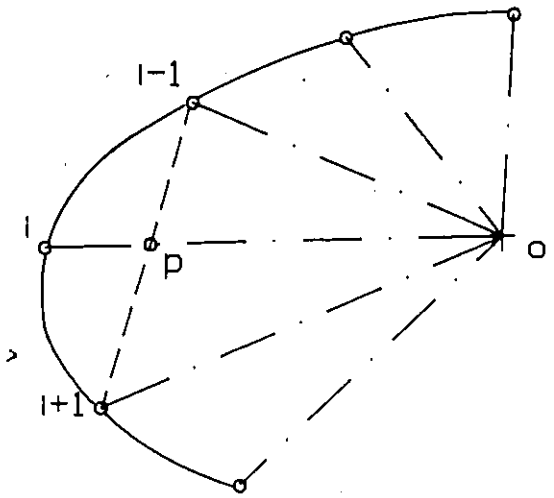
$$n = S_y / \sigma \qquad (4.14)$$

(2) Maximum shear stress theory

$$\sigma_s = \max [(\sigma_1 - \sigma_2)/2, (\sigma_2 - \sigma_3)/2, (\sigma_3 - \sigma_1)/2] \qquad (4.15)$$

$$n = S_{sy} / \sigma_s \qquad (4.16)$$

(3) Modified Mohr theory



$a_l = \overline{op}$ $r_l = \overline{ol}$

Figure 4.9 System function for a line with curvature

Its factor of safety is evaluated by simulating the graphical approach (Shigley, 1983), shown in Figure 4.10,

$$n = OC / OA \qquad (4.17)$$

(4) Coulomb-Mohr theory

Again from Figure 4.10,

$$n = OB / OA \qquad (4.18)$$

The constraint function is,

$$g = \text{minimum } (n) - N \qquad (4.19)$$

where N is the required factor of safety. If g<0, the structure is over stressed.

The two failure theories for brittle materials return only the calculated factor of safety, so to make the procedure consistent, the constraint function is expressed in terms of a factor of safety instead of stress.

4. $D (Displacement of a node)

When the FEM module is called, the displacement d of the node returns,

$$g = \text{allowable displacement} - d \qquad (4.20)$$

5. $B (Bending of a column)

Since this project deals with two-dimensional problem only, the cross sections of the columns are assumed to be rectangles.

When a column is distinguished, the FEM module returns the stress components at the middle of the column, from which the actual compressive force loaded on this

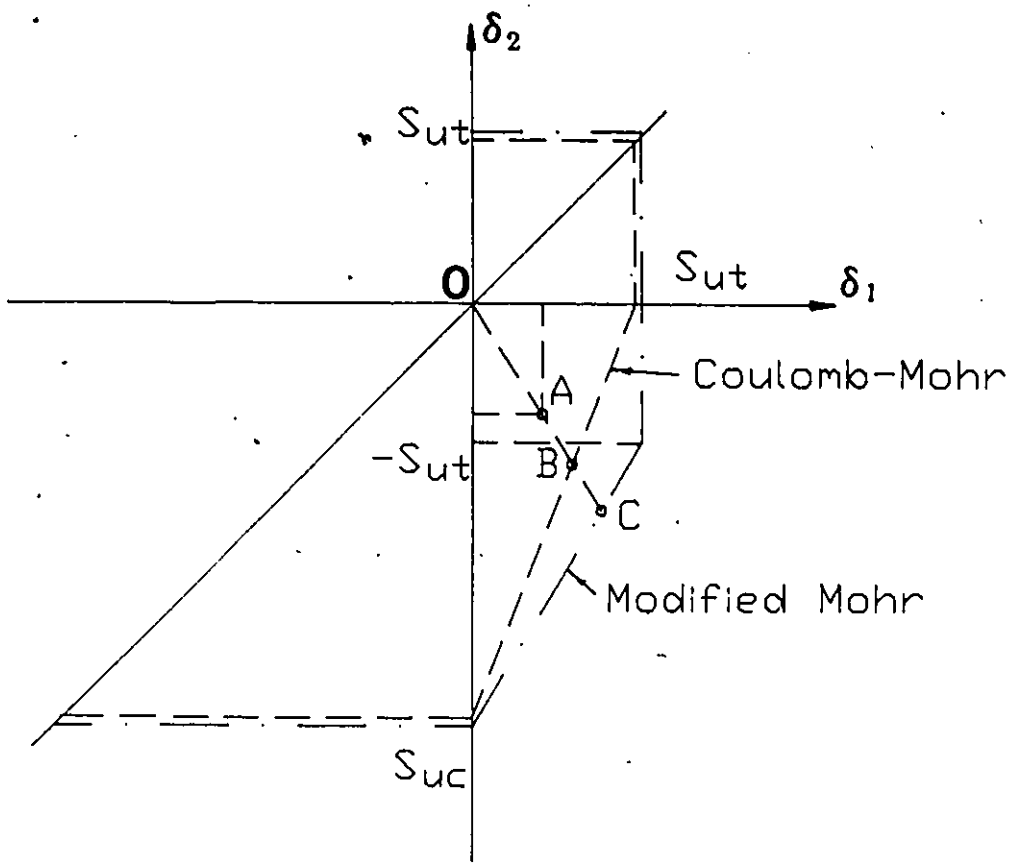


Figure 4.10 Modified Mohr theory and Coulomb-Mohr theory

column can be evaluated.

$$g = \text{critical force} - \text{actual force} \qquad (4.21)$$

$$\text{critical force} = \pi * E * I * n / l^2 \qquad (4.22)$$

where E --- modulus of elasticity

I --- moment of inertia

$$I = \min(t*w^3, w*t^3)/12$$

where t is the thickness, and w is the width in the narrowest point.

n --- 0.25 - 1.2, depending on the end condition

l --- length of the column

$$\text{actual force} = \sigma_c * w_c * t$$

where σ_c -- stress in the middle of the column along its axis

w_c -- width of the column in the middle

The quantities t, w, l and w_c can be evaluated from the geometry of the column. σ_c can be calculated from the stress components. The stress distribution at the middle is far more uniform than at the ends, so the actual force is calculated at the middle.

The last three system functions require an FEM calculation. However it is not called three times separately. All of the information required by these three system functions is returned together in one call.

After the designer has entered all of his design parameters, he will go to section (4.6) for the next design

procedure in consultation with the expert system.

4.3 SPECIFICATION OF THE IMPORTANCE VALUES

This section is relevant to the dependent or semi-dependent mode, in which the designer selects design parameters from the expert system database.

First of all, the designer has to specify the importance values of the performance characteristics which represent the requirements of the design. The expert system shows a bar chart on the screen (Figure 2.7). The height of each bar represents the importance value of a performance characteristic. The original heights represent the standard importance values determined by the human expert in section (2.3.2). The importance values, associated with the performance characteristics are relative values. Instead of requesting them one by one, the expert system displays all of them on a bar chart. The designer can compare them intuitively in one look.

If the designer thinks that all of the standard importance values are satisfactory, no action is required, and the whole procedure in this section is finished. The standard importance values should be basically suitable for many applications.

If the designer finds that some importance values are not quite suitable for his particular design application, he just moves the crosshair to the desired

position on the corresponding bar, then types the M key on the keyboard; the height of the bar representing that importance value will change to the crosshair's position. If the designer types the P key when the crosshair is on a bar, the information about the pertinent performance characteristic will be shown on the top of the screen. For example, if the crosshair is in the bar for cost, the following information appears.

"COST includes the material, labor, cost of manufacture, and so on. If you want the cost to be lower, its importance value should be larger."

If at any time the designer types the H key, the overall information about the bar chart will be displayed.

As mentioned in section (2.3.2), there are some performance characteristics, such as temperature or load, for which it is more convenient and intuitive for the designer to enter the direct values (e.g. value of load) instead of importance values. Let us take load as an example for such a performance characteristic. As shown in Figure (4.11), when the crosshair is on the bar for load, and the designer types an M key, ten interpolation load values will be shown on the left of the load bar. These interpolation values were determined by the human expert in section (2.3.2). For the case shown in Figure 4.11, the load is 850 lb and its importance value is 5.5. The 850 lb is also shown on the top of the screen.

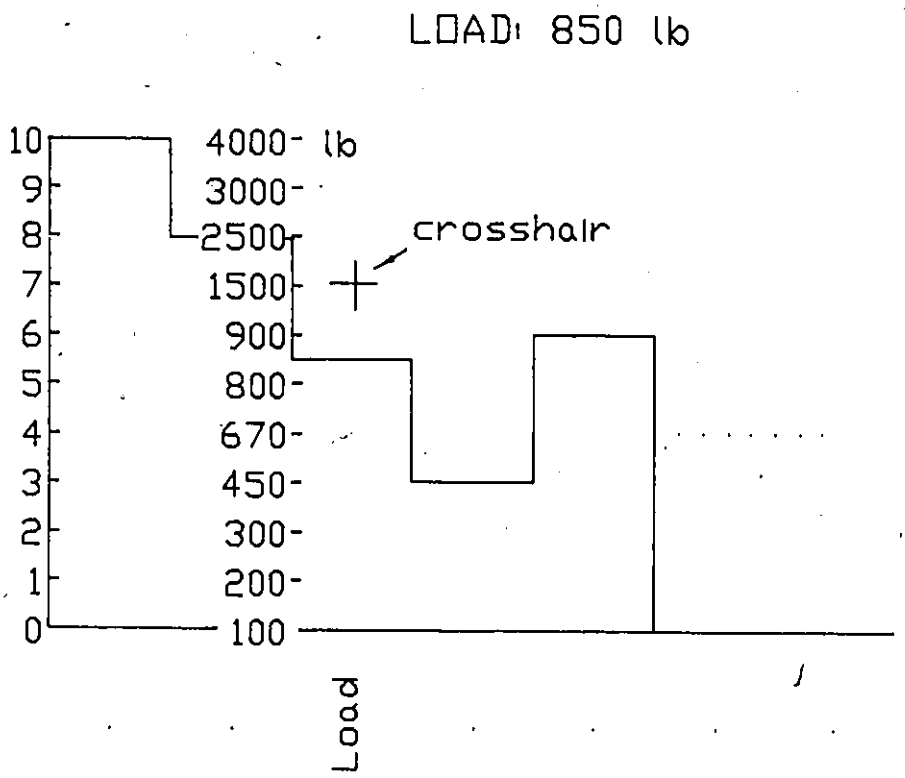


Figure 4.11 Direct values on the bar chart for the importance values

If the designer knows the load in his application is 1500 lb, he moves the crosshair to the position corresponding to this load as in Figure 4.11, and types the M key once again. The equivalent importance value, 7.0, of load will be evaluated by the expert system using linear interpolation. The load value on the top of the screen will be changed to LOAD:1500, and the ten interpolation values of load disappear from the screen.

When the designer finishes modifying the importance values, the system will check them by some interactive rules or number rules provided by the human expert in section (2.3.2). When the condition of a rule is satisfied, the expert system will suggest a modification to the designer, and let him determine whether the modification is accepted or rejected.

4.4 SELECTION OF MATERIAL AND CONFIGURATION

4.4.1 Introduction

This section also applies to the dependent or semi-dependent mode.

The procedures for selecting material and configuration are very analogous. The former are described in some detail first, and then the exceptions of the latter are introduced.

4.4.2 Selection of Material

Figure 4.12 is the flowchart for the selection of material.

1. At the request of the designer, the expert system displays a brief introduction to the three search methods, i.e. the full search method, the discard search method, and the quick search method discussed in section (2.2.4). Then the designer decides on a method.

2. According to the decision of the designer, the expert system adopts a search method to evaluate the goodness of the material candidates in a class level, based on formula (1.1), where the importance values are determined by the designer in section (4.3) and the desirability values are adapted by the super-expert system in section (2.3.4).

3. The names of material classes and the values of properties of a typical material in each class are displayed. They are ranked from top to bottom according to their goodness, i.e. the best material class is on the top. The expert system displays all of the material classes instead of only the best one, in order to give the designer more opportunities for using his own judgement.

4. At the request of the designer, the expert system displays two bar charts as in Figure 4.13, where plain carbon steel is supposed to be the best material class at the moment; other classes are listed at the bottom. The

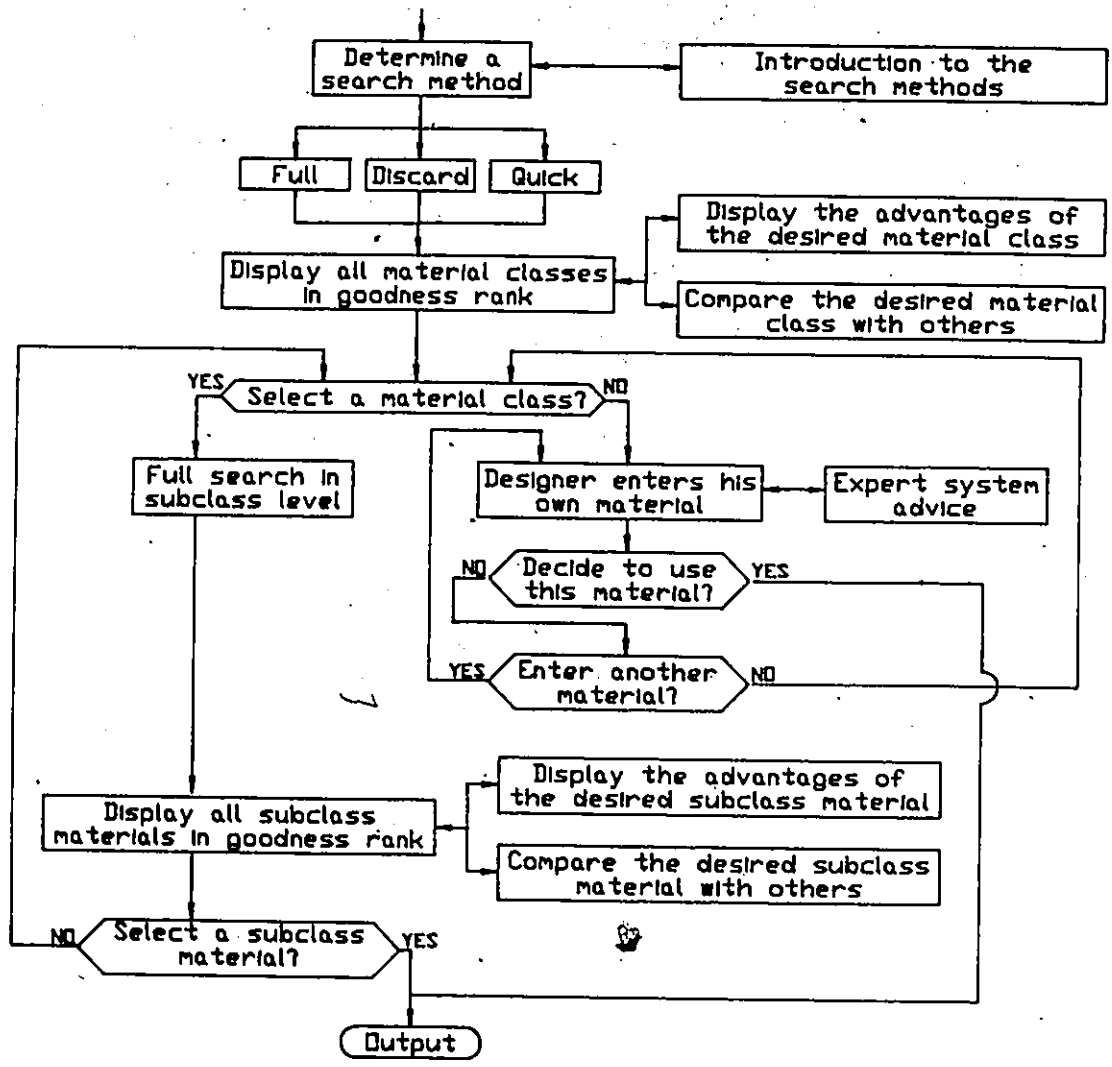
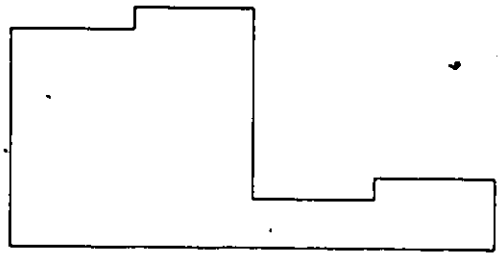


Figure 4.12 Flowchart of the selection of material

Importance values of performance characteristics

Desirability values of PLAIN CARBON STEEL



Other material classes:

1 ALLOY ST 2 STAINLESS ST 3 GRAY IRON

Figure 4.13 Display the advantages of the desired candidate

designer has had some ideas about the bar chart already, since he has used it to specify the importance values of the performance characteristics. So Figure 4.13 is a very intuitive explanation of why a candidate is considered best. In this example, it can be seen that the importance values of cost and accuracy are very high, and plain carbon steel is quite desirable when considering cost and accuracy. The bar chart is superior to most expert systems which explain the results of their deductions by a brief text.

5. If the designer types the number of a material class at the bottom of Figure 4.13, say 2, the expert system will display the bar chart of the desirability values of stainless steel too. Figure 4.13 changes to Figure 4.14, and the merit values of plain carbon steel and stainless steel are displayed together. The designer can recognize immediately that plain carbon steel is superior to stainless steel, due to the lower desirability value of cost of the latter.

6. If the designer does not like any material class in the expert system database, he can enter his own material (new material). He should provide the properties of the new material which must be complete enough for the later FEM analysis. If the designer is able to point out which material class the new material belongs to, the expert

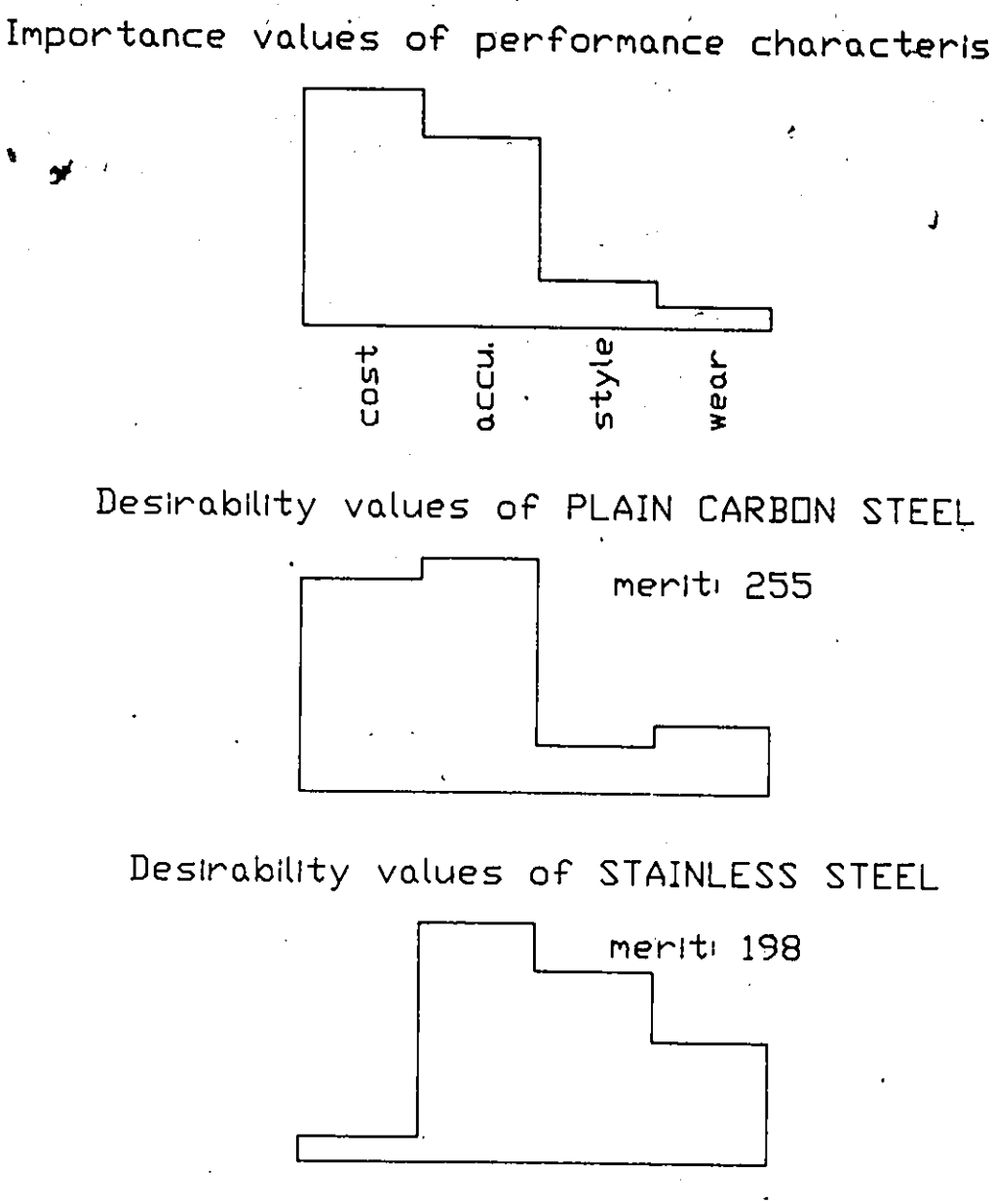


Figure 4.14 Comparison of two candidates

system can find out the closest material within that material class. Now if some properties of the new material are missed, they can be supplemented by referring to the closest material as in section (4.2.1). Then the expert system will provide some useful information to let the designer have some feeling about how good the new material is for the design. Since the new material has no desirability values, the expert system can only make use of some characteristic values. The expert system compares each characteristic value of the new material with that of the material desired by the expert system (desired material) to get a relative suitability value, for example,

$$\text{(Relative suitability value of strength)} = \frac{\text{(Strength of new material)}}{\text{(Strength of desired material)}} * 100\% \quad (4.24)$$

The following suitability values will be dealt with by the expert system.

(1) If (importance value of cost) > 7, consider

(a) Cost/weight

(b) $$\text{Cost/strength} = \frac{\text{(cost/weight)*(unit weight)}}{\text{strength}}$$

(c) $$\text{Cost/stiffness} = \frac{\text{(cost/weight)*(unit weight)}}{\text{modulus of elasticity}}$$

(2) If (importance value of weight) > 7, consider

(a) Unit weight

(b) Weight/(unit strength) = $\frac{\text{unit weight}}{\text{strength}}$

(3) If (importance value of volume) > 7, consider

(a) Strength

(4) If (importance value of stiffness) > 7, consider

(a) Modulus of elasticity

(5) If (importance value of impact) > 7, consider

(a) Elongation

(6) If (importance value of wear) > 7, consider

(a) Hardness

Now the designer should make a decision on one of the following options.

(1) Accept this selected material, go to output.

(2) Enter another material.

(3) Go back to the expert system database and select a material there.

7. If the designer selects a material class in the database, the expert system will search in this material class to evaluate the goodness of all of its subclasses. Since the subclass database is usually small, the full search method is exclusively adopted in this level.

8. Again at the request of the designer, the expert system can show the advantages of the desired subclass material or compare it with others, as in Figures 4.13 and 4.14.

9. Finally, the designer decides on a material, or goes back to consider others.

4.4.3 Selection of Configuration

After evaluating the goodness of all of the configuration candidates, the expert system displays all of them on the screen with a goodness number on the top of each configuration. If the goodness number of a configuration is one, it is the one recommended by the expert system. At the class level, the display includes the first subclass configuration of all of the classes. At the subclass level, it includes all of the subclass configurations in that class. Any configuration in the display can be enlarged to the full size of the screen if the designer wishes.

The designer has several options.

1. He can decide on a configuration from the expert system database without any modification.

2. He can decide on a configuration in the database, but modify it more or less.

3. If he does not like any configuration in the database, he has the alternative of entering a new configuration.

The capability of the designer being able to modify a configuration in the expert system database is very important. Any predetermined configuration can hardly satisfy a new application completely. Now the designer can

pick up from the expert system database a configuration which satisfies his application best and then modify it to the point of satisfying his application completely.

The designer can modify not only the configuration, but also the variables relating to the configuration, and the functions defined in term of these variables.

The same graphics module described in section (4.2.2) is used to do the modifications.

4.5 SELECTION OF FAILURE MODE AND FACTOR OF SAFETY

4.5.1 Selection of Failure Mode

The expert system first advises if the material selected by the designer (determined material) is ductile or brittle. For example, if it is ductile, the following information will be displayed.

1. The two failure modes for ductile material, i.e. the distortion energy theory and the maximum shear stress theory.

2. Brief statements about the advantages and disadvantages of each failure mode.

3. Which failure mode the expert system recommends.

Then the designer finally selects a failure mode.

4.5.2 Selection of Factor of Safety

The designer determines the FS in consultation with the expert system in four steps.

1. The standard material, standard configuration and the reference FS are displayed. They are specified by the human expert in section (2.3.6). Explanations about the procedure for the determination of the reference FS will be given if the designer requests it. Some explanations were provided by the human expert, and some are provided by the super-expert system based on the importance values of some performance characteristics, e.g. reliability, temperature, corrosion, and so on.

2. The standard material and the determined material are then displayed at the same time. If the determined material is selected from the expert system database and is different from the standard material, some modification of the reference FS may have been done by the human expert earlier. Such a modification, if it exists, and the corresponding explanation, will be shown. By comparing the standard and determined material, the designer can either accept the reference FS or modify it by his own judgement.

3. Standard and determined configurations are treated similarly to materials.

4. For the design of certain kind of structures, the human expert has set the standard importance values of performance characteristics. However the importance values specified by the designer (determined importance values) are usually significantly more or less than the standard

importance values. The expert system will check the difference between the standard and determined importance values. If the difference is large, for example, the standard importance value for reliability is 4, the determined importance value is 8, a warning message is given

The standard importance value for reliability is 4
The determined importance value for reliability is 8
Do you wish to increase FS ? (Y/N)

The designer can either ignore the warning message or do some modification. The designer can also modify the FS, based on a consideration of the actual size of the configuration.

4.6 DETAILED SPECIFICATIONS FOR THE APPLICATION

This section includes the specifications for the concrete dimensions of the configuration, the bounds and starting values of the design variables and loading.

4.6.1 Specifying Dimensions

The configuration determined so far (from the expert system database or from the designer's input) is just a conceptual configuration, since the concrete values of dimensions have not been determined yet. The most straightforward algorithm for specifying the dimensions is to specify the x and y coordinates for each node. As in Figure 4.15, it is necessary to specify x1, x2, x3, x4, y1,

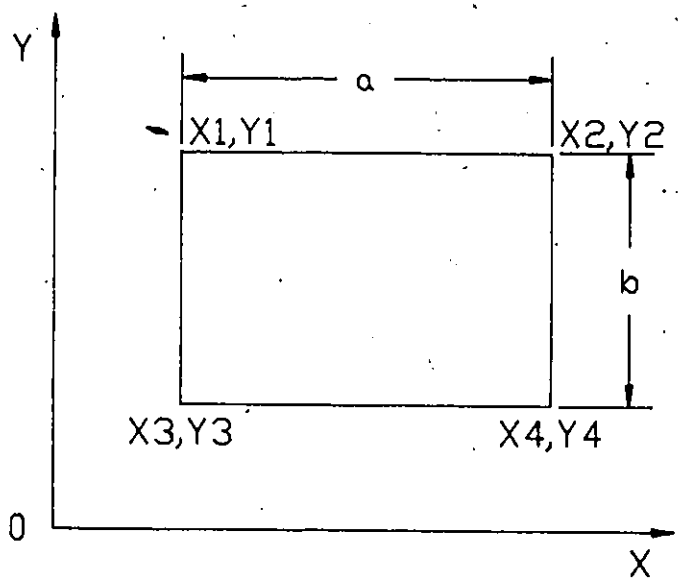


Figure 4.15 Specification of dimensions

y2, y3, y4. However in a conventional design, the designer has to specify only a and b, since it is a rectangle.

So it is necessary to incorporate some intelligence of the designer in conjunction with the expert system procedures in order to make it competitive with conventional design. In our current program, the designer can specify the x or y coordinates of a node; the horizontal or vertical distance between two nodes; the angle and radius of a node on an arc, circle or polar curve; and the thickness of a region. The eventual purpose is to determine the x and y coordinates of all nodes and thickness of all regions. The procedure is as follows.

1. x coordinate

The crosshair is set at a node, X is typed, and the scale line will then appear. The value is then typed in (Figure 4.16).

2. y coordinate

Similar to x coordinate.

3. Horizontal distance between two nodes

The two nodes are successively pointed to, and then H is typed. The value is then entered (Figure 4.17). If the x coordinates of neither or both of these two nodes have been specified, an error message will appear. If the x coordinate of only one node has been specified, the x coordinate of another will be thus determined.

4. Vertical distance between two nodes

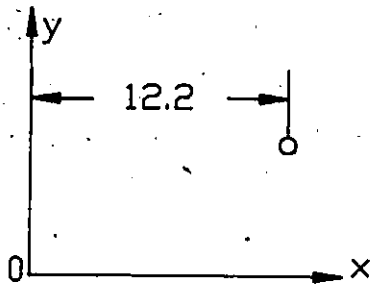


Figure 4.16 Specification of x coordinate

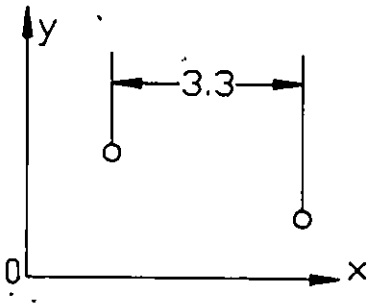


Figure 4.17 Horizontal distance between two nodes

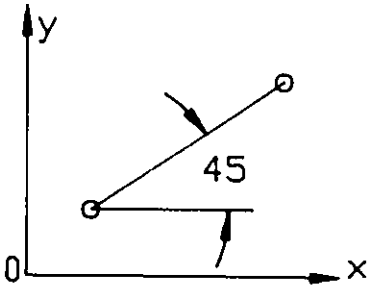


Figure 4.18 Specification of angle

Similar to horizontal distance.

5. Angle

If a node is on a circle, or an arc, or a polar curve (it is called an angular node for short), the user points to it, types A, and then types in the value of its angle with respect to the x axis (Figure 4.18).

6. Radius

The user points to an angular node, types R and then types in the value of its radius. For an angular node, any two of the four dimensions (i.e. x coordinate, y coordinate, angle and radius) can uniquely determine the position of the node. The system will reject the redundant specifications. If two nodes are on a same arc or circle, then when the radius of one node is determined, the radius of another node will take the same value. In Figure 4.19, if A is an angular node with O as center, and the horizontal distance between O and A has already been specified as 12.0, and if the designer now specifies its radius as 10.0, an error message appears, telling the designer that the radius is too short (should be equal to or greater than 12.0). The x and y coordinates of A are calculated as,

$$X_A = X_O + R*\cos(a)$$
$$Y_A = X_O + R*\sin(a) \qquad (4.25)$$

If designer first specifies the x and y coordinates of node O (center), and then specifies the angle and radius of node A, there is no problem. However if the designer first

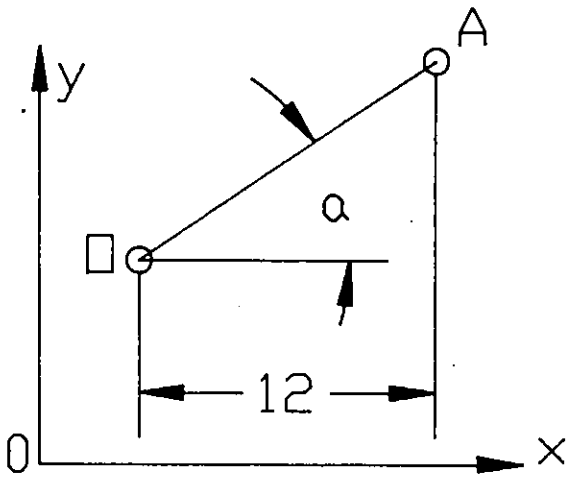


Figure 4.19 Angle node

specifies the angle and radius of node A, then goes on to specify the dimensions of some other nodes, and finally comes back to specify the x, y coordinates of node O (a draftsman may do it this way), some difficulties arise. This is because X_A and Y_A cannot be determined until X_O and Y_O are specified. So R and a must be stored. Later when X_O and Y_O are specified, it will be found that node A takes node O as center, and R and a are available already. So using formula (4.25), X_A and Y_A can be now evaluated. In this program, the sequence of specifications of dimensions is random.

7. Thickness

The region is pointed to, T is typed, and then the value of the thickness is entered.

As mentioned earlier, by utilizing the intelligence of the draftsman, the x and y coordinates of all nodes need not be specified; some of them can be inferred as follows.

1. If a node is on the x (or y) axis, its y (or x) coordinate is zero.

2. If node A and node B are on a horizontal (or vertical) straight line, and the y (or x) coordinate of node A is known, the y (or x) coordinate of node B must have the same value.

3. If node A and node B are symmetric about the x (or y) axis, and the y (or x) coordinate Y_A of node A is known, then $Y_B = -Y_A$ (or $X_B = -X_A$).

4. If the thickness of some regions have not been specified, they can take the value that most of the specified regions have been given.

5. Finally if there are still coordinates of some nodes that have not been determined, the designer is prompted to select a specified node as a benchmark, and the system will evaluate automatically the coordinates of all unspecified nodes as follows.

$$X_u = \frac{x_u}{x_m} * X_m \qquad (4.26)$$

where X_u --- real coordinate of the unspecified node

x_u --- conceptual coordinate of the unspecified node

X_m --- real coordinate of the benchmark

x_m --- conceptual coordinate of the benchmark

So if the conceptual configuration is exactly the same as the real configuration, only the coordinates of the benchmark has to be specified.

When the coordinates of all the nodes have been determined, the coordinates of the dummy nodes can be evaluated by referring to relevant nodes. For example, if dummy node D is on line A-B (Figure 4.20),

$$X_D = (x_d - x_a)*(x_b - x_a)/(X_B - X_A) + X_A \qquad (4.27)$$

where X_A, X_B, X_D ----- real coordinates

x_a, x_b, x_d ----- conceptual coordinates

The dummy node's coordinates need not be specified by the

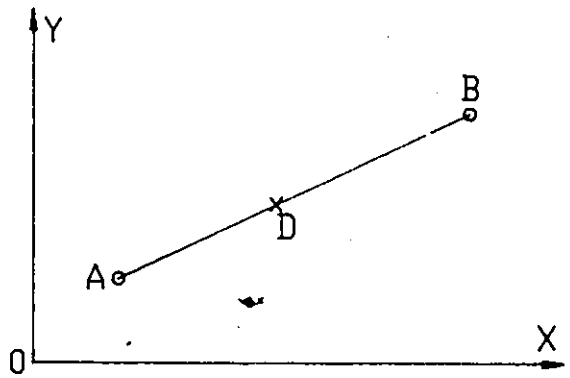


Figure 4.20 A dummy node on a straight line

designer.

The expert system uses a different colour to distinguish the status of the specification of dimensions of nodes at the CRT screen.

unspecified ---------- white

only x specified ------ blue

only y specified ------ cyanosis

fully specified ------- green

At any time, if the designer points to a node or a region, and types an appropriate key on the keyboard, the x coordinate, or y coordinate, or angle, or radius, or thickness will be displayed.

When the specification is finished, the configuration will be re-drawn according to the real coordinates. The scale factor is set automatically so that the configuration always occupies 70% of the total screen. At this time, the designer can still go back to modify any dimensions again.

4.6.2 Specifying Variables

When the real configuration is determined, the designer can specify the lower and upper bounds, and starting values of the variables and constants.

4.6.2.1 Dimensional Variables

Each dimensional variable is highlighted when it is

to be specified. The value specified in section (4.6.1) will be taken as the starting value, and the bounds should be typed in when prompted by the expert system.

Usually a dimensional variable controls more than one dimension. The expert system should be able to find out such implications, called variable induction, which is similar to the induction of configuration dimensions in section (4.6.1). The following are examples of variable induction.

x, y variables

(a) If an x or y coordinate of a node is defined as a variable, the expert system first finds out its symmetric partner. For example, in Figure 4.21, the y coordinate of node 1 is a variable. Node 4 is symmetric with node 1 about the x axis, and the lines (lines 1 and 3, lines 2 and 4), which are connected to nodes 1 and 2 respectively, are symmetric about the x axis too. In this situation, node 4 is called the symmetric partner of node 1. For example, if in an optimization iteration, y_1 equals 10, then y_4 must equal -10.

(b) Another example of a symmetric partner is in Figure 4.22. The x and y coordinates of node 1 are variables. Nodes 2, 3 and 4 are symmetric with node 1 about the x axis, the y axis and the origin respectively. Nodes 1, 2, 3 and 4 are all centers of arcs; the radii of the arcs are the same; and the corresponding angles are symmetric

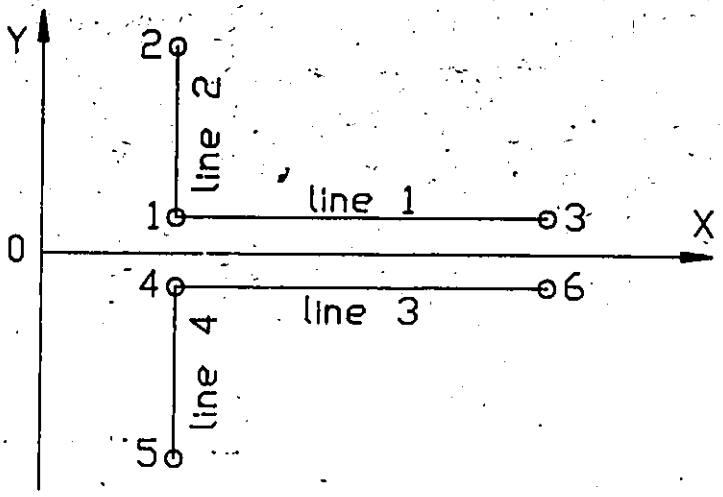


Figure 4.21 Induction of xy coordinates

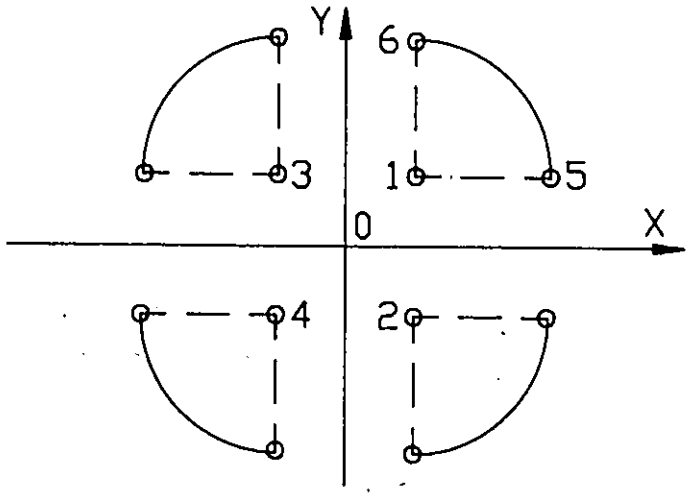


Figure 4.22 Induction of other variables

about the x axis, the y axis, or the origin. So nodes 2, 3 and 4 are all symmetric partners of node 1, and their x or y coordinates relate to the same variables.

(c) In Figure 4.21, since the y coordinate of node 1 and node 4 are determined by a same variable, and line 1 and line 3 are horizontal, the y coordinates of nodes 3 and 6 are also controlled by the same variable.

(d) In Figure 4.22, nodes 5 and 6 are in the same arc. If the x or y coordinate of node 5 is a variable, then the coordinate of node 6 must also be controlled by this variable.

(e) If a node is the center of an arc, circle or polar curve and contains an x or y coordinate variable, then the x or y coordinates of all nodes on the arc, circle or polar curve must be controlled by this variable.

R (radius) variable

If a radius is defined as a variable, all of the radii which have the same value as this radius will be defined by the same variable. For example, in Figure 4.23, r1=r2=r3, and if r1 is defined as a variable, r2 and r3 will be controlled by the same variable.

T (thickness) variable

If the thickness of one region is defined as a variable, the thickness of all of its adjacent regions with the same value of thickness will be controlled by the same

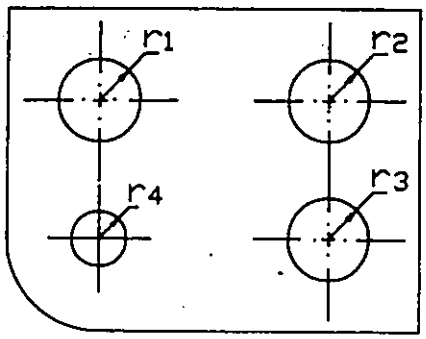


Figure 4.23 Induction of radius variables

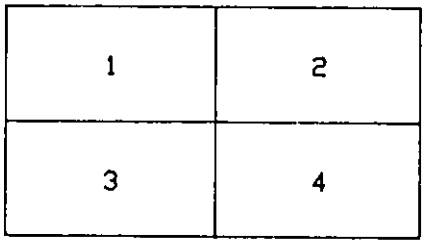


Figure 4.24 Induction of thickness variables

variable. Such induction will involve more and more regions recursively. For example, in Figure 4.24, the thickness of regions 1, 2, 4 are 4.5, and the thickness of 3 is 3.5. If the thickness of region 1 is defined as a variable, then by the above rule, the thickness of its neighbor, region 2, is determined by this variable; as is the thickness of region 4 which is the neighbor of region 2.

The expert system displays the above inductions once for a variable. The logical induction, which implies some expertise, may occasionally make mistakes. So after inspecting the inductions for a variable, the designer has three options.

(1) Accept: the induction is all right.

(2) Add: the designer finds the induction missed something. The variable should control more dimensions. He can accomplish this by moving the crosshair to the nodes or regions and typing the A key.

(3) Delete: the designer finds there are too many inductions. He can move the crosshair to some nodes and type the D key, causing the variable to lose control of the dimension.

There are three purposes for variable induction.

(1) Reducing the burden on the designer to necessarily define all of the coordinate variables and thickness variables.

(2) Reducing the number of independent design variables.

(3) Making all of the dimensions controlled by an independent design variable change synchronously in each optimization iteration.

4.6.2.2 Non-dimensional Variable

When a non-dimensional variable is to be specified, its name, unit and a brief explanation will be shown. The designer types in the lower and upper bounds. The mean value of the lower and upper bounds is taken as the starting value unless the designer wants to specify it.

For example, when the non-dimensional variable X2 in Figure 4.4 is to be specified, the following text will be displayed on the screen.

```
NAME:  NUMB
UNIT:
Number of the holes in the structure
```

The designer types in 2 as a lower bound and 6 as an upper bound; then the mean value 4 is taken as the starting value of X2.

4.6.2.3 Constant

The name, unit and a brief explanation will be displayed. The designer just types in the value for the constant.

When all of the variables and constants have been

specified, the designer may display them one by one, so they can be checked or modified.

4.6.3 Specifying of Load

The expert system prompts the designer to specify the loads one by one. After all of the loads have been specified, the designer can check or modify them by pointing to a node (for concentrated load), or line, arc, curve (for an edge load), or a region (for a body load), and typing an appropriate key.

4.7 DETERMINATION OF THE OPTIMIZATION METHOD AND THE FEM ALGORITHM

All the options of the optimization methods and FEM algorithms, and the related parameters, are displayed to the designer and one is recommended by the expert system as best. The designer makes the decision, after considering the recommendation of the expert system. There is a text for each specification to describe the advantages and disadvantages of each option. These text can be reached at any time by typing the H key.

Finally the expert system will check if all of the selected algorithms for FEM based optimization are valid for use, mainly to check if the computer memory allocated to an algorithm is large enough. If it is not, a message is displayed, and one of following actions is brought into effect.

1. Some algorithms are abandoned due to the shortage of computer memory for large applications.

2. Some parameters of the algorithm are adjusted automatically. For example, if the designer specifies too many safe or fail lines, the system will reduce them a little so that the allocated memory is just large enough. Therefore the safe-fail line method is still valid for adoption with a fewer number of safe or fail lines.

4.8 FORMATION OF THE OPTIMIZATION AND CONSTRAINT FUNCTIONS

In this project, the software **OPTIVAR** (Siddall, 1982) has been used for optimization. A main program and two normally user-written subroutines UREAL and CONST are required. UREAL is called to evaluate the optimization function and CONST is used to evaluate the constraint functions. However in our system, all of the functions and subroutines are created internally.

4.8.1 Implementation of the Main Program

From the expert system database, METHD, NPENAL, NCONS, KPV, KPVN, RMIN, RMAX, XSTRT can be read directly, where

METHD --- optimization method selection index

NPENAL -- penalty function index

NCONS --- number of constraint functions

KPV ----- number of dimensional variables

KPVN ---- number of non-dimensional variables

RMIN ---- lower bounds of variables

RMAX ---- upper bounds of variables

XSTRT --- starting values of variables

The total number of design variables N is,

N = KPV + KPVN

From N, NCONS and the selected optimization method, the expert system can calculate the size of working array W for **OPTIVAR.**

Another task of the main program is to establish a common block which contains the data about nodes, lines, arcs, curves, variables, and so on. Since the main program is separate from subroutines UREAL and CONST in **OPTIVAR**, the data can be communicated through this common block only.

The main program is named OPTZ.FOR.

4.8.2 Implementation of Subroutines UREAL and CONST

The subroutines are created internally by the system. An auxiliary program CONVERT has been developed, which reads in the character array FUN containing the optimization and constraint functions. FUN was defined by the human expert or the designer in section (4.2.5). After some conversion, CONVERT writes FUN into a file, UOC.FOR, which contains the two subroutines UREAL and CONST.

To create each subroutine, say UREAL, CONVERT first writes the following statements into UOC.FOR.

```
      WRITE (1,10)
10    FORMAT (
     *        8X, 'SUBROUTINE UREAL (X,U)'/
     *        8X, 'DIMENSION X(*)')
```

And at the end of a subroutine,

```
      WRITE (1,30)
30    FORMAT (
     *        8X, 'RETURN'/
     *        8X, 'END')
```

The system then creates the optimization function and inserts it into the middle. There are four kinds of functions created: (1) Bound functions, (2) Call statement functions, (3) Expression functions, (4) System functions.

1. Bound function

Bound functions define the upper and lower bounds of the design variables, and are a type of constraint function. Since these bounds are available in the expert system database already, it is easily done. For example, if the lower and upper bounds of the second variable are 2.0 and 4.0 respectively, CONVERT would insert two lines of statements in UOC.FOR to be used in subroutine CONST,

```
PHI(3) = X(2) - 2.0
PHI(4) = 4.0 - X(2)
```

PHI is the array containing all of the constraint functions.

2. Call statement function

Call statement functions allow the designer to define functions involving rather complicated procedures other than the FEM calculation. CONVERT first picks up all of the variables and constants in the input call statement, and substitutes the values for constants. The last argument is exclusively changed to "Z", which will return the function value. For example, if the Ith input constraint function is as follows,

```
CALL ABC (D1, D2, D3, VALUE)
```

Then the expert system finds that D1 and D2 are variables, and D3 is a constant equal to 4.2. Then the following two lines will be written to UOC.FOR

```
CALL ABC (X(1), X(2), 4.2, Z)
PHI(I) = Z
```

If the above call statement is the optimization function, it is written to UOC.FOR as,

```
CALL ABC (X(1), X(2), 4.2, Z)
U = Z
```

where U is the value of the optimization function.

In an application, if call statement functions exist, a supplementary file AUXI.FOR should be prepared by the designer. AUXI.FOR contains the called subroutines, e.g. ABC as above. AUXI is linked to the main program OPTZ later.

3. Expression function

Expression functions are equivalent to the Fortran

assignment statement, which are typed in by the user interactively. An example of an expression function is

D1 + D2 - D3

where D1, D2 and D3 are defined as in the above call statement. If it is an optimization function, it will be converted to

U = X(1) + X(2) - 4.2

If it is the Ith constraint function, it is,

PHI(I) = X(1) + X(2) - 4.2

4. System function

System functions have been built in by the system developer for the convenience of the user, in order to define some complicated functions in a certain domain. There are five system functions which have been introduced in section (4.2.5) already.

$ V calculates the volume of the structure. CONVERT writes a statement in UOC.FOR,

CALL SYSU10 (U)

Subroutine SYSU10 will return the value of the volume of the structure.

$ C is the constraint function related to curvature of the nodes on a polar curve. CONVERT writes two statements in UOC.FOR,

CALL SYSC10 (K, Z)

PHI(I) = Z

Subroutine SYSC10 will return the value of the constraint function through argument Z. K is the curve number.

$ S is the constraint function related to the maximum stress of a region.

$ D is the constraint function related to the displacement of a node.

$ B is the constraint function concerned with the buckling of a column.

The evaluation of the last three system functions requires the FEM calculation, and uses a new subroutine ANFEM(ZZ). The single argument ZZ is an array with 21 elements.

ZZ(I) = (critical load for a column) - (calculated load) I=1,2,...,10

ZZ(I) = (allowable displacement) - (calculated displacement) I=11,12,...,20

ZZ(21) = (calculated FS) - (required FS)

CONVERT first checks if there are $S, $D or $B functions and counts the number of each. Then subroutine ANFEM is called. If there is an $S function, CONVERT writes,

PHI(I) = ZZ(21)

If there are $B funotions, then say for the third one, CONVERT writes,

PHI(I) = ZZ(3)

A similar manipulation is done to displacement constraint functions.

The way the Fortran file UOC.FOR is created is analogous to the creation of a data file. However a Fortran file thus created can be compiled, linked and run exactly as a general Fortran file.

Figure 4.25 is a sample of UOC.FOR. Subroutine ADJUST is called to re-evaluate the coordinates of nodes and thickness of regions, when any dimensional variable has changed. The optimization function is the volume of the structure which is evaluated by calling subroutine SYSU10. PHI(1) to PHI(8) are the constraints about the upper and lower bounds of the four design variables. PHI(9) stems from the expression function

D1 + 10

PHI(10) stems from the call statement function

CALL A (D2, D3, D4, D5, VALUE)

where D5 is a constant, equal to 10.0. PHI(11) is about the curvature of the nodes on curve 1; PHI(12) about the maximum stress; PHI(13) about the buckling of a column; and PHI(14) about the displacement of a node.

4.8.3 Implementation of Subroutines ADJUST, SYSU10, SYSC10 and ANFEM

ADJUST checks whether the design variables have changed when subroutine UREAL or CONST is called. Inside

```
SUBROUTINE UREAL(X,U)
DIMENSION X(*)
CALL ADJUST(X)
CALL SYSU10(U)
RETURN
END


SUBROUTINE CONST(X,NCONS,PHI)
DIMENSION X(*),PHI(*),ZZ(21)
CALL ADJUST(X)
PHI(  1)=X( 1)-     30.00
PHI(  2)=      70.00-X( 1)
PHI(  3)=X( 2)-      0.00
PHI(  4)=      65.27-X( 2)
PHI(  5)=X( 3)-      0.00
PHI(  6)=      53.20-X( 3)
PHI(  7)=X( 4)-      0.00
PHI(  8)=     100.00-X( 4)
PHI(  9)=X(1)+10
CALL A (X(2),X(3),X(4),    10.000,Z)
PHI( 10)=Z
CALL SYSC10( 1,Z)
PHI( 11)=Z
CALL ANFEM(ZZ)
PHI( 12)=ZZ(21)
PHI( 13)=ZZ( 1)
PHI( 14)=ZZ(11)
RETURN
END
```

Figure 4.25 Sample of file UOC

ADJUST, there is an array XO storing the last values of the design variables for this comparison. If none of the design variables has changed, the logic flow will return to UREAL or CONST immediately. If any of the design variable has changed, the coordinates of relevant nodes and the thickness of relevant regions will be modified. Finally, XO is updated.

Subroutines SYSU10, SYSC10 and ANFEM are implemented following the algorithms discussed in section (4.2.5).

All of the above subroutines have very few arguments. The majority of the data are communicated through common block from the main program OPTZ.

4.8.4 Command File SYS.COM

This command file executes the optimization.

```
$ RUN CONVERT
$ FOR UOC
$ LINK OPTZ, AUXI, UOC, EXPERT,-
  SYS$LIBRARY:OPTIVAR/LIB
$ RUN OPTZ
```

CONVERT creates UOC.FOR, which then is compiled and linked to the main program OPTZ. OPTZ is then immediately run.

4.9 GENERATION OF FEM MESH AND DATA FILES

4.9.1 Mesh Generation

4.9.1.1 Distinguishing Constant and Non-constant Substructures

The procedure here is similar to most of today's automatic mesh generators, in which a complex configuration

is divided into several simpler regions interactively at the CRT terminal. It is not a big burden on the designer. The expert system checks each region in turn to see if it is constant. In a constant region, the coordinates of all of its nodes do not change during the optimization search, and the values of stresses and displacements are not needed. The displacements of the FEM nodes in the constant substructure cannot be evaluated during the optimization search. All of the constant regions constitute the constant substructure, and all of the non-constant regions constitute the non-constant substructure.

A structure must be divided into constant or non-constant substructures in order to adopt the substructure method (section (3.4.1)), where the FEM nodes on the mesh must be numbered independently in two substructures.

There are three factors affecting whether a region is constant or not: system functions, dimensional variables and dummy nodes.

1. System function

(1) Buckling

For a buckling system function, the stress of the center of the column must be evaluated. The expert system finds the region containing the center, and sets this region as non-constant.

(2) Displacement

If the displacement of a node is required, this node is classified as non-constant. A region containing a non-constant node must be non-constant.

(3) Maximum stress.

If the maximum stress in a region is required, this region must be non-constant.

2. Variable

If a node is controlled by a variable about its x, y coordinate, or radius, or angle, this node is non-constant. If the thickness of a region is controlled by a variable, this region is non-constant.

3. Dummy node

Dummy nodes are used in order to make more convenient the definition of a region. Their coordinates refer to other nodes. If one of these nodes is non-constant, the dummy node is non-constant too.

What remains after excluding all of the non-constant regions due to any of the above cases are constant regions.

4.9.1.2 Generation of the Mesh

1. Array NOP and CORD

The mesh generator creates two arrays for both constant and non-constant substructures.

NOP(NE,8) --- connectivity array, where NE is the

number of FEM elements in a substructure.

CORD(NP,2) -- coordinate array, where NP is the number of FEM nodes in a substructure.

Figure 4.26 is a region with four FEM elements, each has eight nodes, so

```
NOP(1,1) = 5
NOP(1,2) = 4
NOP(1,3) = 3
NOP(1,4) = 7
NOP(1,5) = 11
NOP(1,6) = 12
NOP(1,7) = 13
NOP(1,8) = 8
.........
CORD(1,1) = X1 = 0.0,    CORD(1,2) = Y1 = 0.0
CORD(2,1) = X2 = 0.0,    CORD(2,2) = Y2 = 1.0
.........
```

There are two kinds of nodes; one is the node for defining the regions (just node for short), another is the finite element node (FEM node for short). Generally, the region nodes are also FEM nodes. However, FEM nodes are not necessarily region nodes.

2. Determination of the Interval Number

From the theoretical point of view, the density of the mesh should be determined by the distribution of stress.

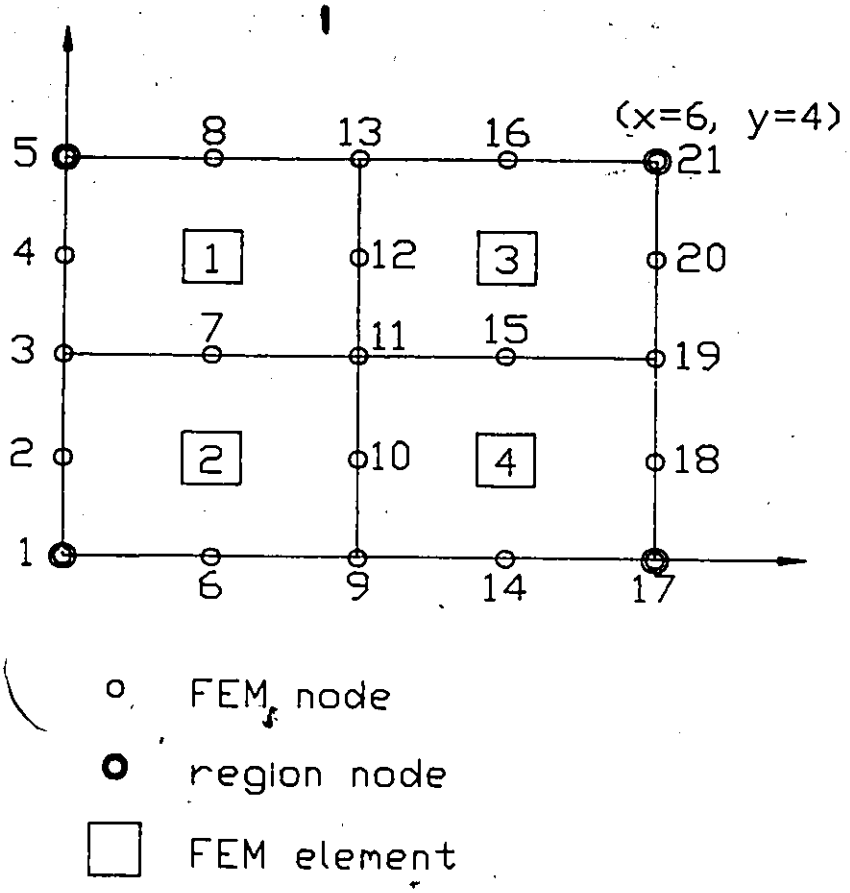


Figure 4.26 A region

However at the very begining of an FEM analysis, little is known about the stress distribution. So the density of a preliminary mesh can be determined only by using intuitive sense or experience.

The designer's procedure is to move the crosshair to an edge (initial edge) of any region (initial region), which is usually the one that is expected to contain the maximum stress, and then type in a number which denotes the number of intervals (initial interval number), i.e. the number of FEM elements in the initial edge. This is the only information the designer need enter. In Figure 4.27, if the designer moves the crosshair to edge 1-2, and types in 3, the mesh generated would be as shown.

The determination of the interval number in other edges is based on the principle that the quadrilateral elements should be made as square as possible to give the most accurate result. The interval number of the edges of the initial region is evaluated first. Following this the neighboring regions are evaluated, and then the neighbors of the neighbor. By this procedure, as a new region is dealt with, the interval number of one of its edges is already known.

In Figure 4.28, l_i is the length of the ith edge of the region, and n_i is its interval number. If the interval number of any edge is unknown, say n_4, then the interval number of its opposite partner is first checked, i.e. n_2, to

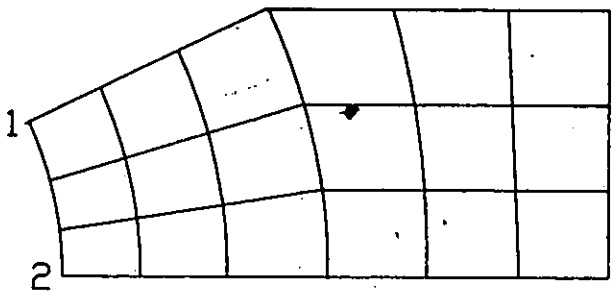


Figure 4.27 Determination of the density of the preliminary mesh

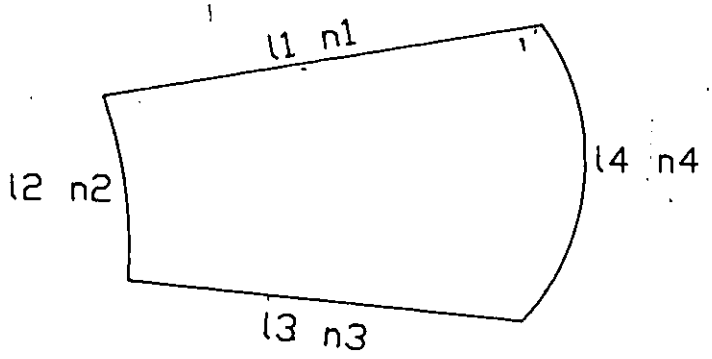


Figure 4.28 Determination of the interval number of an FEM element

determine if it is known. If n_2 is known, then n_4 is set equal to n_3. If n_2 is unknown too, then one of n_1 and n_3 must be known. For illustration, it is assumed that n_3 is known. The following relationship is used,

$$n_4 = n_2 = n_3 * (l_2 + l_4)/(l_1 + l_3) \qquad (4.28)$$

Therefore if the region is a rectangle, each FEM element is theoretically a square. However, since n_i must be rounded to an integer, the elements are usually not exactly square. When the interval numbers of all four edges of a region are determined, the number of FEM elements contained by this region is determined too.

In our present program, the interval numbers of a pair of opposite edges of a region are required to be equal. This requirement may be violated sometimes by the above procedure. Let us look at Figure 4.29. Suppose the designer specifies the interval number for one of the edges of region 1. The interval numbers of other edges are induced in one region after another through the two routes shown as the dashed line. Finally, n_2 may not be equal to n_4 in region 7. If such a conflict occurs, e.g. $n_2 = 6$, $n_4 = 8$, then n_2 and n_4 are reset to their mean.

$$n_2 = n_4 = 0.5 * (6 + 8) = 7 \qquad (4.29)$$

However, n_4 of region 7 and n_2 of region 6 are the same thing, so $n_2 = n_4 = 7$ in region 6. Such "infection" should be continued until n_4 is set to be seven in region 5. Since edge 4 of region 5 is an external edge, the "infection" is

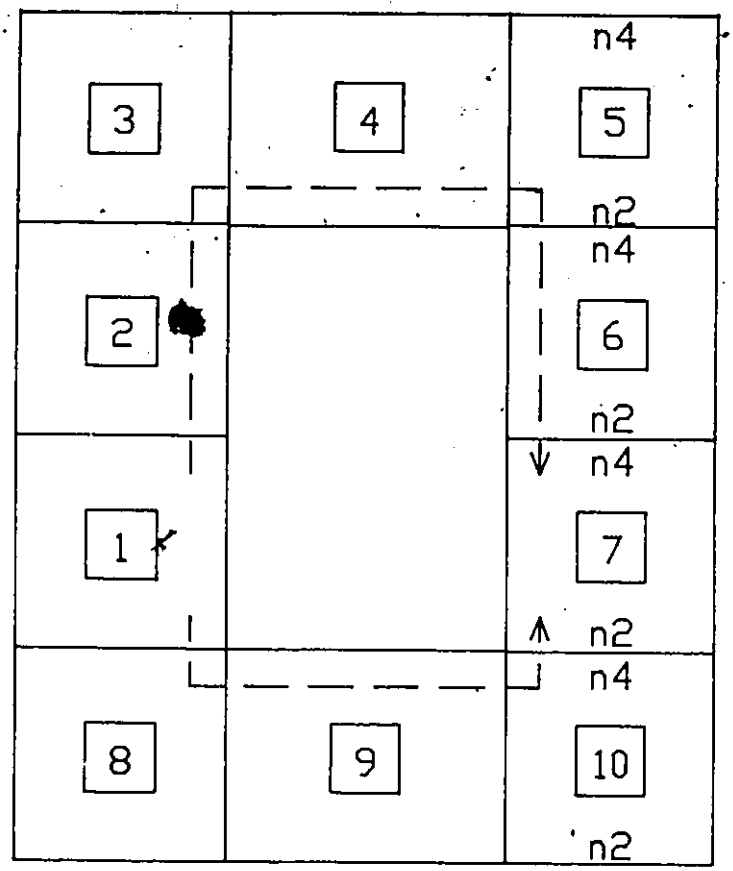


Figure 4.29 Conflict in interval numbers

terminated. Similarly, $n_4 = n_2 = 7$ in region 10.

3. Determination of interval ratio

Figure 4.30 shows a quarter of an annulus. In (a), n_2 has equal interval ratios, and in (b), n_2 has unequal interval ratios, so that the mesh is finer closer to the inner hole. The second case is preferable, since usually in a narrow area, the stress gradient is higher, and the mesh should be finer.

Our algorithm to determine the interval ratios is as follows; also see Figure 4.31. Suppose

$$n_1 = n_3 = 3 \tag{4.30}$$

(1) Locate a_1, a_2, a_3, a_4 and b_1, b_2, b_3, b_4, let

$$a_1a_2 = a_2a_3 = a_3a_4$$

$$b_1b_2 = b_2b_3 = b_3b_4 \tag{4.31}$$

where a_ia_j means the length along the edge, not the straight line distance.

(2) Calculate

$$s_1 = \overline{a_1b_1} + \overline{a_2b_2}$$

$$s_2 = \overline{a_2b_2} + \overline{a_3b_3}$$

$$s_3 = \overline{a_3b_3} + \overline{a_4b_4}$$

$$S = s_1 + s_2 + s_3 \tag{4.32}$$

(3) Calculate interval ratios, and relocate a_i and b_i

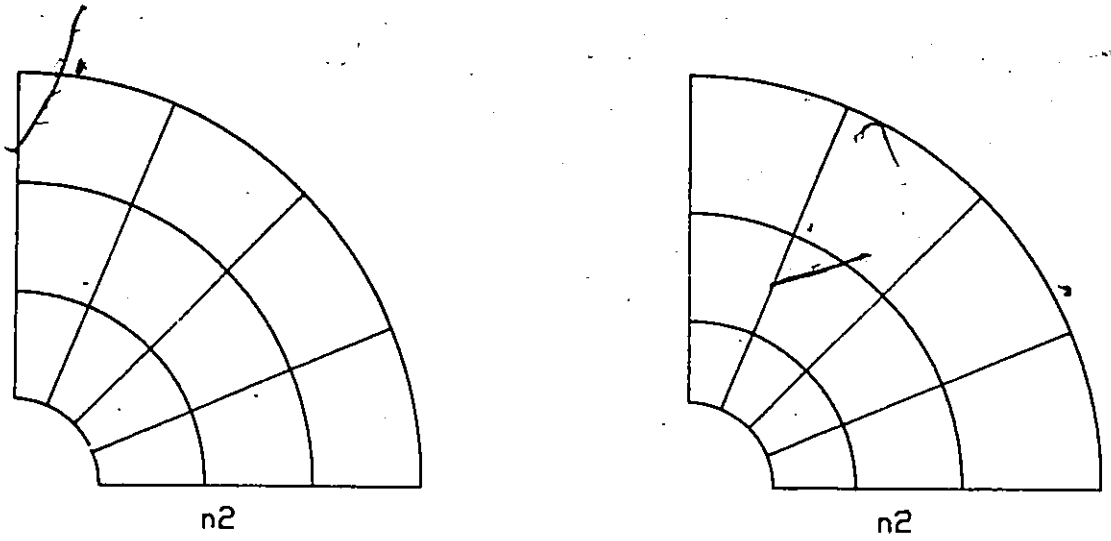


Figure 4.30 Equal and unequal interval ratios

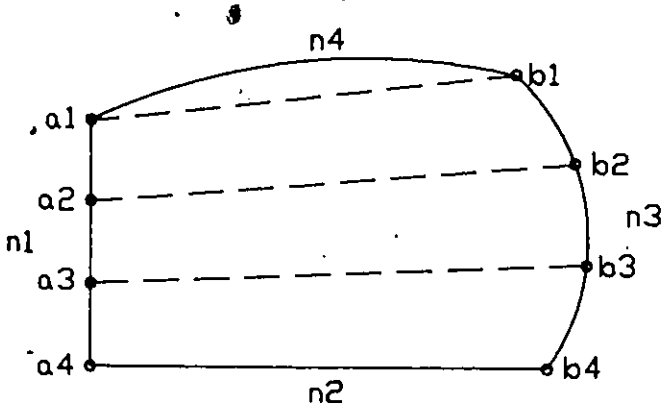


Figure 4.31 Calculation of interval ratio

$$\frac{a_1a_2}{a_1a_4} = \frac{b_1b_2}{b_1b_4} = \frac{s_1}{s}$$

$$\frac{a_2a_3}{a_1a_4} = \frac{b_2b_3}{b_1b_4} = \frac{s_2}{s}$$

$$\frac{a_3a_4}{a_1a_4} = \frac{b_3b_4}{b_1b_4} = \frac{s_3}{s} \qquad (4.33)$$

Usually, an edge is contained by two adjacent regions. Since the interval ratios calculated from different regions are not equal, the average value should be taken. The interval ratios thus determined may not be the best from the theoretical point of view. However they will be adjusted later when stress distribution is returned from FEM calculation.

4. FEM node connectivity

The FEM nodes are first numbered randomly. Later subroutine MBN is called to optimize the numbering so that the bandwidth of the stiffness matrix of the structure tends to be a minimum.

Subroutine MBN was implemented in the author's master project (Wu, 1984), based on the paper of Akhras (1976). Since the interval numbers of edges of the regions are known, the regions can be divided into a certain number of FEM elements, and then the FEM elements and nodes are numbered from top to bottom and from left to right.

However, a configuration can be decomposed as in Figure 4.32. Within the constant substructure, m-n is a common edge between regions 2 and 3, and node k is a common node of regions 1 and 2. The expert system will check out such cases in order to avoid redundantly numbering the common nodes and the nodes on the common edges. It is then very easy to assign values to the array NOP which will be entered into subroutine MBN. Then a new NOP is returned, with the bandwidth minimized.

5. FEM nodal coordinates

The last problem is to evaluate the coordinates of the FEM nodes. A discrete transfinite mapping formulation from Haber (1981) has been adopted. The calculations are complicated, but the basic parameters used remain constant for a certain region. So when dealing with a region, all of its parameters are first calculated and stored. Later, a simple formula is called, which can use these constant parameters.

Arrays NOP and CORD (introduced at the begining of section (4.9.1.2)) can be written to a substructure data file which will be used by the finite element module.

Finally, the FEM mesh generated is displayed on the screen. If the designer is unhappy with it, he can go back to redetermine the initial interval number on an edge of a region, and the whole program runs again.

Figure 4.33 is a sample of the FEM mesh generated by

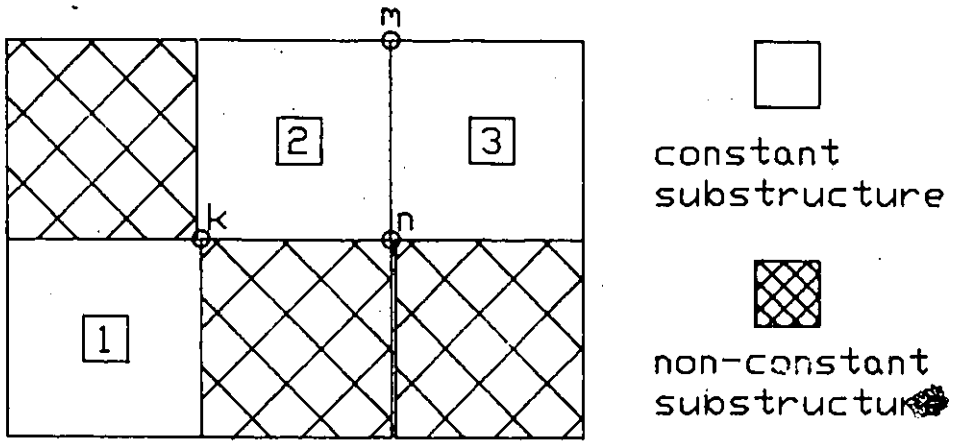


Figure 4.32 Decomposition of a structure

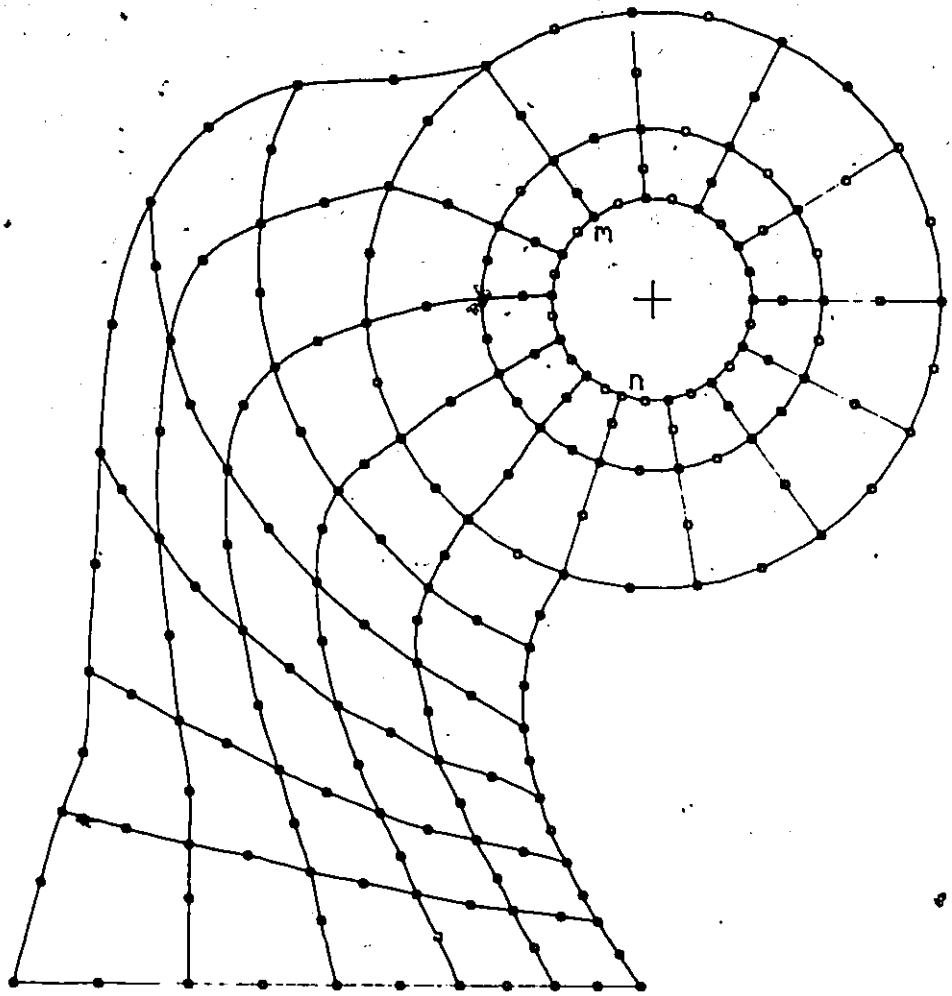


Figure 4.33 Sample FEM mesh

the above procedure. The designer has specified the interval number on edge m-n as five.

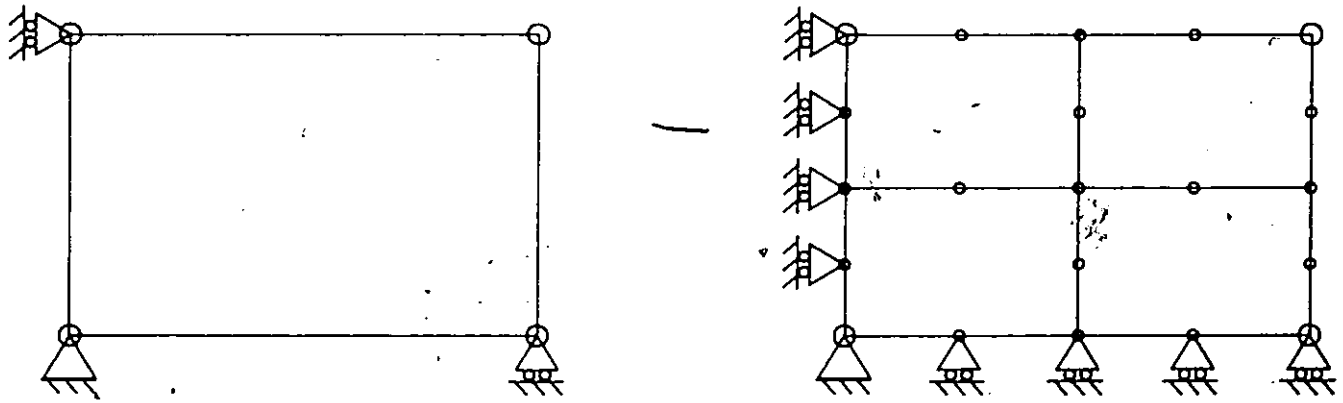
4.9.2 FEM Data File

During the aforementioned procedures in this chapter, a database is available. It contains all of the expertise of the human expert originally, and is updated and supplemented little by little by the specifications from the designer. Now the data must be reorganized into several data files required by the existing finite element package. This presents little difficulty. Only the boundary conditions require some discussion.

The boundary conditions of the structure on region level are defined by the user, as in Figure 4.34(a), where one region node is fixed completely and two are fixed partly. Then the region is divided into several finite elements as in Figure 4.34(b). There are a lot of FEM nodes between the two region nodes on an edge. The following heuristic rules are used to determine the boundary conditions of these FEM nodes.

1. If one region node is completely fixed, the middle FEM nodes have the same boundary conditions as the other region node.

2. If the two region nodes have the same boundary conditions, the middle FEM nodes take this same boundary condition.



○ region node

∘ FEM node

Figure 4.34 Boundary conditions

3. In all other cases, the middle FEM nodes are free.

4.10 PROGRAM STYLE OF THE EXPERT SYSTEM

At this point, we shall sum up the general style of the expert system, which is considered extremely important in an expert system approach. In this section, the term "expert system" implies both the super-expert system or lower level expert system, and we shall call it "system" in short. The term "user" refers to the human expert (in the top level) or the designer (in the lower level).

Generally, our system has several distinct features in program style: sufficient information for the user, convenient input by the user, reliable proof for the user, free judgement from the user. They are summed up in more detail as follows.

1. Graphic Interface. The system makes full use of the CRT graphics terminal. The interfaces between the user and system are visible, intuitive and all on-line. The user need not prepare any files before starting the system.

2. Simple Input. The user enters the specifications in several sections as described in this chapter. Related information items are entered together. The input does not always mean typing a series of numbers. It could be selecting a menu item, pointing to a graphic primitive, adjusting the bar chart, or filling out a table. To specify

the importance values, the user can enter either the importance values or the direct values. For each specification, there is always a default value. At the top level, it is knowledge from the super-expert system. At the lower level, it represents the expertise of the human expert. These default values can save the user's effort in input, since they are usually very adaptable or require only minor modifications for most of the applications. The system can also provide some other aids, such as induction of dimensions and variables, or supplementing missing material properties.

3. Static information. Before the user creates specifications, the system provides adequate introductory information, such as how to use the interactive module to create a configuration, what the importance values mean and how to specify them, what advantages and disadvantages an optimization or FEM algorithm has. This information guides the user in entering specifications or making decisions. Furthermore they can be called again later at any time if the user forgets some detail. The information is all written in a data file BOOK. A piece of information can be reached by identifying its chapter and page number.

4. Dynamic information. After the user has created a specification, the system provides fool-proof checks. This does not only mean simply checking an input number (e.g. 1.25 < factor of safety < 10), but also involves more

complex checking. For instance, it checks if two examples in the training sample are in conflict; or if the interactive rules are recursive; or if an importance value is compatible with others; or if there is enough computer memory for an FEM algorithm. Furthermore the system can evaluate the goodness of a specification. For example, it can compare a user entered material with its own desired material or any others. When the system finds something wrong or unreasonable, a warning message is displayed. No action is enforced; it just reminds the user that the situation should be reviewed. All of the final decisions are made by the user.

5. Intelligent information. If the user wants to know why a candidate, or an algorithm, or a parameter is recommended by the system, the system can trace back to the derivation procedure and answer "why". For example, the system can show the bar charts of the importance values and desirability values in order to explain intuitively why a material or configuration is desired. The system also can show the procedure for initializing the reference factor of safety, and then how it is modified by different considerations.

CHAPTER 5

RUNNING THE APPLICATION

In the last chapter, the model for the FEM based optimization design was established. Some specific features in running this program will now be discussed: finding a feasible starting point; scaling the design variables and functions in order to make the program robust; adjusting and regenerating the FEM mesh to guarantee a high quality FEM analysis; and tracing and adjusting the optimization search in order to let the designer have more control of the optimization procedure.

5.1 FEASIBLE STARTING POINT

Usually, a feasible starting point is more likely to lead to a successful optimization run. Furthermore, some optimization algorithms require a feasible starting point.

The expert system adopts an algorithm suggested by Fox (1971) to find a feasible starting point.

$$U = -\langle g_k(x) \rangle = \text{minimum}$$

$$\text{where} \quad \langle g_k \rangle = \begin{cases} 0, & \text{if } g_k \geq 0 \\ -g_k, & \text{if } g_k < 0 \end{cases} \qquad (5.1)$$

$g_i(x) > 0$, where i includes all of the satisfied constraints.

$g_j(x) - g_j(x_o) > 0$, where j includes all of the unsatisfied constraints except the kth.

All of the unsatisfied constraints are driven to feasibility one by one.

The flowchart for finding a feasible starting point is shown in Figure 5.1. The algorithm has been implemented in the existing optimization subroutine UREAL, and the constraint subroutine CONST, which can be called in two different modes: the optimization mode and the starting-point mode.

In subroutine UREAL, the optimization mode returns the value of the real optimization function of the application. The starting-point mode calls constraint subroutine CONST, and returns $-\langle g_k \rangle$ as the value of the optimization function, where g_k is the unsatisfied constraint being driven at the moment.

In subroutine CONST, when the starting-point mode is evoked, the constraint functions are as follows.

```
PHI(K) = 0
PHI(I) = PHI(I)
PHI(J) = PHI(J) - PHIO(J)
```

where the array PHIO stores the values of the constraint functions in the last iteration.

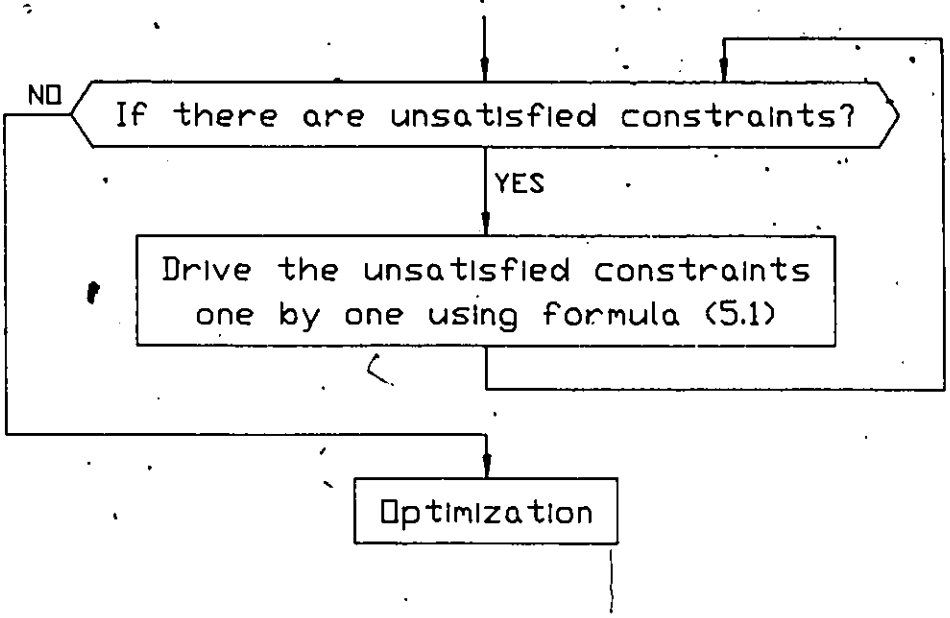


Figure 5.1 Flowchart of finding a feasible starting point

5.2 SCALING VARIABLES AND FUNCTIONS

Scaling the design variables and functions has proven to be very important in optimization practice. The following formula is used to scale all of the design variables to the range from ten to twenty.

$$XS_i = (XO_i - a_i)/ b_i$$

$$a_i = 2 * BL_i - BU_i$$

$$b_i = 0.1 * (BU_i - BL_i) \qquad (5.2)$$

where XO_i --- the original design variable, $BL_i \leq XO_i \leq BU_i$

XS_i --- the scaled design variable, $10 \leq XS_i \leq 20$

BL_i --- lower bound

BU_i --- upper bound

Now inside the optimization search algorithm, all of the design variables are always positive and have the same order of magnitude even when the design point goes beyond the bounds accidentally. The working environment for the algorithm has been improved significantly and a lot of mathematical indeterminate forms and other pitfalls are avoided.

When the optimum solution is obtained, the design variables can be transformed back to their real values as,

$$XO_i = b_i * XS_i + a_i \qquad (5.3)$$

All of the functions (optimization function and constraint functions) are evaluated before the optimization

search in order to obtain the scale factor,

$$S_i = 1./|f_i| \qquad (5.4)$$

where S_i --- scale factor for ith function

f_i --- the value of the ith function at the starting point

Later, each function evaluated in UREAL or CONST is multiplied by its scale factor before its value is returned to the top level program. Therefore the absolute values of all of the functions are initially of the same order of magnitude, and approximately one.

5.3 ADJUSTMENT AND REGENERATION OF THE FEM MESH

5.3.1 Adjusting FEM Mesh

During the optimization search, the values of the design variables vary in small steps, and the coordinates of some FEM nodes and the thickness of some elements are changed correspondingly. The re-evaluation of these coordinates and thickness in each optimization iteration is called mesh adjustment, where the numbers and connectivities of the FEM elements and nodes do not change. The following steps are included in the adjustment procedure.

1. Has any design variable changed during an optimization iteration?

   YES: go to 2

   NO: go to 5

2. Re-evaluate the coordinates of the region nodes and the thickness of the regions which are controlled by the changed design variables.
3. Find out the regions where the coordinates of some of its nodes or its thickness has changed.
4. Loop the regions found in step 3.
   Re-evaluate the thickness of the FEM elements or the coordinates of FEM nodes (section 4.9.1.2).
5. Go to next procedure.

An example is shown in Figure 5.2. Figure 5.2(a) is a region before adjustment. Nodes 1 and 2 are region nodes. The x coordinate of node 1 is the variable X1. By variable induction (section 4.6.2), the expert system knows that X1 controls the x coordinate of node 2 too. Now if X1 changes from 4.0 to 2.0 in an optimization iteration, the x coordinates of node 1 and node 2 are changed accordingly. The coordinates of other FEM nodes in this regions are re-evaluated. The region is uniformly adapted to Figure 5.2(b).

Mesh adjustment is evoked in each optimization iteration.

5.3.2 Regenerating the FEM Mesh

In contrast to mesh adjustment, in mesh regeneration, the number and connectivities of the FEM nodes are redetermined, and the mesh tends to be finer. Since

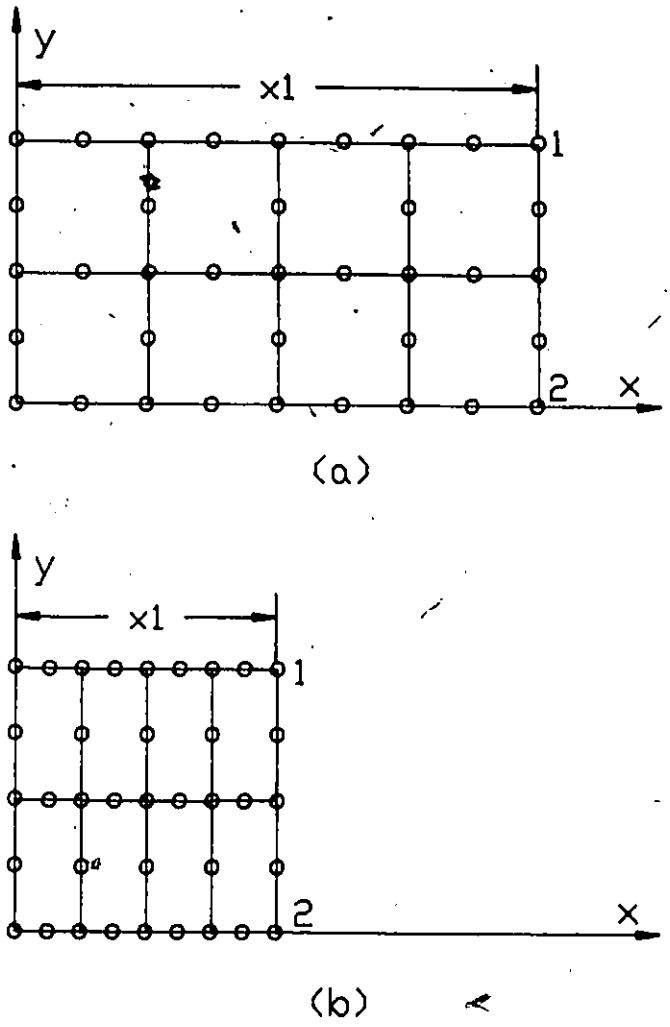


Figure 5.2 Adjustment of FEM mesh

each regeneration takes significant computer time, it is evoked for each ten optimization iterations.

The following two criteria are checked first. If neither of them is satisfied, the actual regeneration procedure is bypassed.

1. Aspect ratio

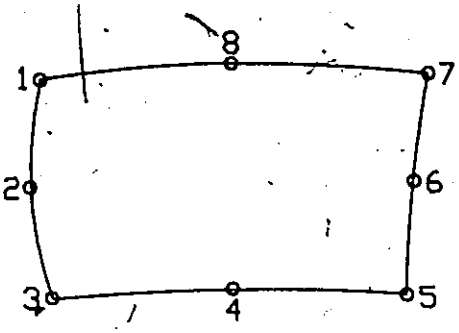
Aspect ratio is the ratio of the lengths of the FEM element in two perpendicular directions. It reflects the geometrical requirement for the FEM elements. When the aspect ratio is too large (Figure 5.3(b)), the accuracy of calculation is low. Even worse, if the severe distortion shown in Figure 5.3(c) occurs, then the program could crash. Here, the aspect ratio r is taken as,

$$r = \max \left(\frac{l_{26}}{l_{48}} , \frac{l_{48}}{l_{26}} \right) \qquad (5.5)$$

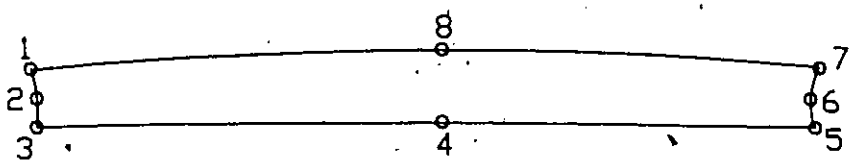
where l_{26} is the distance between FEM node 2 and 6 in Figure 5.3(a). l_{48} has a similar meaning. If $r > R$, the mesh regeneration must be evoked, where R is a predetermined parameter, say 10. There are additional geometric requirements for a high quality FEM element; for example, the four angles should be close to right angles. Since these requirements are not as critical as the aspect ratio, they are neglected in order to reduce calculation.

2. Stress difference

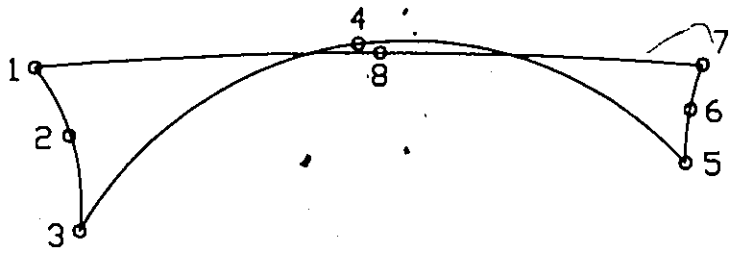
An FEM node is usually a common node of several FEM



(a) Reasonable aspect ratio



(b) Large aspect ratio



(c) Distorted element

Figure 5.3 Aspect ratio

elements. So its stress can be evaluated from different connected FEM elements. As described in section (4.9.1.2), the initial FEM mesh is set up by the designer according to his judgement and experience. If the mesh is not adequately fine, the stresses of an FEM node evaluated from different connected elements have a significant difference. The stress difference d of an FEM node is defined as,

$$d = \frac{\max(s_1, s_2, \ldots, s_n) - \min(s_1, s_2, \ldots, s_n)}{|\min(s_1, s_2, \ldots, s_n)|} \qquad (5.6)$$

where s_1, s_2, ..., s_n are the stresses of the FEM node evaluated from different elements. If d>D, the mesh regeneration must be evoked, where D is a predetermined parameter, say 0.05. The stress difference criterion is related to stress, and reflects the overall requirement for the fineness of the FEM mesh of a structure. The aspect ratio criterion tends to deal with the local distortion of the mesh.

When a criterion for mesh regeneration is met, the expert system will evaluate three numbers IREG, IEDGE and NINT.

1. IREG

The expert system determines the IREG region as the one containing the FEM element which has the largest aspect ratio or stress difference. If both of the criteria are met, the aspect ratio has higher priority, because the severe distortion of the mesh would make the optimization

run crash prematurely.

2. IEDGE

Figure 5.4 shows the IREG region. The following parameters are used.

$$a_{13} = (l_1 + l_3)/ n_1$$

$$a_{24} = (l_2 + l_4)/ n_2 \qquad (5.7)$$

where n_1 and n_2 are the number of FEM elements on edges 1 and 2 of IREG. The parameters a_{13} and a_{24} are average FEM element lengths in edge 1 or 3 and edge 2 or 4 respectively.

If $a_{13} \geq a_{24}$, IEDGE = 1

If $a_{13} < a_{24}$, IEDGE = 2

IEDGE is the edge on which the length of the FEM elements are relatively longer.

3. NINT

NINT is the number of FEM elements on edge IEDGE of region IREG after the mesh regeneration.

$$\text{NINT} = \begin{cases} n_1 + 1 & \text{when IEDGE} = 1 \\ n_2 + 1 & \text{when IEDGE} = 2 \end{cases}$$

Now IREG, IEDGE and NINT can be taken as the initial region, initial edge and initial interval number as defined in section (4.9.1). The FEM mesh of the structure can be regenerated following the algorithm in that section.

The expert system provides another facility for the designer. The FEM mesh of the structure is displayed on the

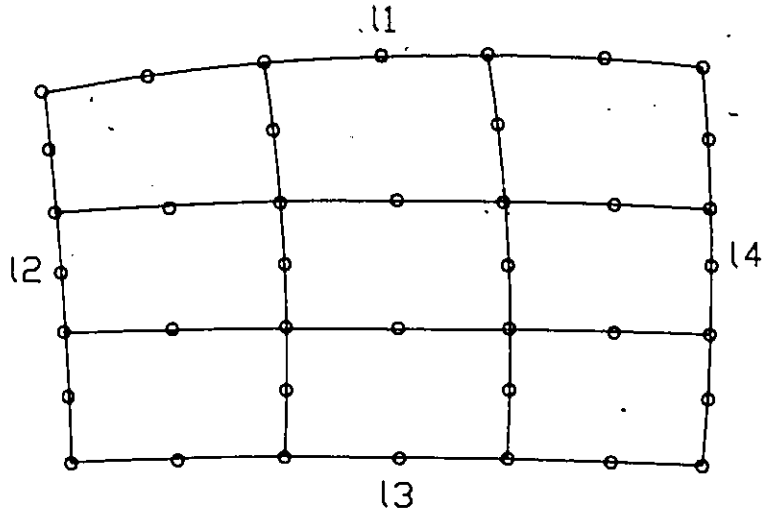
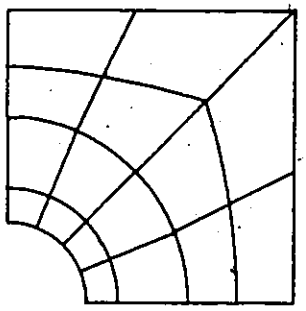


Figure 5.4 IREG region

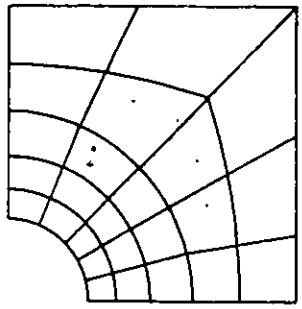
screen after each ten optimization iterations. If the designer feels that the mesh distorts too much based on his experience, he can select a menu item "RESET" from the screen, and the mesh regeneration mechanism will be evoked immediately. Since the designer can look at the mesh only from the viewpoint of geometry, the aspect ratio criterion is used in this case to determine IREG. The predetermined parameter R is one, i.e. the regeneration of the mesh is unconditional.

Figure 5.5 shows an example of mesh regeneration, in which Figure 5.5(a) is the initial mesh. In the first iteration, the stress difference criterion has been met, and the mesh is regenerated into Figure 5.5(b). In the 11th iteration, the stress difference criterion has been met again, and the mesh changes to the form shown in Figure 5.5(c). It remains unchanged until the optimum solution is achieved. The actual mesh adjustment is in effect at each iteration.

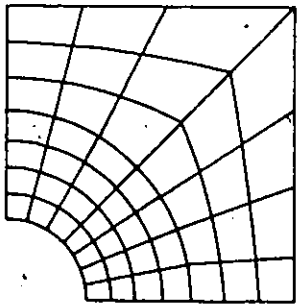
In this example, the aspect ratio criterion was not met, because for an 8-node isoparametric element, the allowable aspect ratio can be as high as ten, and this is seldom exceeded. It should also be noted that when a mesh regeneration is evoked by the stress difference criterion, the stress difference situation is improved, and usually the situation for the aspect ratio improves at the same time. Figure 5.6 is a pseudo example of mesh regeneration due to



(a) Initial mesh

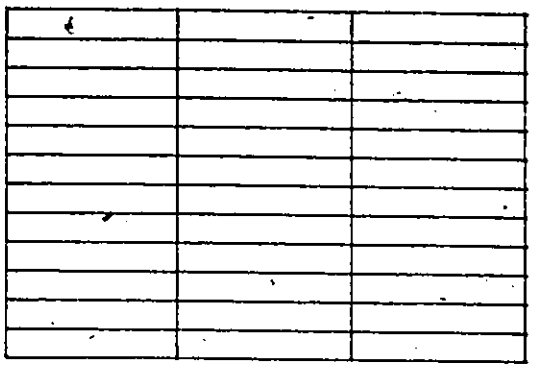


(b) First mesh regeneration

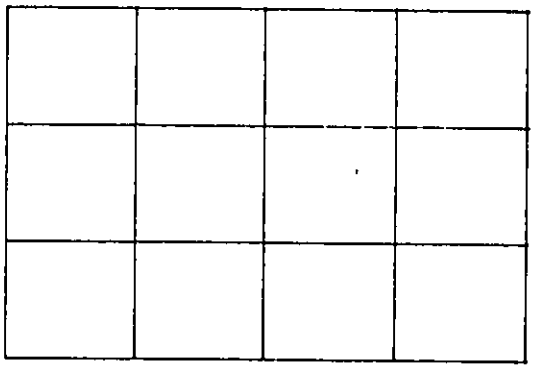


(c) Second mesh regeneration

Figure 5.5 Mesh regeneration due to large stress differences



(a) Before regeneration



(b) After regeneration

Figure 5.6 Mesh regeneration due to a large aspect ratio

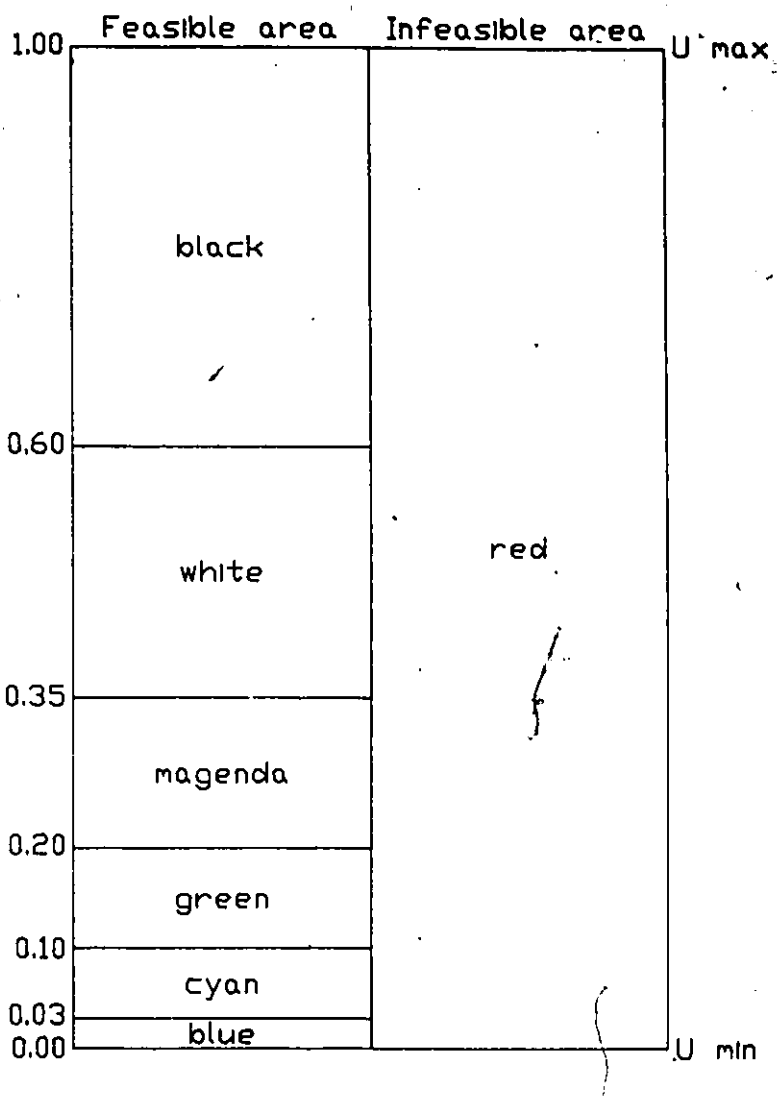
aspect ratio, following the algorithm discussed above.

5.4 TRACING AND ADJUSTING THE OPTIMIZATION SEARCH

5.4.1 Optimization Topography and Search Route

In conventional optimization, when a run is started, the designer has no control and knows nothing about what is happening during the search until a solution, convergent or not, is given. If it is divergent, the designer cannot normally determine why. The expert system improves this situation significantly by displaying the topography of the optimization function and the search route on a two-dimensional plane during the optimization run.

For this purpose, the designer is required to determine two design variables as the important variables. All others are non-important variables. The range of the topography is determined by the bounds of the important variables. Now by fixing the values of all non-important variables at the current design point, and varying the values of the two important design variables within their bounds, the values of the optimization functions and all of the non-FEM constraint functions are evaluated in a set of uniformly distributed points in the 2-dimensional plane. Regions are established with different colours according to their value levels and feasibility, as in Figure 5.7. The computer time spent in calculating the optimization function and non-FEM constraint functions is trivial.



Note: (1) Umin and Umax are the minimum and maximum values respectively of the optimization function evaluated at the uniformly distributed points.

(2) The feasible area and infeasible area are evaluated without considering the FEM constraint functions.

Figure 5.7 Colours of the optimization topography

If the global direction method discussed in section (3.5) has been used, the global unconstrained optimum is available already. It is designated by a red cross "+" on the topography.

The route of the fully constrained optimization search is then drawn on the topography. It transfers very intuitive and visible information to the designer about what is going on. Whenever an infeasible design point is encountered, a small red circle appears. These, together with the optimization search points, gradually identify the constraint lines on the screen.

Another feature for interactive optimization is the use of test points. Any time during the optimization search, the designer can suspend the current search, move the crosshair to any point on the topography, and type key F. All of the constraint functions are then evaluated at this point. If it is feasible, a small white circle appears; otherwise a small red circle appears instead.

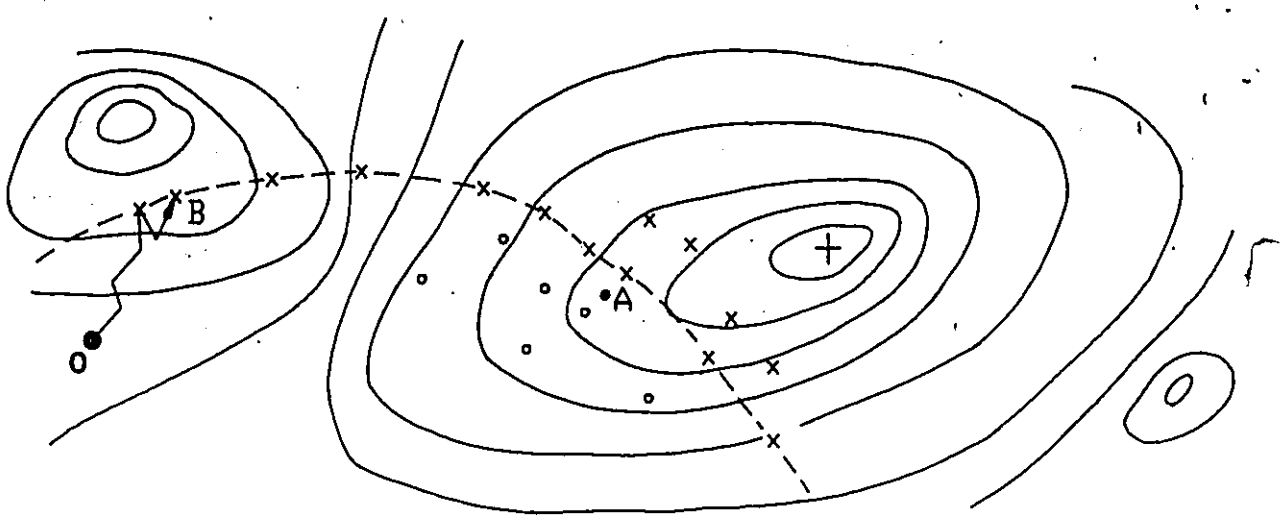
After one or a series of points have been tested, two options are available. First, if the R key is typed, the original optimization search is resumed at the point where it was suspended. Second, the crosshair can be moved to any point on the topography and the S key typed; this point is now a new starting point. The original optimization search is aborted, and a completely new search begins at the new starting point. In conventional

optimization, the designer cannot reselect a starting point until the previous search is terminated. In our system, the designer is able to do that at any time when it seems more likely to lead to a truly global optimum.

In Figure 5.8, the optimization search has been started at point O. When the search point has arrived at point B, the designer has decided, after inspecting the topography, that the search may result in a local optimum. So the search is suspended, and by using a series of test points, an imaginary constraint line is defined. The designer might now be quite sure that point A is close to the constrained optimum. If it were taken as the optimum solution, it would be a very approximate one from the viewpoint of numerical calculation; however it should be acceptable for real design applications.

The main problem is that the topography is relevant to two important variables only. The A point so determined neglects the effect of other variables to some degree. However the point A can be set as the new starting point from which a new optimization search begins. It would lead to a more precise optimum solution in the n-dimensional sense.

When any of the non-important variables change a significant amount, the existing topography becomes out of date, and should be replaced by a new one evaluated at the current design point. Since the test points would be valid



+	unconstrained global optimum
◉	original starting point
x	infeasible test point
∘	feasible test point
---	imaginary constraint line

Figure 5.8 Optimization topography

only in the old topography, they are erased together with the old topography.

The designer may not want to watch the optimization route, or he may wish to leave the terminal during the optimization run. In such circumstances he can select the menu item "HOLD", and the optimization will run without displaying the search route on the topography. Some computer time will be saved. The designer can resume displaying the search route by selecting the menu item "RUN". "HOLD" and "RUN" can be switched at any time during the optimization run.

5.4.2 Showing the Structure and Mesh

The topography and search route can provide only information about the optimization search. The designer would commonly like to know as well the form of the structure at the current design point. The expert system displays the current configuration of the structure together with the FEM mesh after each ten optimization iterations. At this time, the topography and route are turned off temporarily. When the next iteration is finished, the structure and the mesh disappear automatically and the topography and route are redisplayed. Just as for the search route, the structure and mesh display can be omitted in order to save computer time. There is another way that the display of the structure and mesh is avoided. Each time

the expert system is scheduled to display the structure and mesh, it first checks whether the design variables have changed a lot from the design point where the structure and mesh were previously displayed. If they have not, the structure and mesh are not displayed.

5.4.3 Rerunning the Application

If the designer is not satisfied with the first optimum solution, he can rerun the application at different levels.

1. He can reselect a material, a configuration, a failure mode or a factor of safety.

2. He can respecify the application with regard dimensions, loads, design variables, functions, and so on.

3. He can redetermine the optimization method, the FEM algorithm, the penalty function, and some parameters.

The expert system provides a small amount of very general advice for this purpose, and it is the responsibility of the designer to make the major judgements. However the information from the topography, the search route and the test points in the previous run can help the designer to gain an insight into the following problems.

1. Is the starting point suitable? (If not, the original search can be aborted, and a new optimization search can be restarted from a new starting point, as described in section 5.4.1).

2. Are the optimization method, the FEM algorithm, and the penalty function adopted good for this application?

3. Are the step size and some other parameters appropriate?

4. Is any constraint too tight?

5.5 OUTPUT

When the optimization run is finished, all of the data are available for a series of hard copies for the design, all of which can be drawn by a plotter. A table is also displayed on the screen, which includes the following items.

1. Optimum solution
2. Configuration
3. Stress distribution
4. Displacement
5. FEM mesh
6. List of coordinates
7. List of displacements
8. List of stresses
9. List of thickness

The designer can move the crosshair to select the hard copies he requires.

All of the hard copies for an application (case study) will be shown in the next chapter.

CHAPTER 6

CASE STUDY: THE GAS CHAMBER

6.1 INTRODUCTION

The case study is to establish an expert system for the design of a structural family --- a gas combustion chamber with a piston and rod (gas chamber for short), and then use this expert system to design a gas chamber according to a particular set of requirements. The general configuration of this gas chamber is shown in Figure 6.1. Since it is an axisymmetric structure, only half of it is drawn in subsequent figures. Usually an external rib in the middle of the gas chamber is needed as a reinforcement.

Figure 6.2 shows all of the design variables usually considered in the gas chamber. Variables X_1 to X_6 define the radial distances from point 17 of some specified points on the nozzle. Other variables are the external diameter of the opening X_7, the external diameter of the chamber X_8, the diameter X_9 and the height X_{10} of the external rib. All other dimensions are usually treated as constants, e.g. the length and the internal diameter of the chamber. The angles of points 18 to 23 with respect of the positive x direction are set as constants too, therefore the coordinates of these points can be uniquely determined if their radial distances from point 17 are specified.

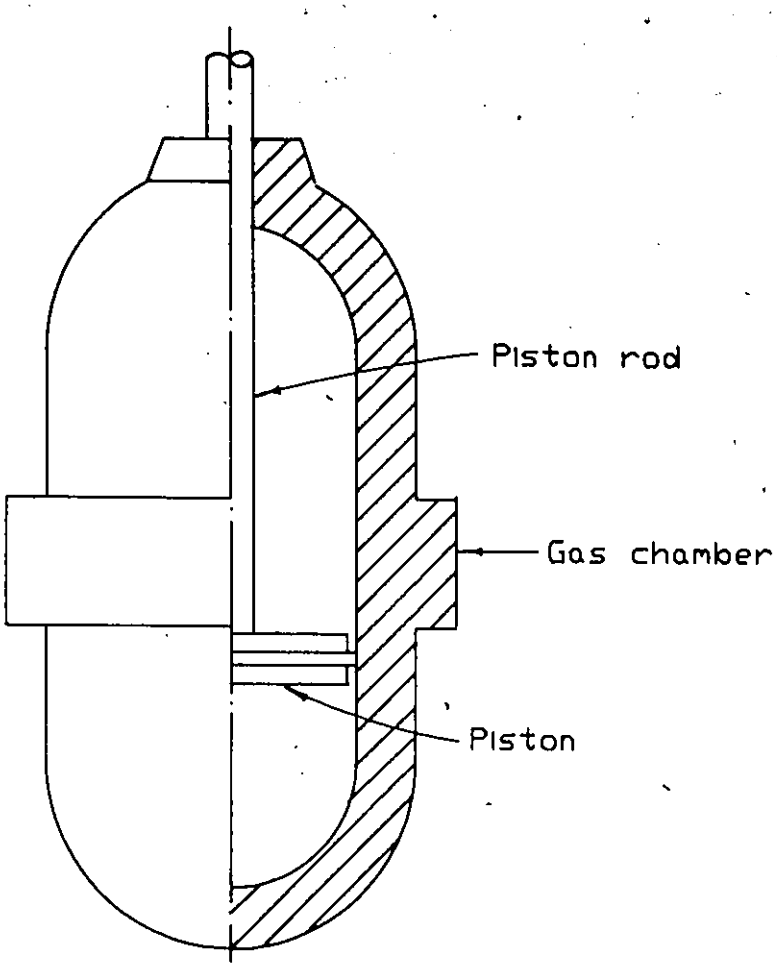


Figure 6.1 General structure of the gas chamber

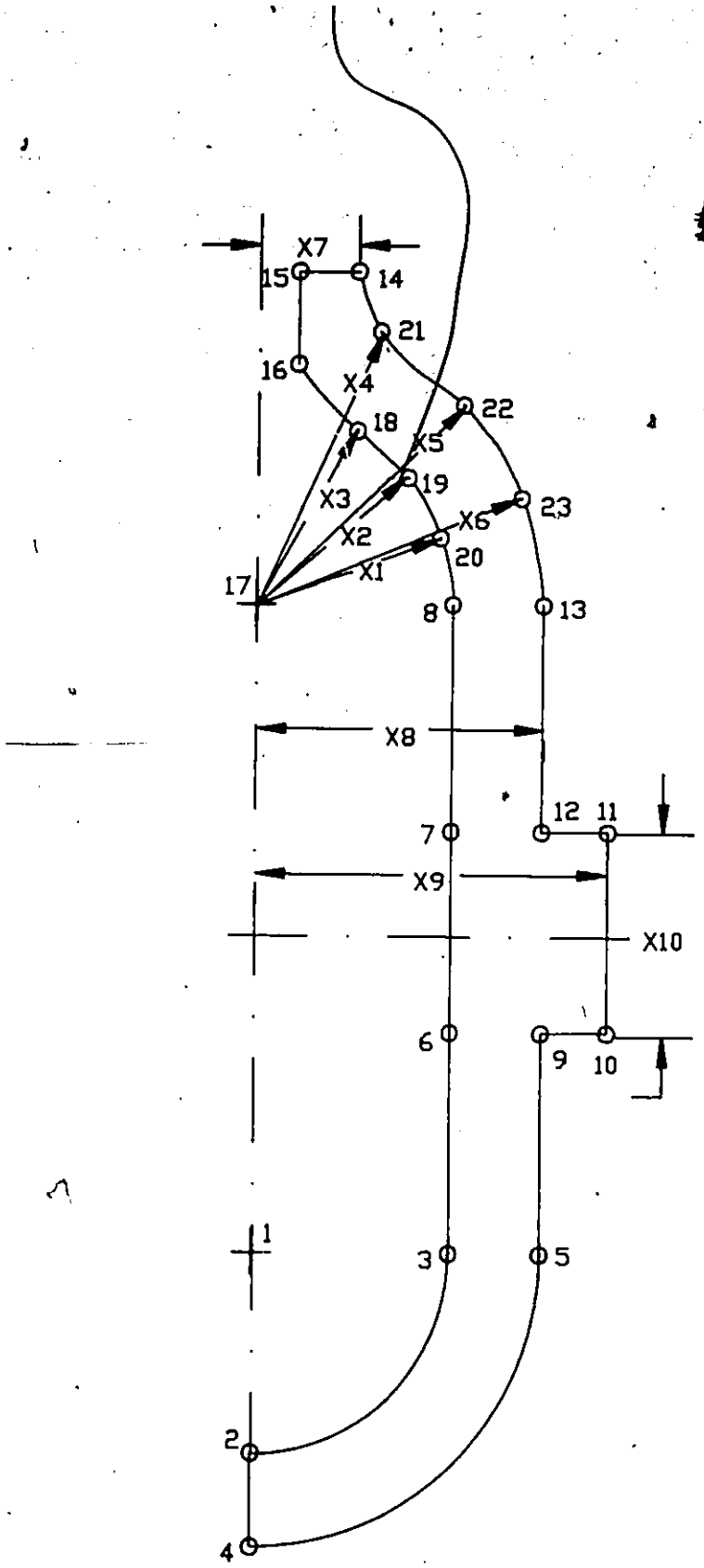


Figure 6.2 The variables of the gas chamber

The most important specification is the internal pressure. Also the working temperature and number of pressure cycles are usually specified.

In many cases, the objective function is to minimize the volume of the structure. Typical constraint functions concern the maximum stress, the curvature of the contours of the opening for the piston rod, and so on. The inside length of the structure is a constant and the width is limited by the upper bound of X_9.

The above description is a general introduction to the design of the gas chamber. Specific design variables and functions are determined by the human expert or the designer. Since the main objective of this project is to study the methodologies of the expert system and design automation, this demonstration case study may not reflect realistically every detailed aspect of an actual design.

6.2 ESTABLISHING THE EXPERT SYSTEM

First the human expert executes the super-expert system, in order to establish the expert system for designing the gas chamber family. The procedure described in section (2.3) is followed.

6.2.1 Fundamental Definition

This section includes defining the gas chamber family as an axisymmetric problem and specifying the number

of material candidates and configuration candidates. In this case, the number of material candidates is nine and the number of configuration candidates is three.

6.2.2 Setting Up Configuration Data Base

The configuration database is set up by interacting with the super-expert system at a CRT terminal. The database includes three configuration classes. Class 1 contains only one subclass; class 2 contains two subclasses; and class 3 contains three subclasses. All of them are shown in Figure 6.3 (a), (b), (c). A practical real life configuration database would be much larger than the one in this demonstration example.

The differences among the configurations occur at the top section. In class 1, both the inner and outer contours are arcs. In class 2, the inner contour is an arc, the outer contour is a curve. In class 3, both the inner and outer contour are curves. The shape of the contour reflects the design requirement. Generally speaking, the curve contour may be expected to save more material and function better, but be more difficult to design and manufacture. The minor differences among subclass configurations in a class can be seen in Figure 6.3. Once the first subclass configuration has been established, it is very easy to set up other subclass configurations in the same class by making the necessary revision directly on the

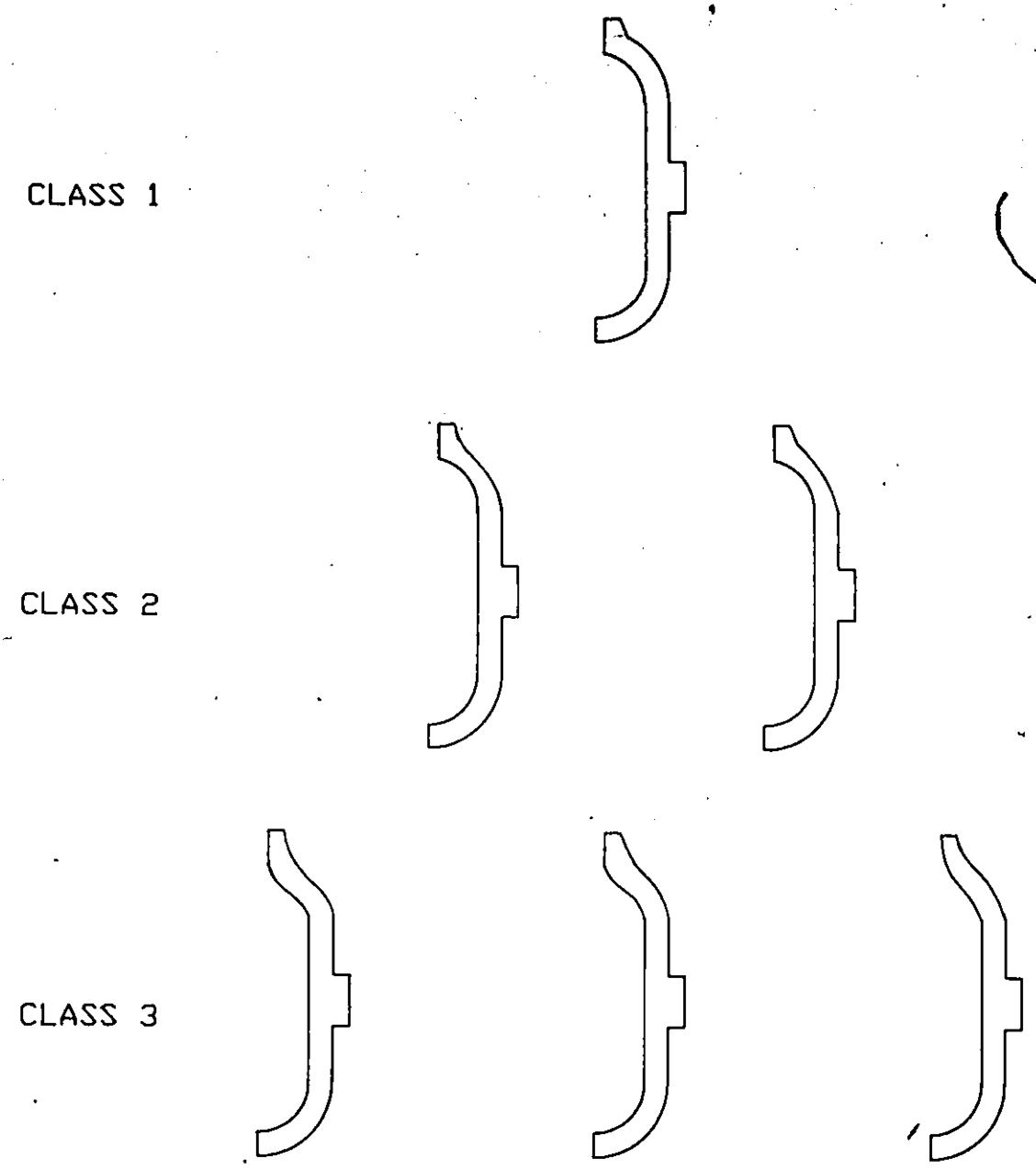


Figure 6.3 Configuration database of gas chamber family

screen.

Since the optimization function, constraint functions and design variables are related closely to the configuration, they are defined at the same time. Typical design variables are shown in Figure 6.2, which is an enlargement of subclass configuration 1 in class 3. The variables can be defined by moving the crosshair to a node and typing an appropriate key as described in section (4.2.5). A typical optimization function is the volume (weight) of the structure. Typical constraint functions are those related to the maximum stress, the curvature of the nodes on the curves, and so forth. All of these functions can be defined easily by using the facilities of system functions described in section (4.2.5). Some additional constraint functions are typed in as Fortran statements. They guarantee that the external diameters are greater than the internal diameters, e.g.

$$X_4 - X_1 > a$$

where a is a constant. The constraint functions about the lower and upper bounds of the design variables are taken care of by the system.

We need not worry about the material database, which is a permanent database in the system. Nine material classes are available, i.e. plain carbon steel, alloy steel, stainless steel, gray iron, ductile iron, wrought Al alloy, cast Al alloy, brass and bronze.

6.2.3 Specifying Standard Importance Values

There are 24 performance characteristics originally in the super-expert system. Since the leakage of the gas will cause the problem of environmental contamination, we have defined an additional performance characteristic, i.e. pollution, whose standard importance value is specified as 5.0. This performance characteristic usually influences the choice of configuration only.

The super-expert system has recommended standard importance values for the 24 original performance characteristics. Some of them are revised for this particular example. For example, the importance value of weight has been increased from 6.0 to 8.0; the importance value of stiffness has been decreased from 6.0 to 4.0. Other changes can be found in Table 6.1. The determination of the importance values in this case study was for demonstration purposes, and they may not all be completely realistic.

The importance values of some performance characteristics are interpolated from the direct values. Some interpolation values for these performance characteristics, e.g. load, have been modified for this example as in Table 6.2.

The super-expert system has recommended some interactive rules about the relationships between the

Table 6.1 Standard importance values

Performance characteristic	Super-expert system	Expert system
cost	10.0	10.0
weight	6.0	8.0
volume	6.0	8.0
accuracy	8.0	8.0
reliability	9.0	9.0
safety	10.0	10.0
low temperature	2.0	2.0
high temperature	5.0	9.0
humidity	2.0	2.0
corrosion	5.0	7.0
stiffness	6.0	4.0
load	8.0	7.0
frequency	5.0	5.0
impact	7.0	8.0
dimension,	3.0	3.0
style	3.0	1.0
noise	0.0	6.0
surface finish	5.0	3.0
wear	7.0	7.0
friction	3.0	7.0
lubrication	2.0	5.0
manufacturability	6.0	7.0
maintainability	7.0	7.0
service life	8.0	5.0
pollution	-	5.0

Table 6.2 Interpolation values for load

Importance value	0	1	2	3	4	5	6	7	8	9	10
Interpolation values in super-expert system	50	60	75	100	140	200	270	350	450	600	800
Interpolation values in expert system	50	60	75	100	500	600	800	1000	1200	1500	2000

importance values of different performance characteristics, and all of them were adopted.

In response to the prompt of the super-expert system, we have defined two number rules about how the number of products affects the importance values of some performance characteristics.

1. If the number of products is larger than 1000, multiply the importance value of accuracy by 1.2.
2. If the number of products is larger than 1000, multiply the importance value of manufacturability by 1.3.

6.2.4 Adapting Desirability Values

Linear programming has been used to adapt the desirability values for material candidates and configuration candidates in both the class level and the subclass level. The samples to be adapted are usually created from design records, which should be acquired from a company or a design institute. However in this case study, they were created by Monte Carlo simulation combined with some judgement. Each sample consists of 100 examples which are divided into five groups (i.e. 20 examples in each group). The only exception is that the sample used to adapt the desirability values for material candidates in class level consists of 500 examples, as these adapted desirability values will be tested intensively later. The

CPU time for adapting this sample was 244 minutes on a VAX 730 computer.

6.2.5 Specifying Failure Mode and Factor of Safety

First of all, according to the specified standard importance values (section (6.2.3)), and the derived desirability values, the super-expert system finds out the most desired material and configuration, and displays them on the screen. In this example, they are plain carbon steel AISI-SAE 1078, and subclass configuration 1 of class 3 (Figure 6.3). They are called standard material and standard configuration, used to determine the reference failure mode and factor of safety for the gas chamber family.

First the failure mode is dealt with. The super-expert system introduces two theories each for ductile and brittle material. The distortion energy theory was selected for the ductile material and the modified Mohr theory for the brittle material.

Noting the standard material and standard configuration, we enter 3.0 as the basic factor of safety, and type in a short text of explanation - "From field tests".

Then the super-expert system reminds us of some facts that may give rise to modifying the basic factor of safety. Some are accepted and others ignored. For example,

the super-expert system reminds us that the importance value of reliability is rather high, thus we increase the factor of safety to 3.5. The super-expert system also tells us that the number of cycles of the alternating internal pressure is large. Fatigue may occur, so we multiply the current factor of safety by 1.2. Finally, the reference factor of safety is 6.0.

Next the super-expert system shows us all of the materials and configurations other than the standard material and standard configuration, and asks whether some modifications concerning the reference factor of safety should be made if a special material or configuration is used. In response, we specify that, if gray iron or cast Al alloy is used, the reference factor of safety should be multiplied by a factor of 1.2, and type in a text of explanation - "brittle material".

6.2.6 Specifying the Optimization Method and the FEM Algorithms

The super-expert system advises us to adopt the Hooke and Jeeve's direct search method ahd Schuldt's penalty function, and these are accepted.

Then we change the values of some parameters provided by the super-expert system, in order to accelerate the convergence by sacrificing a little precision in the optimum solution.

The super-expert system suggests the adoption of all

of the six FEM algorithms, i.e. the global direction method, the substructure method, the skipping method, the safe-fail line method, the quadratic method, and the differential method. Since we are not quite sure about the performance of the global direction method in this example, we decide to adopt only the last five methods. For the parameters relevant to these algorithms suggested by the super-expert system, the only modification made is to decrease the factor of the valid increment for the differential method from 0.02 to 0.00125. Because the curvature constraint functions are rather strict, from our experience, a small increment helps to avoid divergence.

The expert system for the design of the gas chamber family is now complete.

6.3 DESIGNING A GAS CHAMBER USING THE EXPERT SYSTEM

Now we, as a designer, are going to use the above expert system to design a gas chamber, following the procedure described in chapters 4 and 5.

6.3.1 Specifying Importance Values

The expert system displays its standard importance values on the screen. We think they are very suitable to our requirements, so all of them are adopted except increasing the importance value of wear from 7.0 to 8.5.

Then we enter the number of products --- 500. There

are two number rules in the expert system (set up in section (6.2.3)). But, since the number of products is less than the specified critical number of 1000, none of them takes effect.

One interactive rule in the expert system takes effect, i.e.

"If the importance value of wear is larger than 7.0, it is recommended that the importance value of lubrication be increased to 8.0."

We think this is not quite suitable to our situation and thus reject it.

6.3.2 Determination of Material

Now we determine a material, consulting the expert system. The input is the importance values specified in section (6.3.1). Since the database is not large, we ask the expert system to adopt the full search method. After some calculation, the expert system shows us the material classes in the database as follows and informs us that the top one is the best while the bottom one is the worst.

Plain carbon steel

Stainless steel

Alloy steel

Bronze

Gray iron

Ductile iron

Wrought aluminum

Brass

Cast aluminum

We trust the expert system and accept plain carbon steel. Then the expert system displays the four subclass materials and their eleven major properties, and again informs us of their relative goodness.

AISI-SAE 1078

AISI-SAE 1045

AISI-SAE 1020

AISI-SAE 1035

Unfortunately, we find that none of them are satisfactory, and decide to enter our own material. But we only know that the yield strength of the material is 40 ksi and it belongs to plain carbon steel. The expert system finds out in its database that AISI-SAE 1035 is the material in plain carbon steel that is closest to the new material. So its properties are taken to supplement the missing properties of the new material.

Next the expert system compares our material with its recommended material (i.e. plain carbon steel AISI-SAE 1078) and provides some messages about the relative goodness of the new material. Examples of the messages are as follows.

Importance value of cost is 10.0
Relative value of cost/weight is 66.7%

Importance value of weight is 8.0
Relative value of weight/strength is 138.9%

Noting these messages, we still decide to use our own material selection.

6.3.3 Determination of Configuration

The expert system displays the first subclass configuration of each configuration class and informs us that class 3 is the best, class 2 is the second, class 1 is the worst. By moving the crosshair, we select class 3.

Next, the expert system displays the three subclass configurations in class 3, and tells us that subclass configuration 1 is the best, subclass configuration 3 is the second, subclass configuration 2 is the worst. We select subclass configuration 1 (Figure 6.2).

Then the expert system asks if we want to modify this configuration, including the relevant design variables and functions. After inspecting all of them carefully, our answer is "no".

6.3.4 Determination of the Failure Mode and Factor of Safety

Since the material we entered is ductile, the expert system shows us two failure theories, i.e. the distortion energy theory and the maximum shear stress theory, and recommends the adoption of the former. We agree with it.

Then the expert system suggests a reference value 6.0 for factor of safety and asks if an explanation is

required. Since our answer is "yes", the expert system tells us that the basic value of the FS is 3.0 "from field tests". Then it lists other modifications one by one. Examples are as follows.

> The importance value of reliability is high, increase the FS to 3.5
>
> The number of cycles of the alternating load is large, fatigue may occur, multiply the FS by a factor of 1.2

Next, since the material we decide to use is different from that suggested by the expert system, the expert system displays the properties of both materials and asks if any modification of the FS should be considered. Our answer is "no".

The configuration adopted is the one recommended by the expert system, so no modification of the FS due to the specific configuration is prompted.

However, we finally decide to set the FS at 1.0, in despite of the warning from the expert system that the FS is usually larger than 1.25. The reason is that we want to compare the result of this design to that of an existing example.

6.3.5 Detailed Specifications for the Design

The following steps are included.

1. Entering the concrete coordinates of the nodes. If a coordinate of a node is constant, the entered value is

its dimension; if it is a variable, the entered value is its starting value. Actually we need only enter some of them; the expert system can determine other coordinates intelligently. Also, we can enter the radii and angles of some nodes instead of the x, y coordinates.

2. Entering the upper and lower bounds, and the starting values of the design variables.

3. Defining X_8 and X_9 as the two important variables. This information will be used to draw the optimization topography.

4. Inducing design variables. In Figure 6.2, the design variable X_8 is defined as the x coordinate of node 13. The expert system "knows" that this also implies the x coordinate of node 12, but it fails to recognize that X_8 also implies the x coordinates of nodes 9 and 5. We have to indicate this. Similarly, X_{10} is defined as the y coordinate of node 11, and the expert system only knows that it also controls the y coordinates of nodes 12, 10, and 9. It should be made to control the y coordinates of nodes 6 and 7 too.

5. Specifying the normal pressure on the inner surface. It is 10000 psi in this particular design.

6.3.6 Determination of the Optimization Method and FEM Algorithms

The expert system recommends the Hooke and Jeeve's direct search method, Schuldt's penalty function and ten

optimization parameters; all are accepted.

Then the expert system advises us to adopt five FEM algorithms, i.e. the substructure method, the skipping method, the safe-fail line method, the quadratic method and the differential method, and provides the recommended values for relevant parameters; we accept all of them again.

6.3.7 Establishment of the FEM Mesh

We specify the interval number on edge 7-8 in Figure 6.2 as two. The expert system generates the FEM mesh and arranges the FEM nodes to minimize the bandwidth of the stiffness matrix. The FEM mesh that is generated is shown in Figure 6.8.

6.3.8 Running the Optimization

Up to this point, the complete model for an optimization design has been completed. We sum up the data as follows.

Hooke and Jeeve's direct search optimization method

Schuld's penalty function

F:	factor of range as initial step size	0.02
G:	factor of F as minimum step length	0.1
TOL:	convergence criterion	0.01
ZERO:	constraint tolerance	0.002
R:	penalty factor	10.0

Substructure method

Skipping method

Safe-fail line method

Quadratic method

Difference method

NSAFE:	number of safe lines	3
NFAIL:	number of fail lines	1
DIS:	valid radius for quadratic method	0.08
XSTEP:	valid increment for differential method	0.00125

X_1, X_3 negative gradient variable

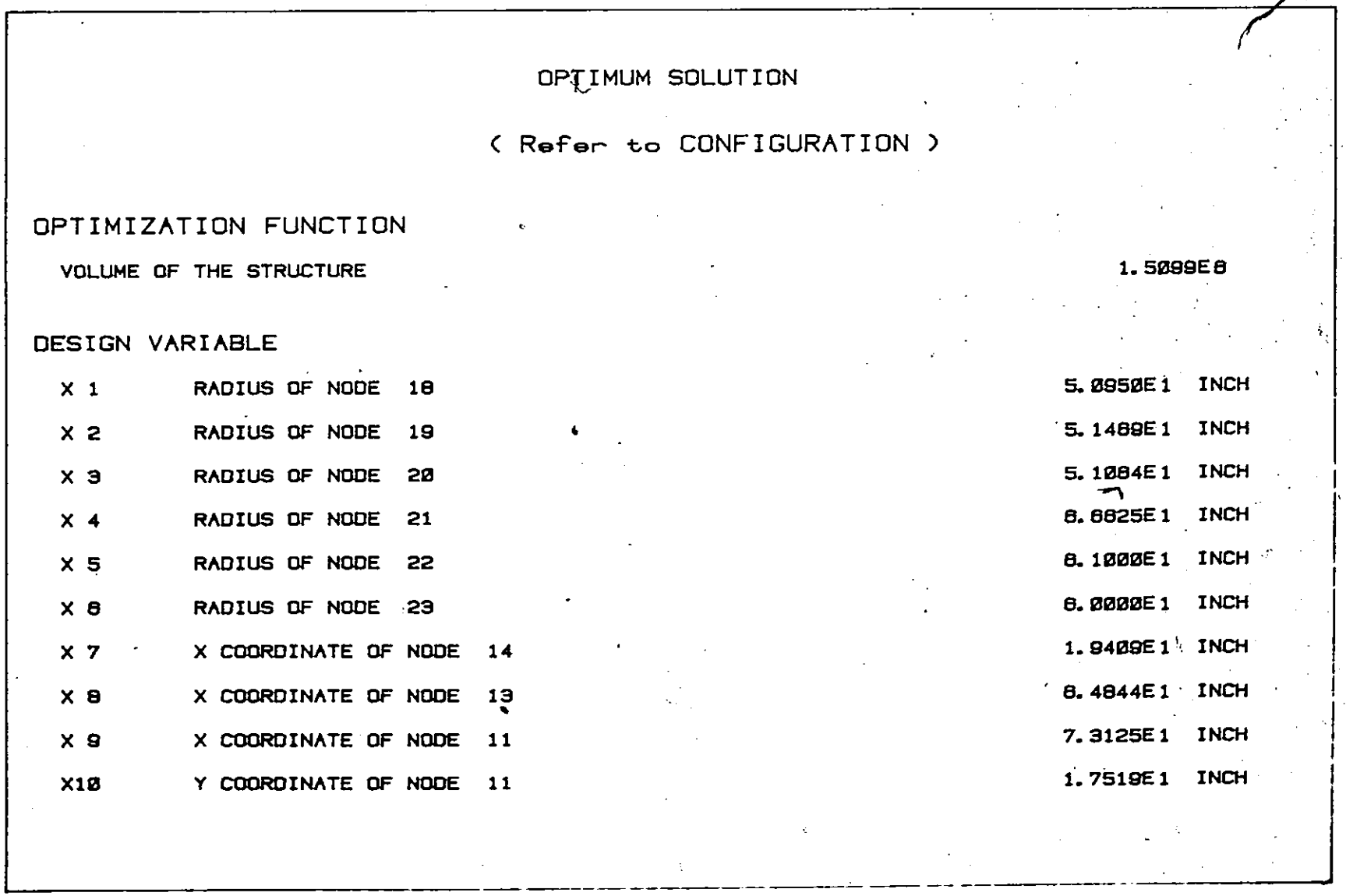
X_2 positive gradient variable

X_4 - X_{10} increase variable

The design variables, optimization function and constraint functions can be seen in Figure 6.2 and Figure 6.4.

Figure 6.2 is the configuration at the starting point. Five hardcopies of the optimum solution are shown in Figure 6.4 to Figure 6.8.

The CPU time used on a VAX 730 is 87.5 minutes. The substructure method actually had no effect, since all of the regions are non-constant; the skipping method took effect 103 times due to the violation of the curvature constraints; the safe-fail line method took effect 229 times (safe line 134 times, fail line 95 times); the differential method, 49



OPTIMUM SOLUTION

(Refer to CONFIGURATION)

OPTIMIZATION FUNCTION

VOLUME OF THE STRUCTURE		1.5099E8

DESIGN VARIABLE

X 1	RADIUS OF NODE 18	5.0950E1	INCH
X 2	RADIUS OF NODE 19	5.1488E1	INCH
X 3	RADIUS OF NODE 20	5.1084E1	INCH
X 4	RADIUS OF NODE 21	8.6825E1	INCH
X 5	RADIUS OF NODE 22	8.1000E1	INCH
X 6	RADIUS OF NODE 23	6.0000E1	INCH
X 7	X COORDINATE OF NODE 14	1.9409E1	INCH
X 8	X COORDINATE OF NODE 13	6.4844E1	INCH
X 9	X COORDINATE OF NODE 11	7.3125E1	INCH
X10	Y COORDINATE OF NODE 11	1.7519E1	INCH

Figure 6.4 Optimum solution of the gas chamber

CONSTRAINT FUNCTION

CURVATURE OF NOD 20 19 ON POLAR CURVE	1.8000E-2
CURVATURE OF NOD 23 22 ON POLAR CURVE	9.2848E-3
FACTOR OF SAFETY ABOUT MAX. STRESS	-1.8345E-3
X4-X1-10.	5.8754E0
X5-X2-7.	2.5315E0
X8-X3-7.	1.9381E0
X9-X8-7.	1.2813E0

MATERIAL

USER SPECIFIED MATERIAL		
ELASTICITY MODULUS	3.0000E1	mpsi
POISSON RATIO	3.0000E-1	unit
UNIT WEIGHT	2.8000E-1	lb/in3
THERM EXPAN COEFFICIENT	6.7000E0	10E-6/F
ULTIMATE STRENGTH	7.2000E1	ksi
YIELD STRENGTH	4.0000E1	ksi
COMPRESSIVE STRENGTH	----	

Figure 6.4 (Cont.)

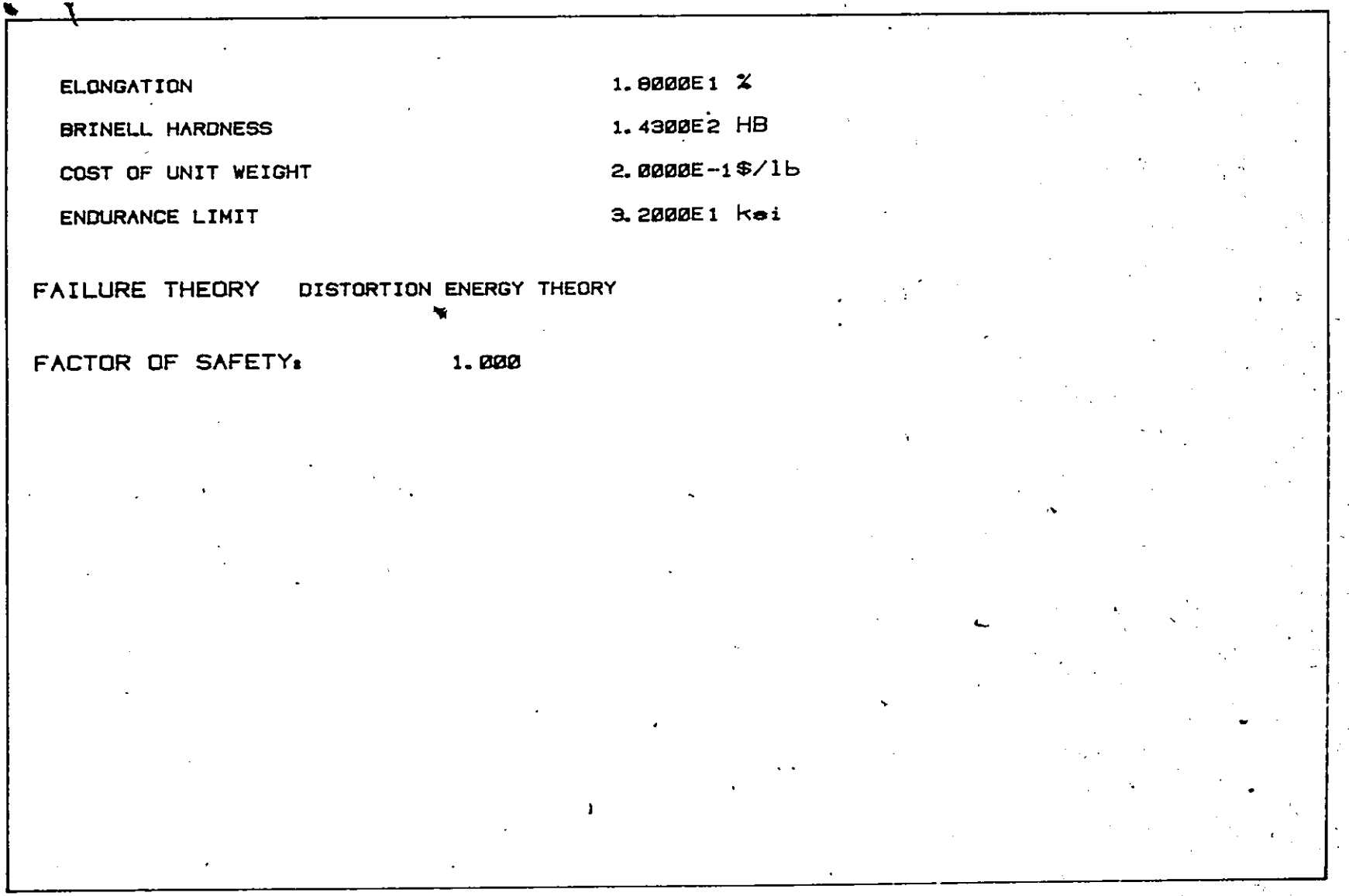


Figure 6.4 (Cont.)

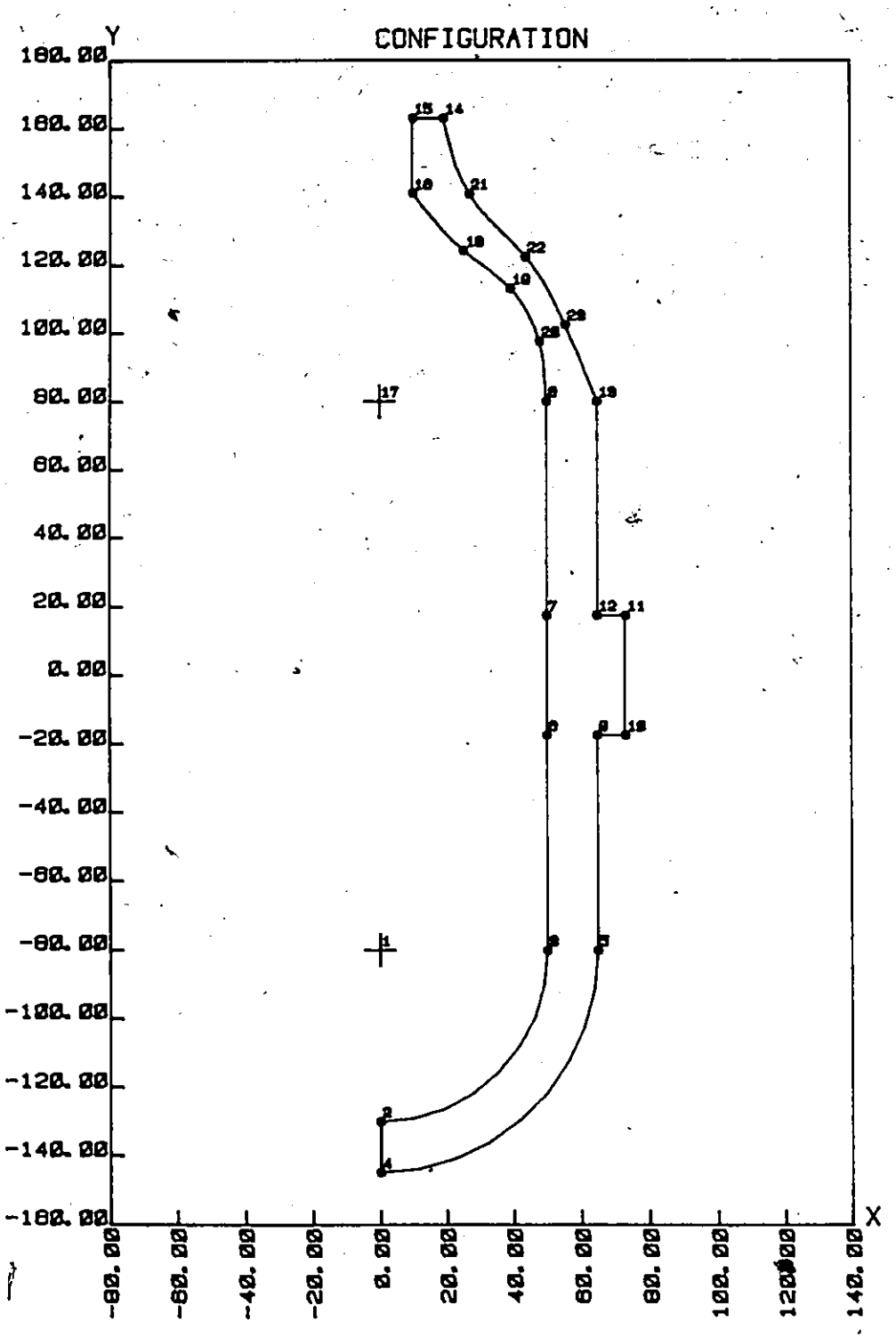


Figure 6.5 Optimum configuration of the gas chamber

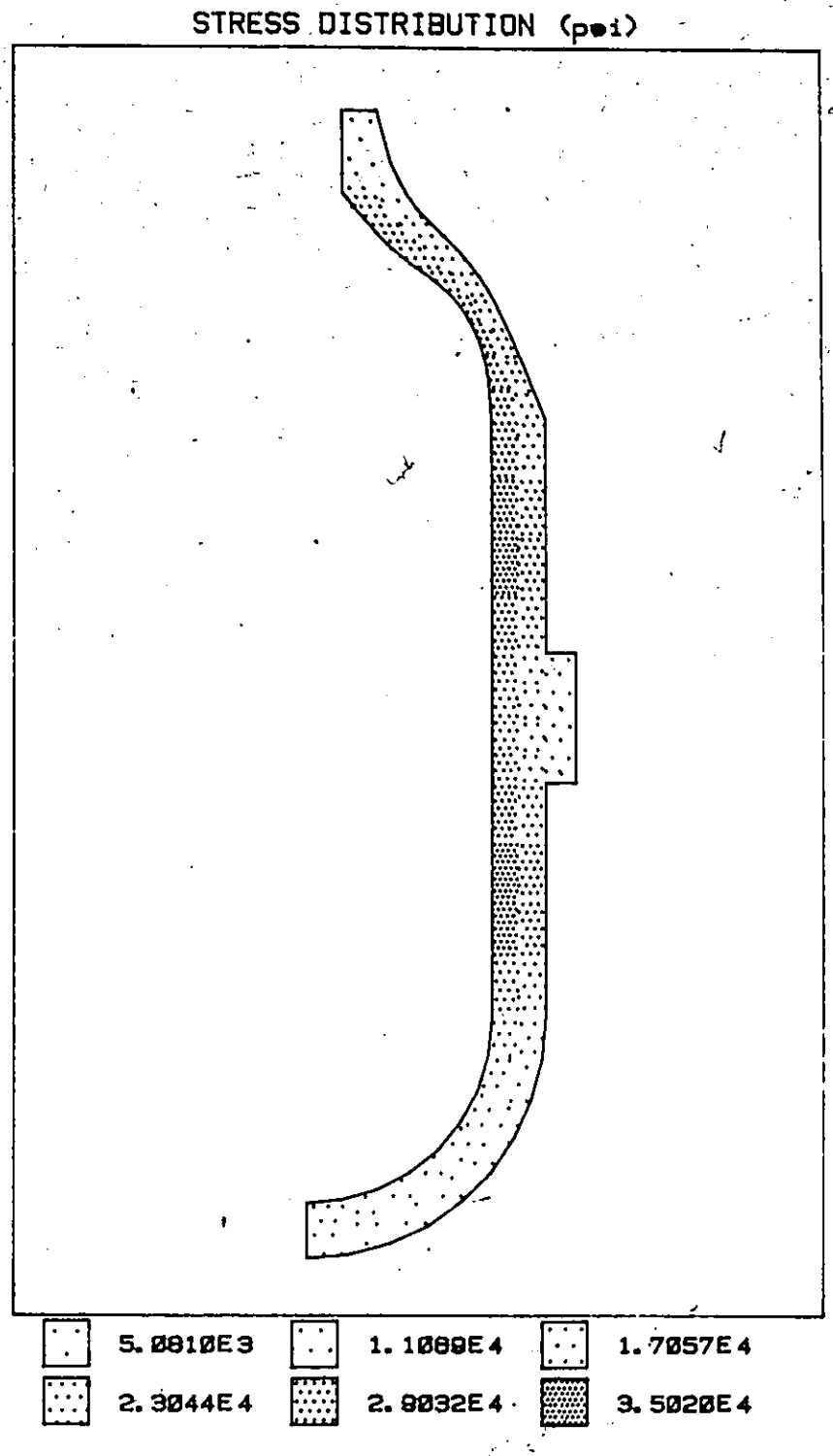


Figure 6.6 Stress distribution of the gas chamber at optimum

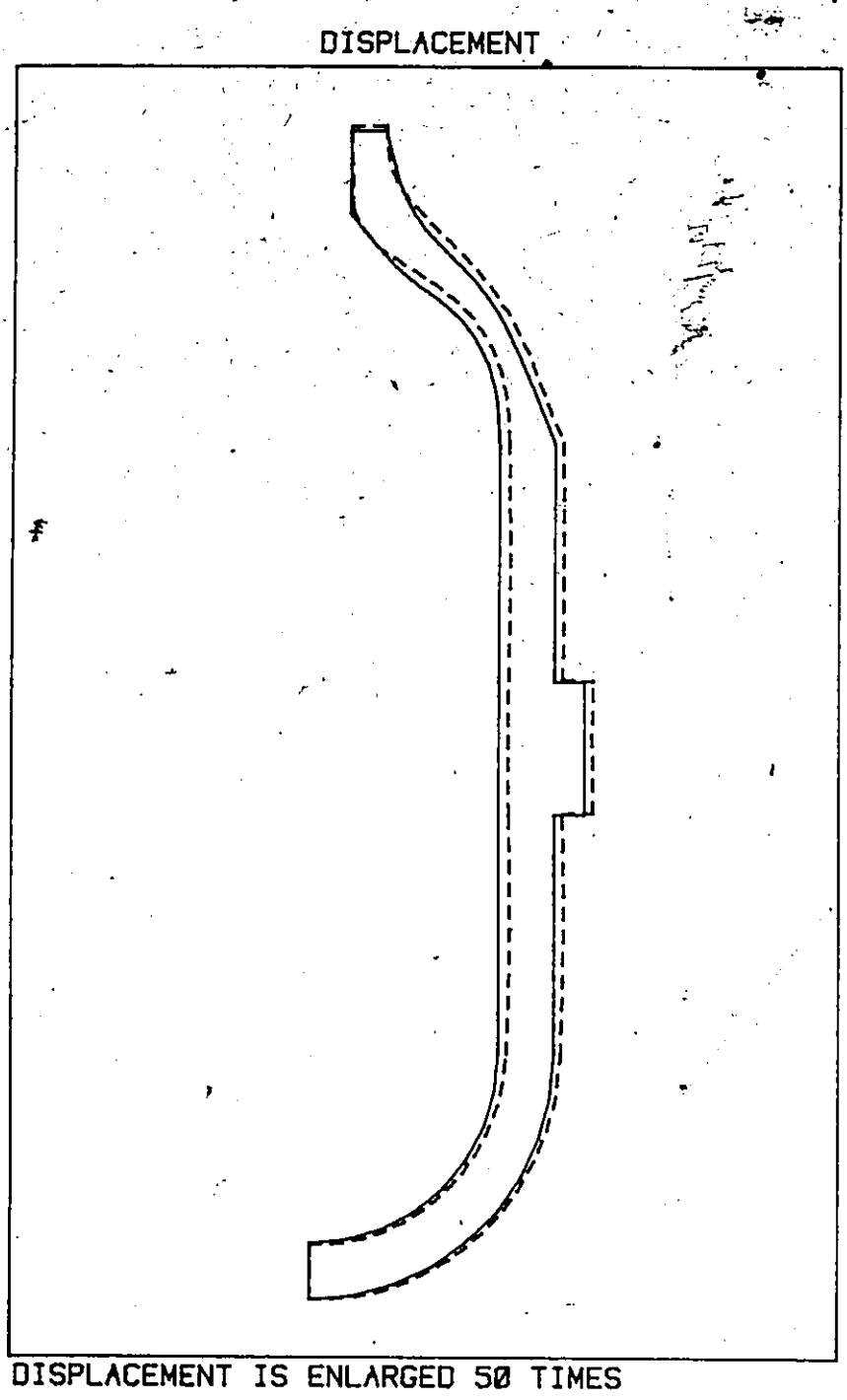


Figure 6.7 Displacement of the gas chamber at optimum

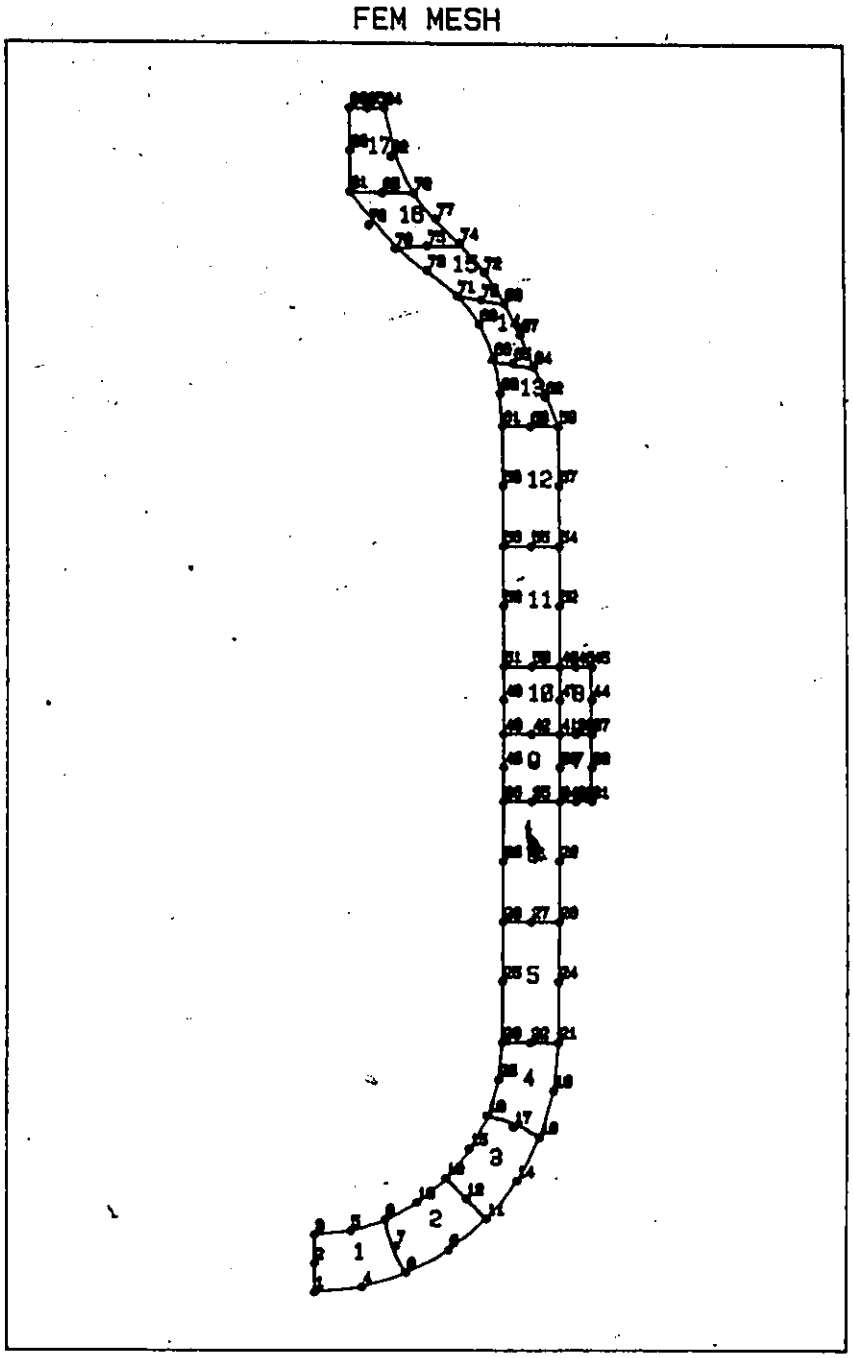


Figure 6.8 FEM mesh of the gas chamber at optimum

times; the quadratic method did not take effect, since the step size is fairly large.

The FEM mesh would be regenerated (refined) automatically during the optimization search under either of two conditions: a large aspect ratio or large stress difference. In this example, the aspect ratio is always small, less than two. The stress difference is relatively large; the maximum value is close to 20%.

Originally the allowable stress difference was set at 10%, giving the FEM mesh shown in Figure-6.9. Such a dense FEM mesh would require a prohibitive CPU time in our mini computer and take a great amount of computer central memory. So at this point, we are forced to relax the allowable stress difference to 20%, therefore the FEM mesh remains unchanged during the whole optimization search. Figure 6.8 is the FEM mesh at the optimum point, where the maximum stress difference is about 15%.

In Figure 6.4, it may be noted that the value of the constraint function related to maximum stress at the optimum point is -0.00163. This minor violation of the constraint function is within the tolerance.

6.4 SOME INTENSIVE STUDIES

For this case study, some more experiments were done on two important issues of this project: (1) testing the validity of the desirability values, the major inference

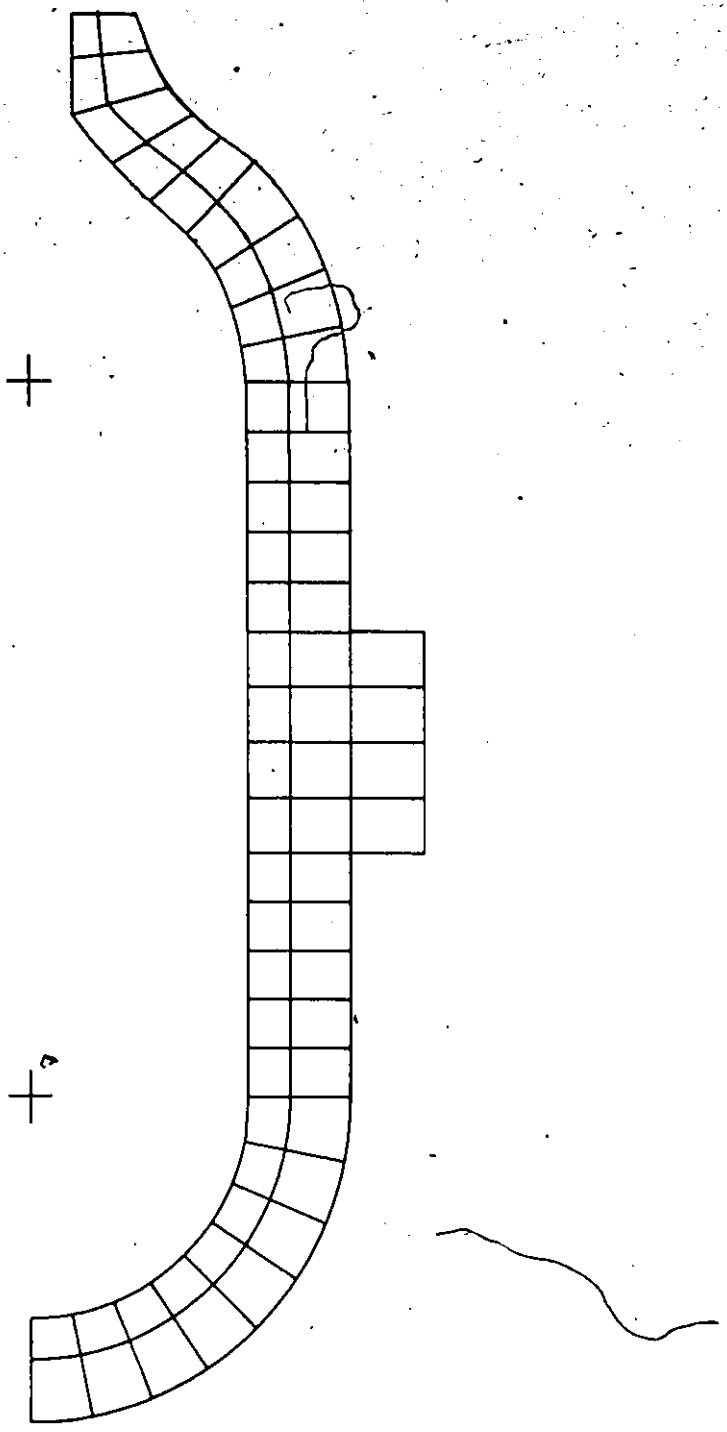


Figure 6.9 Refined FEM mesh of the gas chamber

engine of the expert system (formula (1.1)), and the three search methods, based on the determination of material class; (2) evaluating the CPU time that can be saved in the FEM based optimization by the integrated use of the FEM algorithms.

6.4.1 Evaluating the Reference Engine and Search Methods

The main concern was how accurately the desirability values derived in section (6.2.4) represent the expertise of the human expert.

For this purpose, an "accurate" search method is required. It may be recalled that we adapted the desirability values of the material from a sample consisting of 500 examples. Let us look at the ith example. It includes 25 importance values, P_{ij}, (where $j=1,2,....,25$) and a desired candidate C_i. Now suppose the designer also specifies an input that includes 25 importance values, p_j. Then

$$d_i = \sqrt{\sum_{j=1}^{25} (P_{ij} - p_j)^2} \qquad (6.1)$$

is defined as the distance between the ith example and the designer's input.

Among the 500 examples, if the distance between the kth example and the designer's input is the smallest, then the desired candidate in the kth example, C_k, is taken as the "accurate" solution for the designer's input. This search method does not make use of the desirability values.

This "accurate search method" has not been included in our system because it is only accurate in a certain sense; and furthermore it takes a lot of computer memory to store the whole 500 examples, and requires a lot of CPU time to evaluate the distances between the designer's input and each example in the sample. Here it is just used to compare with the three search methods developed in this project, i.e. the full search method, the discard search method and the quick search method. For an individual designer's input, we shall list the three best candidates found by each search method. This is simulated by entering ten inputs randomly, with the results shown in Table 6.3.

First the candidates found by the accurate search method and the full search method are compared. It can be seen, among the ten inputs, that there are nine times that the best candidates found by both methods are the same. In only one instance the best candidate found by the accurate search method is the second best one found by the full search method. This is strong evidence that the desirability values adapted by linear programming can reflect the essence of the original sample very well, and the reference engine working on the desirability values is able to produce a solution fairly analogous to that of a human expert.

Since the discard search and quick search methods have been developed based on the full search method, it is

Table 6.3 Best candidates found by different search methods

No.	Accurate			Full			Discard			Quick		
1	3	1	9	1	3	9	1	3	9	1	3	5
2	3	2	9	3	4	2	3	2	9	3	1	5
3	1	3	2	1	2	3	1	2	3	2	7	8
4	3	9	1	3	9	1	3	9	1	6	2	7
5	1	3	2	1	3	2	1	3	2	1	3	5
6	1	3	9	1	9	3	1	9	3	1	3	5
7	1	3	9	1	3	9	1	3	9	2	7	8
8	1	3	2	1	2	3	1	2	3	1	2	3
9	1	3	2	1	2	3	1	2	3	2	8	7
10	2	3	9	2	3	1	2	3	1	3	1	5

1 Plain carbon steel

2 Alloy steel

3 Stainless steel

4 Gray iron

5 Ductile iron

6 Wrought aluminium alloy

7 Cast aluminium alloy

8 Brass

9 Bronze

appropriate that the results of these two search methods be only compared with that of the full search method. It is found that the results of the full search and the discard search methods are almost the same except for minor difference in the second input.

The differences between the best candidates found by the full search method and the quick search method are significant. However, some consolation can be found from another viewpoint. We find that among the ten inputs there are five times, i.e. 50%, that the best candidate found by the full search method is contained in the three best candidates (i.e. the desired group) found by the quick search method, and three times, i.e. 30%, that the second best candidate found by the full search method is contained in the three best candidates found by the quick search method. To sum up, among the ten inputs, there are eight times, i.e. 80%, that the first two among a total of nine candidates found by the full search method are contained in the desired group found by the quick search method. Since a full search will be performed on the desired group finally, the quick search method can provide the similar result as the full search method in 80% of the applications.

The computer time spent by each method for one designer's input is as follows.

Accurate method: 16.11 second

Full method: 0.01 second

Discard method: 0.04 second

Quick method: 0.02 second

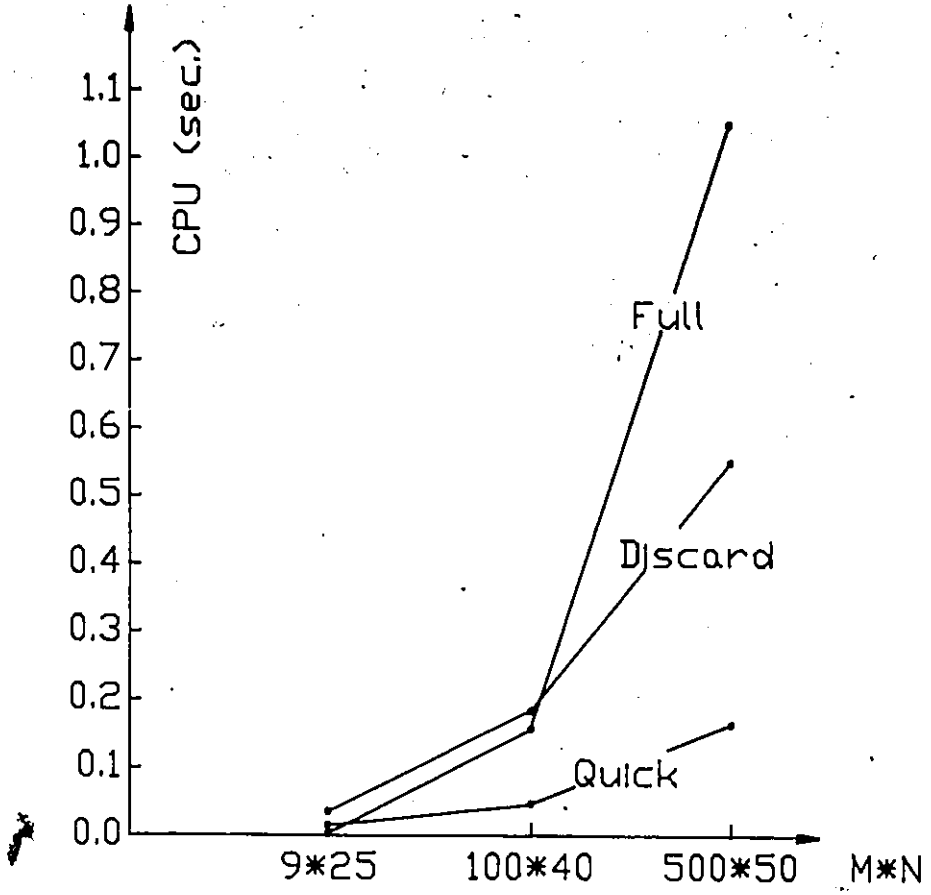
The accurate method is obviously unacceptable in search speed. Since the database is very small in this case study, even the discard and quick methods demonstrate no advantage in speed.

Two pseudo databases were then created to further test the search speed. As expected, the size of the database increased, the discard method and quick method become more and more superior to the full method in search speed. As shown in Figure 6.10, when the database is large, the discard method can save some computer time, almost without any loss in accuracy. The quick method is very fast, though less accurate.

6.4.2 Evaluating the FEM Algorithms in Saving Computer Time

The optimization program of the gas chamber was run two more times to compare the results and the computer times when using different FEM algorithms.

In the second run, none of the FEM algorithms for saving computer time was adopted. The results are shown in Figure 6.11, 6.12 and 6.13. By comparing Figures 6.5 and 6.11, which are the optimum configurations of the first and second runs respectively, it can be seen that the convergence in the second run is a little better at the opening of the gas chamber. Table 6.4 is the comparison of



Note: 1 M is number of candidates; N is number of performance char.

2 The CPU time of the quick search method does not include the time for the final full search in the desired group.

Figure 6.10 Search speed

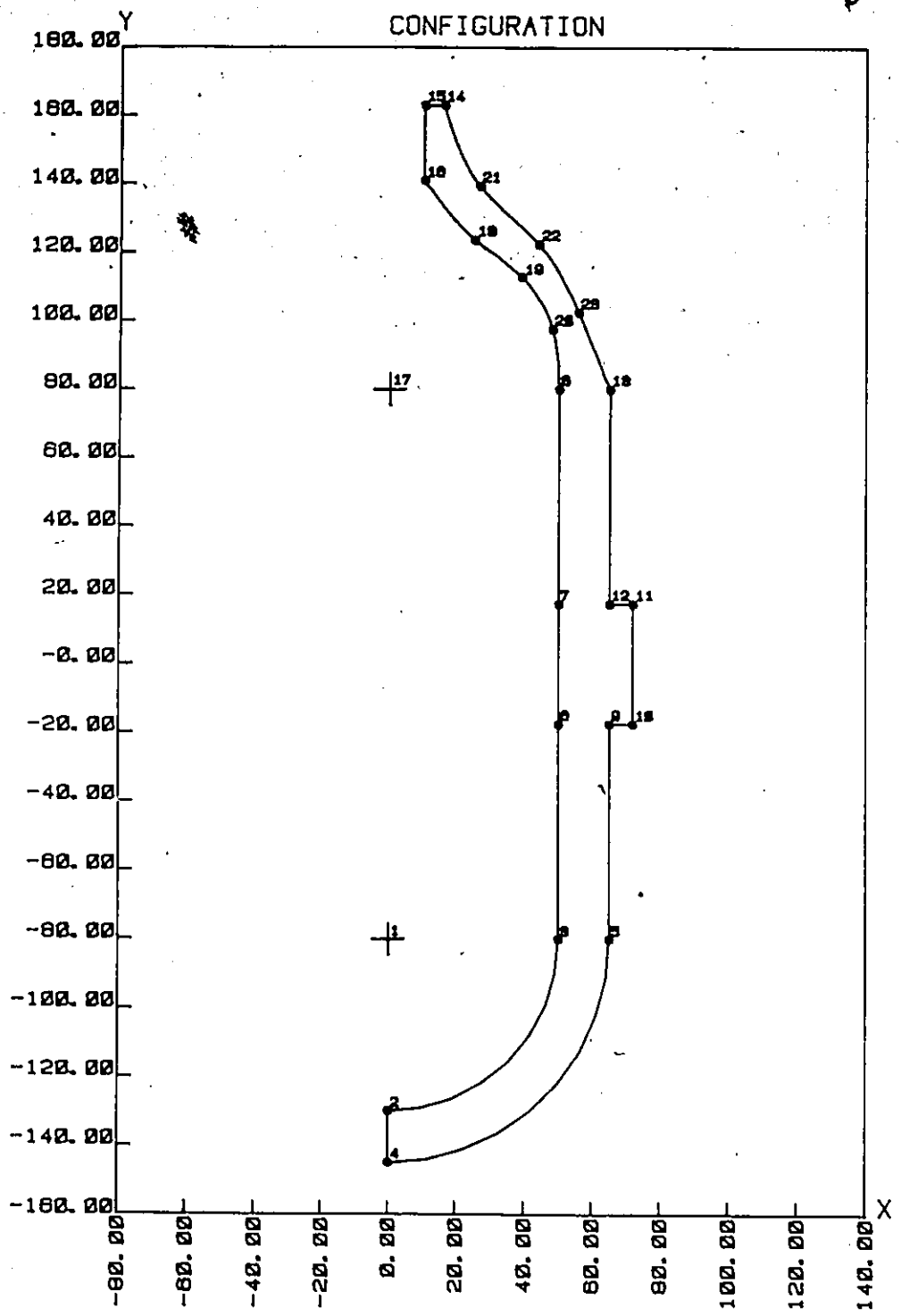


Figure 6.11 Configuration of the gas chamber in the second run

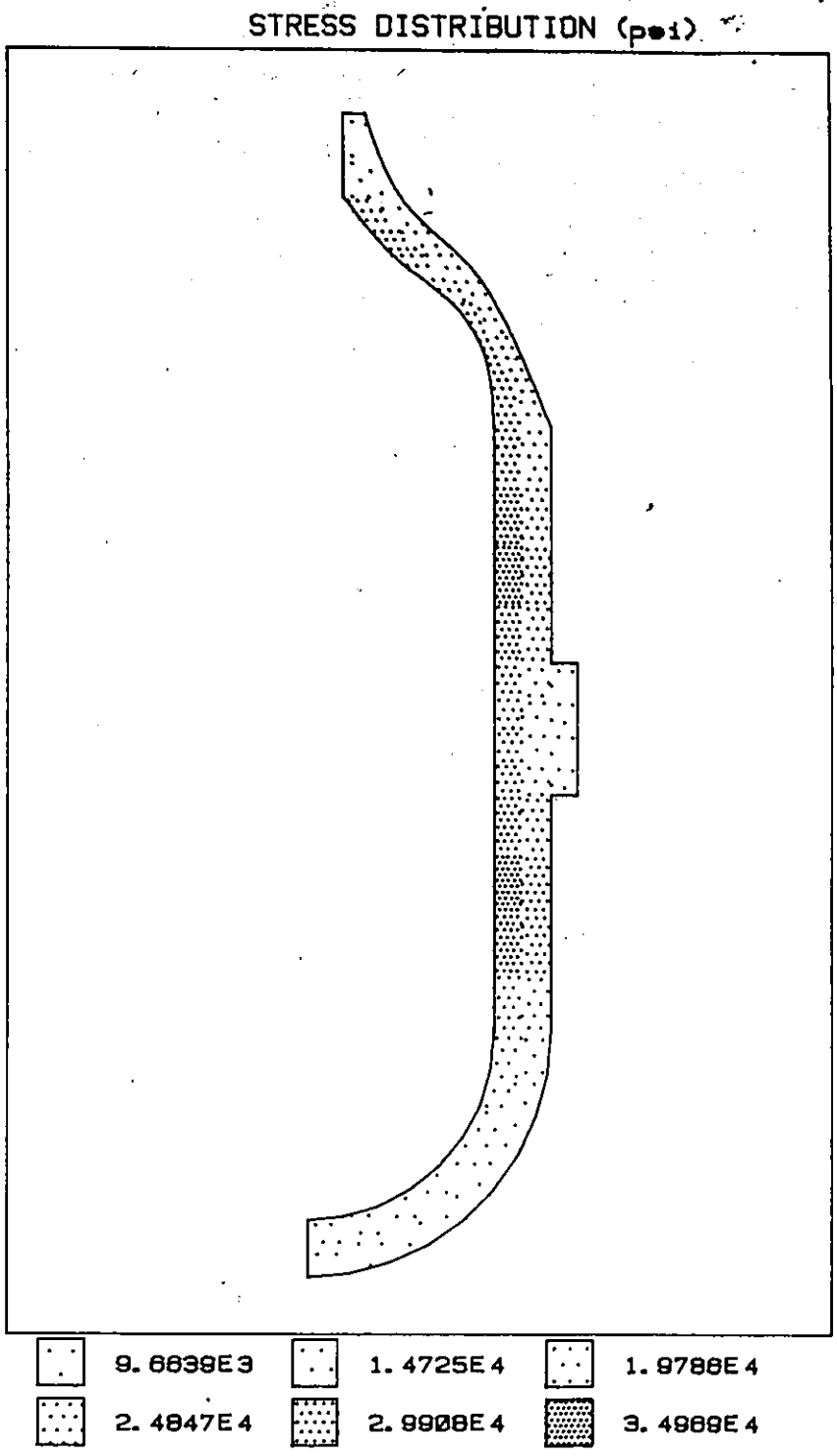


Figure 6.12 Stress distribution of the gas chamber in the second run

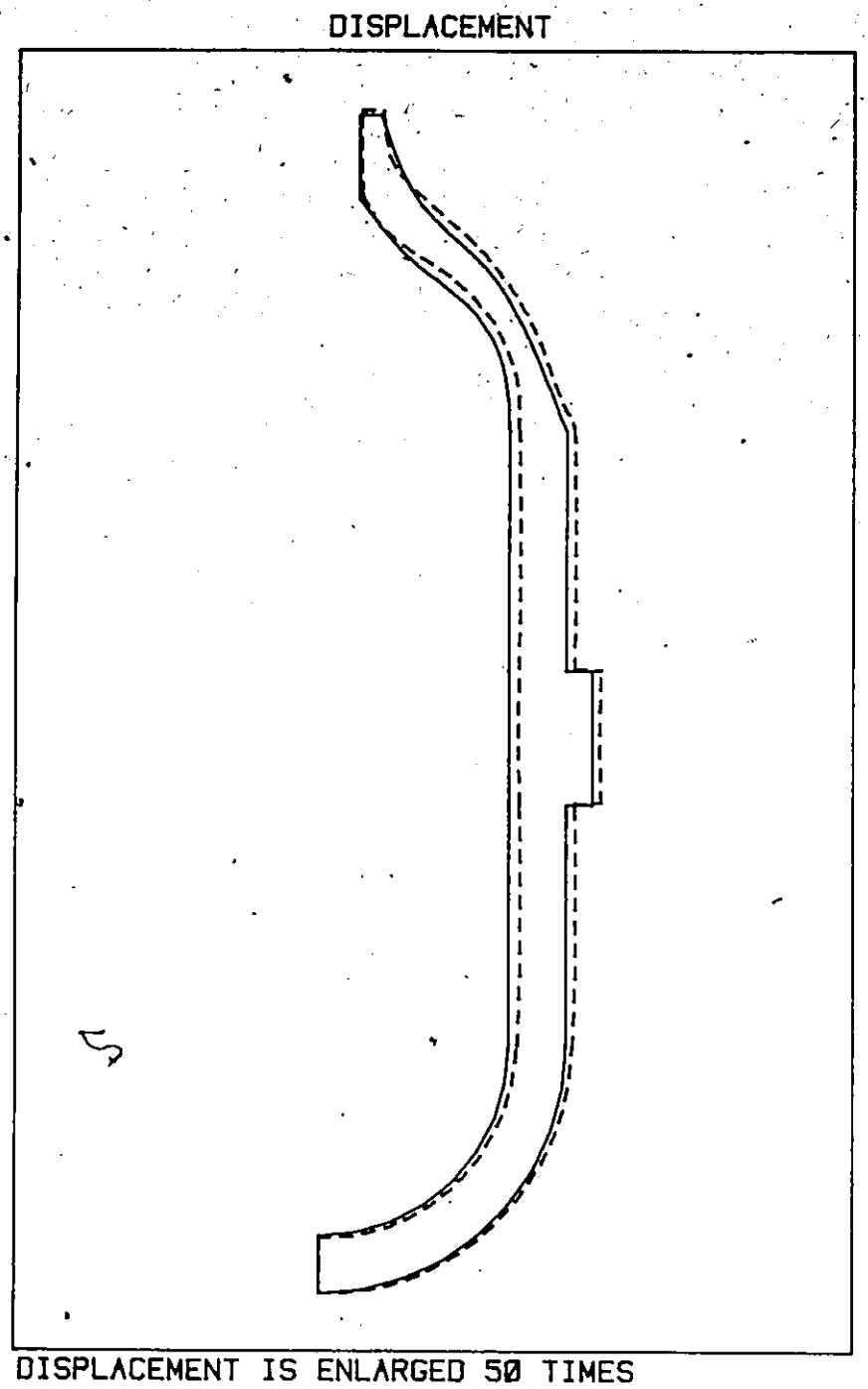


Figure 6.13 Displacement of the gas chamber in the second run

Table 6.4 Comparison of the results of the gas chamber in the first and the second runs

	First run	Second run
Optimum solution	1509937	1489523
Constraint function about maximum stress	-0.00163	0.00136
Number of iterations	674	775
Total CPU time	87.5 min.	281.3 min.
CPU time per iteration	0.13 min.	0.36 min.

the results between the first and the second optimization runs. It is clear that a significant amount of computer time has been saved in the first optimization run, although its convergence is a little bit worse.

Finally the optimization program was run a third time with all of the FEM algorithms for saving computer time adopted, including the global direction method. The global direction method first brought the design to a new starting point as in Figure 6.14 (compared to Figure 6.2). The optimum solution obtained is shown in Figure 6.15. The rather odd result is due to the strict constraints related to the curvature of the nodes on the polar curves, applied to this particular example.

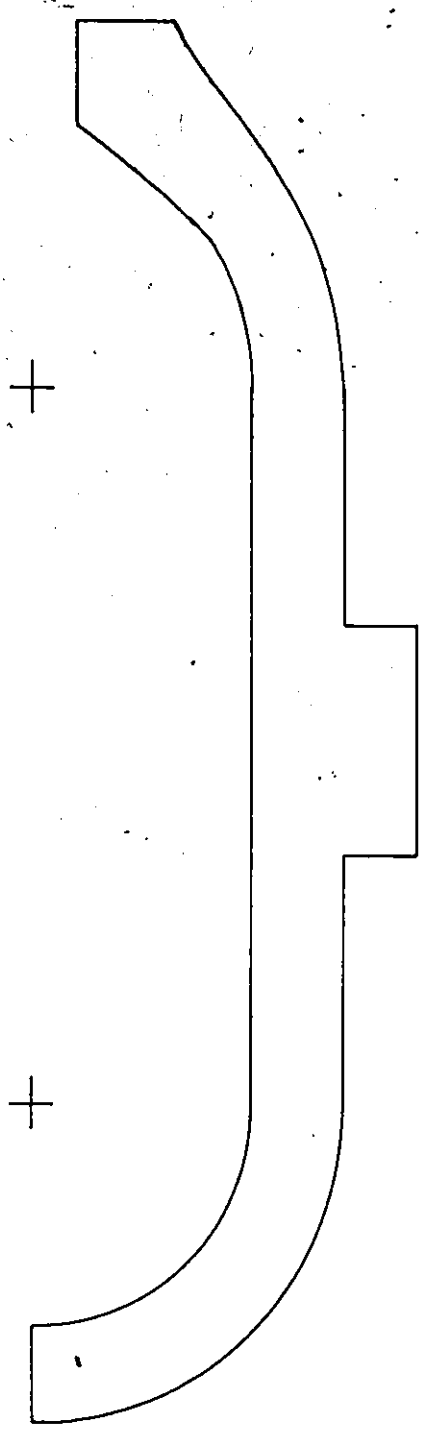


Figure 6.14 New starting configuration of the gas chamber in the third run

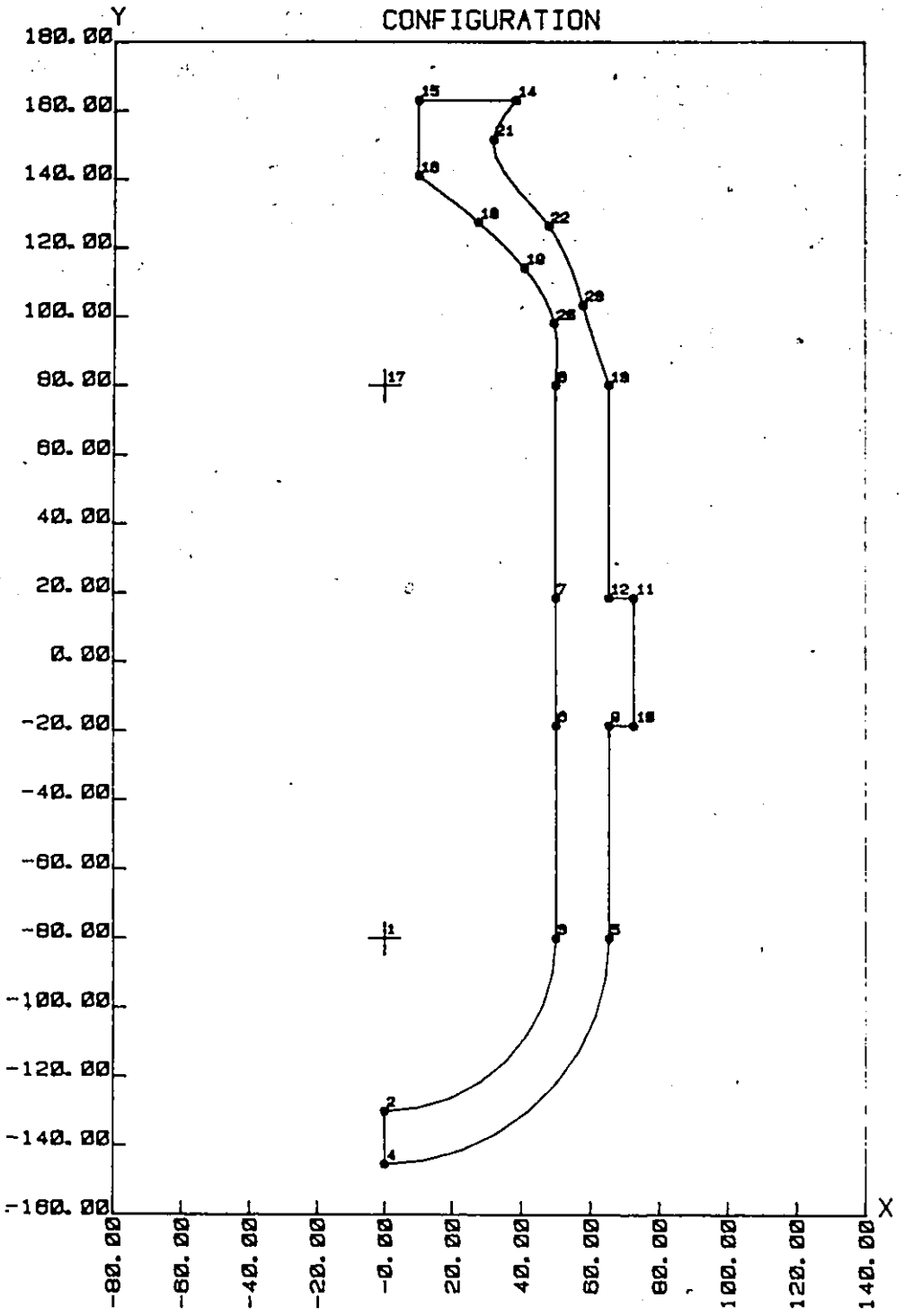


Figure 6.15 Configuration of the gas chamber in the third run

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 CONCLUDING SUMMARY

We have succeeded in developing software that incorporates an expert system with finite element-based optimization; and which has made a number of contributions in a wide area of design automation, both in practice and concept.

In this project, the expert system mainly takes part in the design decision making and design synthesis; the optimization method, based on value theory, guides the whole design process; and the finite element method is the analysis tool. The whole program is totally interactive, and is graphics oriented and user friendly. This project, therefore, introduces the most advanced techniques in design automation to the practicing designer, providing him with a promising and powerful tool for increasing the design efficiency and quality of his routine work.

Four new algorithms were also developed for increasing the efficiency of FEM based optimization significantly.

The original contributions of this project are summarized as follows.

1. Contributions to expert system theory

(1) A complete expert system for structural design has been developed, using optimization and finite element methods. The expert system provides abundant consultations, advice, recommendations and error checks for the designer throughout the whole design procedure, particularly in decision making and synthesis.

(2) By means of the two-level expert system, it is possible to deal with a wide range of structural design problems, and at the same time, particular expertise for a specific design can be built. By adopting the same programming routines in both levels of expert systems, a lot of effort in building the system can be saved. The human expert can concentrate on developing the expertise (specific knowledge) for a specific design problem without worrying about state of the art of design knowledge, and does not need help from a knowledge engineer.

(3) The organization of the database of the expert system into classes and subclasses reduces its development cost (especially for the configuration database) and computer memory, and reduces the computer time in searching for a solution.

(4) A new approach to expert systems has been developed, incorporating machine learning from a sample. It uses linear programming to systematically convert the

expertise from a human expert into desirability values, which represent the expert knowledge, and it is easy to program. An algorithm for creating the sample has been developed. When the number of examples in the sample is large, the examples can be divided into several groups for effective adaptation.

(5) Three search methods, i.e. full search method, discarding search method and quick search method, have been developed. Each of them has a different search accuracy and speed.

(6) The basic inference engine in the expert system is numerically oriented (formula (1.1)). Other logic approaches are also adopted, such as the interactive rules and number rules for adjusting the input importance values.

(7) Expert systems are included for selecting material, configuration, factor of safety, and so on; each has its distinct features.

2. Contributions to integrated CAD

(1) This project also integrates conventional CAD, the optimization method and the finite element method in one program. It is fully interactive, graphics oriented and user friendly. The geometric models and other data are created and manipulated uniformly. The objective function in this project is the total value of a structural design based on multi-objectives, instead of a single design characteristic only, such as the volume.

(2) The system can bypass the expert system and be used flexibly just as a versatile design and analysis tool.

(3) The objective function, constraint functions and design variables are all set up by typing and positioning the crosshair at the CRT screen, and are converted into optimization routines by an interpreter and a set of system functions. The system functions make it possible and convenient to set up complicated objective and constraint functions on line. The design variables can be intelligently induced by the expert system. The introduction of system functions and variable induction is a significant step towards automatic and interactive optimization.

(4) Inside the optimization algorithm, all of the design variables are scaled between 10 to 20 in order to avoid numerical troubles. The optimization procedure can be visualized from the optimization topography at the CRT screen. A distinct feature is to allow the designer to scout any design point and restart a new search from any point at any time.

(5) A finite element mesh generator has been devised, which requires a minimum interface with the user and can adapt or regenerate the mesh during the optimization search, based on aspect ratios and stress differences.

3. Contributions to combining optimization and the FEM

Four new algorithms have been implemented, i.e. the

global direction method, the safe-fail line method, the quadratic method and the differential method, which can be integrated (together with two other algorithms, i.e. the substructural method and the skipping method) and adapted according to different circumstances. Some of these algorithms can be classified as machine learning from memory or experience. The computer time saved by means of these algorithms is significant.

4. Other contributions

(1) The system can deal with incomplete information. For example, it can supplement the missing material properties and induce the unspecified configuration dimensions.

(2) The bar chart is used to make more convenient the entry of importance values of the performance characteristics. It makes possible the use of intuition and comparisons. Since default importance values are available from the system, only the necessary modifications are required.

(3) The system explains its reasoning procedure by displaying the relevant importance value bar chart and desirability value bar chart, also in an intuitive and comparative fashion. Different candidates can be compared in a similar way.

(4) The system can automatically decompose a structure in a configuration class into two parts: the trunk

and the branches, in order to manage them in computer memory economically. The configuration recommended by the expert system from the database can be modified interactively by the designer for a specific application.

7.2 RECOMMENDATIONS FOR FURTHER RESEARCH

While this project unfolds a valuable perspective of a versatile system for automatic optimized design, it is impossible for a single project to pursue deeply all of the fields that are widely covered by this topic. There are a number of worthy and interesting issues left for further studies. We point out some of them as follows.

1. The quality and accuracy of the desirability values derived by linear programming, and machine learning should be further improved by making better use of the adapted sample.

2. Some other reference engines might be incorporated into the expert system in order to make it more flexible and subtle.

3. Some approaches in the study of pattern recognition may be introduced into the field of expert systems. For example, "feature extraction" can be adopted to increase the search speed.

4. Some algorithms for increasing the efficiency of FEM based optimization (e.g. global direction method) need to be improved. A lot of previous studies in this topic,

scattered in the literature, should be reviewed thoroughly in order to establish a unified approach in this important issue.

5. More studies should be done on the development of system functions and variable induction in order to achieve more fully automatic optimization.

6. A more automatic and versatile FEM mesh generator and adjustor should be developed.

7. Much effort can be put into polishing the interface between the system and the user.

This study gives promise that it is possible to develop in the near future a software system for everyday machine design practice that ideally combines design automation with human judgement and creativity.

REFERENCES

Akhra, G. (1976). An Automatic Node Relabelling Scheme for Minimizing a Matrix or Network Bandwidth, Int. J. Num. Meths. in Eng. 10(10), 787-797.

Arora, J. S. (1976). Survey of Structural Reanalysis Techniques, J. of the Stru. Div. ASCE, pp. 783-802

Arora, J. S. and Baenziger, G. (1986). Use of AI in Design Optimization, Computer Methods in Applied Mechanics to Engineerings, Vol. 54, No. 3/March.

Azarm, S. and Papalambros, P. (1984). . A Case for a Knowledge-Based Active Set Strategy. Trans. ASME, J. of Mechanisms, Transmissions, and Automation in Design, Vol. 106, No. 1, March, pp. 77-81.

Barr, A. and Feigenbaum, E. (Eds.) (1982). The Handbook of Artificial Intelligence, Volume 2. William Kaufman, Inc., Los Altos, CA.

Bennet, J. S. and Engelmore, R. S. (1979). SACON: a Knowledge-Based Consultant for Structural Analysis. IJCAI, Vol. 6, pp. 47-49.

Bennett, J. A. and Nelson, M. F. (1979). An Optimization Capability for Automotive Structures, SAE, Oct.

Bennett, J. A. (1982). Optimization in Structural Design, in Modern Automated Structural Analysis, Kamal, M. M., and Wolf, J. A., Jr., (eds.), Van Nostrand Reinhold, New York, 1982.

Bethe, K. (1982). Finite Element Procedures in Engineering Analysis, Prentice-Hall

Boose, J. H. (1986). ETS -- A System for the Transfer of Human Expertise, Knowledge Based Problem Solvings.

Boubez, T. I. and Funnell, W. R. J. (1986). Mesh Generation for Computational Analysis, Computer-aided Eng. J. Oct.

Boyle, C. D. B. (1985). Acquisition of Control and Domain Knowledge by Watching in a Blackboard Environment, in the book of 'Expert System Proceedings of the Fifth Technical Conference of the British Computer Society

Specialist Group on Expert System', Cambridge University Press

Bramer, M. A. (1984). Research and Development in Expert Systems, Cambridge University Press.

Bundy, A., Byrd, L., Lager, C., Mellish, C., and Palmer, M. (1979). Solving Mechanics Problems Using Meta-Level Inference. IJCAI, Vol. 6, PP. 1021-1027.

Carcaillet, R., Dulikravich, G. S. and Kennon, S. R. (1986). Generation of Solution-Adaptive Computational Grid Using Optimization, Computer Methods in Applied Mechanics and Engineering, Vol. 57 No. 3 1986

Cook, R. D. and Tantichaiboriboon, V. (1977). Application of Optimization and Finite Elements to a Cracking Problem in Timber, Int. J. Num. Meths. in Eng. 11(4), 756-758.

Corser, T. and Seireg, A. A. (1985). Optimizing a Design for Production, Inspection and Operation, CIME, Sept. pp. 18-27.

Crawford, T. M. and Marton, V. (1987). A Machine Learning Approach to Expert System for Fault Diagnosis in Complex Equipement, Computer-Aided Engineering J. Feb.

Cuthill, E. H. and Mckee, J. M. (1969). Reducing the Bandwidth of Sparse Symmetric Matrices, Proc. 24th Nat. Conf. ACM, Barndon Systems Press, NJ, pp. 157-172

Day, A. C. (1972). Fortran Techniques with Special Reference to Non-numerical Applications, University Press

Dixon, J. R. and Simmons, M. K. (1983). Computers That Design: Expert Systems for Mechanical Engineers, CIME, 2(3), 10-18

Dixon, J. R. and Simmons, M. K. (1985). Expert Systems for Mechanical Design: a Program of Research, ASME 85-DET-78

Durocher, L. L. (1979). A Versatile Two-Dimensional Mesh Generator with Automatic Bandwidth Reduction, Computer and Structures 10, pp. 561-575

Duta, R. O. and Gaschnig, J. G. (1981). Knowledge-Based Expert Systems Come of Age, BYTE, sept. 1981 Vol. 6.

No. 9 pp. 238-278.

Farag, M. M. (1979). Materials and Process Selection in Engineering, Applied Science Publisher

Flinn, R. A. (1981). Engineering Material and Their Applications, Houghton Mifflin

Foley, J. D. and Dam, A. V. (1982). Fundamentals of Interactive Computer Graphics, Addison-Wesley.

Forsyth, R. (1984). Expert Systems, Principles and Case Studies, Chapman and Hall Computing

Fox, R. L. and Miura, H. (1971). An Approximate Analysis Technique for Design Calculations, AIAA, pp. 177-179

Frey, P. W. (1986). A Bit-Mapped Classifier, BYTE, Nov., Vol. 11, No. 12.

Fulton, R. E. (1987). A Framework for Innovation, CIME, March

Fung, Y. C. (1965). Foundations of Solid Mechanics, PRENTICE HALL

General Motors Advertisement (1983). The Optimum Shape, Mechanical Engineering, Aug. p. 14

Gibbs, N. E., Poole, W. G. and Stockmeyer, P. K. (1976). An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix, SIAM J. Num. Anal., 13, pp. 236-250

Gloudeman, J. F. (1987). Managing Engineering Analysis Data, CIME, March

Goodall, A. (1985). The Guide to Expert System, Learned Information, Oxford and New Jersey.

Grandhi, R. V., Thareja, R. and Haftka, R. T. (1985). NEWSUMT-A: A General Purpose Program for Constrained Optimization Using Constraint Approximation, Journals of Mechanisms, Transmissions, and Automations in Design, April.

Greenberg, M. R. (1980). A Knowledge-Based System for Digital Electronics. AAAI, 1, pp. 283-285.

Haber, R. B., Shephard, M. S., Abel, J. F. (1981). A General Two Dimensional Finite Element Preprocessor Utilizing Discrete Transfinite Mappings, Int. J.

Numer, Methods Eng. 17, 1015-1044

Hanley, D. P. (1980). Introduction to the Selection of Engineering Materials, New York: Van Nostrand.

Hayes-Roth, F., Waterman, D. and Lenat, D. (1983). Building Expert Systems. Addison-Wesley, Reading, Mass.

Himmelblan, D. M. (1972). Applied Nonlinear Programing, McGraw-Hill

Hinton, E. and Owen, D. R. J. (1977). Finite Element Programming, Academic Press

Hornbuckle, J. C., Neville, G. E., and Boykin, W. H. (1975). Structural Optimization Using the Finite Element Method Applied to a Beam, Int. J. Num. Methods in Eng., Vol. 9, No. 1, pp. 101-107.

Horowitz, E. and Sahni, S. (1976). Fundamentals of Data Structure, Computer Science Press

Jiang, B. N. and Carey, G. F. (1987). Adaptive Refinement for Least-Squares Finite Elements with Element-by-Element Conjugate Gradient Solution, Int. J. Numer, Methods Eng. March

Jones, J. E. and Turpin, W. (1986). Developing an Expert System for Engineering, CIME, Nov.

Juvinall, R. C. (1983). Fundamentals of Machine Component Design, New York: Wiley

Kamal, M. M. (1982). Modern Automative Structural Analysis, Van Nostrand Reinhold

Kaufman, J. G. (1985). Meeting a Critical Need - the Materials Property Data Network, Mechanical Engineering, Nov. pp. 38-42

Kavlie, D. and Powell, G. H. (1971). Efficient Reanalysis of Modified Structures, J. of the Stru. Div. ASCE, pp. 377-392

Kela, A., Perucchio, R. and Voelcker, H. (1986). Toward Automatic Finite Element Analysis, CIME, July.

Kelly, D. W., Gago, J. P. DE S. and Zienkiewicz, O. C. (1983). A Posteriori Error Analysis and Adaptive Processes in the Finite Element Method: Part I, Int.

J. for Num. Meth. in Eng. Vol. 19 1593-1619.

Kirsch, U. and Rubinstein, M. F. (1972). Structural Reanalysis by Iteration. Computers and Structures, pp. 497-510

Kirsch, U. and Hofman, B. (1981). Approximate Behavior Models for Optimum Structural Design, The 11th ONR Naval Structure Mechanics Symp.

Ko, F. and Wang, B. P. (1985). Optimum Design of Practical Truss Structures Using Linear Program, Proceedings CAD/CAM, Robotics and Automation International Conference, Feb. 13-15. Tucson, Arizona.

Kolokouris, A. T. (1986). Machine Learning, BYTE, Nov. Vol. 11, No. 12

Kowalik, J. (1986). Knowledge Based Problem Solving, Prentice-Hall, Englewood Cliff, New Jersey.

Lambourne, E. B. (1986). Towards Integration of Computer-Aided Design, Manufacture and Production Management, Computer-aided Eng. J. Dec.

Lee, S. J. and Kapoor, S. G. (1987). On the Automatic Selection of Reanalysis Techniques in Machine Tool Structural Element Optimization, Engineering Optimization, Vol. 10, No. 3

Lenat, D. B., Sutherland, W. R., and Gibbons, J. (1982). Heuristic Search for New Microcircuit Structures: an Application of Artificial Intelligence. AI Magazine, Vol. 3, pp. 17-33.

Li, H. L. and Papalambros, P. (1985). A Production System for Use of Global Optimization Knowledge, J. of Mechanisms, Transmissions, and Automation in Design, Vol. 107 No. 2 June.

Luby, S. C., Dixon, J. R. and Simmons, M. K. (1986). Creating and Using a Features Data Base, CIME, Nov.

Maher, M. L., Sriram, D. and Fenres, S. J. (1984). Tools and Techniques for Knowledge Based Expert System for Engineering Design, Advances in Engng. Software

Maier, G., Zavelani-Rossi, A. and Benedetti, D. (1972). A Finite Element Approach to Optimal Design of Plastic Structures in Plain Stress, Int. J. Num. Methods in Eng., Vol. 4, No. 4, pp. 455-473.

Manuel, T. and Rand, M. B. (1985). Has AI's time come at last? Electronics Week, Feb. 4 pp. 51

Mayne, A. and Wood, M. B. (1983). Introducing Relational Database, Manchester: NCC

McCorduck, P. (1979). Machines Who Think. W. H. Freeman and Company, New York.

Mcdermott, J. (1980). A Rule Based Configurer of Computer Systems, R1 Technical Dept. CMU-CS-80-119. Department of Computer Science. Carnegie-Mellon University, Pittsburgh, Pa.

Michaelsen, R. H., Michie, D. and Boulanger, A. (1985). The Technology of Expert Systems, BYTE, April Vol. 10 no. 4, pp. 303

Michalski R. S. et al (1983). Machine Learning, an Artificial Intelligence Approach Tioga Publishing Company

Michalski R. S. et al (1986). Machine Learning, an Artificial Intelligence Approach II Morgan Kaufmann Publishers, Inc.

Michie, D. (1982). Introductory Readings in Expert Systems New York: Gordon and breach Science Publishers.

Mitchell, T. M., et al. (1983). An Intelligent Aid for Circuit Redesign. Proceedings, National Conference on Artificial Intelligence, Washington, D. C., Aug. (Published by William Kaufman)

Morris, A. J. (1982). Foundations of Structural Optimization: A Unified Approach, New York: Wiley.

Naylor, C. (1983). Building Your Own Expert System, Sigma

Phansalkar, R. S. (1974). Matrix Iteration Method for Structural Reanalysis, Computers and Structures, pp. 779-800

Pickett, R. M., Rubinstein, M. F. and Nelson, R. B. (1973). Automated Structural Synthesis Using a Reduced Number of Design Coordinates, AIAA, pp. 489-494

Preiss, K. (1983). Solving CAD/CAM problem by heuristic programming, CIME, Sept. pp. 56-60

Queau, J. P. and Trompette, P. (1980). Two-Dimensional Shape Optimal Design by the Finite Element Method, Int. J. Num. Meths. in Eng. 15(7), 1603-1612.

Reklaitis, G. V., Rarindran, A. and Ragsdell, K. M. (1983). Engineering Optimization, Methods and Applications, Wiley-Interscience

Requicha, A. A. G and Voelcker, H. B. (1983). Solid modelings: Current Status and Research Directions, Vol. 3, No. 7, pp. 25-37, Oct.

Revelli, V. D. (1985). Getting CAD and FE to cooperate, CIME, Sept. pp. 39-44.

Roswnmcm, M. A. (1986). Expert Systems Applications in Computer-Aided Design, CAD, Dec.

Rusnock, P. S. and R. J. Cipra (1985). Finite Element Mesh Generation of Planar Surfaces Using an Interactive Node point Placement Technique, ASME Mechanisms Transmissions and Automation in Designs, Sept.

Sabourin, R. (1986). GBASE -- a General Database Management System for CAD, Computer-aid Eng. J. Oct.

Schiemeier, J. and Taylor, B. (1987). An Alternate Algorithm for FEA, CIME, Jan.

Schmit, L. A. and Farshi, B. (1974). Some Approximation Concepts for Structural Synthesis, AIAA, pp. 692-699

Schrodt, P. A. (1986). Predicting International Events, BYTE, Nov. Vol. 11, No. 12

Shephard, M. S. and Yerry, M. A. (1986). Toward Automated Finite Element Modeling for the Unification of Engineering Design and analysis, Finite Element in Analysis and Design, Vol. 2 April.

Shigley, J. E. (1983). Mechanical Engineering Design, McGraw-Hill

Shirai, Y. and Tsujii J. (1985). Artificial Intelligence, Concepts, Techniques and Applications, A Wiley-Interscience Publication.

Shortliffe, E. H. (1976). Computer-Based Medical Consultation: MYCIN, New York: American Elsevier.

Siddall, J. N. (1982). Optimal Engineering Design, Principle and Application, New York: M. Dekker

Siddall, J. N. (1984). The Application of Artificial Intelligence to Engineering Design.

Silva, B. M. E., Wang, H. S., Lee, S. J., Ragsdell, K. M. (1985). Modern Computational Techniques for Large Truss Optimization Problem, Proceedings CAD/CAM, Robotics and Automation International Conference, Feb., Tucson, Arizona.

Spencer, R. H. (1985). Interactive Software Design, the Human Touch, CIME, sept. pp. 45-48

Tan, S. T. and Yuen, M. M. F. (1986). Integrating Solid Modelling with Finite Element Analysis, Computer-aided Engineering J. Aug.

Tang, R., Ma, D. and Lu, L. (1987). Some Strategies Related to the Development of a CAD/CAM System for Mechanical Products Based on Solid Modeling Techniques, Computer in Industry, pp. 137-144

Terheyden, A. G. R. V. and Chalcraft, D. A. (1987). Combining Inductive and Deductive Reasoning, Computer-aid Engineering J. Feb.

Thevendran, V. and Thambiratham, D. P. (1986). Minimum Weight Design of Cylindrical Water Tank, Int. J. Numer, Methods Eng. Sept.

Thompson, B. and Thompson, W. (1986). Finding Rules in Data, BYTE, Nov. Vol. 11, No. 12.

Timoshenko, S. P. (1972). Mechanics of Materials, Van Nostrand Reinhold Co.

Tou, J. T. and Gonzalez, R. C. (1974). Pattern Recognition Principles, Addison-Wesley Publishing Co.

Ugural, A. C. and Fenster, S. K. (1975). Advanced Strength and Applied Elasticity, American Elsevier Pub. Co.

Waterman, D. A. and Peterson, M. (1981). Models of Legal Decision-Making, Rand Report R-2717-ICJ. Rand Corp., Santa Monica, Calif.

Weiss, S. M. and Kulikowski C. A. (1984). A Practical Guide to Designing Expert Systems, N. J.: Rowman & Allanheld.

Winston, P. H. and Brown, R. H. (Eds.) (1979). Artificial Intelligence. An MIT perspective. Vol. 1: Expert problem solving, Natural Language Understanding, Intelligent Computer Coaches, Representation and Learning. MIT Press, Cambridge, Mass.

Winston, P. H. and Horn, B. K. P. (1981). LISP, Addison-Wesley Reading, Mass.

William, J. (1981). System Aids in Constructing Consultation Programs, Van Melle.

Wu, M. C. and Lin, C. R. (1985). Automatic Process Planing and Expert Systems, IEEE, pp. 186

Wu, Z. and Siddall, J. N. (1986). A Software System for Structural Optimization Using the Finite Element Method, J. of Mechanisms, Transmissions, and Automation in Design, Sept.

Zienkiewicz, O. C. (1971). The Finite Element Method in Engineering Science, McGraw-Hill

Zienkiewicz, O. C. and Zhu, J. Z. (1987). A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis, Int. J. Numer, Methods Eng. Jan.