3-1-1985

Some Prototype Examples for Expert Systems v.3

K. S. Fu
Purdue University

Follow this and additional works at: https://docs.lib.purdue.edu/ecetr

https://docs.lib.purdue.edu/ecetr/536

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
Some Prototype Examples for Expert Systems

edited by
K.S. Fu

Volume III

TR-EE 85-1
March 1985

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907
# Table of Contents

**Foreward**

**VOLUME 1**

**Part I — Manufacturing**

Chapter 1  
Production Scheduling: A Sub-Aggregate Level Expert Scheduling Module  
*J. G. Maley* .......................................................... 1

Chapter 2  
Expert System for Scheduling  
*D. Ben-Arieh* .......................................................... 19

Chapter 3  
Expert Systems in Quality Control  
*Y. S. Chen* .......................................................... 106

Chapter 4  
Deep Drawing Feasibility Expert System  
*G. Eshel* ................................................................ 167

Chapter 5  
An Expert System For Machine Selection of FMS  
*S. Lan* .................................................................. 237

Chapter 6  
PROLOG EXPERT: A Simple PROLOG based Expert System Framework for Synthesis, the BAGGER Problem as An Example  
*T. Sarjakoski* ......................................................... 270

**VOLUME 2**

**Part II — Robotics**

Chapter 7  
MR1: An Expert System for Configuration of Modular Robots  
*D. Dutta and S. Joshi* ............................................... 282

Chapter 8  
RP – An Expert System for Robot Programming  
*H. Zhang* .............................................................. 317

Chapter 9
Spatial Planner, A Rule-Based System in Robotics
Y. L. Gu.................................................................554

Part III -- Vision

Chapter 10
An Application of Expert System Approach in Detection of Boundaries Between Textures
K. B. Eom and C. Chatterjee ....................................867

Chapter 11
Stripe Pattern Interpreter and Stacker (SPIS): Expert System
H. S. Yang..........................................................477

Part IV -- Management

Chapter 12
An Expert System For New Product Evaluation in Small Companies
Z. Xu and Q. Xue..................................................496

Chapter 13
Expert System for Inventory Models
K. Y. Tam and H. R. Rao ........................................525

VOLUME 3

Part V -- Structural Engineering

Chapter 14
Expert System for Damage Assessment of Existing Structures
X. J. Zhang..........................................................568

Part VI -- Automated Programming

Chapter 15
An Experiment in Parallel Programming Environment:
The Expert Systems Approach
K. Y. Wang..........................................................591

Part VII -- Others
Chapter 16
A Prototype for an Expert System for Morphological Classification of Prehistoric American Pottery
C. Tsatsoulis and K. S. Fu

Chapter 17
Expert System for Contract Bridge Bidding
L. Y. Chang and C. F. Yu

Chapter 18
Air Flight Scheduler Expert System
A. J. Vayda and W. Y. Kim

Chapter 19
Diet Expert System in Hospital
L. Chang and S. J. Lin
FORWARD

This report consists of the nineteen term project reports for the graduate-level course EE695G "Expert Systems and Knowledge Engineering", which was offered for the fall semester of 1984 in the School of Electrical Engineering. The purpose of the term project is to provide each student an opportunity of designing and implementing a prototype expert system. The application area of each of these expert systems was selected by the student(s) working on the projects. This report is published for the purpose of documenting these results for future reference by the students of the above-mentioned course and, possibly, other workers in expert systems.

The nineteen reports are grouped into seven parts based on their application domains. Part I - Manufacturing consists of six reports, and Part II - Robotics contains three. Two reports in each of Part III - Vision and Part IV - Management, and one in each of Part V - Structural Engineering and Part VI - Automatic Programming. The last part, Part VII - Others, consists of four reports with different applications.

I would like to thank Mr. Edward K. Wong for his valuable help in putting the materials together for this report.

K. S. Fu
Instructor, EE695G
February 1985
Lafayette, Indiana
Part V

Structural Engineering
Chapter 14

Expert System for Damage Assessment of Existing Structures

X. J. Zhang
EXPERT SYSTEM FOR DAMAGE ASSESSMENT OF EXISTING STRUCTURES
By X. J. Zhang
School of Civil Engineering
Purdue University

1. INTRODUCTION

The safety of structure is one of the major concerns of structural engineerings. In recent years, although complex structures can be analysis and designed more reliably and more economically than even before, to estimate the reliability and to assess the damage of an existing structure remain a very challenging problem.

The study of damage assessment of existing structures consists of two parts. One is to assess the current safety state and another is to predict future risk. On the other hand, in order to assess the current safety state, it may be need to know the history about the assessing structure, including the information of original design and construction as well as loading history. Therefore, like other complex decision-making process, it is necessary for damage assessment of an existing structure to consider following three sides: past, present and future.

In reality, there exist many uncertainties in considering above three sides. First, there exist discrepancies between the actual structural behavior and its corresponding mathematical representations used in the process of structural analysis and design. Second, it is difficult to predict with great certainty the extraordinary or abnormal loading conditions which may occur.
during the intended lifetime of the structure. Further more, the
demarcations between adjacent damage states are not clearly
defined. For instance, one may have difficulties in distinguishing
the "uncreaked" from "cracked but unyielded" damage states in
actual structures.

Although the damage assessment techniques exist in practice,
the detailed methodology including in the decision-making process
remains as privileged information for a relatively few experts in
the profession. Therefore, development of a rational and sys-
tematic decision-making process to damage assessment problem
would be particularly useful.

In many practical situations, a complex problem can be
divided into a series of simple questions. In 1979, Fu and Yao
[1] suggested that the problem of the damage assessment can be
considered in terms of the theory of pattern recognition. Since
1979, an expert system approach has been developed for a
computer-based damage assessment system [2-6].

Studies relating to the construction of the expert system
basically consists of a knowledge base and an inference machine.
A knowledge base is a storage in a computer, in which useful
knowledge is stored in a stylized form suitable for the inference.
An inference machine is a control process which deduces an
answer from a given problem situation by using the knowledge
stored in the knowledge base.
In order to develop a good expert system, the knowledge collection and representation are very important. A major difficulty in the development of a practical system lies with the effective communication between specialists of expert systems and structural engineering experts. One way is that structural engineers who are interested in expert system learn the fundamentals of knowledge engineering.

The primary aim of this report is to discuss two major parts dealing with knowledge base of an expert system: Uncertainties and Learning by machine-self. In this report attempt is made to define a damage state by normalized fuzzy sets and a learning procedure is suggested.

2. UNCERTAINTY AND FUZZY LOGIC FOR DAMAGE ASSESSMENT

2.1 NORMALIZATION OF FUZZY SETS

In the decision making problems for damage assessment, situations are not always clear and there exist many uncertainties. The fuzzy sets theory and certainty factor can used effectively to deal with these decision making problems. Many attempts have been made and many progresses have been extended in application of fuzzy sets to decision making problems for damage assessment. [7-13]
Nevertheless, questions still remain that need to be explored further. The first and most important question is how to define a damage state of an existing structure. Because of the complexity and uniqueness of each existing structure, it is difficult to define a damage state by one standard. On the other hand, what is the real meaning for a given structure by using some linguistic assessment may be still ambiguous. No wonder that conflicting results are frequently correlated to the clients with conflicting liabilities, when several engineering firms are involved in one damage assessment problem.[14] Therefore, it may be necessary to define a damage state by some standards and to standardize some linguistic expression in damage assessment decision-making process.

In the following, the normalized coordinate of fuzzy sets are suggested.

As shown in Fig.1, the abscissa y is called as a general coordinate system of fuzzy sets which indicate the degree of state and can be expressed as either linguistic or numerical form. In many practical problems, the membership function of fuzzy sets is generally not linear. For example, if events A and B indicate good and bad conditions of a beam separately, and the degree of state is indicated by $\Delta_f/L$, where L is the span of the beam and $\Delta_f$ is the deflection in the middle of span. As some experts suggested, this beam can clearly be classified as (non-fuzzy), being in the "very bad condition" state whenever $y^u >$ some value of $\Delta_f/L$, called upper limit of $\Delta_f/L$. On the other
hand, the beam can be considered as being in the "very good condition" state when $y^1 < \text{lower limit of } \Delta_p/L$.

As an numerical example, the membership function shown in Fig.1 with the following data: $y_1=1$, $y_1=1.625$, $y_2=3.5$, $y_3=6.625$, $y_u=11$. The linear and normalized membership function shown in Fig.2. can be obtained by choosing following translation function:

$$x = \frac{y - y_1}{y_u - y_1}$$

In general case, a general coordinate system of fuzzy sets can be translated to a normalized coordinate system by choosing a suitable translation function:

$$x = F(y^1, y_u, y)$$

In normalized system of fuzzy sets, the normalized coordinate $x$ has a very clear physical meaning, meanwhile a corresponding relationship between numerical and physical meaning is well defined. For example, $x$ may be considered as a degree of damage state when dealing with damage. If we define damage state as five linguistic expressions, say no, slight, moderate, severe and destructive. Saying that $x=0.75$ is equal to say that the structure is considered to be in a "severely damaged" state.

2.1 Fuzzy Relations
The membership function for the intersection of \( n \) fuzzy sets, say \( A_1 \) to \( A_n \), is given as

\[
\mu_{\cap A_i}(x) = \min_{i=1}^{n} \{ \mu_{A_i}(x) \} \tag{3}
\]

On the other hand, the membership function for the union of \( n \) fuzzy sets, say \( A_1 \) to \( A_n \), is as follows:

\[
\mu_{\cup A_i}(x) = \max_{i=1}^{n} \{ \mu_{A_i}(x) \} \tag{4}
\]

In most real-word problems, however, it is difficult to distinguish clearly the intersection of \( n \) fuzzy sets from the union of \( n \) sets. We call such a relation of fuzzy sets as \( F \) relation of \( n \) fuzzy sets. Therefore, investigation of the membership function for \( F \) relation of \( n \) fuzzy sets seems to be more useful. In the following we will show that the value of membership function for \( F \) relation of \( n \) fuzzy sets is between \( \mu_{\cup A_i}(x) \) and \( \mu_{\cap A_i}(x) \).

From the discussion of normalization of fuzzy sets, it has been indicated that in a normalized system of fuzzy sets the relationship between \( x \) and \( \mu(x) \) is well defined and unique. According to this observation, we define the normalized coordinate \( x \) as an effect coefficient of the membership function of \( \mu(x) \).

Effect coefficients of the membership function for the intersection and the union of \( n \) fuzzy sets are easy to extended as:
Now, assume effect coefficient of the membership function for a $F$ relation as:

$$ x_{TA_i} = \frac{n}{\text{Min}_i [x_i]} $$  \hspace{1cm} (5) 

$$ x_{UA_i} = \frac{n}{\text{Max}_i [x_i]} $$  \hspace{1cm} (6) 

Assume,

$$ x_{\text{min}} = \frac{n}{\text{Min}_i [x_i]} $$  \hspace{1cm} (8) 

$$ x_{\text{max}} = \frac{n}{\text{Max}_i [x_i]} $$  \hspace{1cm} (9) 

We can obtain:

$$ x_{F_{A_i}} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \geq \frac{n}{\sum_{i=1}^{n} w_i} x_{\text{min}} = x_{\text{min}} $$  \hspace{1cm} (10) 

Similarly,

$$ x_{F_{A_i}} \leq x_{\text{max}} $$  \hspace{1cm} (11) 

So,

$$ x_{\text{min}} \leq x_{F_{A_i}} \leq x_{\text{max}} $$  \hspace{1cm} (12)
Specially, if keeping the weight coefficient \( w_i^{\text{max}} \) of \( x_{\text{max}} \) equals to 1 and let others equal to 0, we can get

\[
x_{F_{A_i}} = x_{\text{max}}
\]

(13)

Similarly, let \( w_i^{\text{min}} = 1 \) and others = 0,

\[
x_{F_{A_i}} = x_{\text{min}}
\]

(14)

This observation makes it possible to get a fitness solution for expert decision by choosing suitable weight coefficients.

3. LEARNING

3.1 GENERAL REMARK

One of the most important investigations dealing with knowledge base of an expert system is learning. In this report, the following three aspects of learning are considered.

1) Learn New Phenomena and Write New Rules:

During each practical assessment of damage of a building, the machine may ask some questions for new phenomena of damage and search if the phenomena are included in the rule base of the machine. If those are not included in the rule base, the machine write new rules according to those new phenomena. Although the
new rules written by machine may be rather rough, they can be improved continuously through learning.

2) Rule Verification:

In expert system, a complex problem is divided into a series of simple questions. The program often combines several features to determine global character. In designing such a program, it is difficult to know a priori how much weight should be attached to each of the features that is being used. Furthermore, some pieces of the knowledge may be wrong. Therefore, the rules need to be verified and modified.

3) Rule Modification:

Rule modification is included two aspects of work. One is modification of some wrong rules and another is to change the weight coefficients of corresponding elements.

3.2 HOW TO VERIFY AND MODIFY

After one assessment of a building, the machine records automatically the value of damage state for each element. Meanwhile the machine may ask for experts' assessment for final damage state of building. Analysis, verification and modification may be carried out by making a calibration between the two kind of assessments.
Let us assume that the assessment of experts is expressed as a fuzzy value, $v_{ex}$, and the value of $i$th element is expressed as $v_i$.

For example, following Table 1 indicates some records in the computer memory.

Table 1

<table>
<thead>
<tr>
<th>terms</th>
<th>$v_i$</th>
<th>$A_i$</th>
<th>$n_{ai}$</th>
<th>$B_i$</th>
<th>$n_{bi}$</th>
<th>$D_{si}$</th>
<th>$D_{vi}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>elem.1</td>
<td>$v_1$</td>
<td>$A_1$</td>
<td>$n_{a1}$</td>
<td>$B_1$</td>
<td>$n_{b1}$</td>
<td>$D_{s1}$</td>
<td>$D_{v1}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>elem.i</td>
<td>$v_i$</td>
<td>$A_i$</td>
<td>$n_{ai}$</td>
<td>$B_i$</td>
<td>$n_{bi}$</td>
<td>$D_{si}$</td>
<td>$D_{vi}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>elem.m</td>
<td>$v_m$</td>
<td>$A_m$</td>
<td>$n_{am}$</td>
<td>$B_m$</td>
<td>$n_{bm}$</td>
<td>$D_{sm}$</td>
<td>$D_{vm}$</td>
</tr>
</tbody>
</table>

The meanings of symbols are expressed briefly as following:

$v_i$ is the value of damage state of $i$th element in current assessment.

$$n = n_a + n_b \quad (15)$$

where $n$ is the number of assessment of buildings in the record of this table.

$$n_a = n_a^{n-1} + 1 \quad , \quad e_i \geq 0 \quad (16a)$$

$$n_b = n_b^{n-1} \quad , \quad e_i \geq 0 \quad (16b)$$

$$A_i = A_i^{n-1} + e_i \quad , \quad e_i \geq 0 \quad (16c)$$
\[ B_i = B_i^{n-1}, \quad e_i \geq 0 \]  
\[ n_a = n_a^{n-1}, \quad e_i \leq 0 \]  
\[ n_b = n_b^{n-1} + 1, \quad e_i \leq 0 \]  
\[ A_i = A_i^{n-1}, \quad e_i \leq 0 \]  
\[ B_i = B_i^{n-1} + e_i, \quad e_i \leq 0 \]  

Where \( n-1 \) indicates the former record of this table, namely \((n-1)\)th record of assessment for this table.

\[ e_i = v_i - v_i \]  
\[ D_s = \frac{A_i - B_i}{n} \]  
\[ D_v = \frac{A_i + B_i}{n} \]

\( D_s \) and \( D_v \) are define as degree of dispersity and degree of deviation separately. \( D_s \) may be considered as a measure of the accuracy for the rules applied to ith element, and \( D_v \) as a measure of the certainty factor for ith element.

Two ways of modification of the knowledge base may be applied. One is to modify the corresponding rules, when \( D_v \) is to be found greater than critical value. Another is to change the weight coefficients. Those elements that appear to be good predictors of overall success (both \( D_s \) and \( D_v \) are small) will have their weights increased. For such elements that \( D_s \) are to be found greater than some critical values their weights will be decreased.
4. CES-BABY

CES-BABY is an expert system for damage assessment of existing structures. As regards its artificial intelligence, this machine is only a baby. However a baby will grow up and may become a "real expert" through its continuously learning.

4.1 CONSTRUCTION OF CES-BABY

Figure 3 shows the program construction of CES-BABY which basically consists of four parts: inference machine, knowledge base, memory and learning machine.

Useful knowledge for the inference purpose is collected under the organization as shown in Fig. 4. In most engineering problems, engineers prefer using tables and curves for their simplicities and clearness. One table may involve many pieces of knowledge which may be represented by a lot of "IF-THEN" rules. In this report, most of knowledge are represented by using tables.

The memory stores mainly two kinds of information: Information for current assessment and Historical record of assessments. Such as, name of structure, structural material, height or number of stories, areas of floors, shapes, soil condition and foundation, age of the building, building use, design parameters, existence of walls, etc. are stored as reference data of general information for current assessment. Data collected from the inspection and testing of the structure, such as, the size, number, and location of
cracks, the time history of recorded ground motions and structural response in the form of accelerograms, etc. are also stored as information for current assessment.

Historical records consist of two parts: historical records for each structure and historical records for the work of the expert system machine. The former will be used for information of next assessment for same structure. The late is design specially for the purpose of learning.

The learning machine mainly does following three work: learn new knowledge, verify knowledge base and modify knowledge base. Details of learning procedure have been described in section 3.

The inference machine has several methods. Both forward and backward chaining can be used for problem solving according to the different requirement.

4.2 HEURISTIC PROCEDURE FOR MAKING DECISION

In any assessment process the more information we have the more confident decision we make. However, some pieces of information may be difficulty to get or expensive to obtain. For example, some hidden damages are more difficulty to get and to obtain some test data may be very expensive. Therefore to design heuristic procedure for making decision is very important not only for procedure of problem solving but also for practical and economical reasons.
Yao, etc. suggested that a solution may be possible using an iterative procedure as shown schematically in Fig. 5 [15]. As engineers obtain relevant information and test data, analysis and evaluation are performed. If and when results are sufficient for determining the structural condition, the process is complete. Otherwise, more inspection information and test data must be collected for further analysis and evaluation. The process is repeated until the structural condition is assessed with some degree of confidence.

Fig. 6 shows the heuristic procedure of CES-BABY. The order of questions is designed according to the degree of difficulties for obtaining information. At the beginning of asking each category of questions, the machine may first ask following question: Could you give me some information about " *** " category of questions? If user answers YES, the machine will ask this kind of questions. Otherwise, the machine will ask if the user can give other information about next category. Meanwhile, the machine put " *** " category of questions in the tail of the catalogue of questions. If the results are still not sufficient for determining the structural condition when having asked all categories of questions, the machine may suggest user to collect more information which is involved in unanswered categories of questions.

In such a way, the questions to be asked may be more reasonable by adjustment through man-machine dialogue.
4.3 PROGRAM

The program of CES-BABY is written by LISP language, which is currently working on the Purdure CB Unix machine.

5. CONCLUSION

An expert system, CES-BABY, is developed for damage assessment of existing structures. Fuzzy sets theory and certainty factor are used for damage assessment. A method of normalization of fuzzy sets is suggested in this report. In normalized system of fuzzy sets, the normalized coordinate \( x \) has a very clear physical meaning and an unique relation to the membership function. Therefore it is convenient and effective to use normalized system in damage assessment.

A formula for calculating the membership function of \( F \) relation of \( n \) fuzzy sets has been presented. This formula makes it possible to get a solution for expert decision by choosing suitable weight coefficients.

A learning procedure suggested in this report mainly consists of three aspects: learn new knowledge, verify knowledge base and modify knowledge base.

The main purpose of this report is a communication with specialists of expert system and structural engineering experts, and seeking guides of them. It is to be wished that CES-BABY will be
brought up to be a "real expert" in a common effort of both knowledge engineering experts and structural engineering experts.

REFERENCES


**Fig. 1. Membership function in a General coordinate System**

**Fig. 2. Membership Function in a Normalized System**
Fig. 3 Construction of program
Fig 4. Inference Network of Damage Assessment
Fig. 5. Interactive process of structural condition evaluation and interpretation (Following Yao et al. 1984)
Fig. 6. Heuristic Procedure for Making Decision
Part VI

Automated Programming
Chapter 15

An Experiment in Parallel Programming Environment:

The Expert Systems Approach

K. Y. Wang
1. Introduction

1.1 Motivations.

Computing systems are getting faster and more powerful everyday. Programming on these machines, so called supercomputers or parallel computers, is also getting more and more complicated. To take the full advantage of these architectures, users are having to learn special machine dependent features or language primitives, system calls and other tricks. It is also the programmers' responsibility to handle all the synchronization of the concurrent programming.

Different hardware architectures have different views of parallelism. Programs are usually tailored to some non-portable forms in order to explore the parallelism presented by the hardware. Programs developed for one kind of architectures may not be able to run or may be terribly inefficient on other kinds of architectures. Most existing software packages are required to be totally rewritten before they can be used on these new computers.

One promising solution of the problem is to write programs in ordinary sequential languages (for example: FORTRAN), and perform some kind of transformations or program restructuring techniques to transform the program into forms that suit each individual target machines. However, identifying regions of code which can be executed in parallel with minimum synchronization and restructuring of the program to get the best performance on the target machine requires an intimate knowledge of both the program and the underlying hardware on which it is to run. Therefore, it is very desirable to have a convenient programming environment that contains this knowledge and be able to offer some clever assistances to the programmer.

Automatic transformation and parallelism acquiring is still a new research topic, and will not, in the foreseeable future, free the human programmer from thinking about parallelism. However, some optimization and program restructuring techniques have been developed, and powers of these techniques have been demonstrated. Unfortunately, none of the existing systems have successfully integrated this optimization knowledge, special machine primitives and programming requirements together to form a useful programming environment yet.
Languages

Cray Fortran  Applicative Languages

Hardware Features

Vector?

Pipelined?

Multiprocessors?

Array stored in memory?

Array stored in registers?

Memory cycle time?

Vector startup time?

Restructuring Techniques

Reordering?

Forward substituting?

Node splitting?

Loop distribution?

Renaming?

Loop blocking?

Recurrence?

"Who on the earth invented these stuff?"

Our goal is to build a user friendly programming environment that will, when given a sequential source program and the name of the target machine, query the user for necessary information, choose and perform the program restructuring transformations, and give the necessary explanations. It will also evaluate results of the transformations, explore possible parallelism from the program, and produce an equivalent high-level language program that most suits the specified target machine.
1.2 Expert System Implementation.

Restructuring techniques are usually expensive to apply. Deciding when, how, and whether it is worth to apply these transformations is a complicated task. It usually requires extensive testing and some state-of-art decisions. The situation is even worse when we intend to include different kinds of target machines in the system. Different hardware architectures have different views of parallelism, so different criteria and applying sequences of the restructuring are needed for different architectures.

For example, to optimize a program with nested loops on a vector computer, we may move the loop with longest loop bound to the inner most, then we can vectorize the inner most loop and execute other loops sequentially. However, for multi-processors computer, the loop order that allows the outer most loop to match the number of available processors and executes the inner loops on each processors may be the best order. If the computer has both multi-processors and vector instructions then some other criteria are needed to decide the best order of the loops.

All these problems make the implementation of the programming environment a difficult job. We are hoping that by applying expert systems techniques along with human experts' expertise, we can simplify the decision task and give users more flexibility in controlling the restructuring process.

Actually, to evaluate the gains of introducing expert systems and artificial intelligence techniques to parallel compiler restructuring and user friendly programming environments is one of the primary goals of this project.

Our experiences do lead us to believe that by employing A.I. techniques and heuristic knowledges we can rule out some unnecessary, expensive restructuring tests, thus simplify the decision tasks and improve the overall performance of the system. Also, the power of the system can easily be extended by adding new knowledge about parallel programming or new hardware to the knowledge base as soon as they are available.
Parallel Programming Environment Makes Programmers’ Life Easy!!

1.3 Task Domain.

The process of the program restructuring is a repeat procedure of recognizing features or patterns of the program that allow or disallow certain transformations, and basing on these observations and the machine features to choose appropriate transformations that can improve the parallelism in view of the target machine. Thus, the effectiveness of the restructuring process heavily relies on the the quality of knowledge of both the program and the machine.

The features of the target machine are provided by the machine feature database. These features decide the views of parallelism. Therefore, good machine feature classification criteria is very important. But, from the restructuring point of view, once the criteria is chosen, the machine features are simply a list of facts that remains unchanged throughout the process. No inference procedures are needed to acquire more knowledge about machines. However, knowledge about the program is usually implicit to the system and sometimes recognizing some facts that are trivial for the human being requires complicated test for the machine.

Furthermore, the restructuring transformation changes the features of the program. Different ordering of transformations may leads to different results. There are certain cases that it
is very hard to tell which transformation is the best to apply. So, heuristic knowledge plays certain role in choosing the transformations.

The main factor that affects the program features is the data dependence. Data dependence relation dominates the execution order of the program, and is the main obstacle for parallel execution. In many cases, reducing the number of dependences leads to direct reductions in a program's running time.

A nature way to represent the program structure and the dependence relations is the dependence graph. The dependence graph is a directed graph whose nodes represent program components, and whose arcs are the dependence relations between nodes. The program component can be an assignment statement, a for loop header, an expression, or other program constructs.

There are four types of dependence relations: loop dependence, flow dependence, anti-dependence and output dependence.

Definition A program component $C$ (either an assignment statement or a loop header) is said to be loop dependent on a loop header $L$, denoted $C \rightarrow L$, if $C$ is embedded in the loop statement whose header is $L$.

Definition Consider two, not necessary distinct components $S$ and $S'$ and one instance of each, $S(i)$ and $S(j)$, such that $S(i)$ is executed before $S(j)$ in the proper serious execution of the program. We say that:

1. $S(j)$ is flow dependent on $S(i)$, denoted $S(i) \rightarrow S(j)$ iff an output of $S(i)$ is consumed by $S(j)$.
2. $S(j)$ is anti dependent on $S(i)$, denoted $S(i) \rightarrow S(j)$ iff $S(j)$ overwritten a value of a variable after that value is used by $S(i)$.
3. $S(j)$ is output dependent on $S(i)$, denoted $S(i) \rightarrow S(j)$ iff the value computed by $S(i)$ is overwritten by $S(j)$.

Most transformation techniques improve the parallelism of the program by removing or adjusting the dependence relations. Fourteen restructuring techniques are included in the system (not all of them have been implemented at this point), we now proceed to explain them one by one.
Statement Reordering: Statement reordering reorders the statements. If the dependence graph is acyclic or only has single statement self-cycle, property reordering the statements will allow the loop to be vectorizable. Other transformation often needs to reorder the statements in order to collect the statements that are involved in a group of dependence relations.

Renaming: The same memory location used for different purposes at different points could impose unnecessary sequentially constraints on parallel programs. The renaming transformation assigns different names to different uses of the same variable in order to remove unnecessary output and anti-dependence.

Node Splitting: When the dependence graph has cycle, node splitting try to break the cycle by reposition the anti-dependence arcs. This is achieved through breaking an assignment statement into two or more assignment statements.

Expansion: Expansion takes scalar variables that were used in a for loop and expands them into arrays which have one element per iteration of the loop. This process reduces the number of output and anti-arcs that associated with the scalars.

Forward Substitution: Forward substitution eliminates flow dependence arcs from the graph by substituting the right-hand-side expression of an assignment statement into the right-hand-sides of other assignment statements.

Deadcode Elimination: After some transformations have been performed, some code are 'dead', because they are not subsequently used or can't be reached. These code have no effects on the results of the program so can be eliminated.

Loop Distribution: Loop header are replicated and distributed around groups of statements in the loop body. Loop distribution abstracts dependence graphs by finding and merging each strongly connected component in the body of a loop along with the loop header into a component node. (A strongly connected component is a maximal set of nodes such that for any pair of the nodes in the set, there is a path between them.)

Vector Scalarizing: Vector scalarizing is the reverse operation of the scalar expansion, it change a vector into a scalar in order to use less registers or memory space.

If Pattern Matching: To recognizes conditional statements that are equivalent to intrinsic functions such as Max or Min and translates the conditionals into calls to these functions.

If Removal: One way to execute the conditional is to evaluate the conditional in parallel for all iterations of the loop and assign the results to a bit vector called mode vector.
Then the mode vector is used to mask all of the statements in the scope of the conditional. If removal translates the conditionals that were not recognized by if pattern matching into mode vector assignments and masks.

Boolean Recurrence Translation: If a mode vector assignment forms a Boolean linear recurrence, the boolean recurrence translation translates it into calls to a linear system subroutine that solves the recurrence.

Loop Jamming: The loop jamming is the inverse operation of loop distribution. It merges the bodies of two loops to form a single loop. It is necessary that each loop be executed the same number of times and that the indices variable be the same. If these conditions are not met, we may try some transformations like loop distribution and statement reordering to adjust the loops into forms that can be jammed.

Loop Blocking: Loop blocking split the loops that index arrays into a pair of loops, a block loop and a strip loop. The strip loop is the size of the vector registers or number of processors. The block loop steps through the original iteration set in blocks the size of the strip loop. This is used for speeding up loops running on vector machines that stores operands in registers (like Cray) or scheduling the execution of the loop on multiprocessors computers.

Loop Interchanging: Loop interchanging switches inner and outer loops. It is a very powerful transformation. It has a profound effect on the execution order of a loop and can therefore have a large effect on the performance of a loop. No commercially available compiler or translator performs loop interchanging.

Not all transformations mentioned here are applicable for all programs and target machines. Each transformation has its own restrictions and conditions. The most important criterion for applying transformations is that the transformation must preserve the original meaning of the program. Therefore, the basic job of the restructurer is to analyze the dependence graph and other informations to recognize the the conditions or patterns that allow or disallow some transformations and then performs the appropriate transformations.

For more detail about the dependence graph and transformation techniques, please see references [6], [7], [12], [14].
2. The System Overview.

2.1 System Organization.

The system is built on top of the Cprolog interpreter which runs on the UNIX† operating system. It should be able to be moved to any system that supports Prolog without any modification.

The programming environment consists of a parser, a dependence graph analyzer, a machine features data base, a program restructurer, and high-level source code or low level machine code generators.

The parser and high-level source code generator are language dependent and the machine code generators are machine dependent.

The parser parses the input source program and generates the parse tree and symbol table. The dependence graph analyzer analyzes the parse tree and computes the dependence graph of the program.

We classified the features of different machine architectures into categories. For each target machine, there is a list of features in the data base that will be used as the knowledge about the target machine. All decisions about transformations are made based on the features of the architecture, such as the number of processors or having vector instructions, not any particular commercial machine. So the restructurer is independent of the hardware, although it is able to handle many different architectures.

The machine features data base contains informations about the target machines. When given the target machine name, it returns a list of features that are used to characterize the specified target machine.

Adding new target machines to the system is easy, since we can simply include its feature list to the machine features data base.

The main part of the system is the program restructurer. Its an expert system that performs the jobs of choosing the appropriate focuses, applying tests and transformations, giving necessary explanations and evaluating the results.

†UNIX is a Trademark of Bell Laboratories.
Figure 1: The System Configuration.
The restructurer takes the initial knowledge about the source program and target machine (parse tree, symbol table, dependence graph and machine features, etc.) as the initial evidence, and it generates new parse tree and dependence graph that have been restructured as outputs. We will discuss more about the restructurer in the next section.

Based on the resulting dependence graph and other informations generated by the previous phrases, the source code generator generates the program that is identical to the input program.

Since our purpose is to build a source to source program restructuring environment, the system doesn't have to generate the actual machine code. Instead, the system restructures the program a form where the parallelism can be easily recognized by the compiler on the target machines. The output of the system can also be fed back into the restructurer repeatedly until all parallelism has been explored.

Note that if we add the machine code generation phrase for specific machines, the system can be switched into a multi-targets, highly optimized compiler!

2.1.1 User Interface.

The top level of the system is the command mode, in this mode, the users can edit files, reconsult Prolog files, send prolog commands to prolog interpreter, or escape to UNIX shell. They can also parse the source program, compute the dependence graph, and involve the restructurer to produce the parallel code.

The reason that we want to have the extra command level control is that while using the system, the user may need to modify or recompile his programs, change knowledge base (if he is allowed to to so), or execute some system commands. Instead of painfully exiting and restarting the system over and over again, the user can perform these operations in the command mode without exiting the system.

The command mode is the place where the user communicates with the outside world. The system prompts the user with '<COMMAND>', and expecting the user to type in the command, abbreviations of the commands are also accepted. List of available commands are listed in figure 2.
<< Welcome to the Program Restructuring System! >>

COMMAND> : h

Help:
Available commands in command mode.
Characters in brackets are the abbreviations of the commands.

[.] - exit the system
[bye] - exit the system
[consult] - reconsult the prolog file
[edit] - edit file using editor specified in .ressrc
[Edit] - same as edit but edit the file that was last edited
[focus] - list the available focus list
[init] - initialize the system, read in default informations
[level] - select the level to enter
[LD] - clean the screen
[help] - print out this message
[prolog] - send command to prolog interpreter
[restructure] - involve the restructurer
[s]ave - save the current state in a file
[trace] - trace the execution of the named procedure
[T]race - trace all
[f] - escape to UNIX shell

Figure 2: Available comments in command mode.

After the restructurer is invoked, it first asks the user to give the name of input file and target machine. Then it prints out all information about the input and the machine on the screen, and then asks the user to select the degree of explanation that he would like the system to give.

To make interactive easy for users, the display screen was divided into four sections: Actions, Reasons, Recommendations and Information.
The *Actions* section is used to demonstrate the current status of the system. Actions, queries and responses of the user are displayed in this section. It is located on the upper half of the terminal screen.

Right below the *Actions* section is the *Reasons* section, which is used to display the explanations given by the system.

Similarly, the *Recommendations* section shows the recommendations that the system gives.

At the bottom of the screen is the *Information* section. The *Information* section displays the information about the target machine, source program name and the degree of explanation. It remains unchanged during the restructuring process.

The whole screen is refreshed whenever the focus of transformations is changed. The explanations and recommendations sections are cleared when new explanations or recommendations are given. In order to avoid confusion, the explanation and recommendation sections are labeled by rule number.

```markdown
<< Restructurer >>

- [Actions] --------------------------------- Control level: Front end  Focus: Vectorizable --------------------------------- 
  
  (actions and the queries of the system will be displayed here.)

- [Reasons] ---- Rule 1

  (explanations will be shown here.)

- [Recommendations] ---- Rule 1

  (recommendations are given here.)

- [Information]

  Target: Cyber205,  Source: test,  Explanation Degree: 2.
```

Figure 3: The restructurer divides the display on the user's terminal into four display sections.
2.2 The Restructurer.

The program restructurer consists of two parts: the inference machine and the knowledge base.

The knowledge base contains the knowledge about machines and languages, strategy of choosing focuses, criteria of evaluating the results and methods of applying transformations.

The inference machine uses the parse tree, dependence graph and machine features as the initial evidence. It applies knowledge in the knowledge base to guide the flow of reasoning that goes from evidence to conclusions. The conclusions may be used as the evidence of other rules later.

The restructurer may collect additional information by querying the user. For example, when it can not determine the loop bound of a loop without executing the program, it asks the user to give an estimated range.

Figure 4: Diagram of the restructurer.
2.2.2 Knowledge Representations.

There are three kinds of knowledge used in the system, knowledge about the program, knowledge about the target machine, and knowledge about the transformation techniques.

The knowledge about machine is just a list of facts (we called them features). As we mentioned earlier, the criteria of classification of features affect the quality of the system. The way we decided the criteria is that we chose some obvious features first, (for example, number of processors, has vector instructions or not), then we started to build the restructurer by adding new rules to the knowledge base, whenever the existing criteria fails to differ the distinctions between different case, we went back to refine the criteria.

The dependence graph was chosen to represent the knowledge about the properties of the program, because it explicitly represents the data dependence relations of the program and its hierarchy structure is easy to manipulate and modify. Another reason for using dependence graph is that a lot of study about properties and transformations of the dependence graph has been done (see references).

Basically, dependence graph is similar to the semantic nets, it is a directed graph whose nodes represent program components or dependence subgraphs, and whose arcs are the data or control dependence relations between nodes.

Because of its modularity and simplicity, an "IF-THEN" rule structure was chosen to represent the knowledge of analyzing and manipulating dependence graph and the knowledge about transformation techniques. The Prolog built-in matching primitives makes the implementation of the rule-based system more transparent, since each rule corresponding to a prolog clause. However, the rule based system has the disadvantage of inefficiency and opacity. Also, the lexical order of the rules in the knowledge base dominates the execution order of the rules. It is not very easy for the user to alter the control flow.

In order to increase the flexibility and efficiency of the system, we configure the system and knowledge base into three hierarchy layers: control levels, focuses, and rules. By partitioning the knowledge into the hierarchy levels and querying the user whenever the control flow jumps from rules in one group into rules in other groups, the user has very high level of controls. He is free to change the focus or the control level at his will during the restructuring process.
Rules

The rules are knowledge about choosing focuses, restructuring dependence graph of the source program and evaluation of the results.

Some rules are pretty simply. For example, the following rule is used to choose the applicable focuses for multi-processors computers.

Example:

\[
\text{If } \text{MACHINE} = \text{multiprocessors and CONTROL\_LEVEL} = \text{front-end} \\
\text{then applicable focuses are:} \\
\{3 \text{ scalar renaming.} \\
5 \text{ scalar expansion.} \\
8 \text{ loop distribution.} \\
2 \text{ statement reordering.} \\
7 \text{ dead code elimination.} \\
\}.
\]

On the other hand, some rules are fairly complicated. For example, the rule exchange is actually a recursive procedure that generates all the possible loop orders for a nested loops and computes the best one.

Focuses

Focuses are formed by collections of rules that are related to particular restructuring transformations. Deciding which, when and how to apply the focuses, and how to evaluate the results of applying these focuses is the main duty of the expert systems.

The focus is also the basic unit of altering the control flow. During the restructuring process, the user is allowed to select focus or change control level at any point.
Menu:

>> Available transformations at front-end level:
0 <exit>
1 vectorization
2 statement reordering
3 renaming
4 node splitting
5 expansion
6 forward substitution
7 deadcode elimination
8 loop distribution
9 vector scalarizing

>> Available transformations at intermediate level:
0 <exit>
1 if pattern matching
2 if removal
3 Boolean recurrence translation

>> Available transformations at back-end level:
0 <exit>
1 loop distribution
2 loop jamming
3 loop blocking
4 loop interchanging

Figure 5: Available restructuring transformations.
Control Levels

The focuses are grouped into three control level groups according to their function: front-end, intermediate, and rear-end.

The front-end focuses are used to improve the performance of the source program on a general parallel architecture. Most transformations in this group are concentrated on inner loops.

The intermediate focuses deal with recurrence relations and translate conditional statements into function call or mode vectors.

The rear-end focuses consider the loop restructuring and other machine dependent transformations. For example, interchanges the inner and outer loops, distributes the loop header, and merges adjacent loops that have 'similar' loop bound.

2.3 Inference Mechanism.

The Prolog matching and 'infer' primitives are used to build the basic inference mechanism of the system.

For example, the following rule use the Prolog built-in matching mechanism to decide whether the intermediate conclusion nested_loops about the statement ST is true or not.

\[ \text{nested(ST)} : \quad \text{ST} = \text{stm(do\ldots jm}\ldots \text{mstm(do\ldots jm)}\ldots) \]

The advantage of using Prolog built in matching primitives is that the same clause can be used to perform backward chaining and forward chaining depending on which variables are instantiated.

On the other hand, the disadvantage is that when writing the rules, one must bear the Prolog matching principle in mind. In other word, the rules need to incorporate the knowledge about the inference mechanism, which, in our opinion, is the major defect of Prolog.

Conflicts in the knowledge base are resolved by the lexical order of the rules, because the Prolog matching primitive matches the rules according to their order.

Interactivity is an important feature of the system. When the system fail to obtain some necessary information it will try to get helps from the user. For example, if the system fail to estimate the loop bound of a loop, it asks user to give an estimated bound. If the user refuse or unable to give the estimate, a default value will be used.
Based on knowledge in the knowledge base, input program, and the target machine features, the inference machine chooses the current focus, performs the transformations, and evaluates the result. When a transformation is done, the system suggests the next focus to apply and perform the transformation if it is confirmed by the user. The user is allowed to switch focus and control level if he does not agree with the system.

The system is capable of explaining the reasons for all decisions it made. Users can decide whether to accept the result or not, based on their own judgement and the information provided by the system. Users can also choose the degree of the explanations they want the system to give:

- **Degree 0**: This mode is convenient for those people who don't want to learn terms like *If pattern matching* or *Loop jamming* and only care about the result. No explanation will be given by the system, only the result of the transformation will be shown. The system assumes that the user is willing to accept all suggestions given by the system, so it applies the rules without asking for the permission.

- **Degree 1**: The result of the transformation and a short explanation will be given. An one line recommendation is also printed. The user will be asked to confirm the next focus that the system chose.

- **Degree 2**: This mode is useful for novice who wants to look closely how the system performs the restructuring transformations. Reasons will be explained in full detail. Up to four lines of explanations and two lines of recommendations will be given. Also, the user will be informed whenever a new rule is applied.

We viewed the explanation mechanism as an important part of the system. Since only when the user understands what has been done by the system and why they are being done in that way can the user judge if he wants to accept the result of the transformations. The explanation mechanism also makes the system to be a useful teaching tool for training parallel programming experts.

We will use the simple program in figure 6(a) to illustrate the restructuring process.

First, we explain what kind of transformations are needed, then, we go through the restructuring process of the sample program step by step to demonstrate the power of the program restructuring.

In the examples, we try to avoid the uses of formal terms like output dependence to explain the relations, instead, we try to explain the situation in English.

If the target machine is a pipelined vector computer like Cyber 205, our goal is to generate a program with vector instructions that is equivalent to the source program.

A loop is vectorizable if each statement can be executed for all values of the index variable of the loop, before executing any of the statements in the loop following it, and this alternate execution order computes the same results. If a given loop is not vectorizable then we may try transformations to break the data dependence relations that prevent the vectorization.

For the example in figure 6(a), the number of iterations in the inner loop is greater than that of the outer loops, one obvious attempt is to try to vectorize the inner loop and execute the outer loop sequentially. This can be done by distributing the inner loop to each statements then vectorize them. However, a closer look shows that the statement S3 used the value of A[i+1,j+1] which is overwritten in the next iteration by S1. If we vectorize the inner loop directly, as shown in figure 6(b), the loop computes different results, since S1 changes the values of array A before they are used by statement S3. This kind of vectorization is prohibited, because it changes the meaning of the source program.

One way of checking if a loop is vectorizable is by the following procedure mentioned in [2]:

1. Base on the dependence graph we compute the strongly connected components.
2. Any statement that is not in a strongly connected region may be vectorized directly.

Computing the strongly connected components is timely and unnecessary if we are expecting that most input programs can be vectorized directly. The following rule can also be used to check if the loop can be vectorized directly also.
Figure 6:

(a) Original source program, the data dependence of array A between statements S3 and S1 prevents the vectorization of the inner loop.

(b) Example of illegal vectorization, this program computes different results as the program in (a) does.

(c) Program after statement reordering, the data dependence between S1 and S3 still causing problem, but now we are allowed to substitute r.h.s. of S1 into S3.
DO 100 I = 1, 10
   DO 200 J = 1, 1000
   S2:   B[i,j] = R[i,j] / 2
   200    CONTINUE
   100   CONTINUE

(d) Result of Forward Substitution, now, the data dependence from S1 to S3 was eliminated,
but the dependence from S3 to S1 still prevents the vectorization.

DO 100 I = 1, 10
   DO 200 J = 1, 1000
   S2:   B[i,j] = R[i,j] / 2
   200    CONTINUE
   100   CONTINUE

(e) Apply Statement Reordering again, the loop is now vectorizable!

DO 101 I = 1, 10
   DO 201 J = 1, 1000
   201    CONTINUE
   DO 202 J = 1, 1000
   202    CONTINUE
   DO 203 J = 1, 1000
   203    CONTINUE
   101   CONTINUE

(f) Program after distributing the inner loop, each inner loop can be converted into a vector
instruction as in (g).
DO 100 i = 1, 10
S2: \[ C_{i,1:1000} = B_{i,1:1000} \times R_{i,1:1000} + A_{i+1,2:1001} \]
S1: \[ A_{i,1:1000} = B_{i,1:1000} \times R_{i,1:1000} \]
S2: \[ B_{i,1:1000} = R_{i,1:1000} / 2 \]
100 CONTINUE

(g) The resulting vectorized program.

DO 101 i = 1, 10
S3: \[ C_{i,1:1000} = B_{i,1:1000} \times R_{i,1:1000} + A_{i+1,2:1001} \]
101 CONTINUE

DO 201 i = 1, 10
S1: \[ A_{i,1:1000} = B_{i,1:1000} \times R_{i,1:1000} \]
201 CONTINUE

DO 301 i = 1, 10
S2: \[ B_{i,1:1000} = R_{i,1:1000} / 2 \]
301 CONTINUE

(h) Result of distributing the outer loop to each vector statement.

S3: \[ C_{1:10,1:1000} = B_{1:10,1:1000} \times R_{1:10,1:1000} \times A_{2:11,2:1001} \]
S1: \[ A_{1:10,1:1000} = B_{1:10,1:1000} \times R_{1:10,1:1000} \]
S2: \[ B_{1:10,2:1001} = R_{1:10,1:1000} / 2 \]

(i) The vectorized program for target machines that are capable of computing multiple array.

Figure 6. Example of vectorization process for vector computer.
Rule: If a loop $L$ contains statements $S_i$ and $S_j$, where $S_i$ precedes $S_j$ in the loop, and $S_i$ use some data that is computed in previous iterations by $S_j$, or $S_j$ use some data that will be updated by $S_i$ in the following iterations, then the loop $L$ is not vectorizable.

If the loop is not directly vectorizable, we first check if there is any cycle in the dependence graph. If there is no cycle in the dependence graph then Statement Reordering will be able to change the loop into vectorizable. If there is any cycle in the dependence graph then several transformations can be employed to adjust the graph. If the cycle was caused by scalar variables then scalar renaming or scalar expansion is able to break the cycle. If the cycle is caused by array variables (as our example does), then node splitting or forward substitution can break the cycle if the transformation is allowed and does not change the meaning of the computations. If the cycle is broken, after applying the transformation, statement reordering will be able to make the loop vectorizable.

In our example, statement $S_3$ uses value of $A[i,j]$ which are computed by $S_1$ for the same instance of induction variables $i$ and $j$ (we said $S_3$ is flow dependent on $S_1$ in this case). And $S_3$ also uses value of $A[i+1,j+1]$ that is overwritten by $S_1$ in the following iteration of $i$ and $j$ (We called this "$S_1$ is anti-dependent on $S_3$ in the forward direction"). These two data dependence form a cycle in the dependence graph. However, we are not allowed to substitute the right-hand-side of $S_1$ into right-hand-side of $S_3$ directly, because the value of $B[i,j]$, which is on the right hand side of $S_1$, is changed by $S_2$ before it reaches $S_3$. Since statement $S_2$ blocks the way of substitution, we apply statement reordering to move it out of the scope of the cycle in order to allow the substitution. The result of the statement reordering is in figure 6(c).

Now the values that are used in $S_1$ remained unchanged when we execute $S_3$, so we can substitute the $A[i,j]$ in $S_3$ by right-hand-side of $S_1$(which is $B[i,j]*R[i,j]$) without changing the result of the computation. Figure 6(d) shows the result of applying forward substitution.

Although forward substitution has removed the flow dependence arc from $S_1$ to $S_3$, the loop is still not vectorizable. Because the statement $S_3$ is still using the value of $A[i+1,j+1]$ which is overwritten in the next iteration by $S_1$. An easy way to solve the problem is to executed $S_3$ before values of $A$ are changed by $S_1$, so we apply statement reordering again, the resulting loop is in figure 6(e).

Note that each statement in the inner loop of figure 6(e) can be executed for all instance of the induction variables $i$ and $j$ before the execution of statements that are following it in the loop. Therefore, after we apply loop distribution to distribute the inner loop to all statements
(see figure 6(f)), these statements can be converted into vector instructions as in figure 6(g).

We also noted that if the target machine is capable of computing multi-dimensional vectors, we can also distribute the outer loop in figure 6(g) into each vector statements. The resulting program in figure 6(h) is equivalent to the vector code in figure 6(i).

The simulated 'screens' shown in appendix (C) are used to demonstrate the process mentioned above. Each 'screen' shows a short sequence of queries and explanations of the system and the responses of the user. We numbered the actions by integer in order to show the order they are appeared on screen.

The restructuring process for multi-processor computers with lots of processing units is a totally different story. The main implication of this model is that better performance comes from efficient use of processors and minimization of synchronization. So our main concern is how to utilize all available processors and minimize the synchronization.

The natural language structure to examine for parallelism is the DO-loop. A loop body contains a set of computations that are to be done many times with different data indexed by the loop induction variable. So the opportunity for parallelism exists, if we can be sure that the loop contains no data dependence that create synchronization needs which force other processors to idle.

Most compiler on multi-processor machines accept loops like DOALL or FORALL statements. DOALL or FORALL loop is executed by scheduling iterations of the loop on available processors for different values of the induction variable.
(a) If the machine has lots of processors, most processors will be idling during the execution, because only 10 processors are used.

(b) After loop interchange, 1000 processors are used to execute the loop in parallel. However, much more data communication between processors will be needed.

Figure 7. The programs that suit for multi-processor machines. Each loop in the box is scheduled to be executed in a processor for one induction value of j.
A loop can be converted to DOALL loop if there is no communication between iterations. A FORALL loop allows communication between iterations as long as the source of the data dependence relation occurs in the loop lexically before its destination. For multi-processor machines, we will try to convert loops into DOALL or FORALL type of loops.

The loops in figure 6(a) cannot be converted into DOALL or FORALL statements, because the fact that S1 is anti-dependent on S3 but is lexically before S3. (caused by the use of \( A[i+1,j+1] \) in S3). But the program in figure 6(e) are able to be converted to FORALL statements.

For the program in figure 6(e), if we execute the inner loop in parallel (i.e. each processor executes an iteration of the inner loop for different value of j), as in figure 7(a), only 10 processors will be used and other processors will be idling. If we apply loop interchanging to interchange the inner and outer loops, we get the program in figure 7(b). If our target machine has more than one thousand processor units, the program in figure 7(b) get a better utilization of the processors. However, we should note that the loops in figure 7 need some degree of synchronization between processors, because S3 uses values of \( A[i+1,j+1] \) that are overwritten by other iteration of i which is executed in other processors. So the execution of S1 should be delayed until the processor that runs the previous iteration of the loop finishes S3.

If all processors are started at the same time then it is reasonable to expect that they get to the synchronization point (position between S3 and S1) at the same time. For program in figure 7(a), there are 1000 synchronization point between 10 processors, but for program in figure 7(b) there are only 10 synchronization points between 1000 processors. So the execution of program 6(b) in a computer with lots processors will be much faster than that of 6(a).

In general cases, it is not as easy as the example to decide what execution order of the loop is the best order among others. Our approach is as following: first we generate all possible loop orders that preserve the meaning of the original loop, then use some heuristic knowledge to choose the 'best' order. For example, for vector machine, we prefer the order such that the inner most loop has the longest loop bound. For multiprocessors machines that have no vector instruction, the order that move the loop with longest bound to outer most will be the best. If the multi-processor computers has vector instructions or the loops are not perfectly nested then some other criteria may be used. Sometimes, it is really hard to make the decision, so the system breaks the tie arbitrary.

The response time of the system (the time to perform a transformation or between queries) on the VAX 11/780 of our department varies from instant to three seconds during the busy hours. As an interactive system, we judged this delay as acceptable. If in the future, the response time becomes too long we may consider to implement the transformations that has
nothing to do with decision making in some more efficient languages like C.

4. Current status of the system

Apparently, the proposed project is too big for a class project. Groups in other institutions have spent more than a decay in designing parallel programming environment and only a few satisfiable result has been obtained. We feel that the expert system implementation of the programming environment introduced in this report lights up a new way of approaching the problem. In this project, we tried to implement a prototype system that is big enough to demonstrate the power of the system.

Since we have been concentrating on the expert system implementation of the program restructurer, the dependence graph generator, the parser and target code generator parts have not been built yet.

The following transformations have been implemented: vectorization, statement reordering, forward substitution, loop jamming, loop distribution, and loop interchanging.

Since we don't have the dependence graph generator, the user need to provide the dependence graph of the program to the system, which, of course, is very inconvenient.

Currently, CYBER205, CRAY, CRAYII, CRAYXMP, PRINGLE, Cosmic Cube (Intel "personal supercomputer"), and N.Y.U. Ultracomp, have been included in the machine database. Since all the transformation decision are base on the features not on the particular hardware, so new target machine can be added by simply adding the list of their features into the machine database.

5. Future extensions and interesting topics

By adding the machine dependent code generation phrase for specific machines, the system can be changed into a highly optimized compiler.

Since the system employs some A.I. techniques and heuristic knowledges to perform the transformations, it would be interesting to compare the result and efficiency with other existing supercompilers and other programming environments (although there are only a few of them).

The evaluations and improvement of the heuristic knowledge is another interesting topic. By employing software engineering techniques to build a information collector that collects the users' experiences, comments and responses of the system while they are using the system will help us to justify the reliabilities of the rules and gives suggestions to the modification of the knowledge base.
6. Conclusions

The expert system implementation of the programming environment has the following advantages:

- The power of the system can easily be extended by adding new knowledge about parallel programming or new hardware to the knowledge base as soon as they are available.
- It may use heuristic knowledge to rule out some unnecessary, expensive restructuring tests, thus improves the efficiency.
- The explanation mechanism not only helps the users but may also teach them to program in parallel. It can be an useful teaching tool for training parallel programming experts.
- The user has very high level of controls during the restructuring processes. He can switch between focuses or change the control level at his will.
- The result of restructuring usually leads to new chances of further restructurings. As a source to source program restructurer, the output programs can be pumped into the system repeatedly until all parallelism of the problem has been explored.
- By writing the machine independent programs and using the system to translate them into machine dependent parallel code, we can achieve the portability of the programs. The development cost of the programs can be cut down by a significant factor, since the system are capable of generating code that suit various machine architectures.
- The system can run on inexpensive front-end machines (like VAX or even table top work stations) of the supercomputers and uses cheap resources of the host to efficiently utilize the uses of expensive supercomputers.

We hope that the development of the system can lead to new understanding of the parallel programming and new restructuring techniques. Also, a user friendly program environment may ease the programming job for current or future highly parallel computers, and leads to efficient use of the expensive supercomputers.
References


APPENDIX 1: Displays of the Example shown in Section 3.

<<< \$\$ Welcome to the restructurer programming environment: >>>

>>> Input file name? test.
test.pt reconsulted 768 bytes 0.4666668 sec.
\$\$ Source program name : test
>>> Target machine? cyber205.
$ machine name : cyber205
$ machine type : vector
$ pipeline? : 1
$ startuptime : large
$ operand : inMEM
$ arraystore : inROW
$ penumber : 1
$ configuration : [1,1]
$ special features : has not been initialized

>>> How much explanation do you like me to give you during the execution? (3)
\$\$ You have the following choices
(0) No, thanks.
(1) A little bit.
(2) In detail.

>>> Give me a number between 0 to 2: 2
$ explanation degree : 2

Screen 1: This screen shows process of initializing the system. Features of the target machine are printed on screen. Number in parathesis represents the order that the action appeared.
Screen 2: The system asks the user to confirm its choice of focus and control level. The user is free to change the focus if he doesn't agree the choice made by the system. The first operation chosen by the system is the vectorization. Actually, the term "vectorization" is somewhat misleading here, because in this focus we determine the sequence of transformations for all kinds of target machines. Notice that the screen is divided into four sections: Actions, Reasons, Recommendations, and Informations sections.

Screen 3: The given loop is not directly vectorizable. Notice that whenever an explanation or recommendation is given, the system waits until the user hits the <RETURN> key to guarantee that they have enough time to read the informations.
<< Restructurer >>
--- [Actions] ----------------- Control level: Front end ------ Focus: Vectorizable ------
$ Applying rule 0.
$ Applying rule 1.
$ loading the inner most loop.
$ innermost loop loaded.
$ Press <Return> to continue.
$ Applying rule 2.
$$ The loop has cycle.
Press <return> to continue. (user hit <return> key)
>> Do you agree to apply forward substitution then statement reordering ? (y/n) y
$$ New focus : << forward substitution >>

--- [Reasons] ----------------- Rule 2 ------------------------------
The loop has cycle.
We may apply Forward Substitution to reduce number of statements in the cycle!
After apply Forward Substitution, we need to apply Statement Reordering to vectorize the loop.

--- [Recommendations] ---- Rule 2 ------------------------------
Apply Forward Substitution and then Statement Reordering.

--- [Informations] ------------------------------
Target: Cyber205, Source: test, Explanation Degree: 2.

Screen 4 : This screen is the continuation of the previous one. Since the loop has cycle, forward substitution was suggested to apply.

<< Restructurer >>
--- [Actions] ----------------- Control level: Front end ------ Focus: forward substitution ------
$ applying rule 1
$ Statement 3 is a candidate for substituting.
$ Press <return> to continue.

--- [Reasons] ----------------- Rule 1 ------------------------------
Statement 3 is the only statement in cycle has forward dependence relation.
So Forward substitution may be allowed for this statement.

--- [Recommendations] ---- Rule 1 ------------------------------
Try to substitute the r.h.s. of statements that dominate it into its r.h.s.

--- [Informations] ------------------------------
Target: Cyber205, Source: test, Explanation Degree: 2.

Screen 5 : Trying to break the cycle by substituting r.h.s. of definitions of the variables that were used in S3 into S3.
Screen 6: The statement S2 blocks the way of substitution, so apply statement reordering to move it out of the cycle.

Screen 7: After forward substitution, statement reordering is needed.
Screen 8: The loop is now vectorizable, there is no need to apply other transformations in this level. So the system suggests to exit the current level.

Screen 9: End of demonstration.
Part VII

Others
Chapter 16

A Prototype for an Expert System for Morphological Classification of Prehistoric American Pottery

C. Tsatsoulis and K. S. Fu
A Prototype for an Expert System for
Morphological Classification of Prehistoric
American Pottery
Costas Tsatsoulis and King-sun Fu

INTRODUCTION

1. Problem Statement

The purpose of the work presented was to design the prototype of an expert system that will arrive to a set of hypotheses about the forming and decorating techniques used in designing a vessel, given a description of it. The vessels (or sherds) studied are limited to prehistoric pottery of the Southwestern US and Central American regions (at the present state of the system a space limitation is not very necessary). The hypotheses the system arrives at can also be used as intermediate hypotheses of a larger system that does space/time classification of pottery. The designed prototype should have a good user interface, some explanation capabilities, should be fast by utilizing an efficient control strategy and be also able to handle uncertainty.

2. Importance of Pottery Studies in Archeology

Archeologists draw their knowledge of prehistoric cultures largely from the remains of material things.

Pottery sherds, being the most abundantly found artifact, are very important in archeological research and help establish relative chronologies by serializing and correlating finds. The dependence of the research on sherds has forced archeologists to pay more attention to secondary features, such as rim shape, and to physical properties, such as hardness, luster and paste color. Technological analysis was a necessary development in the period of concentration on the features that can be judged from sherds, leading to a close cooperation between archeology and analytical sciences like chemistry and petrography. The evidences collected threw light to trade relations, contacts of people and interactions of cultures, and, of course, they became fundamental to pottery classification. The specialization that followed drew the criticism that taxonomy might become an end to itself. In order to avoid that, in studying ceramic technology one should not only stress its accuracy and reliability, not the data it recovers, not even the special advantages it offers for determining the sources and relationships of pottery, for these contributions are already recognized; rather, one
should bring out an awareness of the potter’s role, and view pottery both as a facet of culture and as a product of human skill and intelligence.

All the above make pottery an exiting and necessary study subject, but, at the same time, a subject presenting many problems. The most important problem facing the study of ceramic technology is the lack of objectivity: “There is a deeply rooted, popular idea that the expert recognizes subtle characteristics that are significant but indefinable. This conviction is not infrequently shared by archeologists whose experience in pottery classification seems to support it...To the archeologist, the immediate, unreasoned recognition of licknesses and differences, frequently called ‘pottery sense’, seems to be a special gift without which he would be unable to get meaning from the bushels upon bushels of potsherds that pass under his scrutiny...It seems to me that we must recognize the value of the ability called ‘pottery sense’ and at the same time acknowledge its limitations...Pottery sense is characterized by a sensitive and receptive state of mind, and fortunately it is developed by experience. Its most conspicuous limitation is that it places pottery analysis on an individual, and therefore unscientific, basis. Often it does not aid in transmission of information.” [1]. A standardized method for pottery analysis would help some of the problems presented by relying on ‘pottery sense’, but many have criticized that standardized methods are too mechanical, and not even objective. Even if this were true, it does not alter the fact that the possibility of agreement and of meaningful reporting are enhanced by the establishment of common standards. The recognition of the limitations of objective methods should not blind us to the fact that they afford data that cannot be obtained otherwise, and that they approach a common language more nearly than any other methods of description.

An expert system would help standardize the ceramic technology study and, eventually, pottery classification. It would also free archeologists from the tedious and repetitious work of sherd studying, without hindering further developments in the field; it may even add enough controversy to accelerate them.

A short description of ceramic technologies and pottery forming are in Appendix A.
Knowledge Base

1. Observations, Certainty, Importance and RRI

The observations are in the form of LISP symbols. At the initialization stage, the system defines every observation name it will accept, and connects it with a list of observations contradictory to it. So, for example, the observation `fully-oxidizing-firing-atmosphere` will be connected to the NOT list (`partially-oxidizing-firing-atmosphere, reducing-firing-atmosphere`). This is necessary for the rule description used, as will be explained later. Every observation carries also a certainty value C, in the range between 0 and 1.

Rules can be viewed as frames, each with two subframes IF and THEN. Each subframe contains n slots for the antecedents and consequents respectively. There are 98 such rule frames, named r1 to r98, and listed in Appendix B. Each slot in the IF subframe is bound to an importance value I, in the range between 1 and 10. These values indicate the importance of the antecedent filling the slot in the rule conclusion. (importance values can be used even in the case of a single antecedent rule, and they then define the importance of the whole rule).

Each slot in the THEN subframe is bound to a certainty value CC, in the range from 0 to 1. These values indicate the certainty of the rule conclusions.

The strategy for rule firing is based on the Relative Rule Importance (RRI), where RRI can be defined as the ratio of the sum of the product of the observation certainty times the antecedent importance, over the sum of the antecedent importance. So:

$$RRI_r = \frac{\sum_{i=1}^{n} (C_j \times I_j)}{\sum_{i=1}^{n} I_i}$$

where RRI is the relative rule importance of rule r, C_j the certainty of the observation in slot j with importance I_j, and n the total number of slots in the IF frame of rule r.
This method was preferred over the probability and fuzzy set based methods, since they would both not fire in the case of a missing antecedent (C=0). Using the RRI method even rules with incomplete antecedent lists may be fired.

The certainty of an observation as a consequent is simply calculated as the product of the rule RRI and the certainty of the consequent slot it is filling. If the observation already has a certainty value, the largest of the two (existing and new) is selected.

2. Rules

The prototype is rule based, with a total of 98 rules, written in a PROLOG-like style, with the main difference that consequents are not restricted to an ORed format. The consequents are always a conjunction, and cases where the consequents are a disjunction are handled by splitting them into two or more separate rules (e.g. $A \lor B \rightarrow C \Rightarrow A \rightarrow C$, $B \rightarrow C$). Consequents can be in any kind of format (conjunction or disjunction). When a rule is fired, all consequents are added to the inferred facts list and at the end of the session the list is searched for inconsistencies. In this phase, if the inferred facts list contains both an observation and an observation belonging to the first one’s NOT list, the observations are ORed. So, returning to the previous NOT list example, if both reducing-firing-atmosphere and fully-oxidizing-firing-atmosphere were part of the inferred facts list, they would be ORed at the end phase.

An example of rule firing would be as follows:

Suppose the observed facts list was [paint-sinks-in-clay [0.85], paint-is-a-suspension [0.45]], and the rule to be fired was r70.

RULE 70

IF (paint-sinks-in-clay) 8.0
(paint-luster-and-texture-same-as-that-of-clay) 10.0
THEN (paint-is-a-suspension) 1.0
(paint-is-iron-oxide) 1.0

Then, RRI=(8.0*0.85) / (8.0+10.0) = 0.36. And we have the new facts paint-is-a-suspension [(0.36*1.0) = 0.36], and paint-is-iron-oxide [(0.36*1.0) = 0.36]. Since the previous certainty of the observation paint-is-a-suspension was higher than the new one, it retains its old
certainty value.

3. Control Strategy

The rules are divided into 11 context lists: firing atmosphere, forming and shaping methods, finishing techniques, luster, non-plastic inclusions, strength, fracture, plastic decoration, paint type, painting techniques and vessel form. A forward chaining procedure is used, but by handling missing antecedents, the technique also closely resembles backward chaining. As an example, if the rules were:

RULE-1:
IF (temper-color-grey) 10.0 (clay-color-clear) 10.0
THEN
(reducing-firing-atmosphere) 0.9

RULE-2:
IF (clay-color-red) 10.0
THEN
(clay-color-clear) 1.0

and the observed facts list were (clay-color-red [0.8], temper-color-grey [0.9]), then by firing RULE-1 first we would get: (clay-color-red [0.8], temper-color-grey [0.9], reducing-firing-atmosphere [0.405]). By firing RULE-2 and RULE-1 next, we would get: (clay-color-red [0.8], temper-color-grey [0.9], reducing-firing-atmosphere [0.765], clay-color-clear [0.8]). This example (even though it is practically impossible given the existing rule ordering), is very similar to backward chaining.

As mentioned above, a basic rule ordering strategy was used, wherever possible, and rules are ordered in such a way so that the conclusions of previous rules are antecedents of the rules following. This strategy proved not to be sufficient in limiting the iterations through the rule lists, so further controls were added.

If the object of the study is a sherd, the vessel-form rule list is not fired at all. After this first check, the system runs through all context lists, looping inside a context until no more rules are fired. A rule is fired if its RRI is greater than 0.2 and if its fire-flag is TRUE. Initially, all fire-flags are set to TRUE, but if a rule is fired with RRI greater than 0.98 the flag is set to FALSE. Some rules are also grouped, so that if one rule in the group is fired, the rest have their fire-flags set to FALSE. A representative group is the one concerning clay color. Obviously, if
the clay color is red, there is no need to fire rules concerned with orange, white, black, etc, clay colors.

The above mixed control strategy proved to be very effective, since for the prototype with maximum observed facts the response time was less than 3 minutes (actual user waiting time; cpu time was less).
USER INTERFACE

1. Introduction

An effort was made to make the system user friendly and accessible. So, when entering the system, the user is given the options of getting information about the system, viewing the rules, getting examples of questions the system will ask, starting the session or exiting. (Flags -a and -s allow skipping of the introductory stages for experienced users). Before starting the session the system asks also for the name of the user, to make him/her feel a little bit more at home.

At the end of a session the user can choose between viewing the results, adding the results to a database, viewing a trace of the system's inference procedure, starting another case or exiting.

2. Knowledge Acquisition

The system obtains the information it needs by asking the user questions. There are two main types of questions, information acquisition and questioning control. In the first type belong questions that directly ask the user for information about the object of study. The second type contains questions that control the sequence of questions asked (e.g. A positive response to the question "Is the vessel painted?", would be followed by questions concerning painted vessels. A negative response would make the system skip these questions).

There are four kinds of information acquisition questions: yes/no, multiple choice, check list and certainty acquisition. The examples that follow demonstrate these four types of questions.

YES/NO QUESTION EXAMPLE

---------------------

** Q1 (y/n)
Is the plastic technique used clearly stamping?

** Q2 (give the ONE appropriate response)

Does the vessel have a:

a. square and even break
b. conchoidal break
c. ragged break

** Q3 (give all the appropriate)

How would you describe the texture of the vessel surface?

1. grainy
2. smooth and compact
3. lustrus
4. matte

YES/NO questions expect either 'y' (yes) or 'n' (no) as response. Multiple choice questions expect only one answer, and have a lower case letter in front of every possible response. Check list questions accept any number of answers separated by commas, and have a number in front of every possible response. If the user prefers not to answer a question a 0 has to be given as response (actually, any kind of 'illegal' response will produce the same effect with the 0, but consistency should be preferred).
After every answer to an information acquisition question, the same certainty acquisition question appears:

** CERT

You can say that the correctness of your observation(s) could be described as:
1. you can't get more certain than that
2. very certain
3. certain
4. a bit certain
5. uncertain
6. strongly uncertain
g. I prefer to give a numerical value

The question is in check list format to support multiple entries. A typical example would be (where ? is the system’s prompt):

** Q3  (give all the appropriate)

How would you describe the texture of the vessel surface ?

1. grainy
2. smooth and compact
3. lustrus
4. matte

?1, 3, 2

** CERT

You can say that the correctness of your observation(s) could be described as:
1. you can't get more certain than that
2. very certain
3. certain
4. a bit certain
5. uncertain
6. strongly uncertain
    g. I prefer to give a numerical value

\[ g, 2, g \]
\[ 0.58 \]
\[ 0.96 \]

The above example adds to the observed facts list the observations grainy, lustrus, smooth and compact with respective certainty values 0.58, <very certain> and 0.96. (<very certain> is decoded as 0.85).

3. Displaying Results, Tracing and Adding to Database

The results are displayed as english text, in user-friendly format, with the certainty value following each observation and with contradicting facts as disjunctions.

The trace of the system's inference procedure is very simple, and displays only the rules fired and the order in which they were fired.

The addition to the database adds the results to a mere library, as english text, and still needs modifications to become a real database.
COMPUTER IMPLEMENTATION

1. User Interface

The user interface program was written in C language, mainly as a matter of personal preference of the language’s word and text processing capabilities. A LISP implementation would also be possible, if an environmental consistency is preferred or required. A complete listing of the user interface program is included in Appendix C.

2. Inference Machine

The program for the inference machine was written in FRANZ LISP language, but it is not dialect limited. The mixed C and LISP environment presents no problem on the VAX, but, in times of heavy load, it slightly increases the response times.

As previously described, observations are LISP symbols with one property list, NUMBER. NUMBER is one of f1 to f181, and stands for ‘fact number’. The fact numbers have properties: CERTAINTY, NOT, PRINT? (printing flag), CONTEXT and FACT. This representation connects both a fact “fact” with its fact number “fn” through the NUMBER property, and a fact number “fn” with its fact “fact” through the FACT property. The above method was preferred, to make programming less prone to typing errors (“f12” is easier to type than “volcanic-sand-nonplastic-inclusions”).

Rules are represented as r1 to r98 and have property lists IF, THEN, IMPORTANCE, CONSEQ-CERTAINTY and FIRE? (firing flag). IF contains a list of the antecedents; THEN contains a list of the consequents; IMPORTANCE a list with the antecedent importance values; CONSEQ-CERTAINTY a list with the consequent certainty values. A property ’context’ is not necessary, since rules are grouped into context lists.
EXPERIMENTAL RESULTS

Initially artificial cases were used for testing, and the results indicated that the system worked according to specifications. Then, sherd cases from Pulltrouser Swamp and the Chichen Itza area of Yucatan were used for testing, and the results showed that the majority of the system rules were leading to correct conclusions, but also that some of the rules had to be rewritten and some more should be added to produce correcter results. The main problem of the system shown by the testing phase was that some of the rules were too general and gave ambiguous results.
DISCUSSION

Since the initial goal was the design of a prototype that would provide intermediate hypotheses to a larger system, it can be said that it has been fulfilled. Still, a simple, rule base system may not be the answer if the expansion of the present prototype is desired.

The user interface is also not optimal. It should allow editing of the information given, addition or modification of rules, should have better explanation capabilities and a link to a real database. It should also give the option to experienced users to skip the time consuming question answering session, and input their observations directly.

The problems facing an expansion of this prototype to a pottery classification system are many. The disagreement of experts on many issues forces the designer to almost become an expert himself, since this is the only way to distinguish between fundamental, conceptual differences and simple disagreements. One way to reduce such conflicts would be to focus on only morphological classification, avoiding controversial conclusions about cultural relations, trade, etc. Still, it will not be an easy task. Another big problem is the general distaste of the humanistic sciences towards computers and mechanized processes. The discussion in part 1 has already given an answer to this.
ACKNOWLEDGEMENTS

Special thanks are due to Prof. R. Fry of the Purdue Anthropology Department, for his help, suggestions, expert advice and time.
LIST OF REFERENCES


APPENDIX A

A FEW WORDS ABOUT PREHISTORIC POTTERY ANALYSIS

1. Prefiring Treatment

1.1. Clay and temper

Clay is very difficult to exactly define, since the word has been applied to a variety of materials, differing both in origin and composition. It can be defined broadly as fine-grained, earthly material, that develops plasticity when mixed with water. Its chemical components may be silica, alumina, water, iron, alkalies and alkaline earths.

Temper is the word archeologists have used to designate nonplastic materials mixed with clay to counteract excessive shrinkage of ceramic bodies in drying and firing. It is readily identified microscopically and is one of the best means of distinguishing wares and sources of pottery types. Temper material has effect on the strength of the vessel, fracture type and color.

1.2. Forming & shaping techniques

In the region and time of study, the potter’s wheel was not used. Main forming methods were coiling, where clay coils are added over each other to form the vessel, molding, where a vessel is formed using a stone mold (usually one part of the vessel molded, the other part formed by coiling) and the more rare technique of paddle and anvil. Each technique leaves distinguishing marks (coiling marks, junctures, anvil depressions) on both the exterior and interior of the vessel. Good finishing of the vessel surface, though, may destroy these marks.

1.3. Finishing techniques

Strictly speaking, the surfaces can be divided into polished and unpolished, and there are several polishing techniques. Polished surfaces usually have luster and the unpolished ones are matte. Depending on the finishing technique used, different marks are left on the surface. A hard and smooth tool (like a pebble) leaves streaks and
depressions, while a hand or a wet cloth leaves the surface matte and possibly grainy.

2. FIRING

The changes of clay and temper during firing of a vessel depend on the temperature and duration of firing. Usually clay contains iron and in a fully oxidizing atmosphere, the creation of iron oxides leads to bright clay colors. Partially oxidizing and reducing firing atmospheres lead to dull colors, or clear clay colors and dull temper colors, when the firing time was not enough for the oxidation process to penetrate the whole vessel. Smudging was also been used and has to be distinguished from naturally black or grey clays. Refiring a sherd and clay and temper color offer the most widely used methods for determining the firing conditions.

3. POSTFIRING TREATMENT

3.1. Luster

The application of luster can be done both during and after firing. In the first case the lustrous material is a glass-like glaze which leaves tool marks matte on the surface. In the second case the lustrous material is a plant extract that burns and blackens by refiring.

3.2. Plastic decoration

Plastic decoration can be clay figures welded on the surface (rare) and plastic lines. Plastic lines can be mainly done either by stamping (obvious by a repetition of exactly the same pattern) or incising. Deep, narrow and strongly curved lines are results of a pointy tool; wider, asymmetric lines are result of a knife or gauge. We can also determine post- or pre-firing incision by comparing the color of the lines to the color of the clay.

3.3. Paint types

The paint used was either a plant extract solution, which penetrates the surface, burns by refiring and is usually durable, or an oxide suspension, which deposits on the clay and usually becomes worn and powdery with time.
3.4. Painting styles

The major painting styles are positive and negative. In the negative painting style we can define background (figures painted on the background), foreground (background painted around figures) and resist painting techniques.

4. VESSEL FORM

There are four characteristic points on a vessel: end points, inflection points, vertical tangent points and corner points. A vessel can be classified as unrestricted (if it allows access to its interior), independent restricted and dependent restricted (if it has a neck). It can be further classified as simple, composite, inflected and complex, depending on the characteristic points it has. Vessels are usually represented as a combination of the following basic solids and surfaces: ovalloid, ellipsoid, sphere, cylinder, cone and hyperboloid. Rims are defined as direct (if there is no elaborate rim) and deviate, and different methods of rim forming can also be identified.

The above analysis descriptions are overly simplified, and by no means complete, and are only intended to give a brief feeling of the work involved. A completer analysis would also have to include plastic and painted decoration forms (something that can be extremely complicated), brush stroke types, colors, decoration fields, exact rim forms (since rims often served as the 'signature' of the potter), dimensions and proportions, exact type and percentage of nonplastic inclusions, etc. The complexity and diversity of the above may even necessitate some kind of a graphic interface or simple grammar.
FIRING ATMOSPHERE

------------------------------

RULE 1
IF (clay-color-red) 10.0
THEN (clay-color-clear) 1.0

RULE 2
IF (clay-color-yellow) 10.0
THEN (clay-color-clear) 1.0

RULE 3
IF (clay-color-orange) 10.0
THEN (clay-color-clear) 1.0

RULE 4
IF (clay-color-buff) 10.0
THEN (clay-color-clear) 1.0

RULE 5
IF (clay-color-clear-cream) 10.0
THEN (clay-color-clear) 1.0

RULE 6
IF (clay-color-light-grey) 10.0
THEN (clay-color-grey) 1.0

RULE 7
IF (clay-color-dark-grey) 10.0
THEN (clay-color-grey) 1.0

RULE 8
IF (temper-color-red) 10.0
THEN (temper-color-clear) 1.0

RULE 9
IF (temper-color-yellow) 10.0
THEN (temper-color-clear) 1.0

RULE 10
IF (temper-color-orange) 10.0
THEN (temper-color-clear) 1.0

RULE 11
IF (temper-color-buff) 10.0
THEN (temper-color-clear) 1.0

RULE 12
IF (temper-color-clear-cream) 10.0
THEN (temper-color-clear) 1.0

RULE 13
IF (temper-color-light-grey) 10.0
THEN (temper-color-grey) 1.0

RULE 14
IF (temper-color-dark-grey) 10.0
THEN (temper-color-grey) 1.0

RULE 15
IF (clay-color-clear)
   (temper-color-clear)
THEN (fully-oxidizing-firing-atmosphere) 1.0

RULE 16
IF (clay-color-light-grey)
   (temper-color-dark-grey)
THEN (partially-oxidizing-firing-atmosphere) 1.0

RULE 17
IF (clay-color-brown)
   (temper-color-brown)
   (color-changed-by-refiring)
THEN (partially-oxidizing-firing-atmosphere) 1.0

RULE 18
IF (color-changed-by-refiring)
THEN (partially-oxidizing-firing-atmosphere) 1.0
(reducing-firing-atmosphere) 1.0

RULE 19
IF (clay-color-whitish)
   (temper-color-whitish)
THEN (partially-oxidizing-firing-atmosphere) 0.9

RULE 20
IF (color-unchanged-by-refiring)
THEN (fully-oxidizing-firing-atmosphere) 1.0

RULE 21
IF (clay-color-black)
   (temper-color-clear)
THEN (fully-oxidizing-firing-atmosphere) 1.0
(smuudging) 1.0

RULE 22
IF (clay-color-black)
   (temper-color-black)
THEN (natural-smudging) 1.0

RULE 23
IF (clay-color-grey)
   (unevenly-colored)
THEN (smudging) 0.9

FINISHINGTECHNIQUES
--------------------------
Rule 24

IF (grainy-surface)
THEN (no-finishing-tool-used) 10.0
THEN (soft-and-yielding-finishing-tool-used) 0.8
THEN (unslipped-and-unpolished-surface) 0.7

Rule 25

IF (smooth-and-compact-surface)
THEN (hard-and-smooth-finishing-tool-used) 6.0

Rule 26

IF (lustrus-surface)
THEN (polished-and-unslipped-surface) 10.0

Rule 27

IF (matte-surface)
THEN (polished-and-unslipped-surface) 10.0

Rule 28

IF (tool-marks-on-surface)
THEN (unslipped-and-unpolished-surface) 10.0

Rule 29

IF (strong-color-contrast-of-surface)
THEN (slipped-surface) 10.0

Rule 30

IF (polished-surface)
THEN (fine-nonplastics) 10.0
THEN (very-fine-nonplastics) 0.9
THEN (soft-and-yielding-finishing-tool-used) 0.8

Rule 31

IF (polished-and-unslipped-surface)
THEN (hard-and-smooth-finishing-tool-used) 10.0

Luster

Rule 33

IF (surface-burned-by-refiring)
THEN (lustrus-surface) 10.0
THEN (luster-applied-varnish) 0.7

Rule 34

IF (tool-marks-on-surface)
THEN (lustrus-surface) 10.0
THEN (surface-rubbed-with-smooth-tool-before-firing) 0.9

Rule 35

IF (no-tool-marks-on-surface)
THEN (lustrus-surface) 10.0
THEN (luster-applied-during-firing) 0.8
THEN (luster-due-to-clay-quality) 0.3

Rule 36

IF (matte-surface) 10.0
THEN (no-luster-applied) 0.8

RULE 37
IF (strong) THEN (pebble-nonplastics) 6.0 (granule-nonplastics) 0.45 (coarse-nonplastics) 0.45 (very-coarse-nonplastics) 0.45 (luster-applied-during-firing) 0.8

RULE 38
IF (weak) THEN (medium-nonplastics) 6.0 (fine-nonplastics) 0.45 (very-fine-nonplastics) 0.45 (slit-nonplastics) 0.45

RULE 39
IF (square-and-even-break) THEN (slit-nonplastics) 7.0 (very-fine-nonplastics) 0.8

RULE 40
IF (conchoidal-break) THEN (hard-clay) 7.0 (fine-nonplastics) 0.7

RULE 41
IF (ragged-break) THEN (very-coarse-nonplastics) 7.0 (pebble-nonplastics) 0.75 (granule-nonplastics) 0.75

RULE 42
IF (junctures-on-surface) THEN (vessel-formed-using-coiling) 10.0

RULE 43
IF (junctures-on-interior) THEN (vessel-formed-using-coiling) 10.0

RULE 44
### IF (coiling-marks-on-surface) THEN (vessel-formed-using-coiling)

**RULE 45**

IF (coiling-marks-on-interior) THEN (vessel-formed-using-coiling)

**RULE 46**

IF (weld-ridges-on-surface) THEN (vessel-formed-using-molding)

**RULE 47**

IF (weld-ridges-on-interior) THEN (vessel-formed-using-molding)

**RULE 48**

IF (one-part-of-vessel-asymmetric) THEN (vessel-formed-using-molding)

**RULE 49**

IF (anvil-depressions-on-interior) THEN (vessel-formed-using-paddle-and-anvil)

---

### PLASTIC DECORATION TECHNIQUES

**RULE 50**

IF (stamping) THEN (cord-wrapped-stamp-used)

**RULE 51**

IF (stamping) THEN (roulette-used)

**RULE 52**

IF (animal-figure-welded) THEN (figure-hand-worked)

**RULE 53**

IF (human-figure-welded) THEN (figure-hand-worked)

**RULE 54**

IF (human-figure-welded) THEN (molding)
RULE 55
IF (animal-figure-welded) 10.0
THEN (modeling) 1.0
THEN (molding) 1.0

RULE 56
IF (strongly-curved-lines) 10.0
IF (sharp-edged-lines) 10.0
IF (edge-elevation) 10.0
IF (deep-lines) 10.0
IF (narrow-lines) 10.0
THEN (pointy-tool-used) 1.0

RULE 57
IF (sharp-edged-lines) 10.0
IF (edge-elevation) 10.0
THEN (circular-patterns) 8.0
THEN (pointy-tool-used) 1.0

RULE 58
IF (sharp-edged-lines) 10.0
IF (edge-elevation) 10.0
THEN (up-and-down-strokes) 8.0
THEN (pointy-tool-used) 1.0

RULE 59
IF (sharp-edged-lines) 10.0
IF (edge-elevation) 10.0
THEN (back-and-forth-strokes) 8.0
THEN (pointy-tool-used) 1.0

RULE 60
IF (line-has-asymmetric-cut) 10.0
THEN (knife-used) 0.7

RULE 61
IF (line-inclination-curvature-center-independent) 10.0
THEN (knife-used) 1.0

RULE 62
IF (end-of-stroke-towards-worker) 10.0
THEN (knife-used) 1.0

RULE 63
IF (line-inclination-towards-curvature-center) 10.0
THEN (gauge-used) 1.0

RULE 64
IF (end-of-stroke-away-from-worker) 10.0
THEN (gauge-used) 1.0

RULE 65
IF (clay-pushed-inside-lines) 10.0
THEN (prefiring-incising) 1.0

RULE 66
IF (line-edge-chipping) 10.0
THEN (postfiring-incising) 1.0

RULE 67
IF (lighter-colored-plastic-lines) 10.0
  (smudging) 10.0
THEN (postfiring-incising) 0.8

PAINTTYPES

RULE 68
IF (paint-burned-by-refiring) 10.0
THEN (paint-applied-after-firing) 1.0
     (paint-is-organic) 1.0
     (paint-is-a-solution) 1.0

RULE 69
IF (paint-not-burned-by-refiring) 10.0
THEN (paint-is-iron-oxide) 1.0
     (paint-is-a-suspension) 1.0

RULE 70
IF (paint-sinks-in-clay) 8.0
     (paint-luster-and-texture-same-as-that-of-clay) 10.0
THEN (paint-is-a-suspension) 1.0
     (paint-is-iron-oxide) 1.0

RULE 71
IF (paint-deposit-on-surface) 8.0
     (paint-luster-and-texture-different-than-clay) 10.0
THEN (paint-is-a-suspension) 1.0
     (paint-is-iron-oxide) 1.0

RULE 72
IF (paint-has-blurred-outlines) 5.0
THEN (paint-is-organic) 0.9
     (paint-is-a-solution) 0.9

RULE 73
IF (vessel-unevenly-colored) 5.0
THEN (paint-is-organic) 0.9
     (paint-is-a-solution) 0.9

RULE 74
IF (center-of-line-lighter-than-edge) 5.0
THEN (paint-is-organic) 0.9
     (paint-is-a-solution) 0.9

RULE 75
IF (chipping-of-painted-surface) 5.0
THEN (paint-is-organic) 0.9
     (paint-is-a-solution) 0.9

RULE 76
IF (beading-of-painted-surface) 5.0
THEN (paint-is-iron-oxide) 0.9
     (paint-is-a-suspension) 0.9

RULE 77
IF (paint-has-red-or-brown-tones) 5.0
THEN (paint-is-iron-oxide) 0.75
(paint-is-a-suspension) 0.75

RULE 78
IF (painted-surface-is-patchy) 5.0
THEN (paint-is-iron-oxide) 0.85
(paint-is-a-suspension) 0.85

RULE 79
IF (paint-is-worn) 5.0
THEN (paint-is-iron-oxide) 0.8
(paint-is-a-suspension) 0.8

RULE 80
IF (paint-is-soft-and-powdery) 5.0
THEN (paint-is-iron-oxide) 0.9
(paint-is-a-suspension) 0.9

RULE 81
IF (paint-left-only-in-clay-pores) 5.0
THEN (paint-is-iron-oxide) 0.95
(paint-is-a-suspension) 0.95

================================================================================

PAINTING TECHNIQUE

================================================================================

RULE 82
IF (dark-background) 10.0
(light-painted-figures) 10.0
THEN (negative-painting-style) 1.0

RULE 83
IF (light-background) 10.0
(dark-painted-figures) 10.0
THEN (positive-painting-style) 1.0

RULE 84
IF (foreground-painting-technique) 10.0
THEN (paint-is-a-solution) 1.0
(paint-is-organic) 1.0

RULE 85
IF (painted-figure-is-simple) 5.0
THEN (background-painting-technique) 0.6

RULE 86
IF (carbon-background) 5.0
THEN (resist-painting-technique) 0.6

RULE 87
IF (resist-painting-technique) 10.0
THEN (vessel-painted-after-firing) 1.0

================================================================================

VESSEL SHAPE
---------

RULE 89
IF (negative-or-zero-tangent-at-top) (no-neck) THEN (vessel-of-unrestricted-form)

RULE 89
IF (positive-tangent-at-top) (no-neck) THEN (vessel-of-dependent-restricted-form)

RULE 90
IF (neck) THEN (vessel-of-independent-restricted-form)

RULE 91
IF (no-I-point) (no-C-point) THEN (vessel-is-simple)

RULE 92
IF (no-I-point) (C-point) THEN (vessel-is-composite)

RULE 93
IF (no-C-point) (I-point) THEN (vessel-is-inflected)

RULE 94
IF (C-point) (I-point) (VT-point) THEN (vessel-is-complex)

RULE 95
IF (wall-carried-to-top-with-no-break) THEN (rim-is-direct)

RULE 96
IF (rim-is-elaborated) THEN (rim-is-deviate)

RULE 97
IF (abrupt-rim-thickening) THEN (rim-done-by-adding-coils)

RULE 98
IF (smooth-rim-thickening) THEN (rim-done-by-adding-coils) (rim-done-by-wall-redistribution)
APPENDIX C  Sample Consultation Session

---more--(95%)

Hello, and welcome to PEGASUS

I, the experimental Expert System for Prehistoric American Pottery classification.

To get information about the usage of this system type 'info'
To see a complete rule list type 'rules'
To start working immediately type 'start'
To exit the system type 'exit'
Pegasus-I is the very first Expert System for Prehistoric American Pottery classification. It is rule based and supported by a C and FRANTZ LISP environment. In its present state, it is only an experimental model that can infer the way a vessel was formed, given information about its clay and temper color, type of plastic and painted decorated lines, etc. Pegasus-I provides answers to questions about vessel forming techniques, firing methods, finishing and luster applying techniques, paint types and painting styles and form classification.

It does not use any analytic data information, since the very acquisition of such data requires an expert, which nullifies the need of an Expert System. Pegasus-I will ask the user to provide him with the information he needs using YES/NO, check list and multiple choice.

P.S. PEGASUS-I was conceived sometime in early September 1984 and brought into the world in November 1984, by Costas Tsatsoulis, who is solely responsible for the moods, behavior, and performance of the system.

P.P.S. "PEGASUS" stands for pegasus, "I" stands for 1.

To get information about the usage of this system type 'info'
To see a complete rule listing type 'rules'
To start working immediately type 'start'
To exit the system type 'exit'

? start

Hello. As you probably know my name is Pegasus-I, but my friends call me simply Pegasus. Before we start working together, could you please give me your name? (end your name with a ",")

?Costas

O.K. Costas

I need to know as much as you do about the vessel or sherd you want me to study with you. In order to do that I will have to ask you some questions. They will be yes/no, multiple choice or check list questions. After you give me an answer, I will ask you for how certain you are about it. If you make any errors, you will have a chance to view your responses and change them at the end of the questioning session.
If you want to see the examples following, press '?' otherwise press any other key.

O.K., Costas.
The first thing I need to know is the case name/number we are studying.

 powdered

Are we working with a vessel (or a big piece from a vessel) or just a sherd? Please, type 'vessel' or 'sherd' accordingly.

sherd

Costas, we will first work on the forming and shaping technique information.

** Q1 (give the ONE appropriate response)

On the exterior of the sherd in question, there are:
   a. weld ripples
   b. coiling marks
   c. junctures

* CERT

The correctness of your observation(s) could be described as:
   1. you can't get more certain than that
   2. very certain
   3. certain
   4. a bit uncertain
   5. uncertain
   6. strongly uncertain
   a. I prefer to give a numerical answer

2

** Q2 (give the ONE appropriate response)

On the interior of the sherd in question, there are:
   a. weld ripples
   b. coiling marks
   c. junctures
   d. anvil depressions

0

QUESTIONS 3 AND 4 DO NOT APPLY

Thank you, Costas. Let's talk now about firing conditions.

** Q5 (give the ONE appropriate response)

The clay color is:
   a. red
   b. orange
   c. yellow
   d. buff
   a. dark grey
   b. black
   c. brown
   d. whitish
The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
q. I prefer to give a numerical answer

The temper color is:

a. red
b. orange
c. dark grey
d. black
e. yellow
f. brown
g. buff
h. whitish
i. clear cream
j. white
k. light grey

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
q. I prefer to give a numerical answer

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
q. I prefer to give a numerical answer
Is the clay unevenly colored? (Please note the difference between clay...)

Thank you, Costas. As you see, the whole thing is easier than it sounds. Let's go to finishing techniques.

** Q9 (give all the appropriate responses)

** How would you describe the texture of the sherd surface?

1. grainy
2. smooth and compact
3. lustrus
4. matte

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

** Q10 (y/n)

Would you say that there are tool marks on the surface? (such as streaks, grooves, pits or depressions)

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

Thank you, Costas. Let's talk now about luster.

** Q11 (y/n)

Was the unpainted (!) surface of the clay burned by re-firing?

Thank you, Costas. The next question concerns the strength of the... we are studying. It is very subjective, but don't worry out it too much.

Q12 (give the ONE appropriate response)

Very roughly, would you say that the sherd we are studying is:
a. strong
b. weak
?

Thank you, Costas. The next question concerns the fracture type of the sherd:

**Q13** (give the ONE appropriate response)

Does the sherd have a:

- a. square and even break
- b. conchoical break
- c. ragged break
?

**CERT**

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

Thank you, Costas.

Now we are getting to the tricky part. First, let's find out a few basic things about the plastic decoration style.

***Q14** (give all the appropriate responses)

Are there any:

1. human figures welded on the sherd
2. animal figures welded on the sherd
?

**QUESTION 15 DOES NOT APPLY**

***Q16 (y/n)**

Is there any other plastic decoration on the sherd, except welded figures?
?

***Q17 (y/n)**

Would you say that the technique used was clearly stamping?
?

***Q18 (y/n)**

Are the stamped lines continuous?
?

**CERT**

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer.

Q.50

Q.19 (y/n)

Is stamping the only technique used?

Q.20a

(give all the appropriate responses)

How would you describe the incised plastic lines?

1. strongly curved
2. sharp edges
3. have clear edge elevation
4. deep
5. narrow
6. have up-down stroke
7. have back-forth stroke
8. have arrow
9. have curved
10. have clod red color
11. have back-forth stroke
downstroke

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

Q.20b

(give all the appropriate responses)

How would you describe the incised plastic lines?

1. they have asymmetric cut
2. they incline toward the center of curvature
3. they incline independently from the center of curvature
4. the end of the stroke is towards the worker
5. the end of the stroke is away from the worker
6. the clay is pushed inside the lines
7. there is chipping of the line edges
8. the color of the plastic lines is lighter than the clay

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
q. I prefer to give a numerical answer

Thank you, Costas. Is the sherd we are studying painted (press 'y' or 'n')?

*** Q21 (y/n)

Was the painted area burned by re-firing?

\[ n \]

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

** Q22 (give the ONE appropriate response)

Would you say that:

a. the paint sinks in the clay
b. the paint deposits on the clay surface

\[ b \]

** CERT

The correctness of your observation(s) could be described as:

1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

** Q23 (give the ONE appropriate response)

Would you say that:

a. the painted surface has luster and texture different than that of the unpainted surface
b. the painted surface has luster and texture same as that of the unpainted surface

\[ a \]

** Q24 (give all the appropriate responses)

Would you say that:

1. the paint has blurred outlines
2. the center of painted line lighter than edge
3. there is blurring of lines
4. there is blurring of objects
5. I prefer to give a numerical answer
4. there is teading of the paint
5. the paint has red or brown tones
6. the painted surface is patchy
7. the paint is really worn
8. the paint is soft and powdery
9. the paint left only inside clay pores

CERT

The correctness of your observation(s) could be described as:
1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

Q 25 (y/n)

Is the clay surface unevenly painted?

Thank you, Costas.

Few words about painting techniques, now.

Q 26 (give the ONE appropriate response)

Is the background:

a. dark
b. light

CERT

The correctness of your observation(s) could be described as:
1. you can't get more certain than that
2. very certain
3. certain
4. a bit uncertain
5. uncertain
6. strongly uncertain
7. I prefer to give a numerical answer

Q 27 (give the ONE appropriate response)

Are the figures:

a. dark
b. light

CERT

The correctness of your observation(s) could be described as:
1. you can't get more certain than that
2. very certain
3. certain
**Q28** (give the ONE appropriate response)

Would you say that the painting technique used was:

a. foreground
b. resist

**Q29** (y/n)

Are the painted figures simple (squares, circles, dots, straight lines, etc)?

**Q30** (y/n)

Is there a carbon background?

Thank you, Costas. I have everything I need.

I will start working now. Usually the whole thing lasts a few minutes only. I'll let you know when I'm done.

---

Franz Liszt, Opus 38.69

---

**WORKING. PLEASE WAIT..........**

::: I am finished! :::::

To see the complete information (supplied and inferred) about the vessel/sherd type 'view'
To put the information in the database type 'add'
To start working on a new case type 're-start'
To trace the execution of the rules type 'trace'
To exit the system type 'exit'

? trace
r18
r16
r14
r36
r34
r39
r67
r65
r54
r51
r46
r61
r81
to see the complete information (supplied and inferred) about
the vessel/sherd type 'view'
put the information in the database type 'add'
start working on a new case type 're-start'
trace the execution of the rules type 'trace'
exit the system type 'exit'

>>> CASE : c0001 [ ]

Firing atmosphere <<<

- The temper color is grey  (1.0)
- The temper color is dark grey  (1.0)
- The clay/temper color is changed by refiring  (1.0)
- The firing atmosphere was partially oxidizing  (1.0)

- The firing atmosphere was reducing  (1.0)

Forming techniques <<<

- There are weld ridges on the surface  (0.85)
- The vessel was formed using molding  (0.85)

Temper, Strength and Fracture <<<

- The break is square and even  (0.25)

Luster and Finishing techniques <<<

- The surface is grainy  (1.0)
- The surface is matte  (0.85)
- The nonplastic inclusions are of very fine size  (0.2)

- The nonplastic inclusions are of slit size  (0.2)
- There are tool marks on the surface  (0.85)
- The surface was rubbed with a smooth tool before firing  (0.22)
- There was no luster applied  (0.68)

Plastic decoration techniques <<<

- A cylinder seal was used for stamping  (0.25)
- A flat seal was used for stamping  (0.25)
- A roulette was used for stamping  (0.25)
- The plastic stamped lines are continuous  (0.5)
The incised lines were made by a pointy tool (0.54)
The incised lines are narrow (0.7)
The incised lines are deep (1.0)
The incised lines are strongly curved (1.0)
The clay is pushed inside the incised lines (0.4)
The plastic lines are of lighter color than the clay (0.67)
The vessel was incised before firing (0.4)
OR
The vessel was incised after firing (0.268)

>>> Paint type and Painting techniques <<<

The paint is a suspension (1.0)
The paint is iron oxide (1.0)
The paint was not burned by refiring (1.0)
The paint deposits on the clay surface (1.0)
The paint is left only inside the clay pores (0.85)
The painted background is dark (1.0)
The painted figures are light (1.0)
The painting style is negative (1.0)

To see the complete information (supplied and inferred) about the vessel/sherd type 'view'
To put the information in the database type 'acc'
To start working on a new case type 're-start'
To trace the execution of the rules type 'trace'
To exit the system type 'exit'
? add

>>>> Database has been updated <<<<

To see the complete information (supplied and inferred) about the vessel/sherd type 'view'
To put the information in the database type 'acc'
To start working on a new case type 're-start'
To trace the execution of the rules type 'trace'
To exit the system type 'exit'
? exit

BYE
Chapter 17

Expert System for Contract Bridge Bidding

L. Y. Chang and C. F. Yu
Expert System for Contract Bridge Bidding

L. Y. Chang and C. F. Yu

1. Introduction

The game of bridge is played by 4 players often designated in the literature as North, South, East and West depending on their positions around the card table. North and South are partners, playing against the partnership of East and West. The goal in bridge is to maximize the score or minimize the loss for your partnership by an appropriate sequence of bidding followed by a proper sequence of card playing following the agreement of a contract by one partnership. Card playing may be classified as declarer playing or defense playing depending on which partnership opens the bidding. Although declarer playing and defense playing are important aspects of bridge game, it has been found that in approximately 60% of the games the result depends on bidding. The reason for this is that bidding limits the the best outcome that may be achieved, regardless of the skills exhibited in declarer playing or defense playing. Also, bidding is the first activity carried out during a game of bridge and should be the first portion of a
bridge playing system. Even so, the situation is very complex in the presence of possible interference by the opposing partnership in the bidding process. To restrict the scope of the problem to that solvable in a semester project we have only considered bidding when the opponents always pass.

In chess and many other games, complete information is always available to the player. In bidding the information available to the player is incomplete. Only the thirteen cards that comprises the player's hand and the previous bids are known to the player. In such circumstances, heuristics are necessary for a good bidding system.

There are several widely used bidding systems. We have used a variation of American Standard Bidding System. In this bidding system, most bids have their natural meanings, that is, a player has a good suit when he bids the suit. The exceptions are:

1. strong 2 club opening
2. weak 2 (spades, hearts and diamonds) opening
3. Stayman inquiry after notrump opening, and
4. 3 notrump opening.

Some of the bids that follow these artificial bids may also be artificial.
2. **Work Distribution**

The division of labor in this project was partly determined by the knowledge of the game possessed by L. Y. Chang and the lack of by C. F. Yu at the initiation of this project. Thus, the definition of the game variables and the rules in pidgin English was carried out by L. Y. Chang. C. F. Yu was responsible for translating these into Lisp as well as the rest of the Lisp code. L. Y. Chang also designed the examples used to test the program.

3. **Issues in the Design of a Bridge Bidding System**

3.1. **Knowledge Representation**

Production rules were chosen as the books on bridge that we were able to obtain expressed their advice in the "IF-THEN" form. Typical of such advice is

If you have a even distributed hand and high card point is between 16 and 18, then open 1 notrump.

In addition, production rules are relatively easy to add, delete and modify during testing and debugging. This is useful since we believe that this project is oriented towards producing a prototype which will be subjected to various revisions as the participants in this project gain experience.
3.2. **Inference mechanism**

The appropriate next bid depends largely on the own hand held by the player and the previous bids. Thus, forward reasoning is employed in the system.

3.3. **Knowledge acquisition**

The knowledge was extracted from several books on bidding in bridge ([1], [2], [3]). This extracted knowledge was supplemented with the personal knowledge of L. Y. Chang, who plays the game of bridge.

The advice given in bridge books are normally expressed as English statements which contain qualitative descriptions. These qualitative descriptions have to be translated into the equivalent values or range of values for the game parameters which describe the hand held by the player. An example would be the statement If partner responds 2 notrump to your weak two opening and the opening is sound, you may show a side feature such as K-J-x, or Q-x-x-x if you do not have a solid suit or a strong suit with a sure re-entry.

To write this as a production rule, the terms "solid suit", "strong suit", "sound weak two opening" and "re-entry" have to be rewritten using the game parameters. The "sound weak two opening hand" becomes a hand with

\[
i \quad 9 \leq \text{HCP} \leq 11, \text{ and}
\]
11 a side suit with length \( \geq 3 \) and HCP in the suit is greater 2

while "re-entry" implies the presence of a quick trick (Ace or KQ) in a suit. Similarly, a "solid suit" indicates a suit with

1 length of suit \( \geq 7 \), and

11 contains an Ace, a King and a Queen;

while a "strong suit" is a suit that contains a AKQ, AKJ, AQJ, or KQJ. The game parameters used are described in further detail in Appendix 1.

3.4 Control Strategy

3.4.1 Conflict Resolution

The nature of the problem domain suggests the use of context limiting as a conflict resolution strategy. The rules that are applicable at any time depends on the round of bidding at that particular instant. The set of bidding rules, \( P \), can thus be partitioned into disjoint subsets \( P_1, P_2, \ldots, P_n \) where \( P_i \) is the set of applicable rules in round \( i \). The first four rounds are often also referred to as the opening, response, opener's rebid and responder's rebid respectively. \( P_i \) can be further partitioned into disjoint subsets (which we shall refer to as clusters) \( P_{i1}, P_{i2}, \ldots, P_{ik} \) based upon the previous bids. For example,
the set of rules applicable during the response may be partitioned into: no-trump opening response, strong two opening response, weak two response and one level suit opening response based upon the opening bid. Thus, the rules should be organized into clusters and the system only allows rules in the current cluster to participate in the matching process.

Within each cluster, the conflict resolution scheme chosen is rule ordering. The rules in each cluster are arranged according to their priority whereby rule 1 will have higher priority than rule (i+1). The rule set is scanned linearly from rule 0 and the first rule triggered will be fired. This reduces the overhead in the search process within each cluster. The disadvantage of this scheme is the need to maintain the ordering of the rules whenever rules are modified or added to the cluster.

4. Implementation

At the time of this report, we have implemented a prototype of the bidding system. This prototype contains rules for four rounds of bidding. The current prototype is our third prototype and should be regarded as one of a whole series of prototypes. The bidding system prototype was written in Franz Lisp on a VAX 11/780 running 4.2BSD Unix.

The organization of the implemented
Figure 1: System Organization
system is shown in Figure 1. The global database consists of the game, history and bookkeeping variables while the rule base contains the production rules. Based upon the values of the variables in the global database the control system selects the appropriate rule and fires it. The trace mechanism shows the sequence of firings that lead to a particular decision as well as other pertinent information. We hesitate to call this an explanation system as it is rather primitive in its present form and requires knowledge of the system as well as some effort to interpret. A proper explanation system will require a filter between the user and the trace system.

The system operates in two modes, default and dual. The normal mode is the mode in which it will operate when and if it progresses beyond the prototype stage. The system only plays the position designated by the user, that is, either South or North. This mode is cumbersome for prototyping since it queries the user for the bid by its partner. To get round this, the dual mode can be used. In the dual mode, the system plays both North and South. The user is only required to tell the system whether to continue or to stop (to prevent the system from continuing into the rounds for which the rules have not yet been entered).
4.1. The Global Database

In the default mode, the global database consists of 6 (lisp) vectors named Mst, Cst, Spades, Hearts, Diamonds and Clubs. Mst contains the system bookkeeping and history variables. Cst consists of the game and history variables that describe the entire hand while Spades, Hearts, Diamonds and Clubs contains the game variables that describe each suit of the hand. The vectors Spades, Hearts, Diamonds and Clubs are also accessible as 1suit, 2suit, 3suit, and 4suit. 1suit points to the suit vector corresponding to the longest suit and 4suit to that of the shortest suit. Ties are broken by the rank of the suit. A complete description of these vectors can be found in Appendix 3.

In the dual mode, the global database consists of 11 vectors Mst, nCst, nSpades, nHearts, nDiamonds, nClubs, sCst, sSpades, sHearts, sDiamonds and sClubs. As before, the suit vectors can also be accessed via nsuit, n2suit, n3suit, n4suit, nsuit, s2suit, s3suit and s4suit. This increase in the number of vectors is due to the need to retain game variables for both North and South. To retain compatibility with the code for the default mode, Cst, Spades, Hearts, Diamonds, Clubs, 1suit, 2suit, 3suit and 4suit are retained and point to the appropriate lisp objects depending on whether South or North is bidding.
4.2. Rule Base

As discussed before, context limiting is used as a conflict resolution strategy and rules are collected into clusters. Each cluster is implemented as a lisp vector. Each element of this vector is bound to a list representing the corresponding rule in the cluster. A rule is implemented as a lisp list of 2 elements. The first element of this list is the condition portion of the rule. The second element is the action portion of the rule. Each of these elements are themselves lists of clauses. There is an implied conjunction between the clauses in each list. The clauses are valid lisp statements. For example, the pidgin English rule

\[
\text{IF } (16 \leq CP \leq 18) \\
\text{(suit is major)} \\
\text{(length of suit} \geq 5) \\
\text{THEN (raise to 3)}
\]

resides in the rule base as

```lisp
; if
((member (vref Cst Cp) '(16 17 18))
 (member (vref Cst Csuit) ' (Spades Hearts))
 (> (vref (eval (vref Cst Csuit)) xLnth) 4))
; then
([vset Mst Bsuit (vref Cst Csuit)])
[vset Mst Blevel 3]))
```
The rules in the rule base are of two types: bidding rules and context-determination rules. Context-determination rules are used to determine the correct context while bidding rules determine the bid. Context limiting was also used for the context-determination rules. We have used the term super-cluster to refer to the disjoint subsets of context-determination rules.

4.3. Control System

The context-determination process is carried out in two steps:

i. the super-cluster corresponding to the round is determined

ii. the correct context (rule cluster) is determined by firing the triggered rule in the super-cluster.

Having determined the correct context, the triggered bidding rule is then fired. The global database is the updated. The above three steps are repeated until the bidding terminates or is terminated by the user.

4.4. Trace System

Three levels of tracing are available on the system: default, verbose and screen. In the default mode, the identity of each rule that was fired is printed.
More information is provided in the verbose mode. The verbose mode is invoked by

\[-\rightarrow\text{(verbose)}\]

and deactivated by

\[-\rightarrow\text{(unverbose)}\]

When it is either North's or South's turn to bid, the contents of the global database are printed. This will be followed by the identity each fired rule, the variables changed by the firing of the rule and their new values. A sample of the output produced in the verbose mode is shown in Figure 2.

Although, the verbose mode provides sufficient information to debug the rules, it does so in an inefficient fashion. The user has to create a script of the session and print out the script on a printer to extract the required information since the information exceeds the limited display area of a terminal. This problem is partially alleviated with the screen mode.

The screen mode activates a primitive screen oriented monitor of system execution. The screen mode is invoked by

\[-\rightarrow\text{(scrn)}\]

and deactivated by

\[-\rightarrow\text{(unscrn)}\]
Figure 2a: Example of output produced in the verbose mode (continued in Figure 2(b)).

<table>
<thead>
<tr>
<th>Cards</th>
<th>1suit</th>
<th>2suit</th>
<th>3suit</th>
<th>4suit</th>
</tr>
</thead>
<tbody>
<tr>
<td>xSuit</td>
<td>Spades</td>
<td>Clubs</td>
<td>Hearts</td>
<td>Diamonds</td>
</tr>
<tr>
<td>xLnth</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>xStrn</td>
<td>moderate</td>
<td>weak</td>
<td>weak</td>
<td>weak</td>
</tr>
<tr>
<td>xHcp</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>xRank</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>xSup</td>
<td>verygood</td>
<td>moderate</td>
<td>little</td>
<td>little</td>
</tr>
<tr>
<td>xTrck</td>
<td>1.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

| Spades | (Ace Queen 10 8 7 2) |
| Hearts | (Jack 6) |
| Diamonds | (9 7) |
| Clubs  | (Queen 10 9) |

<table>
<thead>
<tr>
<th>Name</th>
<th>south</th>
</tr>
</thead>
<tbody>
<tr>
<td>Csuit</td>
<td>nil</td>
</tr>
<tr>
<td>Clevel</td>
<td>nil</td>
</tr>
<tr>
<td>Part</td>
<td>north</td>
</tr>
<tr>
<td>Hcp</td>
<td>9</td>
</tr>
<tr>
<td>Open</td>
<td>nil</td>
</tr>
</tbody>
</table>

| Dist   | (6 3 2 2) |
| Cp     | nil       |
| Vuln   | nil       |

| Rnd   | nil       |
| Play  | south     |
| Ihand | 1         |
| mSet  | nil       |

| mOpenr| nil       |
| mGame | nil       |
| Bsuit | nil       |
| Blevel| nil       |

| r0    | nil       |
| r1    | nil       |
| r2    | nil       |
| r3    | nil       |

| r4    | nil       |
| r5    | nil       |
| r6    | nil       |
| r7    | nil       |

Rset0 : rule 0 fired
Mst:mSet = (Rset0)
Rset0 : rule 0 fired
Cst: Dv = 2
Cst: Cp = 11
Mst:mSet = (Rset0)
Rset10 : rule 25 fired
Mst:Blevel = 0
Mst: Bsuit = pass
*** Player south bids 0pass ***
Continue?? y/n : y
1 Figure 2b: Example of output produced in the verbose mode (continued)

*** Player west bids pass ***
Continue ??? y/n : y

<table>
<thead>
<tr>
<th>xSuit</th>
<th>1suit</th>
<th>2suit</th>
<th>3suit</th>
<th>4suit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clubs</td>
<td>Spades</td>
<td>Diamonds</td>
<td>Hearts</td>
</tr>
<tr>
<td>xLnth</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>xStrn</td>
<td>weak</td>
<td>weak</td>
<td>weak</td>
<td>weak</td>
</tr>
<tr>
<td>xHcp</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>xRank</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>xSup</td>
<td>verygood</td>
<td>moderate</td>
<td>moderate</td>
<td>little</td>
</tr>
<tr>
<td>xTrck</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Spades (Jack 5 3)
Hearts (King 8)
Diamonds (Queen 5 4)
Clubs (King 7 5 3 2)

Name      north | Clevel nil | Part | south | Psuit | pass | Plevel 0
---|---|---|---|---|---|---|---
Even t     Dist (5 3 3 2) | Hcp 8 | Open nil
Qtrck 1.0   Dv nil | Cp nil | Vuln nil

Rnd nil     Play north | lhand 3 | mSet nil
mOpenr nil   mGame nil | Bsuit pass | Blevel 0
r0 nil      r1 nil | r2 nil | r3 nil
r4 nil      r5 nil | r6 nil | r7 nil

==) Rset0 : rule 0 fired
Mst:mSet = (Rset0)
==) Rset0 : rule 0 fired
Cst:Dv = 1 Cst:Cp = 9
Mst:mSet = (Rset10)
==) Rset10 : rule 25 fired
Mst:Blevel = 0 Mst:Bsuit = pass

*** Player north bids pass ***
Figure 3: Example of video terminal screen in the screen mode.
An example of the screen display produced in the screen mode is shown in Figure 3. Three windows are assigned to Cst, Mst and the suit vectors respectively. A fourth window is assigned to the display of rules. Currently all windows are displayed simultaneously since there is sufficient space on the terminal to display them simultaneously. As each rule is executed, the clause and affected variables are highlighted for a short period to allow for an easy check of why a rule fired or did not fire. This facility is extremely primitive at this stage. It only allows the user to watch. The extension of this monitor to an interactive debugging environment has not been carried out at the time of this report.

5. Performance Evaluation

As noted earlier, we were only able to implement the first 4 rounds of bidding. At first glance, 4 rounds of bidding seem insufficient to lead to any testable results. In practice, 4 rounds of bidding may be sufficient to reach game if the hands held by North and South are strong enough. This occurs when the opener opens with either a notrump or a two-level (weak) bid. When this occurs, both the high card points and the distribution of the hand of the opener are made known to his partner and the desired result should be determined within 2 or 3 rounds of bidding. Similarly, a response of a limited bid (e.g., simple raise, double raise, bid 1 or 2 notrump over partner's 1 level suit opening)
indicates a particular range for the high card points and a particular set of possible distributions. Based upon this information, the opener can then decide to continue bidding, bid game directly, inquire more about partner's holding or simply pass in the following few rounds. Bidding will also end within 2 or 3 rounds if both hands are weak.

Four rounds are not enough when both hands are fairly strong. Limited bids are not longer suitable to describe the hands and more investigation or exchange of information is needed leading to more than 4 rounds.

We have tested the program with more than a hundred examples. Some of these consist of weak or uninteresting hands (which result in a throw in, that is, four consecutive passes) leaving 35 interesting examples. These 35 examples and the corresponding response are shown in Appendix 2. In each of these examples, the cards for North are shown above those for South. The bidding sequence shown is that generated by the program. A "-" represents a bid of pass. The optimal contract is the desired result as determined by L. Y. Chang (the "expert"). The conclusion is the result of comparing the bidding sequence with the optimal contract. A conclusion of "successful" indicates that the contract reached by the bidding sequence is the same as the optimal contract. The conclusion will be "incomplete" when more than 4 rounds of bidding are required and "unsuccessful" when the bidding sequence deviates from
the optimal sequence.

Of the 35 examples, 25 of them is successful, 3 are incomplete, 7 are unsuccessful. This suggests that further debugging and refinement of the rules are necessary. This has not been completed at the time of this report. Also, at about the time that testing was carried out, it was realized that a more compact set of rules can be obtained if intermediate values of the deduced hand of the partner are used explicitly. It was decided that the affected portions of the rule set should be replaced. This is discussed in further detail in the next section.

6. Concluding Remarks

There are over 400 rules in our bidding system. It is estimated that another 300 or so rules will be needed to complete the bidding system; if the opposing partnership is allowed to interfere with the bidding process, an additional 800 rules may be needed.

For future revisions of this prototype intermediate conclusions should be used. This will reduce the number of rules required. Some of these should be

i) Distribution of partner's hand (1 suit, 2 suits, 3 suits and even distribution)

ii) Strength of partner's hand
Current status (forcing one round, forcing to game, non-forcing, slamming, game reached)

Suit determined (spades, hearts, diamonds, clubs, some combinations of the suits, notrump or undetermined)

Stopper in each suit

The use of these intermediate conclusions will represent a significant departure from the normal approach used in bridge books.
Appendix 1

Parameter definition

[1] HCP: high card point (A=4, K=3, Q=2, J=1)

[2] DV: distribution value (void +5, singleton +3, doubleton +1, 4-3-3-3 -1).


[4] Strength of suit

solid: length of suit greater than or equal to 7, suit contain AKQ

strong: length of suit greater than or equal to 4, contain AKQ, AKJ, or KQJ

moderate: length of suit greater than or equal to 4, contain AK, AQ, AJ, KQ, KJ, QJ10

weak: length of suit greater than or equal to 4, other than strong or moderate

[5] Distribution

even: (5-3-3-2), (4-4-3-2), or (4-3-3-3)

uneven: otherwise

[6] Quick trick

1/2: for Kx, QJx, KJx

1: for A, KQ

1-and-1/2: for AQ, AJ10

2: for AK, AQJ, KQJ

2-and-1/2: for AKJ

3: for AKQ

4: for AKQJ

[7] Support of suit

no: void or singleton
little: Jx, xx, xxx, Qx, Kx, Ax, QJ, KJ
moderate: Jxx, Qxx, xxx, Kxx, Axx, QJ, AQ, AK, AJ
good: QJx, Jxxx, Qxxx, KJx, Kxxx, Axx, AJx, AQx, AKx, KQx
very-good: QJxx, KJxx, KQxx, AJxx, AQxx, AKxx (and longer)

[8] Game reached: all contract above(include) 3NT except 4C, 4D
[9] Identity: the hand to bid (south or north).
[10] History of bidding sequence
[12] Last bid by own hand
[13] Last bid by partner
Appendix 2
More examples in evaluating the system

Ex1:
S: A
H: A K 10 8 7
D: 8 5
C: A J 9 8 2
S: K 4 3 2
H: 6
D: Q J 10 6 3 2
C: K 5

Optimal contract: 3NT
Conclusion: unsuccessful

Ex2:
S: Q J 7 6
H: K Q 8 7 5
D: K Q
C: K Q
S: A 10 9 5
H: J 9 4
D: J 10
C: 9 8 6 4

Optimal contract: 4H or 4S
Conclusion: successful

Ex3:
S: 10 4
H: J 9 7 6 2
D: A K J 9 5
C: 10
S: K J 5
H: K 8 5
D: 0 10 6
C: K 8 4 2

Optimal contract: 2H or 3H
Conclusion: unsuccessful

Ex4:
S: A 7
H: K 10 8 6 2
D: 8 7 3
C: Q J 5
S: K Q 8 4
H: A 9
D: 10 4 2
C: A K 8 3
Optimal contract: 3NT
Conclusion: successful

Ex5:
S: K 9 8 2
H: K Q J 7 6
D: A 7
C: 9 3

S: J 6 5
H: A 10 4 2
D: 0 6 3
C: A 7 2

Optimal contract: 4H
Conclusion: successful

Ex6:
S: K 9 8 6
H: Q J 5 3
D: K 8 4
C: 5 3

S: A J 7 5
H: K 10 3
D: A Q 4 2
C: Q 6

Optimal contract: 4S
Conclusion: successful

Ex7:
S: 0 10 7
H: A 3
D: 8 6 4
C: K Q 9 7 3

S: A K 9 8
H: K 10 3
D: A Q J
C: A J 8

Optimal contract: 6NT
Conclusion: successful

Ex8:
S: 4 2
H: K 5 3 2
D: J 6 4
C: 10 8 5 4

S: A K Q 10 7 6 3
H: A 4
D: K Q 5
C: 7

Optimal contract: 4S
Conclusion: successful

EX9:
S: J 5 3
H: K 8
D: Q 5 4
C: K 7 5 3 2
S: A Q 10 8 7 2
H: J 6
D: 9 7
C: 0 10 9
Optimal contract: 2S, but probably are not allowed to do

EX10:
S: K Q J 7 5 4
H: 9 7 2
D: 8
C: K Q J
S: A 9 6
H: A J 6
D: K Q 7 3
C: 9 6 5
Optimal contract: 4S
Conclusion: successful

EX11:
S: A K J 8 6
H: J 3
D: 6 2
C: K Q J 10
S: 0 10 3 2
H: 10 7 4
D: K Q 8 7 4
C: 4
Optimal contract: 3S
Conclusion: successful

EX12:
S: A 10 9 5
H: J 9 4
D: J 10
C: 9 8 6 4
S: Q J 7 6
H: K Q 8 7 5
D: K Q
C: K Q
Optimal contract: 4H or 4S
Conclusion: successful

EX13:  
S: Q 10 9 3
H: J 6 2
D: Q 5 4
C: K 7 5 3 2

```
S W N E
2N - 3C -
3H - 3N -
```

S: A J 6 5
H: A K 9 3
D: K Q J
C: A 3

Optimal contract: 3NT
Conclusion: successful

EX14:  
S: K 7 4
H: 9 5
D: A K 8 5 2
C: 7 5 2

```
S W N E
1H - 2D -
3N - - -
```

S: A J 3
H: A K Q 10 2
D: 7 4
C: A Q 3

Optimal contract: 3NT
Conclusion: successful

EX15:  
S: 3
H: A K 8 5 2
D: A 10 7 4
C: 10 8 4

```
S W N E
1H - 2H -
```

S: K Q 8 7
H: G J 10 9 4
D: 8 3
C: A 9

Optimal contract: 4H
Conclusion: unsuccessful

EX16:  
S: K J 4 2
H: Q 8 2
D: K 3
C: 7 5 4 2

```
S W N E
1H - 1S -
2H - - -
```

S: A Q 3
H: A K 10 7 5 4
D: J 10 5
C: 6

Optimal contract: 4H
Conclusion: unsuccessful

EX17:
S: Q 8 7
H: Q J 5
D: 7 2
C: A K 10 6
S: K 3
H: 10 5 2
D: A K Q 10 9 6 5
C: 3

Optimal contract: 3NT
Conclusion: successful

EX18:
S: J 6 4
H: 9 5 3 2
D: K 9 6 5
C: J 6
S: A 9 2
H: A Q 7
D: A 8 4
C: Q 7 3 2

Optimal contract: 3NT
Conclusion: successful

EX19:
S: K Q 7 6
H: A K 9 2
D: A J 6 5
C: 10
S: A J 3
H: 8 6 4
D: 7
C: A J 9 8 4 2

Optimal contract: 4S
Conclusion: incomplete bidding

EX20:
S: A K Q
H: 9 4
D: Q 7 6 2
C: K 10 8 5

Conclusion: unsuccessful
Optimal contract: 3NT
Conclusion: successful

EX21:
S: A J 9 7 6
H: Q J 7 5
D: 7 2
C: K 7

S: K 2
H: A 10 6
D: J 5 3
C: A Q J 6 2

Optimal contract: 2H or 2S
Conclusion: successful

EX22:
S: Q 5 3
H: K Q 9
D: 4 2
C: A K 7 5 4

S: K 9 8
H: A 8
D: A K 8 3
C: Q 9 8 3

Optimal contract: 3NT
Conclusion: successful

EX23:
S: 7 5
H: J 6 4
D: A K 8 5 4
C: K J 7

Optimal contract: 2D or 3D
Conclusion: successful

EX24:
S: A
H: 10 8 2
D: A Q J 9 8 2
Optimal contract: 3S or 3D
Conclusion: successful

EX25:
S: K
H: Q 10 9 7 5 2
D: A Q J
C: A 4 2

S: J 10 8 6 5
H: K 3
D: K 6 2
C: 9 8 3

Optimal contract: 2H or 3H
Conclusion: unsuccessful

EX26:
S: Q 7 6
H: A J 9 7 6 5
D: 10 2
C: Q J

S: 5
H: K Q 4 3
D: A 5 3
C: K 7 6 3 2

Optimal contract: 4H
Conclusion: successful

EX27:
S: A 8 4
H: K 8 6 4
D: K 9 8 2
C: 10 5

S: Q 6 3
H: A Q 7 2
D: A Q 10
C: A 4 2

Optimal contract: 3NT
Conclusion: successful

EX28:
S: 10 8 6 5
H: 7 6 3
D: A K Q 5
C: 8 4

S: A K 9
H: K J 9 4 2
D: 8 7 3
C: 6 5

Optimal contract: 2H
Conclusion: unsuccessful

EX29:
S: -
H: 9 8 6
D: K Q J 4
C: A K J 10 5 3

S: A K 4
H: Q 10 5 2
D: A 10 8 6
C: 8 4

Optimal contract: 3NT
Conclusion: incomplete bidding

EX30:
S: 9 6 5
H: K 10 3 2
D: A K Q 10
C: 8 3

S: A K 8 3
H: 9 7
D: Q J 7 6 2
C: 3 2

Optimal contract: 2D
Conclusion: unsuccessful

EX31:
S: K Q 8 2
H: 9 2
D: Q 10 8 4
C: Q 4

S: -
H: A J 3
D: A J 9 7 2
C: A K 9 8 6

Optimal contract: 5D
Conclusion: incomplete bidding
EX32:
S: K J 9 8 6 4  
H: A Q 5  
D: 7 5  
C: Q 7  

S: Q 5  
H: K J 8 4 2  
D: J 10 8 3  
C: A 10  

Optimal contract: 4H  
Conclusion: successful

EX33:
S: 10 9 7 6  
H: 8 4 3  
D: 10 2  
C: A Q 5 4  

S: J 5  
H: A K 10 9  
D: A 9 5  
C: K J 8 3  

Optimal contract: 1NT  
Conclusion: successful

EX34:
S: Q 10 8  
H: K 10 9 8  
D: A 10 6  
C: Q 6 3  

S: J 8 6 5  
H: A J 7 4  
D: K J 4  
C: J 2  

Optimal contract: 2H  
Conclusion: unsuccessful

EX35:
S: K 8  
H: 10 8 7 4 2  
D: Q 10 3  
C: J 5 2  

S: A 9 2  
H: A Q 7  
D: A 8 4  
C: Q 7 3 2  

Optimal contract: 1NT  
Conclusion: successful
Appendix 3: Description of Global Database Vectors

1. Mst contains the following elements:

- **Rnd**: Name of player who is supposed to bid
- **Play**: Number of consecutive passes
- **Suit**: Suit of current bid
- **Level**: Level of current bid
- **mSet**: Context list
- **mOpener**: Name of player who opens the bidding
- **mGame**: Has game being reached?
- **Shist**: History of suit of bids of all players
- **Lhist**: History of level of bids of all players
- **r0 - r7**: Scratch registers

2. Cst contains the following elements:

- **Even**: Is the hand even?
- **Hcp**: Total HCP of hand
- **Dist**: Distribution of hand
- **Dv**: DV of hand
<table>
<thead>
<tr>
<th>5</th>
<th>Cp</th>
<th>Total CP of hand</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Open</td>
<td>Is Hcp $\geq 11$</td>
</tr>
<tr>
<td>7</td>
<td>Vuln</td>
<td>Vulnerable or not?</td>
</tr>
<tr>
<td>8</td>
<td>Name</td>
<td>Name of owner of this hand</td>
</tr>
<tr>
<td>9</td>
<td>Part</td>
<td>Name of partner</td>
</tr>
<tr>
<td>10</td>
<td>Csuit</td>
<td>Suit of last bid by owner</td>
</tr>
<tr>
<td>11</td>
<td>Clevel</td>
<td>Level of last bid by owner</td>
</tr>
<tr>
<td>12</td>
<td>Psuit</td>
<td>Suit of last bid by partner</td>
</tr>
<tr>
<td>13</td>
<td>Plevel</td>
<td>Level of last bid by partner</td>
</tr>
<tr>
<td>14</td>
<td>Qtrck</td>
<td>Sum of quick tricks in hand</td>
</tr>
<tr>
<td>15</td>
<td>Phst</td>
<td>History of bids by partner</td>
</tr>
<tr>
<td>16</td>
<td>Chst</td>
<td>History of bids by partner</td>
</tr>
</tbody>
</table>

3 Spades, Hearts, Diamonds and Clubs have the following elements

| 1 | xSuit | Suit represented by this vector |
| 2 | xLnth | Number of cards in suit |
| 3 | xHcp | HCP in suit |
| 4 | xCard | List of cards in this suit |
| 5 | xStrn | Strength in this suit |
6 xRank  Rank of this suit 1suit -> 1, 2suit -> 2, ...

7 xSup  Support in this suit

8 xTrck  Quick tricks in this suit
Chapter 18

Air Flight Scheduler Expert System

A. J. Vayda and W. Y. Kim
1. INTRODUCTION

1.1 STATEMENT OF PROBLEM

A serious problem facing the airline industry today is the increasing demand that is being placed on the airports. Following the deregulation of the industry there has been a significant increase in the number of airlines in the business and in the number of flights that are made. This increase has continued to such levels that the demands for service at many airports have severely strained their resources.

The primary symptom of this problem is the delay of most flight departures and arrivals because of limited runway facilities and air traffic controllers. Currently there is no method for airlines and airports to work together to address this problem. The standard procedure at all airports is to provide service to incoming and outgoing flights on a first-come-first-serve basis. This often results in taxiways full of planes waiting to take-off and a number of planes in holding patterns around the airport waiting to land. A tremendous amount of plane fuel is wasted because of this, and it is impossible to measure the cost of the waste of time (not to mention the irritation) for the people involved. While most airlines are concerned about this problem there is not much that can be done about it. The intense competition in the
industry makes it difficult for cooperation in the scheduling of flights. Each airline has too much to lose in allowing a competitor a slightly better flight schedule.

1.2. MOTIVATION

The need for a universal flight schedule for all airlines and airports is evident but the necessity that it be fair to everyone has prevented it from happening. The job is too complex to be handled by any standard scheduling algorithm. The many interrelationships that must be taken into account require a scheduling system that makes decisions as a human would but can handle the overwhelming complexity of the data. What is required is a system to schedule a time slot for each flight's departure and arrival at the appropriate airports. The system should flatten out the peaks and valleys in the flight schedule so that a steady load is maintained on the airport and its maximum capacity is never exceeded. Airports would still serve flights on a first-come-first-serve basis, but the flight load would be steady instead of cyclical as is currently the case. There will always be some variability in flight times due to weather conditions and other factors but the congestion due to uncoordinated flight scheduling is the biggest problem. An expert system is well suited for a task such as this.
The major problem to be overcome in the development of an expert system for this task is that there are no experts currently working on this topic. The only people who have any experience at all in this area are airline executives and airport managers. However, these people only have experience in their area of the problem. Thus it became necessary for us to become experts ourselves [1,2]. This required a great deal of research of Federal Aviation Administration (FAA) reports, Department of Transportation (DOT) reports, and conversations with an air-traffic controller at the West Lafayette airport and a professor of Aeronautical Engineering [3-9].

1.3. GROUP MEMBER RESPONSIBILITIES

Responsibilities were divided evenly between the group members throughout the execution of the project. Whenever a task needed to be done one of the partners would volunteer to take the primary responsibility for it. The other partner would then offer help as necessary as well as offering suggestions and criticism. For program code there was a continual tradeoff of programs where each person made improvements to the other's code. Kim took primary responsibility for the initial research and interviews, and in the programming of the scheduling expert. Alan took primary responsibility for the programming of the knowledge base procedures, the weight experts, and the
input, output, and user interface procedures.

2. SYSTEM ORGANIZATION

This system falls into the basic category of design or resource allocation applications where the goal is to configure a group of objects under various constraints. The block diagram of the system is shown in Fig. 1. R1 is an example of another system in this category. The main difference between our system and R1 is that R1 configures a variety of components with different constraints for each component [1]. Our system configures flights which all are affected by the same type of constraints. Each flight to be scheduled has a requested time as a goal but it is expected that not every flight will be scheduled at its requested time. (otherwise the system would not be necessary) The constraints are the availability of runways for liftoff and landing and are affected by all other flights.

2.1 EVALUATION

A simple measure that could be used to evaluate the success of the system in scheduling a group of flights is the factor that is obtained by dividing sum of all of the differences in scheduled flight times from the requested times by the total number of flights scheduled. A more useful evaluation function would need to take into account
the importance of the various flights. It is expected that less important flights (those assigned lower weights by the system) will be moved more frequently and farther in time than the more important flights. Both of these methods also overlook the difficulty of the original scheduling task. One measure of the difficulty of the original scheduling problem is obtained by dividing the number of flights requesting location in time slots where there is no room by the total number of flights to be scheduled. This is a very simplistic approach but it might provide some useful information.

The third method that can be used for evaluation is that of comparing example schedules with schedules prepared by scheduling experts. This is the most reasonable approach because the measurement of the difficulty and the success of the scheduling problem and solution is actually just as complex as the scheduling task itself.

2.2. IMPLEMENTATION

The program is implemented in Lisp. It contains over 750 lines of code which include approximately 90 function definitions.

3. KNOWLEDGE BASE
The knowledge base consists of three parts: the flight data base, the airport data base, and the schedule data base.

The flight data base consists primarily of information supplied by each airline in the form of requests for flights to be scheduled. It contains:

1) flight number - The flight number is a combination of numbers and letters which identifies a specific flight and should be distinct from all other flight numbers. The flight number is used as the key for accessing the other information in the data base.

2) airline name - The airline name is the name of the airline making the flight.

3) origin - The origin is the name of the airport at the origin of the flight.

4) destination - The destination is the name of the airport at the destination of the flight.

5) departure time - The departure time is the desired departure time as requested by the airline.

6) flight time - The flight time is the length of the flight from the origin to the destination. The arrival time may be calculated by adding the flight time to the departure time.
7) scheduled departure time — The scheduled departure time is the actual departure time that has been assigned by the scheduling system.

8) weight — The weight is a weighting factor that is assigned by the scheduling system and is used to guide scheduling decisions for resolving conflicts.

Items one through six are obtained directly from each airline. The scheduled departure time and the weight are assigned by the system.

The airport data base consists primarily of information obtained from each airport. It contains:

1) airport name — The airport name is the name of the airport. The airport name should match with the origin and destination elements in the flight data base.

2) maximum rate of arrivals and departures — The maximum rate of arrivals and departures is the maximum number of flights that may takeoff or land at the airport in a time slot of given size.

3) time zone — The time zone is the time zone that the airport is located in. The time zone is used to make conversions so that arrival and departure times are always correct for the time zone in that location.
4) departure list and arrival list — The departure list and the arrival list contain entries for each flight scheduled for arrival or departure at the airport. The flights are grouped and sorted according the appropriate time slot.

5) weight — The weight is a weighting factor assigned by the scheduling system and is used to guide scheduling decisions for resolving conflicts.

Items one through three are obtained directly from each airport. The departure and arrival lists and the weight are created by the system. The time slot is a basic system parameter which defines the grain of the scheduling system. It is the interval of time between two adjacent locations on the schedule. Normally the size of the time interval is in the range of one to ten minutes. This parameter would have to be agreed upon by all airlines and airports because it is used in the data files that they must supply.

The schedule database is actually implemented through extensions to the flight and airport data bases. It consists of:

1) scheduled flight times — from the flight data base

2) flight weights — from the flight data base
3) departure and arrival lists - from the airport data base

4) airport weights - from the airport data base

These values are assigned by the system and vary as the system proceeds in the scheduling of all flights. They define the state of the flight schedule at any time.

Data is accessed and stored through the use of "get" and "put" routines which store the information as property values for each flight and airport. For flights, the flight number is used as the key. For airports, the airport name is used as the key. The flight lists are stored in the form of association lists. The actual manner of storage in property lists is shielded from the rest of the program by a group of "get" and "put" routines so that a different or more efficient data storage format may be implemented by simply changing the access and storage routines. Property lists were used for the sake of flexibility in the design of the system.

4. INFERENCE ENGINE

The rule based approach to expert system design was ruled out for this project for a number of reasons. The large number of flights and airports would create a very large data base. The relationships between the flights
and airports are sufficiently complex to make the formulation of rules a very difficult task. The large search space combined with the large number of very complex rules would produce an inefficient system. Instead it was decided to build the system around a "scheduling expert" that would be designed to perform the scheduling in a manner similar to the way a human expert would. It would not use brute force trial and error, but a group of effective heuristics. In order to make the system flexible the scheduler was designed to make decisions regarding conflicts in the scheduling of flights by checking a weight value for each flight. This method allows the scheduler to work without the need to access many different values from various databases. It only needs access to the weights of the flights. The scheduler could then be designed without worrying about the changes in the rest of the system.

4.1 SCHEDULING EXPERT

The basic method that the scheduler follows is to schedule all flights to meet the constraints at the busiest airport, and then repeating the process for the rest of the airports in an order defined by the airport weight expert. The scheduling of any airport could cause conflicts to arise at other airports so the scheduler continues to loop through all airports as long as any conflicts
are present. The flow chart of scheduling expert is shown in Fig. 2.

For each airport the scheduler schedules flights for each time slot in order. Typically it would start at midnight and schedule flights through the day until the next midnight. As each time slot is examined to make the necessary scheduling a group of four rules are used to make the decisions. These rules implement a strategy that is comparable to buoyancy. Flights with heavier weights sink to the bottom of the slots and are less likely to be moved than flights with lighter weights which will float to the top of the time slots. Flights above the maximum level for flights in a slot will be moved to adjacent slots until they sink below the maximum level.

1) The number of flights to be scheduled in the time slot is compared to the maximum number allowed at the airport.

a) If it is less than or equal to the maximum then the flights are scheduled as requested. The scheduler then moves on to the next time slot.

b) If the number of flights to be scheduled in the time slot is greater than the maximum allowed then one flight at a time is selected to be scheduled. The flight to be scheduled is selected by sorting all of
the flights in the slot according to weight and choosing the one that is ranked at the position one greater than the maximum number of flights that may be scheduled in the slot. The scheduler then moves on to the rest of the rules to schedule this flight.

2) If the current slot is full then the number of available locations in the preceding and following time slots are compared.

   a) If there are openings in either or both slots then the flight is scheduled in the slot with the most openings.

   b) If there are the same number of openings in both slots then the flight is scheduled in the slot where its weight has the the best rank.

   c) If the rank is the same for both slots then schedule the flight in the later slot.

   d) If the flight has been scheduled then the scheduler selects the next flight to be scheduled for the time slot and repeats the process.

   e) If there are no openings in either slot then the scheduler moves on to the rest of the rules.
3) If both the preceding and following slots are filled with flights then the rank of the weight of the flight in both slots is compared with the maximum allowed in those slots.

a) If the rank in either slot places the flight under the maximum then schedule the flight in the slot where it will be placed the lowest.

b) If the rank of both flights is the same and is under the threshold then schedule the flight in the later slot.

c) If the flight has been scheduled then the scheduler will use the same procedure to schedule the flights in the slot where the flight has been scheduled. The scheduler will then return to select the next flight to be scheduled for the current time slot and repeat the process.

d) If the flight cannot be scheduled in either slot then the scheduler moves on to the next rule.

4) The flight cannot be scheduled in either the earlier or later time slot so the scheduler will use the same rules (1, 2 and 3) to attempt to schedule the flight in time slots one slot earlier and one slot later.

a) The scheduler will continue to expand the area of
testing for scheduling a flight until it schedules the flight or reaches a specified limit. The limit would typically be one hour from the requested time. If the flight cannot be scheduled within that range then its weight is increased and the process starts over at the requested time slot.

The scheduler uses both iteration and recursion to carry out the scheduling process. It iterates through all of the time slots of an airport in order. It calls itself recursively when it schedules a flight in a slot that is over the limit or if there are more flights to be scheduled in the current slot. It uses iteration to expand the distance of the slots that it tests from the requested slot. This method of combining iteration and recursion results in a depth first type of scheduling scheme. It has reasonable efficiency and handles the scheduling in much the same way as a human expert would.

4.2. FLIGHT WEIGHT EXPERT

The use of weights by the scheduler requires that another part of the system assign the weights to each flight. The "flight weight expert" part of the system is used to make initial assignments of weights to each flight. It follows the basic rule based method and is designed so that rules can be modified easily. It uses
the information provided by the airlines and airports to calculate a weight for each flight. The rules are divided into two groups. The first group is used to make weight adjustments due to factors that are particular to each flight. The second group is used to make weight adjustments for all flights at a particular airport.

Each weight is a number greater than or equal to one. The scale is open ended but typical weights are in the range from one to two. Weights are usually modified through multiplication by a factor in the range of one to 1.25. The scheduler may modify the weights of flights during the scheduling process so that the weights at the end of the process are different from those assigned at the beginning.

One of the rules used increases the weight of a flight by an amount that depends on the airline for that flight. Bigger airlines get higher preference. A second rule increases the weight of a flight by an amount that depends on the length of the flight. Longer flights have get higher preference. Another rule increases the weights of flights that have as an origin or destination one of the busier airports. Although these rules seem deceptively simple they have a large affect on the scheduling of the flights.
4.3. AIRPORT WEIGHT EXPERT

The airport weight expert assigns weights to each airport that are then used to set the order in which the scheduler schedules the airports. Currently this system uses only one rule to make its assignments. It assigns weights according to a factor obtained by dividing the total number of flights using the airport by the maximum number of flights serviced per time slot. This factor is an excellent indicator of how busy each airport is. The use of one rule does not make this an actual expert. It is expected that this part of the system would be expanded to expert status by making use of more information available from the airports. Other factors which could be considered are: variations due to seasonal weather conditions, the importance of the airport as a hub for connecting flights, and air traffic that is not covered by the scheduling system.

5. KNOWLEDGE ACQUISITION

The input for this system is provided directly by the airports and airlines. It is assumed to be error free as there is no method to check for errors. Data is provided in files that are loaded into the system on request.

Currently the system operates only in a batch mode. All flight and airport information must be entered before
any scheduling may be done. It would not be too difficult to modify the system so that it could be used in an incremental mode. This would allow new flights and/or airports to be added to the schedule with minimum disruption of the rest of the flights which have already been scheduled. Eventually the system could be modified to handle real time operations. Then it could use information about current flights and conditions to adjust the schedule on a minute to minute basis. This could be very useful when flights are disrupted by bad weather or a closed airport. It could be used to automatically reroute flights to other airports and make other necessary adaptations to the schedule.

6. EXPERIMENTAL RESULTS

A demonstration of the system scheduling 20 flights at 4 airports is useful in the understanding of the operation of the system. It is complex enough to make the system do some work but easy enough for someone to understand. For the sake of simplicity, the time slot size has been selected to be one hour and each flight has a length of one, two, or three hours.

Table 1 contains a table of the flights that the airlines have requested to be scheduled. Table 2 contains a table of the flights as scheduled by the system. Table 3 contains the schedule output by the system for the
airports. Table 4 contains the schedule output by the system for the airlines. Appendix A contains a trace of the program with messages printed at various parts of the scheduling process. Some definitions that are used in the trace:

1) length - the number of flights to be scheduled in the current time slot

2) thresh - the maximum input and output rate for the airport

3) flight-list-in-flat-aux - the list of flights that is given to the procedure flat-aux

4) time-slot - the time slot that the scheduler is currently trying to schedule

5) sorted-flight-list - when rule 3 is called the system sorts the weights of the flights in the earlier and later slots in order to compare them with the weight of the current flight

6) early-num and later-num - the number of spaces available for flights in earlier or later time slot

It can be seen from the trace how the scheduler attempts to schedule each time slot in order but when it is resolving conflicts it calls itself recursively.
Table 1. Requested flight schedule

<table>
<thead>
<tr>
<th>Schedule Designed by System</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>500</th>
<th>600</th>
<th>700</th>
</tr>
</thead>
<tbody>
<tr>
<td>NY</td>
<td>*n3t5</td>
<td>n3h4</td>
<td>t2n4</td>
<td>c2n5</td>
<td>*h3n4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>*n4t5</td>
<td>n2h3</td>
<td>n3c6</td>
<td>c1n4</td>
<td>*n4t6</td>
<td>t4n6</td>
<td>*h4n5</td>
</tr>
<tr>
<td>PITT</td>
<td>t2c3</td>
<td>h2t3</td>
<td>*n3t5</td>
<td>c4t5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>t2n4</td>
<td>c2t3</td>
<td>*n3t5</td>
<td>t4n6</td>
<td>*t4h5</td>
<td>*c3t4</td>
<td>*n4t6</td>
</tr>
<tr>
<td>CHI</td>
<td>c2n5</td>
<td>*h21c4</td>
<td>*h22c4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>c1n4</td>
<td>c2t3</td>
<td>t2c3</td>
<td>c4t5</td>
<td>*c3t4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHIL</td>
<td>*h21c4</td>
<td>*h4n5</td>
<td>n2h3</td>
<td>c4t5</td>
<td>*t4h5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The symbol "*" designates the flight number from original request.

Table 2. Scheduled flight by system Air-Scheduler

Airport: NY

Departures:

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>Flight-Number</td>
<td>Airline</td>
<td>Origin</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>300</td>
<td>n3c6</td>
<td>KAL</td>
<td>CH</td>
</tr>
<tr>
<td>500</td>
<td>n3h4</td>
<td>TWA</td>
<td>PH</td>
</tr>
<tr>
<td>500</td>
<td>n4t6</td>
<td>EST</td>
<td>PT</td>
</tr>
</tbody>
</table>

**Arrivals:**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>c1n4</td>
<td>EST</td>
<td>CH</td>
</tr>
<tr>
<td>500</td>
<td>t2n4</td>
<td>KAL</td>
<td>PT</td>
</tr>
<tr>
<td>600</td>
<td>c2n5</td>
<td>TWA</td>
<td>CH</td>
</tr>
<tr>
<td>700</td>
<td>h4n5</td>
<td>JAL</td>
<td>PH</td>
</tr>
</tbody>
</table>

**Airports:**

**PT**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>n4h4</td>
<td>EST</td>
<td>NY</td>
</tr>
<tr>
<td>400</td>
<td>t2n4</td>
<td>KAL</td>
<td>NY</td>
</tr>
<tr>
<td>400</td>
<td>t2c3</td>
<td>USA</td>
<td>CH</td>
</tr>
<tr>
<td>500</td>
<td>t4n6</td>
<td>JAL</td>
<td>NY</td>
</tr>
<tr>
<td>500</td>
<td>t4h5</td>
<td>USA</td>
<td>PH</td>
</tr>
</tbody>
</table>

**PH**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>n3t5</td>
<td>KAL</td>
<td>NY</td>
</tr>
<tr>
<td>500</td>
<td>c4t5</td>
<td>USA</td>
<td>CH</td>
</tr>
<tr>
<td>600</td>
<td>c3t4</td>
<td>JAL</td>
<td>CH</td>
</tr>
<tr>
<td>700</td>
<td>n4t6</td>
<td>EST</td>
<td>NY</td>
</tr>
<tr>
<td>Time</td>
<td>Flight-Number</td>
<td>Airline</td>
<td>Destination</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>100</td>
<td>h21c4</td>
<td>TWA</td>
<td>CH</td>
</tr>
<tr>
<td>200</td>
<td>h2t3</td>
<td>KAL</td>
<td>PT</td>
</tr>
<tr>
<td>300</td>
<td>h22c4</td>
<td>TWA</td>
<td>CH</td>
</tr>
<tr>
<td>500</td>
<td>h3n4</td>
<td>KAL</td>
<td>NY</td>
</tr>
<tr>
<td>600</td>
<td>h4n5</td>
<td>JAL</td>
<td>NY</td>
</tr>
</tbody>
</table>

**Arrivals:**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>n4h5</td>
<td>EST</td>
<td>NY</td>
</tr>
<tr>
<td>300</td>
<td>n2h3</td>
<td>EST</td>
<td>NY</td>
</tr>
<tr>
<td>400</td>
<td>n3h4</td>
<td>TWA</td>
<td>NY</td>
</tr>
<tr>
<td>600</td>
<td>t4h5</td>
<td>USA</td>
<td>PT</td>
</tr>
</tbody>
</table>

**Departures:**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>c1n4</td>
<td>EST</td>
<td>NY</td>
</tr>
<tr>
<td>200</td>
<td>c2t3</td>
<td>EST</td>
<td>PT</td>
</tr>
<tr>
<td>200</td>
<td>c2n5</td>
<td>TWA</td>
<td>NY</td>
</tr>
<tr>
<td>400</td>
<td>c4t5</td>
<td>USA</td>
<td>PT</td>
</tr>
<tr>
<td>500</td>
<td>c3t4</td>
<td>JAL</td>
<td>PT</td>
</tr>
</tbody>
</table>

**Arrivals:**

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>t2c3</td>
<td>USA</td>
<td>PT</td>
</tr>
<tr>
<td>500</td>
<td>h21c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
<tr>
<td>600</td>
<td>h22c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>t2c3</td>
<td>USA</td>
<td>PT</td>
</tr>
<tr>
<td>500</td>
<td>h21c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
<tr>
<td>600</td>
<td>h22c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>t2c3</td>
<td>USA</td>
<td>PT</td>
</tr>
<tr>
<td>500</td>
<td>h21c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
<tr>
<td>600</td>
<td>h22c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Flight-Number</th>
<th>Airline</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>t2c3</td>
<td>USA</td>
<td>PT</td>
</tr>
<tr>
<td>500</td>
<td>h21c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
<tr>
<td>600</td>
<td>h22c4</td>
<td>TWA</td>
<td>PH</td>
</tr>
</tbody>
</table>

**Table 3. Airport Flight Schedule**
# AIRLINE FLIGHT SCHEDULES

<table>
<thead>
<tr>
<th>Airline</th>
<th>Flight Number</th>
<th>Liftoff-Time</th>
<th>Origin</th>
<th>Dest.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EST</td>
<td>c1n4</td>
<td>100</td>
<td>CH</td>
<td>NY</td>
</tr>
<tr>
<td></td>
<td>n4h5</td>
<td>100</td>
<td>NY</td>
<td>PH</td>
</tr>
<tr>
<td></td>
<td>c2t3</td>
<td>200</td>
<td>CH</td>
<td>PT</td>
</tr>
<tr>
<td></td>
<td>n2h3</td>
<td>200</td>
<td>NY</td>
<td>PH</td>
</tr>
<tr>
<td></td>
<td>n4t6</td>
<td>500</td>
<td>NY</td>
<td>PT</td>
</tr>
<tr>
<td>JAL</td>
<td>t4n6</td>
<td>400</td>
<td>PT</td>
<td>NY</td>
</tr>
<tr>
<td></td>
<td>c3t4</td>
<td>500</td>
<td>CH</td>
<td>PT</td>
</tr>
<tr>
<td></td>
<td>h4n5</td>
<td>600</td>
<td>PH</td>
<td>NY</td>
</tr>
<tr>
<td>KAL</td>
<td>h2t3</td>
<td>200</td>
<td>PH</td>
<td>PT</td>
</tr>
<tr>
<td></td>
<td>n3t5</td>
<td>200</td>
<td>NY</td>
<td>PT</td>
</tr>
<tr>
<td></td>
<td>t2n4</td>
<td>200</td>
<td>PT</td>
<td>NY</td>
</tr>
<tr>
<td></td>
<td>n3c6</td>
<td>300</td>
<td>NY</td>
<td>CH</td>
</tr>
<tr>
<td></td>
<td>h3n4</td>
<td>500</td>
<td>PH</td>
<td>NY</td>
</tr>
<tr>
<td>TWA</td>
<td>h21c4</td>
<td>100</td>
<td>PH</td>
<td>CH</td>
</tr>
<tr>
<td></td>
<td>c2n5</td>
<td>200</td>
<td>CH</td>
<td>NY</td>
</tr>
<tr>
<td></td>
<td>h22c4</td>
<td>300</td>
<td>PH</td>
<td>CH</td>
</tr>
<tr>
<td></td>
<td>n3h4</td>
<td>300</td>
<td>NY</td>
<td>PH</td>
</tr>
<tr>
<td>USA</td>
<td>t2c3</td>
<td>200</td>
<td>PT</td>
<td>CH</td>
</tr>
<tr>
<td></td>
<td>c4t5</td>
<td>400</td>
<td>CH</td>
<td>PT</td>
</tr>
<tr>
<td></td>
<td>t4h5</td>
<td>500</td>
<td>PT</td>
<td>PH</td>
</tr>
</tbody>
</table>

Table 4. Airline Flight Schedule
7. CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

7.1. CONCLUSIONS

The goal of this project was to build a prototype expert system that would do a good job of scheduling flights and was flexible enough to be improved to a truly expert level of proficiency without the necessity of making major modifications to the program. We have been successful in achieving this goal. Although there is always room for improvement, the system does an outstanding job of constructing a flight schedule. The program was designed for generality and adaptability throughout so it should be easy to make improvements.

The scheduling expert received the most attention during the development of the project. It is believed that this part of the system does not need many more improvements. The flight weight expert and the airport weight expert on the other hand were designed to be flexible as it is expected that these parts of the system will need to be altered as conditions in the industry change. Our development of these experts was not as complete as the development of the scheduling expert. The completion of the flight weight expert and the airport weight expert would require the agreement of all airport managers and airline executives involved. The expertise that this system is attempting to capture is actually the consensus of
the expertise of the managers and executives. The few rules that are used do produce a very proficient system. It is expected that more rules will not significantly improve the schedules developed by the system but will help the system adapt the schedules to the desires of the airports and airlines. This is the type of fine tuning that can be carried out very easily on our expert system.

7.2 SUGGESTIONS FOR FURTHER WORK

The first area that should be considered for further work is the addition of more rules for the flight weight expert and the airport weight expert. The rationale for this has been explained above. A second area for improvement is the user interface. The system should be switched from a menu driven system to a simpler "start it and watch it go" type of system. This would allow inexperienced users access to the system. It would only require the answering of a few prompts by the user to make the system work. A third area that could be improved is the formatting of the output by the system. Currently only two output formats are available. One output format is designed for use by the airlines and the other is designed to be used by the airports. It would be useful to have the user specify exactly what is desired in the output and how it should be arranged. The creation of a time zone compensation routine is the fourth improvement that should be made
to the system. Currently a dummy routine is used which makes no compensations. This simplifies the evaluation of the output schedules because all airports are assumed to be in the same time zone.

The next major step in the evolution of the system would be to make it operate in an incremental mode. This would allow it to make additions and corrections to a flight schedule with a minimum disruption of currently scheduled flights. A long term goal for the system would be the conversion to real time operation. As a real time flight control system it would use information on the status of all flights, weather conditions, unexpected delays, and unscheduled aircraft to make decisions that are currently made by air traffic controllers. These would be decisions involving when and on what runway to land, what route to take to the destination, and when a flight should be delayed because a connecting flight has been delayed. This type of system would be particularly useful in emergency situations when there are major disruptions in the flight schedule.
REFERENCES


[8] FAA, USDT "FAA Plan and programs for Future Airport and Air Traffic Control System", 431-c-8, 15954-M

Block diagram of AIR-SCHEDULER

Figure 1
Flow-chart of Flat-aux

Figure 2.

c-f: current flight to be rescheduled

c-s: current working slot

max: max-i-o rate

take the 1st flight exceeding max-i-o rate as current flight

look for the room at the first level neighborhood
APPENDIX A

A Demonstration

Please enter the number of the desired option

1 - Exit
2 - Initialize global variables
3 - Load flight data file
4 - Load airport data file
5 - Set time slot size
6 - Schedule all flights as requested
7 - Compute weights for airports and sort list
8 - Print weights for airports
9 - Compute weights for all flights
10 - Print weights for all flights
11 - Flatten all airports
12 - Print schedules for airports
13 - Print schedules for airlines

Option: 11

FLAT-ALL called

Flattening NY Airport

Time slot: 0
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is : nil
### <slot-they-were-scheduled-in> is : 0

Time slot: 100
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is : nil
### <slot-they-were-scheduled-in> is : 100

Time slot: 200
Flight list in flat-aux: (n2h3)
Length: 1  Thresh: 2
### <flights_scheduled_in_slot> is : (n2h3)
### <slot-they-were-scheduled-in> is : 200

Time slot: 300
Flight list in flat-aux: (n3t5 n3c6 n3h4)
Length: 3  Thresh: 2
Sorted flight list: (n3c6 n3h4 n3t5)
Current flight: n3t5
k: 1
### <earlier-time-slot> is : 200
### <later-time-slot> is : 400
### <earlyphrnum> is : -1
### <latternum> is : 3
### <weight-of-cflight> is : 1.333904
### <weight-list-in-early> is : (1.273272)
### <weight-list-in-later> is :
(1.420848 1.297758 1.333904 1.308736 1.273272)
### <rank-of-c-flight-in-early> is : 1
Time slot: 300
Flight list in flat-aux: (n3c6 n3h4)
Length: 2
### <flights_scheduled_in_slot> is: (n3c6 n3h4)
### <slot-they-were-scheduled-in> is: 300

Time slot: 400
Flight list in flat-aux: (c1n4 h3n4 t2n4 n4t6 n4h5)
Length: 5
Sorted flight list: (c1n4 t2n4 n4t6 h3n4 n4h5)
Current flight: n4t6
k: 1
### <earlier-time-slot> is: 300
### <later-time-slot> is: 500
### <early-num> is: 0
### <later-num> is: 0
### <weight-of-c-flight> is: 1.308736
### <weight-list-in-early> is: (1.448172 1.34673)
### <weight-list-in-later> is: (1.50282 1.248786)
### <rank-of-c-flight-in-early> is: 3
### <rank-of-c-flight-in-later> is: 2
### <Scheduled-in-later> is: n4t6

Time slot: 500
Flight list in flat-aux: (c2n5 h4n5 n4t6)
Length: 3
Sorted flight list: (c2n5 n4t6 h4n5)
Current flight: h4n5
k: 1
### <earlier-time-slot> is: 400
### <later-time-slot> is: 600
### <early-num> is: 2
### <later-num> is: -1
### <weight-of-c-flight> is: 1.248786
### <weight-list-in-early> is: (1.420848 1.297758 1.333904 1.273272)
### <weight-list-in-later> is: (1.28356A)
### <rank-of-c-flight-in-early> is: 5
### <rank-of-c-flight-in-later> is: 2
### <Scheduled-in-later> is: h4n5

Time slot: 500
Flight list in flat-aux: (c2n5 n4t6)
Length: 2
### <flights_scheduled_in_slot> is: (c2n5 n4t6)
### <slot-they-were-scheduled-in> is: 500

Time slot: 400
Flight list in flat-aux: (c1n4 h3n4 t2n4 n4h5)
Length: 4
Sorted flight list: (c1n4 t2n4 h3n4 n4h5)
Current flight: h3n4
k: 1
### <earlier-time-slot> is: 300
### <later-time-slot> is: 500
### <early-num> is: 0
### <later-num> is: 0
### weight-of-c-flight: 1.297758
### weight-list-in-early: (1.448172 1.34673)
### weight-list-in-later: (1.50282 1.308736)
### rank-of-c-flight-in-early: 3
### rank-of-c-flight-in-later: 3

k: 2
### earlier-time-slot: 200
### later-time-slot: 600
### early-num: 0
### later-num: 0
### weight-of-c-flight: 1.297758
### weight-list-in-early: (1.273272 1.333904)
### weight-list-in-later: (1.283568 1.248786)
### rank-of-c-flight-in-early: 2
### rank-of-c-flight-in-later: 1
### Scheduled-in-later: h3n4

**Time slot: 600**
- Flight list in flat-aux: (t4n6 h4n5 h3n4)
  - Length: 3
  - Threshold: 2
  - Sorted flight list: (h3n4 t4n6 h4n5)
- Current flight: h4n5

k: 1
### earlier-time-slot: 500
### later-time-slot: 700
### early-num: 0
### later-num: -2
### weight-of-c-flight: 1.248786
### weight-list-in-early: (1.50282 1.308736)
### weight-list-in-later: (0)
### rank-of-c-flight-in-early: 3
### rank-of-c-flight-in-later: 1
### Scheduled-in-later: h4n5

**Time slot: 600**
- Flight list in flat-aux: (t4n6 h3n4)
  - Length: 2
  - Threshold: 2
### flights-scheduled-in_slot: (t4n6 h3n4)
### slot-they-were-scheduled-in: 600

**Time slot: 400**
- Flight list in flat-aux: (c1n4 t2n4 n4h5)
  - Length: 3
  - Threshold: 2
  - Sorted flight list: (c1n4 t2n4 n4h5)
- Current flight: n4h5

k: 1
### earlier-time-slot: 300
### later-time-slot: 500
### early-num: 0
### later-num: 0
### weight-of-c-flight: 1.273272
### weight-list-in-early: (1.448172 1.34673)
### weight-list-in-later: (1.50282 1.308736)
### rank-of-c-flight-in-early: 3
### rank-of-c-flight-in-later: 3
k: 2
### earlier-time-slot: 200
### later-time-slot: 600
### early-num: 0
### later-num: 0
<<< weight-of-cflight > is : 1.273272
<<< weight-list-in-early > is : (1.273272 1.333904)
<<< weight-list-in-later > is : (1.283568 1.297758)
<<< rank-of-c-flight-in-early > is : 3
<<< rank-of-c-flight-in-later > is : 3
k: 3
<<< earlier-time-slot > is : 100
<<< later-time-slot > is : 700
<<< early-num > is : -2
<<< later-num > is : -1
<<< weight-of-cflight > is : 1.273272
<<< weight-list-in-early > is : (0)
<<< weight-list-in-later > is : (1.248786)
<<< rank-of-c-flight-in-early > is : 1
<<< rank-of-c-flight-in-later > is : 1
<<< Scheduled-in-early > is : n4h5

Time slot: 400
Flight list in flat-aux: (c1n4 t2n4)
Length: 2  Thresh: 2
<<< flightsScheduled_in_slot > is : (c1n4 t2n4)
<<< slot_they_were_scheduled_in > is : 400

Time slot: 500
Flight list in flat-aux: (c2n5 n4t6)
Length: 2  Thresh: 2
<<< flightsScheduled_in_slot > is : (c2n5 n4t6)
<<< slot_they_were_scheduled_in > is : 500

Time slot: 600
Flight list in flat-aux: (t4n6 h3n4)
Length: 2  Thresh: 2
<<< flightsScheduled_in_slot > is : (t4n6 h3n4)
<<< slot_they_were_scheduled_in > is : 600

Time slot: 700
Flight list in flat-aux: (h4n5)
Length: 1  Thresh: 2
<<< flightsScheduled_in_slot > is : (h4n5)
<<< slot_they_were_scheduled_in > is : 700

Time slot: 800
Flight list in flat-aux: nil
Length: 0  Thresh: 2
<<< flightsScheduled_in_slot > is : nil
<<< slot_they_were_scheduled_in > is : 800

Time slot: 900
Flight list in flat-aux: nil
Length: 0  Thresh: 2
<<< flightsScheduled_in_slot > is : nil
<<< slot_they_were_scheduled_in > is : 900

Time slot: 1000
Flight list in flat-aux: nil
Length: 0  Thresh: 2
<<< flightsScheduled_in_slot > is : nil
<<< slot_they_were_scheduled_in > is : 1000
Flattening PT Airport

Time slot: 0
  Flight list in flat-aux: nil
  Length: 0 Thresh: 2
  ### < flights_scheduled_in_slot > is : nil
  ### < slot-they-were-scheduled-in > is : 0

Time slot: 100
  Flight list in flat-aux: nil
  Length: 0 Thresh: 2
  ### < flights_scheduled_in_slot > is : nil
  ### < slot-they-were-scheduled-in > is : 100

Time slot: 200
  Flight list in flat-aux: (t2n4 t2c3)
  Length: 2 Thresh: 2
  ### < flights_scheduled_in_slot > is : (t2n4 t2c3)
  ### < slot-they-were-scheduled-in > is : 200

Time slot: 300
  Flight list in flat-aux: (c2t3 h2t3)
  Length: 2 Thresh: 2
  ### < flights_scheduled_in_slot > is : (c2t3 h2t3)
  ### < slot-they-were-scheduled-in > is : 300

Time slot: 400
  Flight list in flat-aux: (c3t4 n3t5 t4h5 t4n6)
  Length: 4 Thresh: 2
  Sorted flight list: (n3t5 t4n6 t4h5 c3t4)
  Current flight: t4h5
    k: 1
  ### < earlier-time-slot > is : 300
  ### < later-time-slot > is : 500
  ### < early-num > is : 0
  ### < later-num > is : -1
  ### < weight-of-c-flight > is : 1.2501216
  ### < weight-list-in-early > is : (1.2265344 1.2269712)
  ### < weight-list-in-later > is : (1.2737088)
  ### < ranke-of-c-flight-in-early > is : 1
  ### < ranke-of-c-flight-in-later > is : 2
  ### < Scheduled-in-later > is : t4h5

Time slot: 400
  Flight list in flat-aux: (c3t4 n3t5 t4n6)
  Length: 3 Thresh: 2
  Sorted flight list: (n3t5 t4n6 c3t4)
  Current flight: c3t4
    k: 1
  ### < earlier-time-slot > is : 300
  ### < later-time-slot > is : 500
  ### < early-num > is : 0
  ### < later-num > is : 0
  ### < weight-of-c-flight > is : 1.2029472
  ### < weight-list-in-early > is : (1.2265344 1.2269712)
  ### < weight-list-in-later > is : (1.2737088)
  ### < ranke-of-c-flight-in-early > is : 3
  ### < ranke-of-c-flight-in-later > is : 3
  k: 2
  ### < earlier-time-slot > is : 200
Time slot: 400
Flight list in flat-aux: (n3 t5 t4 n6)
Length: 2  Thresh: 2
### <flights_scheduled_in_slot> is: (n3 t5 t4 n6)
### <slot-they-were-scheduled-in> is: 400

Time slot: 500
Flight list in flat-aux: (c4 t5 t4 h5)
Length: 2  Thresh: 2
### <flights_scheduled_in_slot> is: (c4 t5 t4 h5)
### <slot-they-were-scheduled-in> is: 500

Time slot: 600
Flight list in flat-aux: (c3 t4)
Length: 1  Thresh: 2
### <flights_scheduled_in_slot> is: (c3 t4)
### <slot-they-were-scheduled-in> is: 600

Time slot: 700
Flight list in flat-aux: (n4 t6)
Length: 1  Thresh: 2
### <flights_scheduled_in_slot> is: (n4 t6)
### <slot-they-were-scheduled-in> is: 700

Time slot: 800
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is: nil
### <slot-they-were-scheduled-in> is: 800

Time slot: 900
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is: nil
### <slot-they-were-scheduled-in> is: 900

Time slot: 1000
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is: nil
### <slot-they-were-scheduled-in> is: 1000

Flattening PH Airport

Time slot: 0
Flight list in flat-aux: nil
Length: 0  Thresh: 2
### <flights_scheduled_in_slot> is: nil
### <slot-they-were-scheduled-in> is: 0
Time slot: 100
Flight list in flat-aux: nil
Length: 0 Thresh: 2
### <flights_scheduled_in_slot> is: nil
### <slot-they-were-scheduled-in> is: 100

Time slot: 200
Flight list in flat-aux: (n4h5 h2t3 h22c4 h21c4)
Length: 4 Thresh: 2
Sorted flight list: (n4h5 h2t3 h22c4 h21c4)
Current flight: h21c4
k: 1
### <earlier-time-slot> is: 100
### <later-time-slot> is: 300
### <early-num> is: -2
### <later-num> is: -1
### <weight-of-cflight> is: 1.385208
### <weight-list-in-early> is: (0)
### <weight-list-in-later> is: (1.5279264)
### <rank-of-c-flight-in-early> is: 1
### <rank-of-c-flight-in-later> is: 2
### <Scheduled-in-early> is: h21c4

Time slot: 200
Flight list in flat-aux: (n4h5 h2t3 h22c4)
Length: 3 Thresh: 2
Sorted flight list: (n4h5 h2t3 h22c4)
Current flight: h22c4
k: 1
### <earlier-time-slot> is: 100
### <later-time-slot> is: 300
### <early-num> is: -1
### <later-num> is: -1
### <weight-of-cflight> is: 1.385208
### <weight-list-in-early> is: (1.385208)
### <weight-list-in-later> is: (1.5279264)
### <rank-of-c-flight-in-early> is: 2
### <rank-of-c-flight-in-later> is: 2
### <Scheduled-in-later> is: h22c4

Time slot: 200
Flight list in flat-aux: (n4h5 h2t3)
Length: 2 Thresh: 2
### <flights_scheduled_in_slot> is: (n4h5 h2t3)
### <slot-they-were-scheduled-in> is: 200

Time slot: 300
Flight list in flat-aux: (n2h3 h22c4)
Length: 2 Thresh: 2
### <flights_scheduled_in_slot> is: (n2h3 h22c4)
### <slot-they-were-scheduled-in> is: 300

Time slot: 400
Flight list in flat-aux: (n3h4)
Length: 1 Thresh: 2
### <flights_scheduled_in_slot> is: (n3h4)
### <slot-they-were-scheduled-in> is: 400

Time slot: 500
Flight list in flat-aux: (h3n4)
Flattening CH Airport

All done!
Chapter 19

Diet Expert System in Hospital

L. Chang and S. J. Lin
A. INTRODUCTION

A.1. Statement of the Problem

The object of this system is implement a small scale expert system, demonstrating the feasibility of a computer-based model of expert reasoning for diet diagnosis.

This problem can be considered as an extension of the system MYCIN, which takes the result of the diagnosis from the system MYCIN as the input and generate a diet for a patient as the output. To simplify the problem, the uncertainty of the diagnosis is not used in our system. That is, the data from the diagnosis is considered reliable with 100% certainty.

This expert system for diet can tackle real-life problems with realistic models of reasoning.

A.2. Motivation

A diet whose aim is to maintain a healthy person in a state of nutritive sufficiency. It should provide amounts of energy, protein, vitamins, minerals, and other nutrients sufficient to meet the needs of the individual in his life cycle. Diet should be prescribed in qualitative and quantitative terms. The diet may be modified in individual situation, especially when one is ill. In this project, we design the expert system with knowledge base for qualitative consultation, the data base for the quantitative would not be considered so far.

In order to be practical and meaningful, a standard such as the Recommended Dietary Allowances must be translated into guidelines for the proper selection of diet.

However how to modify the normal diet to fit the individual in his particular situation? This technique need an expert as
dietitian to make the decisions based on the experience which is constructed in knowledge base.

The most important reason to implement the diet expert system is the formalization and clarification of knowledge that results from having the human expert make his reasoning explicit. Another reason for building expert systems is the possibility of combining the expertise from many human experts into a shared knowledge base that can be then studied for consistency and reliability of its advice.

B. SYSTEM ORGANIZATION

A variety of languages, representations and tools are used to build expert systems and a particular one is selected for the special purpose. Using LISP as programming language, hierarchical production system as representations and backward chaining as tools. Production system are used to describe the knowledge and use the knowledge by inference machine in a consultation session. An expert system requires two major components: an inference engine and a body of rules (knowledge base).

B.1. Knowledge Base

The basic problem is to express the knowledge about choosing a source-destination strategy in the rule language. This is generally best done in a top-down (hierarchical) fashion, the first step being to identify the top-level hypotheses. In the example, this is done as follows:

H1: diet knowledge base is inadequate
H2: full liquid diet for oral feeding
H3: tube feeding

Evidence exists that might support or rule out each of these
top-level hypotheses and one usually tries to identify a small number of general factors that bear on each hypothesis.

Hierarchical methods have been used in several expert systems, so does in this diet expert system. Since the intermediate-level concepts are the most important tool available for organizing the knowledge base. Abstract descriptions for intermediate hypothesis can be used to achieve hierarchical reasoning. The advantage for the hierarchical reasoning is to reduce the search in many kinds of problem-solving and applies a powerful pruning rules early in the deduction process. generation process.

B.2. Knowledge Representation

This problem-solving systems are based on matching rules, called production system. The rules can have two parts to its conditions: a uncertain precondition check and a certainty condition in the production. In this system whose knowledge-carriers(rules) have a great deal of structure, knowledge acquisition becomes a process of filling in the schemata. The rule is corresponding inference, rather than as a static statement. The classification model has proven to be an excellent representation for expert system problem of diagnosis or interpretation.

What we are interested in the type of pattern-invoked program is production rule, a degenerate program of the form.

\[
\text{IF condition THEN primitive action}
\]

\[
\text{(rule <name> (if <...>) (then <...>))}
\]

The basic control strategy employed in the production is backward-chaining, and one can infer that the rules are not too branchy in that direction. The goal of backward is to determine the variant diet of patient by using the tree search procedure,
starting with the hypotheses. The intermediate hypotheses (conclusions), which is used, is the feature of the expert system. No uncertainty is associated with hypotheses in our system. The advantage of the knowledge representation (hierarchical rule-based) in our system is modularity, explicit knowledge flexibility and efficiency.

Another control strategy employed by this project is conflict resolution which is guided by specificity ordering (or, and, default in order). If more than one rule's IF parts are satisfied, use a conflict resolution strategy to eliminate all but one.

B.3. Inference Machine

Much of the power of an expert system comes from properly applying good reasoning techniques to a large store of problem-specific knowledge. The inference engine does pattern matching for retrieval and rule application and controls the process of deduction and transformation of the data base. The semantic network description is constantly trying to establish the possible existence of situations.

A consultation narrows down the list of goal hypotheses continually working to establish the true or falsity of the most promising ones. In order to use the knowledge in the consultation session, ranking and selection of conclusion ordering of the rule should be done at first. Once the goal-hypotheses H has been chosen, the program enters the question-asking mode (in some kind of ordering). A "yes/no" answer as expression of the observation would then be taken to establish the existence satisfiable description.

C. KNOWLEDGE ACQUISITION

Knowledge acquisition process is one of the most difficult
phases of expert system building. In this system, the application is a diagnostic problem (not interpretive problem).

Since much of our knowledge about decision-making can be expressed in rules of if-then form, there are often other kinds of information that human experts tell us are important in reasoning: hypotheses, functional relations among evidence. This knowledge typically underlies and supports the inferential knowledge expressed in rules. The systematic type of description as rules are used in the expert system. The rules is a modular collection of personal expertise, knowledge.

Knowledge acquisition is the transformation of problem-solving expertise from some knowledge base. The expertise to be elucidated is a collection of specialized facts, procedures, and judgmental rules about the narrow domain or common sense knowledge about the world.

C.1. Observation (findings or evidences)

The yes/no question is used as observation expression. Strategy for asking question:

1. Ask the least costly question at first
2. Ask the question affect the current highest conclusion
3. Ask the question affect the current highest finding
4. Significant increasing or decreasing the current conclusion
5. More popular one should be asked later
C.2 RULE

R01—If the age of the patient is less than 18
Then the "Diet knowledge base" is inadequate for the patient.

R02—If the patient is pregnant
Then the "Diet knowledge base" is inadequate for the patient.

R03—If the patient is surgery medicine
Then the "Diet for surgery" is needed for the patient.

R04—If the patient is internal medicine
Then the "Diet for internal medicine" is needed for the patient.

R05—If the "Diet for internal medicine" is needed
and "The diagnosis is acute illness"
Then the "Diet for acute illness" is needed for the patient.

R06—If the "Diet for internal medicine" is needed
and "The diagnosis is chronic illness"
Then the "Diet for chronic illness" is needed for the patient.

R07—If the "Diet for surgery" is needed for the patient.
and "oral surgery"
Then "Full liquid diet for oral feeding"

R08—If the "Diet for surgery" is needed for the patient.
and "plastic surgery of face and neck"
Then "Full liquid diet for oral feeding"

R09—If the "Diet for surgery" is needed for the patient.
and "surgical treatment of pharyngeal areas"
Then "tube feeding" is needed for the patient

R10—If the "Diet for surgery" is needed for the patient.
and "sever burned patients"
Then "tube feeding" is needed for the patient

R11—If the "Diet for surgery" is needed for the patient.
Then the "High protein, high kilocalorie diet" is needed for the patient.

R12—If the "Diet for surgery" is needed for the patient.
and "gastrectomy operation"
Then "Carbohydrate restricted diet" is needed, for the patient.

R13—If the "Diet for acute illness" is needed
and "severe inflammatory bowel disease"
Then "Total parental nutrition" is needed for the patient.

R14—If the "Diet for internal medicine" is needed
and "constipation"
Then "High fiber diet" is needed for the patient.

R15—If "Diet for acute illness" is needed for the patient.
and "intolerance for food"
Then "Clear liquid diet for oral feeding" is needed for the patient

R16—If "Diet for acute illness" is needed for the patient
and "severe gastroenteritis"
Then "Sucrose restricted diet" is needed for the patient

R17—If "Diet for acute illness" is needed for the patient
and "ulcerative colitis"
Then "Fiber restricted diet" is needed for the patient

R18—If "Diet for acute illness" is needed for the patient
Then "general diet for acute illness" is needed for the patient

R19—If the "Diet for the chronic illness" is needed
and "the diagnosis of chronic duodenal ulcer disease"
Then "Milk rich bland diet for oral feeding" is needed for the patient

R20—If the "Diet for the chronic illness" is needed
and "chronic pancreatitis"
Then "Fat restricted diet" is needed for the patient

R21—If the "Diet for the chronic illness" is needed
and "gallbladder disease"
Then "Fat restricted diet" is needed for the patient

R22—If the "Age of the patient is less than 55"
and "obesity"
Then "Special diet for individual patient"

R23—If the "Diet for the chronic illness" is needed
and "Coronary heart disease"
Then "diet for coronary heart disease" is needed

R24—If the "Diet for the coronary disease" is needed
and "Age of the patient is less than 55"
Then "Cholesterol and fat restricted diet" is needed

R25—If the "Diet for the coronary disease" is needed
and "Age of the patient is over 55"
Then "diet for elder" is needed

R26—If the "diet for elder" is needed
and "Kidney disease"
Then "Special diet for individual patient"

R27—If the "diet for elder" is needed
and "Diabetic disease"
Then "Special diet for individual patient"

R28—If "Diet for acute illness" is needed for the patient
and "Diet for chronic illness" is needed
then "Composition diet" is needed for the patient

C. Hypothesis
The hypothesis have been arranged as follows:
"Diet knowledge base" is inadequate for the patient.
"Full liquid diet for oral feeding"
"Tube feeding" is needed for the patient
"High protein, high kilocalorie diet" is needed for the patient
"Carbohydrate restricted diet" is needed for the patient
"Total parental nutrition" is needed for the patient
"High fiber diet" is needed for the patient
"Clear liquid diet for oral feeding" is needed for the patient
"Sucrose restricted diet" is needed for the patient
"Fiber restricted diet" is needed for the patient
"General diet for acute illness" is needed for the patient
"Milk rich bland diet for oral feeding" is needed for the patient
"Fat restricted diet" is needed for the patient
"Controlled protein, potassium and sodium diet"
"Cholesterol and fat restricted diet" is needed for the patient
"Special diet for individual patient"
"composition diet for different classification" is needed for the patient.

C.3.1 Intermediate hypotheses
the "diet for elder" is needed
the "Diet for surgery" is needed for the patient.
the "Diet for internal medicine" is needed for the patient.
the "Diet for acute illness" is needed for the patient.
the "Diet for chronic illness" is needed for the patient.

C 4 Conflict Resolution Strategies
The specificity ordering is used in conflict resolution strategies for this system. One of the purposes of conflict resolution is it can allow the interpreter to do some of the processing automatically.

As the number of rules in the expert system grows, it becomes less practical to test the conditions of every rule each time the situation changes slightly. The specificity ordering (or, and, default) is the way to find the relevant rules quickly.
D. EXPERIMENTAL RESULTS

D.1. Demonstrative Example
CASE STUDY

(Please refer to Flow Chart G.1)

I. SIMPLE CASE STUDY

(1) -> (diet)
  "Age younger 15 (y/n)? Ans: "y"

Diet is not suitable

(2) -> (diet)
  "Age younger 15 (y/n)? Ans: "n"
  "Pregnant (y/n)? Ans: "y"

Diet is not suitable

(3) -> (diet)
  "Age younger 15 (y/n)? Ans: "n"
  "Pregnant (y/n)? Ans: "n"
  "Is the fact: (the patient is surgery medicine)?
   y/n? Ans: n

These facts have been checked so far:
  1 (the patient is surgery medicine)
  2 (the patient is internal medicine)

(4) -> (diet)
  "Age younger 15 (y/n)? Ans: "n"
  "Pregnant (y/n)? Ans: "n"
  "Is the fact: (the patient is surgery medicine)?
   y/n? Ans: y

** Following stack shows facts from where intermediate conclusion is deduced: **
  1. (the patient is surgery medicine)
rule 101
is true. The conclusion is (diet for surgery)

Is the fact : (oral surgery)
y/n ? Ans : n

these facts have been checked so far
1 (oral surgery)

Is the fact : (plastic surgery for face and neck)
y/n ? Ans : n

these facts have been checked so far
1 (plastic surgery for face and neck)
2 (oral surgery)

Is the fact : (surgical treatment of pharyngeal areas)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (surgical treatment of pharyngeal areas)
2. (diet for surgery)
3. (the patient is surgery medicine)

rule 107
is true. The conclusion is (tube feeding)

** following stack shows facts
from where intermediate conclusion is deduced **
1. (tube feeding)
2. (surgical treatment of pharyngeal areas)
3. (diet for surgery)
4. (the patient is surgery medicine)

The suggestion is (tube feeding)

(5) -> (diet)---------------------------------------------------(default case)

" Age younger is (y/n) ? Ans : "n
" pregnant (y/n) ? Ans : "n

Is the fact : (the patient is surgery medicine)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (the patient is surgery medicine)

rule 101
is true. The conclusion is (diet for surgery)
Is the fact: (oral surgery) y/n ? Ans: n

des these facts have been checked so far 1 (oral surgery)

Is the fact: (palstic surgery for face and neck) y/n ? Ans: n

des these facts have been checked so far 1 (palstic surgery for face and neck) 2 (oral surgery)

Is the fact: (surgical treatment of pharyngeal areas) y/n ? Ans: n

des these facts have been checked so far 1 (surgical treatment of pharyngeal areas) 2 (palstic surgery for face and neck) 3 (oral surgery)

Is the fact: (sever burned patients) y/n ? Ans: n

des these facts have been checked so far 1 (sever burned patients) 2 (surgical treatment of pharyngeal areas) 3 (palstic surgery for face and neck) 4 (oral surgery)

Is the fact: (gastrectomy operation) y/n ? Ans: n

des these facts have been checked so far 1 (gastrectomy operation) 2 (sever burned patients) 3 (surgical treatment of pharyngeal areas) 4 (palstic surgery for face and neck) 5 (oral surgery)

rule id10
is true. The conclusion is (high protein and high kilocalorie diet)

** following stack shows facts from where intermediate conclusion is decuced  
1. (high protein and high kilocalorie diet)  
2. (diet for surgery)  
3. (the patient is surgery medicine) 
The suggestion is (high protein and high kilocalorie diet)
" Age younger 15 (y/n) ? Ans : "n
" pregnant (y/%n) ? Ans : "n
Is the fact : (the patient is surgery medicine)
y/n ? Ans : n
these facts have been checked so far
1 (the patient is surgery medicine)
Is the fact : (the patient is internal medicine)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is (diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (constipation)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id12
is true. The conclusion is (high fiber diet)

** following stack shows facts from where intermediate conclusion is deduced **
1. (high fiber diet)
2. (constipation)
3. (diet for internal medicine)
4. (the patient is internal medicine)

The suggestion is (high fiber diet)

(7) -> (diet)
" Age younger 15 (y/n) ? Ans : "n
" pregnant (y/%n) ? Ans : "n
Is the fact : (the patient is surgery medicine)
y/n ? Ans : n
these facts have been checked so far
1 (the patient is surgery medicine)
Is the fact : (the patient is internal medicine)
y/n ? Ans : n
these facts have been checked so far
1 (the patient is internal medicine)
2 (the patient is surgery medicine)

Is the fact: (obesity) y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (obesity)

Is the fact: (age of the patient is less than 55) y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (age of the patient is less than 55)
2. (obesity)

rule id19 is true. The conclusion is (special diet for individual patient)

** following stack shows facts from where intermediate conclusion is deduced **
1. (special diet for individual patient)
2. (age of the patient is less than 55)
3. (obesity)

The suggestion is (special diet for individual patient)

(8) -> (diet)
" Age younger 15 (y/n) ? Ans: "n"
" pregnant (y/n) ? Ans: "n"
Is the fact: (the patient is surgery medicine) y/n ? Ans: n

these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact: (the patient is internal medicine) y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2 is true. The conclusion is (diet for internal medicine)

Is the fact: (constipation)
Is the fact (the diagnosis is acute illness) y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id3 is true. The conclusion is (diet for acute illness)

Is the fact (severe inflammatory bowel disease) y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (severe inflammatory bowel disease)
2. (diet for acute illness)
3. (the diagnosis is acute illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id11 is true. The conclusion is (total parental nutrition)

** following stack shows facts from where intermediate conclusion is deduced **
1. (total parental nutrition)
2. (severe inflammatory bowel disease)
3. (diet for acute illness)
4. (the diagnosis is acute illness)
5. (diet for internal medicine)
6. (the patient is internal medicine)

The suggestion is (total parental nutrition)

II. COMPLICATED CASE STUDY

(9) -> (dict)
  " Age younger 15 (y/n) ? Ans: "n"
  " pregnant (y/n) ? Ans: "n"

Is the fact (the patient is surgery medicine) y/n? Ans: n

these facts have been checked so far
1 (the patient is surgery medicine)
Is the fact: (the patient is internal medicine) y/n? Ans: y

** following stack shows facts
from where intermediate conclusion is deduced  **
1. (the patient is internal medicine)

rule 162
is true. The conclusion is (diet for internal medicine)

Is the fact: (constipation) y/n? Ans: n

these facts have been checked so far
1. (constipation)
2. (the patient is surgery medicine)

Is the fact: (the diagnosis is acute illness) y/n? Ans: n

these facts have been checked so far
1. (the diagnosis is acute illness)
2. (constipation)
3. (the patient is surgery medicine)

Is the fact: (chronic illness) y/n? Ans: y

** following stack shows facts
from where intermediate conclusion is deduced  **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule 164
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic duodenal ulcer disease) y/n? Ans: n

these facts have been checked so far
1. (the diagnosis of chronic duodenal ulcer disease)
2. (the diagnosis is acute illness)
3. (constipation)
4. (the patient is surgery medicine)

Is the fact: (chronic pancreatitis) y/n? Ans: n

these facts have been checked so far
1. (chronic pancreatitis)
2. (the diagnosis of chronic duodenal ulcer disease)
3. (the diagnosis is acute illness)
4. (constipation)
5. (the patient is surgery medicine)

Is the fact: (gall bladder disease)
these facts have been checked so far
1 (gall bladder disease)
2 (chronic pancreatitis)
3 (the diagnosis of chronic duodenal ulcer disease)
4 (the diagnosis is acute illness)
5 (constipation)
6 (the patient is surgery medicine)

Is the fact: (coronary heart disease)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (coronary heart disease)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id20
is true. The conclusion is (diet for coronary heart disease)

Is the fact: (age of the patient is less than 55)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (age of the patient is less than 55)
2. (diet for coronary heart disease)
3. (coronary heart disease)
4. (diet for chronic illness)
5. (chronic illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

rule id21
is true. The conclusion is (cholesterol and fat restricted diet)
The suggestion is (cholesterol and fat restricted diet) (diet)

(10) Age younger 15 (y/n)? Ans: "n"
" Pregnant (y/n)? Ans: "n"
Is the fact: (the patient is surgery medicine)
y/n? Ans: n

these facts have been checked so far
1. (the patient is surgery medicine)

Is the fact: (the patient is internal medicine)
y/n? Ans: y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule 102
is true. The conclusion is (diet for internal medicine)

Is the fact: (constipation)
y/n? Ans: n

these facts have been checked so far
1. (constipation)
2. (the patient is surgery medicine)

Is the fact: (the diagnosis is acute illness)
y/n? Ans: n

these facts have been checked so far
1. (the diagnosis is acute illness)
2. (constipation)
3. (the patient is surgery medicine)

Is the fact: (chronic illness)
y/n? Ans: y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule 104
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic appendicitis)
y/n? Ans: n

these facts have been checked so far
1. (the diagnosis of chronic appendicitis)
2. (the diagnosis is acute illness)
3. (constipation)
4. (the patient is surgery medicine)

Is the fact: (chronic appendicitis)
these facts have been checked so far
1 (chronic pancreatitis)
2 (the diagnosis of chronic duodenal ulcer disease)
3 (the diagnosis is acute illness)
4 (constipation)
5 (the patient is surgery medicine)

Is the fact : (gall bladder disease) 
y/n ? Ans : n

these facts have been checked so far
1 (gall bladder disease)
2 (chronic pancreatitis)
3 (the diagnosis of chronic duodenal ulcer disease)
4 (the diagnosis is acute illness)
5 (constipation)
6 (the patient is surgery medicine)

Is the fact : (coronary heart disease) 
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (coronary heart disease)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id20
is true. The conclusion is (diet for coronary heart disease)

Is the fact : (age of the patient is less than 55) 
y/n ? Ans : n

these facts have been checked so far
1 (age of the patient is less than 55)
2 (gall bladder disease)
3 (chronic pancreatitis)
4 (the diagnosis of chronic duodenal ulcer disease)
5 (the diagnosis is acute illness)
6 (constipation)
7 (the patient is surgery medicine)

Is the fact : (obesity) 
y/n ? Ans : n

these facts have been checked so far
1 (obesity)
2 (age of the patient is less than 55)
3 (gall bladder disease)
4 (chronic pancreatitis)
5 (the diagnosis of chronic duodenal ulcer disease)
6 (the diagnosis is acute illness)
7 (constipation)
8 (the patient is surgery medicine)
Is the fact: (age of the patient is over 55) 
y/n ? Ans : n

these facts have been checked so far
1 (age of the patient is over 55)
2 (obesity)
3 (age of the patient is less than 55)
4 (gall bladder disease)
5 (chronic pancreatitis)
6 (the diagnosis of chronic duodenal ulcer disease)
7 (the diagnosis is acute illness)
8 (constipation)
9 (the patient is surgery medicine)

rule id28
is true. The conclusion is (information from doctor)

** following stack shows facts from where intermediate conclusion is deduced **
1. (information from doctor)
2. (diet for coronary heart disease)
3. (coronary heart disease)
4. (diet for chronic illness)
5. (chronic illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

The suggestion is (information from doctor) 

-> (diet)
" Age younger 15 (y/n) ? Ans : "n
" pregnant (y/n) ? Ans : "n
Is the fact : (the patient is surgery medicine) 
y/n ? Ans : n

these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine) 
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id22
is true. The conclusion is (diet for internal medicine)

Is the fact : (constipation) 
y/n ? Ans : n
these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact: (the diagnosis is acute illness) 
y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule 103
is true. The conclusion is (diet for acute illness)

Is the fact: (severe inflammatory bowel disease)
y/n ? Ans: n

these facts have been checked so far
1 (severe inflammatory bowel disease)
2 (constipation)
3 (the patient is surgery medicine)

Is the fact: (intolerance for food)
y/n ? Ans: n

these facts have been checked so far
1 (intolerance for food)
2 (severe inflammatory bowel disease)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact: (severe gastroenteritis)
y/n ? Ans: n

these facts have been checked so far
1 (severe gastroenteritis)
2 (intolerance for food)
3 (severe inflammatory bowel disease)
4 (constipation)
5 (the patient is surgery medicine)

Is the fact: (ulcerative colitis)
y/n ? Ans: n

these facts have been checked so far
1 (ulcerative colitis)
2 (severe gastroenteritis)
3 (intolerance for food)
4 (severe inflammatory bowel disease)
5 (constipation)
6 (the patient is surgery medicine)

Is the fact: (chronic illness)
y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for acute illness)
3. (the diagnosis is acute illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule ic4
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic cuodenal ulcer disease) y/n? Ans: n

these facts have been checked so far
1. (the diagnosis of chronic cuodenal ulcer disease)
2. (ulcerative colitis)
3. (severe gastroenteritis)
4. (intolerance for food)
5. (severe inflammatory bowel disease)
6. (constipation)
7. (the patient is surgery medicine)

Is the fact: (chronic pancreatitis) y/n? Ans: n

these facts have been checked so far
1. (chronic pancreatitis)
2. (the diagnosis of chronic cuodenal ulcer disease)
3. (ulcerative colitis)
4. (severe gastroenteritis)
5. (intolerance for food)
6. (severe inflammatory bowel disease)
7. (constipation)
8. (the patient is surgery medicine)

Is the fact: (gall bladder disease) y/n? Ans: n

these facts have been checked so far
1. (gall bladder disease)
2. (chronic pancreatitis)
3. (the diagnosis of chronic cuodenal ulcer disease)
4. (ulcerative colitis)
5. (severe gastroenteritis)
6. (intolerance for food)
7. (severe inflammatory bowel disease)
8. (constipation)
9. (the patient is surgery medicine)

Is the fact: (coronary heart disease) y/n? Ans: n

these facts have been checked so far
1. (coronary heart disease)
2. (gall bladder disease)
3. (chronic pancreatitis)
4. (the diagnosis of chronic cuodenal ulcer disease)
5. (ulcerative colitis)
6. (severe gastroenteritis)
7. (intolerance for food)
8 (severe inflammatory bowel disease)
9 (constipation)
10 (the patient is surgery medicine)

Is the fact: (obesity)
\text{y/n} \text{ ? Ans: n}

these facts have been checked so far
1 (obesity)
2 (coronary heart disease)
3 (gall bladder disease)
4 (chronic pancreatitis)
5 (the diagnosis of chronic duodenal ulcer disease)
6 (ulcerative colitis)
7 (severe gastroenteritis)
8 (intolerance for food)
9 (severe inflammatory bowel disease)
10 (constipation)
11 (the patient is surgery medicine)

rule id29
is true. The conclusion is (composition diet for different classification)

** following stack shows facts
from where intermediate conclusion is deduced **
1. (composition diet for different classification)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for acute illness)
5. (the diagnosis is acute illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

The suggestion is (composition diet for different classification)

11) -> $\text{diet}\text{y}$

"Age younger 15 (y/n)? Ans: "n"
"pregnant (y/n)? Ans: "n"

Is the fact: (the patient is surgery medicine)
\text{y/n} \text{ ? Ans: n}

these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact: (the patient is internal medicine)
\text{y/n} \text{ ? Ans: y}

** following stack shows facts
from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule 102
Is the fact: (constipation)
y/n? Ans: n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact: (the diagnosis is acute illness)
y/n? Ans: n

these facts have been checked so far
1 (the diagnosis is acute illness)
2 (constipation)
3 (the patient is surgery medicine)

Is the fact: (chronic illness)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule 1d4
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic duodenal ulcer disease)
y/n? Ans: n

these facts have been checked so far
1 (the diagnosis of chronic duodenal ulcer disease)
2 (the diagnosis is acute illness)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact: (chronic pancreatitis)
y/n? Ans: n

these facts have been checked so far
1 (chronic pancreatitis)
2 (the diagnosis of chronic duodenal ulcer disease)
3 (the diagnosis is acute illness)
4 (constipation)
5 (the patient is surgery medicine)

Is the fact: (gall bladder disease)
y/n? Ans: n

these facts have been checked so far
1 (gall bladder disease)
2 (chronic pancreatitis)
3 (the diagnosis of chronic duodenal ulcer disease)
4 (the diagnosis is acute illness)
5 (constipation)
6 (the patient is surgery medicine)
Is the fact: (coronary heart disease)  
y/n? Ans: n

Is the fact: (obesity)  
y/n? Ans: n

Is the fact: (the patient is surgery medicine)  
y/n? Ans: n

Rule 1d28
is true. The conclusion is (information from doctor)

** Following stack shows facts from where intermediate conclusion is deduced  **
1. (information from doctor)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

The suggestion is (information from doctor)

(12) -> (diet)
" Age younger 15 (y/n)? Ans: "n"
" pregnant (y/n)? Ans: "n"
Is the fact: (the patient is surgery medicine)  
y/n? Ans: n

these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact: (the patient is internal medicine)  
y/n? Ans: t  
typo please reenter y/ny
** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is (diet for internal medicine)

Is the fact: (constipation)
y/n? Ans: n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact: (the diagnosis is acute illness)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id3
is true. The conclusion is (diet for acute illness)

Is the fact: (severe inflammatory bowel disease)
y/n? Ans: n

these facts have been checked so far
1 (severe inflammatory bowel disease)
2 (constipation)
3 (the patient is surgery medicine)

Is the fact: (intolerance for food)
y/n? Ans: n

these facts have been checked so far
1 (intolerance for food)
2 (severe inflammatory bowel disease)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact: (severe gastroenteritis)
y/n? Ans: n

these facts have been checked so far
1 (severe gastroenteritis)
2 (intolerance for food)
3 (severe inflammatory bowel disease)
4 (constipation)
5 (the patient is surgery medicine)

Is the fact: (ulcerative colitis)
y/n? Ans: n

these facts have been checked so far
1 (ulcerative colitis)
2 (severe gastroenteritis)
3 (intolerance for food)
4 (severe inflammatory bowel disease)
5 (constipation)
6 (the patient is surgery medicine)

Is the fact: (chronic illness)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for acute illness)
3. (the diagnosis is acute illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id4 is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic duodenal ulcer disease)
y/n? Ans: n

these facts have been checked so far
1 (the diagnosis of chronic duodenal ulcer disease)
2 (ulcerative colitis)
3 (severe gastroenteritis)
4 (intolerance for food)
5 (severe inflammatory bowel disease)
6 (constipation)
7 (the patient is surgery medicine)

Is the fact: (chronic pancreatitis)
y/n? Ans: n

these facts have been checked so far
1 (chronic pancreatitis)
2 (the diagnosis of chronic duodenal ulcer disease)
3 (ulcerative colitis)
4 (severe gastroenteritis)
5 (intolerance for food)
6 (severe inflammatory bowel disease)
7 (constipation)
8 (the patient is surgery medicine)

Is the fact: (gall bladder disease)
y/n? Ans: n

these facts have been checked so far
1 (gall bladder disease)
2 (chronic pancreatitis)
3 (the diagnosis of chronic duodenal ulcer disease)
4 (ulcerative colitis)
5 (severe gastroenteritis)
6 (intolerance for food)
7 (severe inflammatory bowel disease)
8 (constipation)
9 (the patient is surgery medicine)
is the fact : (coronary heart disease)  
   y/n ? Ans : n

these facts have been checked so far  
1 (coronary heart disease)  
2 (gall bladder disease)  
3 (chronic pancreatitis)  
4 (the diagnosis of chronic duodenal ulcer disease)  
5 (ulcerative colitis)  
6 (severe gastroenteritis)  
7 (intolerance for food)  
8 (severe inflammatory bowel disease)  
9 (constipation)  
10 (the patient is surgery medicine)

is the fact : (obesity) 
   y/n ? Ans : n

these facts have been checked so far  
1 (obesity)  
2 (coronary heart disease)  
3 (gall bladder disease)  
4 (chronic pancreatitis)  
5 (the diagnosis of chronic duodenal ulcer disease)  
6 (ulcerative colitis)  
7 (severe gastroenteritis)  
8 (intolerance for food)  
9 (severe inflammatory bowel disease)  
10 (constipation)  
11 (the patient is surgery medicine)

rule 1029
is true. The conclusion is (composition diet for different classification)

** following stack shows facts from where intermediate conclusion is deduced  **
1. (composition diet for different classification)  
   2. (diet for chronic illness)  
   3. (chronic illness)  
   4. (diet for acute illness)  
   5. (the diagnosis is acute illness)  
   6. (diet for internal medicine)  
   7. (the patient is internal medicine)

The suggestion is (composition diet for different classification)

(13) => (diet)
   " Are younger 15 (y/n) ? Ans : "n"
   " pregnant (y/n) ? Ans : "n"
   is the fact : (the patient is surgery medicine)  
   y/n ? Ans : n
these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule 1c2
is true. The conclusion is (diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans : n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact : (the diagnosis is acute illness)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule 1d3
is true. The conclusion is (diet for acute illness)

Is the fact : (severe inflammatory bowel disease)
y/n ? Ans : n

these facts have been checked so far
1 (severe inflammatory bowel disease)
2 (constipation)
3 (the patient is surgery medicine)

Is the fact : (intolerance for food)
y/n ? Ans : n

these facts have been checked so far
1 (intolerance for food)
2 (severe inflammatory bowel disease)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact : (severe gastroenteritis)
y/n ? Ans : n

these facts have been checked so far
1 (severe gastroenteritis)
2 (intolerance for food)
3 (severe inflammatory bowel disease)
4 (constipation)
5 (the patient is surgery medicine)
Is the fact: (ulcerative colitis)  
y/n? Ans: n

des these facts have been checked so far
1 (ulcerative colitis)  
2 (severe gastroenteritis)  
3 (intolerance for food)  
4 (severe inflammatory bowel disease)  
5 (constipation)  
6 (the patient is surgery medicine)

Is the fact: (chronic illness)  
y/n? Ans: y

** Following stack shows facts **
from where intermediate conclusion is deduced  **
1. (chronic illness)  
2. (diet for acute illness)  
3. (the diagnosis is acute illness)  
4. (diet for internal medicine)  
5. (the patient is internal medicine)

rule 104
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic duodenal ulcer disease)  
y/n? Ans: n

des these facts have been checked so far
1 (the diagnosis of chronic duodenal ulcer disease)  
2 (ulcerative colitis)  
3 (severe gastroenteritis)  
4 (intolerance for food)  
5 (severe inflammatory bowel disease)  
6 (constipation)  
7 (the patient is surgery medicine)

Is the fact: (chronic pancreatitis)  
y/n? Ans: n

des these facts have been checked so far
1 (chronic pancreatitis)  
2 (the diagnosis of chronic duodenal ulcer disease)  
3 (ulcerative colitis)  
4 (severe gastroenteritis)  
5 (intolerance for food)  
6 (severe inflammatory bowel disease)  
7 (constipation)  
8 (the patient is surgery medicine)

Is the fact: (gall bladder disease)  
y/n? Ans: n

des these facts have been checked so far
1 (gall bladder disease)  
2 (chronic pancreatitis)  
3 (the diagnosis of chronic duodenal ulcer disease)  
4 (ulcerative colitis)
5 (severe gastroenteritis)
6 (intolerance for food)
7 (severe inflammatory bowel disease)
8 (constipation)
9 (the patient is surgery medicine)

Is the fact : (coronary heart disease)
y/n ? Ans : n

these facts have been checke so far
1 (coronary heart disease)
2 (gall bladder disease)
3 (chronic pancreatitis)
4 (the diagnosis of chronic duodenal ulcer disease)
5 (ulcerative colitis)
6 (severe gastroenteritis)
7 (intolerance for food)
8 (severe inflammatory bowel disease)
9 (constipation)
10 (the patient is surgery medicine)

Is the fact : (obesity)
y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is concluded  **

1. (obesity)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for acute illness)
5. (the diagnosis is acute illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

Is the fact : (age of the patient is less than 55)
y/n ? Ans : n

these facts have been checke so far
1 (age of the patient is less than 55)
2 (coronary heart disease)
3 (gall bladder disease)
4 (chronic pancreatitis)
5 (the diagnosis of chronic duodenal ulcer disease)
6 (ulcerative colitis)
7 (severe gastroenteritis)
8 (intolerance for food)
9 (severe inflammatory bowel disease)
10 (constipation)
11 (the patient is surgery medicine)

rule 1029
is true. The conclusion is (composition diet for different classification)
** Following stack shows facts from where intermediate conclusion is deduced **
1. (composition diet for different classification)
2. (obesity)
3. (diet for chronic illness)
4. (chronic illness)
5. (diet for acute illness)
6. (the diagnosis is acute illness)
7. (diet for internal medicine)
8. (the patient is internal medicine)

The suggestion is (composition diet for different classification)

(14) -> (diet)

Age younger 15 (y/n) ? Ans: "n"
pregnant (y/n) ? Ans: "n"
Is the fact : (the patient is surgery medicine)
y/n ? Ans: n

these facts have been checked so far
1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine)
y/n ? Ans: y

** Following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is (diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans: n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact : (the diagnosis is acute illness)
y/n ? Ans: y

** Following stack shows facts from where intermediate conclusion is deduced **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id3
is true. The conclusion is (diet for acute illness)

Is the fact : (severe inflammatory bowel disease)
y/n ? Ans: n

these facts have been checked so far
1 (severe inflammatory bowel disease)
2 (constipation)
Is the fact: (intolerance for food)  
y/n? Ans: n

these facts have been checked so far
1. (intolerance for food)  
2. (severe inflammatory bowel disease)  
3. (constipation)  
4. (the patient is surgery medicine)

Is the fact: (severe gastroenteritis)  
y/n? Ans: n

these facts have been checked so far
1. (severe gastroenteritis)  
2. (intolerance for food)  
3. (severe inflammatory bowel disease)  
4. (constipation)  
5. (the patient is surgery medicine)

Is the fact: (ulcerative colitis)  
y/n? Ans: n

these facts have been checked so far
1. (ulcerative colitis)  
2. (severe gastroenteritis)  
3. (intolerance for food)  
4. (severe inflammatory bowel disease)  
5. (constipation)  
6. (the patient is surgery medicine)

Is the fact: (chronic illness)  
y/n? Ans: y

** Following stack shows facts from where intermediate conclusion is deduced **
1. (chronic illness)  
2. (diet for acute illness)  
3. (the diagnosis is acute illness)  
4. (diet for internal medicine)  
5. (the patient is internal medicine)

rule id4  
is true. The conclusion is (diet for chronic illness)

Is the fact: (the diagnosis of chronic caecal ulcer disease)  
y/n? Ans: n

these facts have been checked so far
1. (the diagnosis of chronic caecal ulcer disease)  
2. (ulcerative colitis)  
3. (severe gastroenteritis)  
4. (intolerance for food)  
5. (severe inflammatory bowel disease)  
6. (constipation)  
7. (the patient is surgery medicine)

Is the fact: (chronic pancreatitis)
these facts have been checked so far
1. (chronic pancreatitis)
2. (the diagnosis of chronic duodenal ulcer disease)
3. (ulcerative colitis)
4. (severe gastroenteritis)
5. (intolerance for food)
6. (severe inflammatory bowel disease)
7. (constipation)
8. (the patient is surgery medicine)

Is the fact: (gall bladder disease)
y/n? Ans: n

these facts have been checked so far
1. (gall bladder disease)
2. (chronic pancreatitis)
3. (the diagnosis of chronic duodenal ulcer disease)
4. (ulcerative colitis)
5. (severe gastroenteritis)
6. (intolerance for food)
7. (severe inflammatory bowel disease)
8. (constipation)
9. (the patient is surgery medicine)

Is the fact: (coronary heart disease)
y/n? Ans: n

these facts have been checked so far
1. (coronary heart disease)
2. (gall bladder disease)
3. (chronic pancreatitis)
4. (the diagnosis of chronic duodenal ulcer disease)
5. (ulcerative colitis)
6. (severe gastroenteritis)
7. (intolerance for food)
8. (severe inflammatory bowel disease)
9. (constipation)
10. (the patient is surgery medicine)

Is the fact: (obesity)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (obesity)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for acute illness)
5. (the diagnosis is acute illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

Is the fact: (age of the patient is less than 55)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (age of the patient is less than 55)
2. (obesity)
3. (diet for chronic illness)
4. (chronic illness)
5. (diet for acute illness)
6. (the diagnosis is acute illness)
7. (diet for internal medicine)
8. (the patient is internal medicine)

rule id19
is true. The conclusion is (special diet for individual patient)

** following stack shows facts from where intermediate conclusion is deduced **
1. (special diet for individual patient)
2. (age of the patient is less than 55)
3. (obesity)
4. (diet for chronic illness)
5. (chronic illness)
6. (diet for acute illness)
7. (the diagnosis is acute illness)
8. (diet for internal medicine)
9. (the patient is internal medicine)

The suggestion is (special diet for individual patient)

-> 0
Goodbye
$ 0$

script done on Thu Nov 29 00:55:35 1984
Franz Lisp, Opus 38.69
-> (include li.l)
[Load li.l]

-> (diet)

"Age younger 15 (y/n)? Ans: " n
"pregnant (y/n)? Ans: " n

Is the fact: (the patient is surgery medicine)
  y/n? Ans: n

these facts have been checked so far [autoload /usr/lib/lisp/loop]
[fast /usr/lib/lisp/ioop.o]
[fast /usr/lib/lisp/machacks.o]

1 (the patient is surgery medicine)

Is the fact: (the patient is internal medicine)
  y/n? Ans: y

** following stack shows facts
  from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2 is true. The conclusion is (diet for internal medicine)

Is the fact: (constipation)
  y/n? Ans: y

** following stack shows facts
  from where intermediate conclusion is deduced **
1. (constipation)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id12 is true. The conclusion is (high fiber diet)

** following stack shows facts
  from where intermediate conclusion is deduced **
1. (high fiber diet)
2. (constipation)
3. (diet for internal medicine)
4. (the patient is internal medicine)

The suggestion is (high fiber diet)
-> ^D
Script started on Sun Dec 2 15:15:14 1984

Franz Lisp, Opus 38.69
-> (include li.*l)
[load li.*l]
t
-> (diet)
" Age younger 15 (y/n) ? Ans : " n
" pregnant (y/n) ? Ans : " n
Is the fact : (the patient is surgery medicine)
y/n ? Ans : n

these facts have been checked so far [autolaod /usr/lib/lisp/loop]
[fasl /usr/lib/lisp/loop.o]
[fasl /usr/lib/lisp/machacks.o]

1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced  **
1 (the patient is internal medicine)

rule id2
is true. The conclusion is (diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans : n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact : (the diagnosis is acute illness)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced  **
1. (the diagnosis is acute illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id3
is true. The conclusion is (diet for acute illness)

Is the fact : (severe inflammatory bowel disease)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced  **
1. (severe inflammatory bowel disease)
2. (diet for acute illness)
3. (the diagnosis is acute illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)
rule id11
is true. The conclusion is (total parental nutrition)

** following stack shows facts from where intermediate conclusion is deduced **
1. (total parental nutrition)
2. (severe inflammatory bowel disease)
3. (diet for acute illness)
4. (the diagnosis is acute illness)
5. (diet for internal medicine)
6. (the patient is internal medicine)

The suggestion is (total parental nutrition)
-> ^D
Goodbye
$ ^D
script done on Sun Dec 2 15:18:02 1984
Script started on Sun Dec 2 15:35:48 1984

Franz Lisp, Oous 38.69
(inc-> lude li.tl)
[load li.tl]
t
-> (diet)
" Age younger 15 (y/n) ? Ans : " n
" pregnant (y/n) ? Ans : " n
Is the fact : (the patient is surgery medicine)
y/n ? Ans : n
t
these facts have been checked so far [autoload /usr/lib/lisp/loop]
[fasl /usr/lib/lisp/loop.home]
[fasl /usr/lib/lisp/machacks.o]
1 (the patient is surgery medicine)
Is the fact : (the patient is internal medicine)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is(diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans : n
t
these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)
Is the fact : (the diagnosis is acute illness)
y/n ? Ans : n
t
these facts have been checked so far
1 (the diagnosis is acute illness)
2 (constipation)
3 (the patient is surgery medicine)
Is the fact : (chronic illness)
y/n ? Ans : y

** following stack shows facts
from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id4
is true. The conclusion is(diet for chronic illness)

Is the fact : (the diagnosis of chronic duodenal ulcer disease)
y/n ? Ans : n
these facts have been checked so far
1 (the diagnosis of chronic duodenal ulcer disease)
2 (the diagnosis is acute illness)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact : (chronic pancreatitis)
y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (chronic pancreatitis)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id17
is true. The conclusion is (fat restricted diet)

** following stack shows facts from where intermediate conclusion is deduced **
1. (fat restricted diet)
2. (chronic pancreatitis)
3. (diet for chronic illness)
4. (chronic illness)
5. (diet for internal medicine)
6. (the patient is internal medicine)

The suggestion is (fat restricted diet)
-> ^D
Goodbye
$ ^D
script done on Sun Dec 2 15:42:11 1984
$ lisp
> (include li.l)
Cload li.l
C
-> (diet)
" Age younger 15 (y/n) ? Ans : " n
" pregnant (y/n) ? Ans : " n
Is the fact : (the patient is surgery medicine)
y/n ? Ans : n

these facts have been checked so far [autoload /usr/lib/lisp/loop]
C[fast /usr/lib/lisp/loop.o]
C[fast /usr/lib/lisp/machackso]
1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine)
y/n ? Ans : y

** following stack shows facts
* from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is(diet for internal medicine)

Is the fact : (constipation)
y/n ? Ans : n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact : (the diagnosis is acute illness)
y/n ? Ans : n

these facts have been checked so far
1 (the diagnosis is acute illness)
2 (constipation)
3 (the patient is surgery medicine)

Is the fact : (chronic illness)
y/n ? Ans : y

** following stack shows facts
* from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id4
is true. The conclusion is(diet for chronic illness)

Is the fact : (the diagnosis of chronic duodenal ulcer disease)
y/n ? Ans : n
these facts have been checked so far
1 (the diagnosis of chronic duodenal ulcer disease)
2 (the diagnosis is acute illness)
3 (constipation)
4 (the patient is surgery medicine)

Is the fact: (chronic pancreatitis) y/n ? Ans: n

these facts have been checked so far
1 (chronic pancreatitis)
2 (the diagnosis of chronic duodenal ulcer disease)
3 (the diagnosis is acute illness)
4 (constipation)
5 (the patient is surgery medicine)

Is the fact: (gall bladder disease) y/n ? Ans: n

these facts have been checked so far
1 (gall bladder disease)
2 (chronic pancreatitis)
3 (the diagnosis of chronic duodenal ulcer disease)
4 (the diagnosis is acute illness)
5 (constipation)
6 (the patient is surgery medicine)

Is the fact: (coronary heart disease) y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (coronary heart disease)
2. (diet for chronic illness)
3. (chronic illness)
4. (diet for internal medicine)
5. (the patient is internal medicine)

rule id=20
is true. The conclusion is (diet for coronary heart disease)

Is the fact: (age of the patient is less than_55) y/n ? Ans: n

these facts have been checked so far
1 (age of the patient is less than 55)
2 (gall bladder disease)
3 (chronic pancreatitis)
4 (the diagnosis of chronic duodenal ulcer disease)
5 (the diagnosis is acute illness)
6 (constipation)
7 (the patient is surgery medicine)

Is the fact: (obesity) y/n ? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (obesity)
2. (diet for coronary heart disease)
3. (coronary heart disease)
4. (diet for chronic illness)
5. (chronic illness)
6. (diet for internal medicine)
7. (the patient is internal medicine)

Is the fact: (age of the patient is over 55)? y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (age of the patient is over 55)
2. (obesity)
3. (diet for coronary heart disease)
4. (coronary heart disease)
5. (diet for chronic illness)
6. (chronic illness)
7. (diet for internal medicine)
8. (the patient is internal medicine)

rule id22
is true. The conclusion is (diet for elder)

Is the fact: (kidney disease)? y/n? Ans: y

** following stack shows facts from where intermediate conclusion is deduced **
1. (kidney disease)
2. (diet for elder)
3. (age of the patient is over 55)
4. (obesity)
5. (diet for coronary heart disease)
6. (coronary heart disease)
7. (diet for chronic illness)
8. (chronic illness)
9. (diet for internal medicine)
10. (the patient is internal medicine)

rule id23
is true. The conclusion is (special diet for individual patient)

** following stack shows facts from where intermediate conclusion is deduced **
1. (special diet for individual patient)
2. (kidney disease)
3. (diet for elder)
4. (age of the patient is over 55)
5. (obesity)
6. (diet for coronary heart disease)
7. (coronary heart disease)
8. (diet for chronic illness)
9. (chronic illness)
10. (diet for internal medicine)
11. (the patient is internal medicine)

The suggestion is (special diet for individual patient)
-> ^0

Goodbye
$ ^D

script done on Sun Dec 2 15:35:18 1984
Script started on Sun Dec 2 15:43:35 1984
$ lisp
Franz Lisp, Opus 38.69

-> (include li.1)
[load li.1]
t
-> (diet)
" Age younger 15 (y/n) ? Ans : " n
" pregnant (y/n) ? Ans : " n
Is the fact : (the patient is surgery medicine)
 y/n ? Ans : n

these facts have been checked so far [autoload /usr/lib/lisp/loop]
[fasl /usr/lib/lisp/loop.o]
[fasl /usr/lib/lisp/machacks.o]

1 (the patient is surgery medicine)

Is the fact : (the patient is internal medicine)
 y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (the patient is internal medicine)

rule id2
is true. The conclusion is(diet for internal medicine)

Is the fact : (constipation)
 y/n ? Ans : n

these facts have been checked so far
1 (constipation)
2 (the patient is surgery medicine)

Is the fact : (the diagnosis is acute illness)
 y/n ? Ans : n

these facts have been checked so far
1 (the diagnosis is acute illness)
2 (constipation).
3 (the patient is surgery medicine)

Is the fact : (chronic illness)
 y/n ? Ans : y

** following stack shows facts from where intermediate conclusion is deduced **
1. (chronic illness)
2. (diet for internal medicine)
3. (the patient is internal medicine)

rule id4
is true. The conclusion is(diet for chronic illness)

Is the fact : (the diagnosis of chronic duodenal ulcer disease)
 y/n ? Ans : n
these facts have been checked so far  
1 (the diagnosis of chronic duodenal ulcer disease)  
2 (the diagnosis is acute illness)  
3 (constipation)  
4 (the patient is surgery medicine)  

Is the fact : (chronic pancreatitis)  
y/n ? Ans : n  

these facts have been checked so far  
1 (chronic pancreatitis)  
2 (the diagnosis of chronic duodenal ulcer disease)  
3 (the diagnosis is acute illness)  
4 (constipation)  
5 (the patient is surgery medicine)  

Is the fact : (gall bladder disease)  
y/n ? Ans : n  

these facts have been checked so far  
1 (gall bladder disease)  
2 (chronic pancreatitis)  
3 (the diagnosis of chronic duodenal ulcer disease)  
4 (the diagnosis is acute illness)  
5 (constipation)  
6 (the patient is surgery medicine)  

Is the fact : (coronary heart disease)  
y/n ? Ans : n  

these facts have been checked so far  
1 (coronary heart disease)  
2 (gall bladder disease)  
3 (chronic pancreatitis)  
4 (the diagnosis of chronic duodenal ulcer disease)  
5 (the diagnosis is acute illness)  
6 (constipation)  
7 (the patient is surgery medicine)  

Is the fact : (obesity)  
y/n ? Ans : y  

** following stack shows facts from where intermediate conclusion is deduced  **  
1. (obesity)  
2. (diet for chronic illness)  
3. (chronic illness)  
4. (diet for internal medicine)  
5. (the patient is internal medicine)  

Is the fact : (age of the patient is less than 55)  
y/n ? Ans : n  

these facts have been checked so far  
1 (age of the patient is less than 55)  
2 (coronary heart disease)  
3 (gall bladder disease)  
4 (chronic pancreatitis)  
5 (the diagnosis of chronic duodenal ulcer disease)  
6 (the diagnosis is acute illness)
7 (constipation)
8 (the patient is surgery medicine)

rule id26
is true. The conclusion is (information from doctor)

** following stack shows facts
from where intermediate conclusion is deduced **
1. (information from doctor)
2. (obesity)
3. (diet for chronic illness)
4. (chronic illness)
5. (diet for internal medicine)
6. (the patient is internal medicine)

The suggestion is (information from doctor)

-> ^D

Goodbye

script done on Sun Dec 2 15:49:47 1984
D.2. Performance Evaluation

There are two approaches in evaluating expert system performance (anecdotal and empirical approach). The second approach is chosen to compare the expert system's results with the human expert.

A data base of cases can be used not only to evaluate the application model, but also provide a wealth of information for guiding the model designer in improving the performance of a model.

Performance is evaluated by matching the expert's conclusion to the model's conclusion in each case. The results are organized according to final conclusions and show the number of cases in which the model's conclusion matches the expert's conclusion. The overall performance measure for the data base of cases will generally suggest to the model designer a single conclusion for which system performance needs improvement.

The representation described in section C is sufficient to represent much of the core reasoning structure of an expert system. This system is quite typical of classification models. There are two major issues to evaluate the classification model.
1. Reaching accurate conclusions.
2. Asking reasonable questions which aid in the diagnosis (classification).

Other issues such as quality of the system's decision and advice, quality of the human-computer interaction, efficiency, and cost-effectiveness should also be considered.

The questioning order may be important. With the questionnaire, control is handled by very simple mechanisms such as FF, FH rules. The addition of confidence measure will produce a somewhat more complicated analysis.
It is important to measure the performance of human experts in a field if they are assessed by the same standards to be used in the evaluation of the expert system. The expert may be expected not only the right answer but also the right way (i.e., reasonable inference) of getting the answer. Diagnostic decision-making in diet is one of the classification problem.

D.2.1 Comparison

ROSIE supports three types of inference mechanisms:

1. State-driven - the state of the system directly causes a rule to fire.
2. Goal-driven - backward-chaining is used to find rules that will verify predicates in the rule conditions.
3. Change-driven - a data-base change causes a rule to fire.

Comparing the system organization with ROSIE, they are similar to our system, except that our system doesn't supply the data base.

D.2.2 Correct Reasoning

The question is whether the programs were reaching decisions using reasoning equivalent to that used by comparable human experts. From the programs and the uses interaction, the interface between knowledge engineering and human's reasoning is great and appropriate.

Unlike many conventional programs, expert systems do not usually deal with problems for which there is clearly a right or wrong answer. As a result it is seldom easy to demonstrate in a straightforward fashion that a system's answers are correct. However, reviewing the result of section D.1, even the one who is not an expert in diet still can conclude that the diagnosis from our system is quite reasonable.
D.2.3 Discourse (I/O content; Asking question)

As the designer of a model, one would like to somehow give the system a minimum amount of information and hope that a general strategy could be programmed so that the computer would always choose the best next question to ask of the user. One of the keys to having a good questioning strategy is to give the questions as much structure as possible.

The nature of the discourse between the expert system and the user is particularly important in our system.
* The choice of sentences used in the question and responses generated by the program is easy to understand and hierarchical.
* The ability of the expert system to give advice to the user in a congenial fashion and in the user's own terms.

From section D.1, it is shown that the asking question is structive, hierarchical and easy to understand.

D.2.4 Hardware Environment

The conventional typewriter keyboard are adequate to support an interaction between intended users and the expert system. In our system, the user just answers the question on the screen by typing in yes/no. If one have the error typing, system will allow you to answer the question again.

For the model designer, most people prefer to display the set of rules which were satisfied for a particular hypothesis for the production rule system as the most direct form of explanation. From section D.1, we can see the source rule of the result is printed out.

The impact of an expert system on the process of decision-making in the users' environment must also be analyzed during the system's evaluation. i.e. powerful CPU, efficiently designed
searching scheme. Since hierarchical hypothesis are used, the goal of the efficiency is reached.

D.2.5 Sensitivity studies

As the rule becomes more flexible, more time is needed to get the reasoning. In addition, system control is made more accessible by introducing a set of control rules in the model. (see E.3)

E. CONCLUSION AND DISCUSSIONS

E.1 Conclusion

(a) Overall system structure

This system is seen to be both knowledge-base (not model-based) and goal-driven and this project presents a diet problem that is based on the design of a rule-based expert system. In addition to the knowledge rules, the highly constrained backward-chaining control structure are also employed. The searching performance is implemented by LISP as programming language. Since this system is only small scale, it does not cover the spectrum of all possibilities. The system has the flexibility to improve the spectrum (enlarge the scope).

(b) The rule-based model

Heuristics that formed the basis of rules were extracted and organized with a hierarchical representation. The actions of the rules are independent entities that are carried out by the processing modules. The conditions and actions provide for a wide range of data manipulation and modification. The ability to modify and expand the knowledge base is an important contribution that is a result of the rule-based structure.

(c) Experimental results

A ranking of the rules was established based on their effectiveness in maximizing the performance measures and their fre-
frequency of firing. The contribution of these experiments is the selection of an optimal set of rules over a large collection of states.

(d) Measurement

Performance is judged by comparing the diagnosis from this system to the dietitian.

The expert's knowledge provides the key to expert performance, while knowledge representation and inference schemes provide the mechanisms for its use.

An expert consultation program will give advice on the diet treatment for a patient. The system would interpret data from various symptoms.

E.1.1 Weakness

Like human intelligence, we expect that the performance of an expert system will improve with experience over time, but the current expert system lacks powerful learning capabilities. There is some tools which can help the designer evaluate the performance of a model such as sensitivity studies (checking the performance of results from the model changes) and biasing/blinding studies. But changing a model is a heavy burden, and thus the sensitivity is not studied. And it is impossible to invite the domain expert to evaluate this system for financial reasons. As mentioned before, the data from the diagnosis is considered reliable with 100% certainty. The uncertainty can be introduced to expand our system and the ability to quantitatively measure performance have not been constructed yet.

E.2. Discussion

E.2.1 Potential development

If the data base is added to this system, and the uncertainty is considered, this system can be connected to the MYCIN. It can
form a large automatic expert system in the hospital. In medicine, symptoms are often qualified to denote mild, moderate and severe manifestations or confidence in the presence of diseases (e.g. possible, probable or definite). Similarly, the composition of the two different diseases or distinguish the different quality (mild, moderate and severe) can be updated in our system by increasing the number of rules and increasing the number of hierarchical levels. The main reason is that our goal is to implement the reasoning of knowledge through the software, not the robustness of the system.

In order to expand the application to the non-hospital utilization, like the diet for special purposes such as the diet for infant, pregnant woman and athlete by updating the rules.

F. REFERENCES

5. P. H. Winston and B. K. P. Horn, "LISP".
E.3 More extension for sensitivity studies

An important tool in designing and evaluating expert models is a consistency checking procedure which lists all differences between the conclusions for the stored cases before and after a model has been modified. A common modification would be to change a production rule so as to correct the conclusions of the case. The forward chaining is chosen to check the consistency, and comparing the time performance with our system (with backward chaining).

The forward chaining stars with a collection of facts and tries all available rules over and over, adding new facts as it goes, until no more rule applies.

Characteristics of forward chaining:
1. No hypothesis is assumed. (that is different from backward chaining)
2. The question is asked from the IF part of each rule.
3. The most important technique is to "prune" the search tree in order to reduce the search space (procedure gar is responsible for collecting relevant information).
4. The program will terminate if no more relevant rules can be checked further.
5. Collecting irrelevant facts is very time consuming, especially for conflict case like those we have.
   Forward chaining can't do it efficiently.
6. The forward-chaining control strategy only can get one solution, but optimal solution could be got by backward-chaining. Thus, the forward-chaining is more efficient.

The power, generality are important aspects of an expert sys-
tem, and the distinct tools (backward-chaining v.s. forward-chaining) have special purpose. The efficiency and power of inference are tradeoff. The conclusion is backward chaining is more powerful as a tool of expert system.