AN EXPERT SYSTEM ASSISTING DECISION-MAKING AT LDC FOR THE NATIONAL ELECTRICAL GRID FAULTS.

A DISSERTATION SUBMITTED IN PARTIAL
FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER SCIENCE & ENGINEERING
BY

MD. ANISUR RAHMAN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BANGLADESH UNIVERSITY OF ENGINEERING & TECHNOLOGY,DHAKA.
AUGUST,1990

AN EXPERT SYSTEM ASSISTING DECISION -MAKING AT LDC FOR NATIONAL ELECTRICAL GRID FAULTS.

A Thesis

By

MD. ANISUR RAHMAN

Accepted as satisfactory as to style and contents for partial fulfillment for the degree of M.Sc. Engineering in Computer Science and Engineering on 29-8-90.

BOARD OF EXAMINERS

DR. SYED MAHBUBUR RAHMAN
Associate Professor and Head,
Computer Science and Engeneering Dept.
Bangladesh University of Engineering and
Technology, Dhaka, Bangladesh.

Chairman
and
Supervisor

DR.MD. SEKENDAR ALI
Director,
Bangladesh Institute of Technology (BIT)
Dhaka.

Member
and
Co-supervisor

DR. MD. SHAMSUL ALAM
Associate Professor,
Computer Science and Engineering Dept.
Bangladesh University of Engineering and
Technology, Dhaka, Bangladesh.

Member

DR.SYED FAZL-I RAHMAN
Professor
Electrical and Electronic Engineering Dept.
Bangladesh University of Engineering and
Technology, Dhaka, Bangladesh.

Member
(External)

TO

THE MEMORIES

OF

MY FATHER

CERTIFICATE OF RESEARCH

This is to certify that the work presented in this thesis is the result of the investigation carried out by the candidate under the supervision of **Dr. Syed Mahbubur Rahman** Associate Professor & Head, Department of Computer Science & Engineering, Bangladesh University of Engineering & Technology, Dhaka, Bangladesh.

Counter signed by
Supervisor

(DR.SYED MAHBUBUR RAHMAN)

Signature of the
Candidate

(MD. ANISUR RAHMAN)

DECLARATION

I do hereby declare that neither this thesis nor any part thereof has been submitted or is being concurrently submitted in candidature for any degree at any other university.

Candidate

ACKNOWLEDGMENT

I would like to express my thanks to the authorities of Bangladesh Institute of Technology (B.I.T), Khulna, especially to the Director **prof. M.A. Hannan** for awarding me UNDP stipend for M, Sc, Engineering program.

I express my deep sense of gratitude to my supervisor **Dr. Syeed Mahbubur Rahman** Head, Computer Science & Engineering Department, Bangladesh University of Engineering & Technology, Dhaka, for his close support, unwavering encouragement and overall guidance during the course of this work. This research would not have been even in this stage but only for his continual guidance.

I am also indebted to my Co-supervisor **Dr. Md. Sekendar Ali**, Director, Bangladesh Institute of Technology (B.I.T), Dhaka, for his valuable suggestions and close guidance in fault analysis of power system.

Finally I would like to express my gratitude to **Md. Bazlur Rahman**, Chief Engineer transmission & system protection, P.D.B, and **Md. Illius**, Deputy Director, operation LDC, for their contributions during all phases of the present work as domain expert.

ABSTRACT

Artificial Intelligence has undergone tremendous development as well as revolutionary change in computer science since its birth at Dartmouth college, USA in 1966. As a result of these development, Expert systems have emerged as an applied branch of AI. The application of AI methodologies to the power system is a new area of research.

The behavior of the modern interconnected grid system has become more and more complicated, causing decision making process to be increasely difficult. This study is aimed to develop an expert system to assist the decision making and to suggest the necessary actions by applying the expert knowledge in order to ensure the national grid supply normal.

With reference to the system analysis, the power Development Board (PDB) of Bangladesh has a Load Dispatch Center (LDC). The objective of this center is to make co-ordination between the demand and generation through out the whole country. In order to do so the LDC has to make liaison with more or less thirty (30) power stations near about sixty four (64) sub-stations located at different places of Bangladesh and to face the related electrical faults so as to ensure normal power supply in the grid.

In this contest the first part of this work is to make study and survey about the expert system and then to examine the possibilities to develop an expert system for the proposed field. Since acquiring the knowledge is a heart of an expert system, so

the next portion of this work describes the knowledge acquisition techniques from domain expert. This knowledges is incorporated in the data base in terms of production rules and finally the program was developed by PROLOG.

It is expected that the developed expert system will offer quick optimum decision both by text and in graphics during the time of electrical faults to the system operators and will be a valuable tool for Load Dispatch Center of PDB.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1

INTRODUCTION TO EXPERT SYSTEM.

1.1 A SHORT HISTORY OF EXPERT SYSTEM:

It is difficult to pinpoint an exact starting date for what is commonly called expert system a sub-class of AI. Perhaps credit for the birth of AI should be given to A.M TURNING for his invention of the stored program computer in the early of 1950.

Turning's recognition that a program could be stored as data in the computers memory and executed later formed the basis for all modern computers. The storing of programs allowed the computer's function to be changed quickly and easily simply running a new program. This capability implies that a computer might be able to change its own functions ---that is to learn or think.[1]

The term "Artificial Intelligence" is usually regarded as having been invented by JOHN MCCARTHY in 1956, then assistance professor of Mathematices, Dartmouth College in Hanover N.H. At that time he convened a conference which was the begining of AI as a separate branch of computer science. [1]

The term "Artificial Intelligence" is generally credited to MARVIN MINSKY of M.I.T, who in 1961 wrote a paper entitled, "steps towards artificial intelligence". The 1960 were a period of intense optimism over the possibility of making a computer think.

In 1964 JOSEPH EIZEN BAUM of M.I.T published the most famous and highly publicated "ELIZA" a mini expert system program that acted like a Rogerian phychoanalyst.

By the end of the 1970, several success such as natural language processing, knowledge representation and problem solving had been achieved in the specific area of A.I. Thus the stage was set for the introduction of the first commertial product, the expert system.

During the 1970 EDWARD FEIGENBAUM, at Standford developed the first expert system "DENDRAL" used for analysis of mass spectography data. At the middle of 1970 one of the most successfull expert system was "MYCIN", developed at standford university and which was created to help physician diagnose illness.[17]

Today many large companies -- IBM, Digital Equipment Corporation, Hitachi etc, have set up AI research laboratories and important research is being conducted in many institute and universities. The Japanese plans for fifth-generation computer require massive finding in AI research, development and increasing commercial emphasis is being given to the development of particular expert system for specific purpose. Now-a-days in U.S.A. more than one thousand expert systems are working.[6]

1.2 DEFINITION OF EXPERT SYSTEM:

Expert systems, a sub-class of AI, are being developed for a variety of particular purpose. Expertise is collected from human beings and fed into systems with a capacity to store and to manipulate the knowledge in response to the subsequent user inquiries.[17]

Expert systems are one of the key development contributing to the international fifth-generation program which may be regarded as a means of recording and accessing human competence in a particular specialist field.[17]

Most robust interpretation (e.g. Dudaet at 1980) suggested that an expert system is capable of human performance and can serve there by as an replacement expert.

Less ambitiously, Expert system may be seen as exhibiting competence in a relatively narrow agreed domain serving as tools to aid communication between human experts.[11]

The British Computer Society's Committee of the specialist group on Expert system has defined an expert system as---

" The embodiment within a computer of a knowledge-base component form an expert skill in such a form that the machine can offer intelligent advice or take an intelligent decision about a processing function. A desirable additional characteristic, which may be regarded as fundamental, is the capacity of the system to justify its own line of reasoning in a manner directly intelligible to the inquirer. The style adopted to attain these characteristic is rule-based programming".[17]

In conclusion we can say the expert systems are sophisticated computer programs that manipulate knowledge to solve problems efficiently and effectively in a narrow problem area. Like real human experts, these systems use symbolic logic and heuristics--rules of thumb-- to find solutions. And like real experts, they make mistake but they have the capacity to learn from their errors.[2]

1.3 ADVANTAGES OF EXPERT SYSTEM:

There are some excellent reasons for using artificial expertise to human reasoning. Some of these advantages are summerized as follows, figure 1 illustrates these advantages.

a) PERMANENCE: One advantage of artificial expertise is its permanence. Human expertise can quickly fad, regardless of whether it involves mental or physical activity. An expert must constantly practice and rehearse to maintain proficiency in some problem area.

Any significant period of disuse can seriously affect the expert's performance. The adage "use it or lose it" certainly applies here. However, the adage does not apply to artificial expertise. Once it is acquired, it is around for ever, barring catastrophic accidents related to memory storage. Its permanence is not related to its use.

b) TRANSFERABLE: Another advantage of artificial expertise is then ease with which it can be transferred or reproduced. Transferring knowledge from one human to another is the

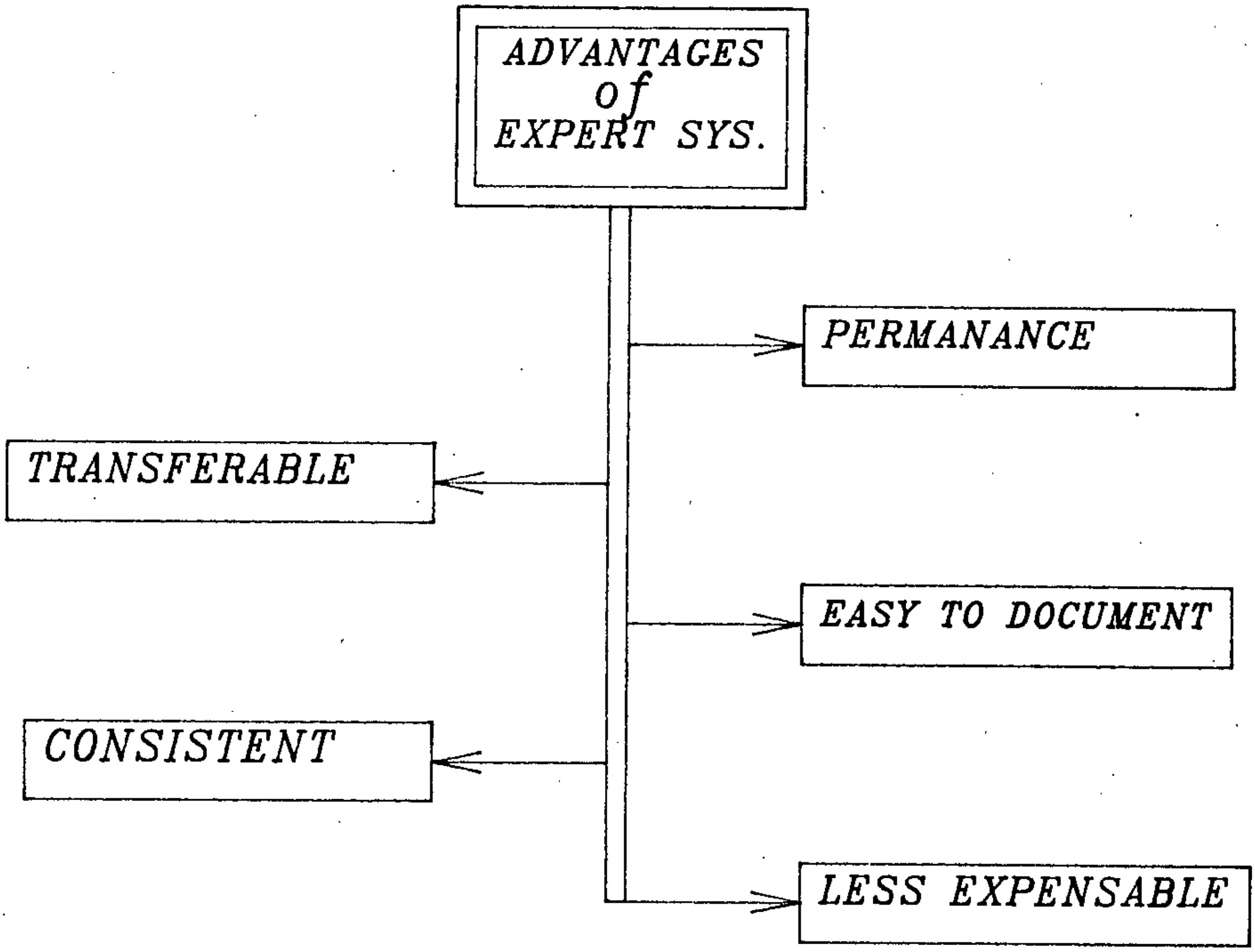


FIG:1 THE ADVANTAGES OF AN EXPERT SYSTEM laborious, lengthy and expensive process called education. Transferring artificial expertise is the trivial process of copying or cloning a program or data file.

c) **EASY TO DOCUMENT:** Artificial expertise is also much easier to document. Documenting human expertise is extremely difficult and time-consuming, as any experienced knowledge engineer will verify. Documenting artificial expertise is relatively easy. There is a straightforward mapping between the way in which the expertise is represented in the system and natural language description of that representation.

d) **CONSISTENT:** Artificial expertise produces more consistent, reproducible results than does human expertise. A human expert may make different decisions in identical situations because of emotional factors. For example a human may forget to use an important rule in crisis situation because of time pressure or stress. An expert system is not susceptible to these distractions.

e) **LESS EXPENSIVE:** A final advantage of artificial expertise is its low cost. Human experts, especially the top-notch ones, are very scarce, and hence very expensive and they demand large salaries. Expert systems, by contrast, are relatively inexpensive.

The expert systems are costly to develop but cheap to operate. Their operating cost is just the nominal computer cost of running the program. Their high development cost (years of effort by high priced knowledge engineers and domain experts) is offset by their low operating cost and the ease with which new copies of the system can be made.

1.4 OBJECTIVES OF PRESENT RESEARCH:

The Power Development Board (PDB) of Bangladesh has central load dispatch unite. The objectives of this center are as follows

* To maintain the co-ordination between the demand and generation of national electrical grid.

* To ensure the quality of supply i,e, the voltage and current rating should not exceed the limits (+10, V,-5).

* Safety of the equipments connected with the grid system.

* To maintain the security of the working man.

* Overall to make the system reliable.

In order to do so the L D C has to make liaison with more or less thirty power stations and near about sixty four sub-stations located at different places of Bangladesh. Figure 2 shows the important power stations of Bangladesh which are under direct controll of LDC. These power stations and sub-stations are interconnected with 66KV, 132KV and 230KV power lines, all of these associately called the NATIONAL ELECTRICAL GRID SYSTEM, as illustrated in figure 3.

To make co-ordination between the stations and sub-stations

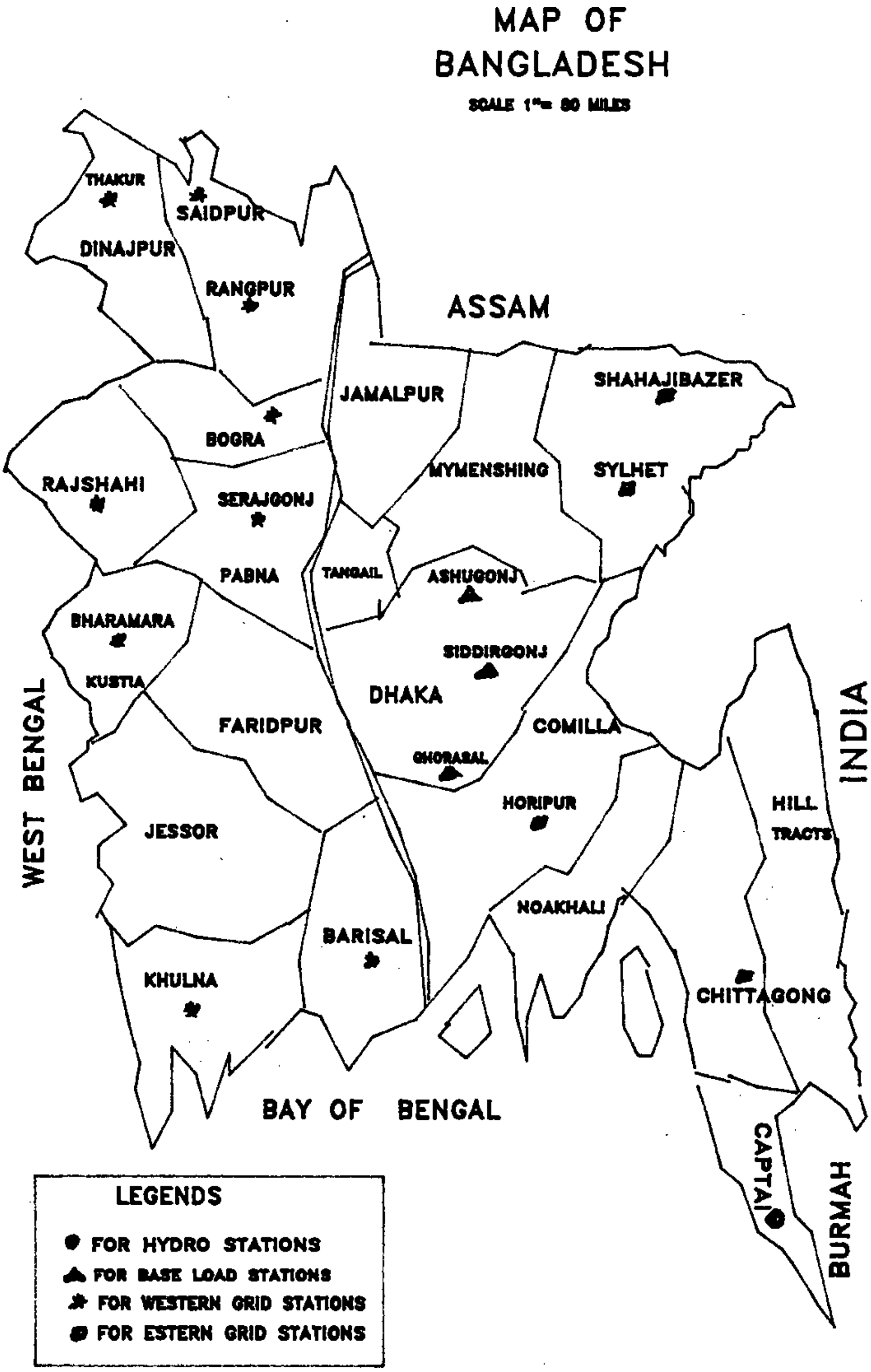


FIG: 2 THE IMPORTANT POWER STATIONS OF BANGLADESH

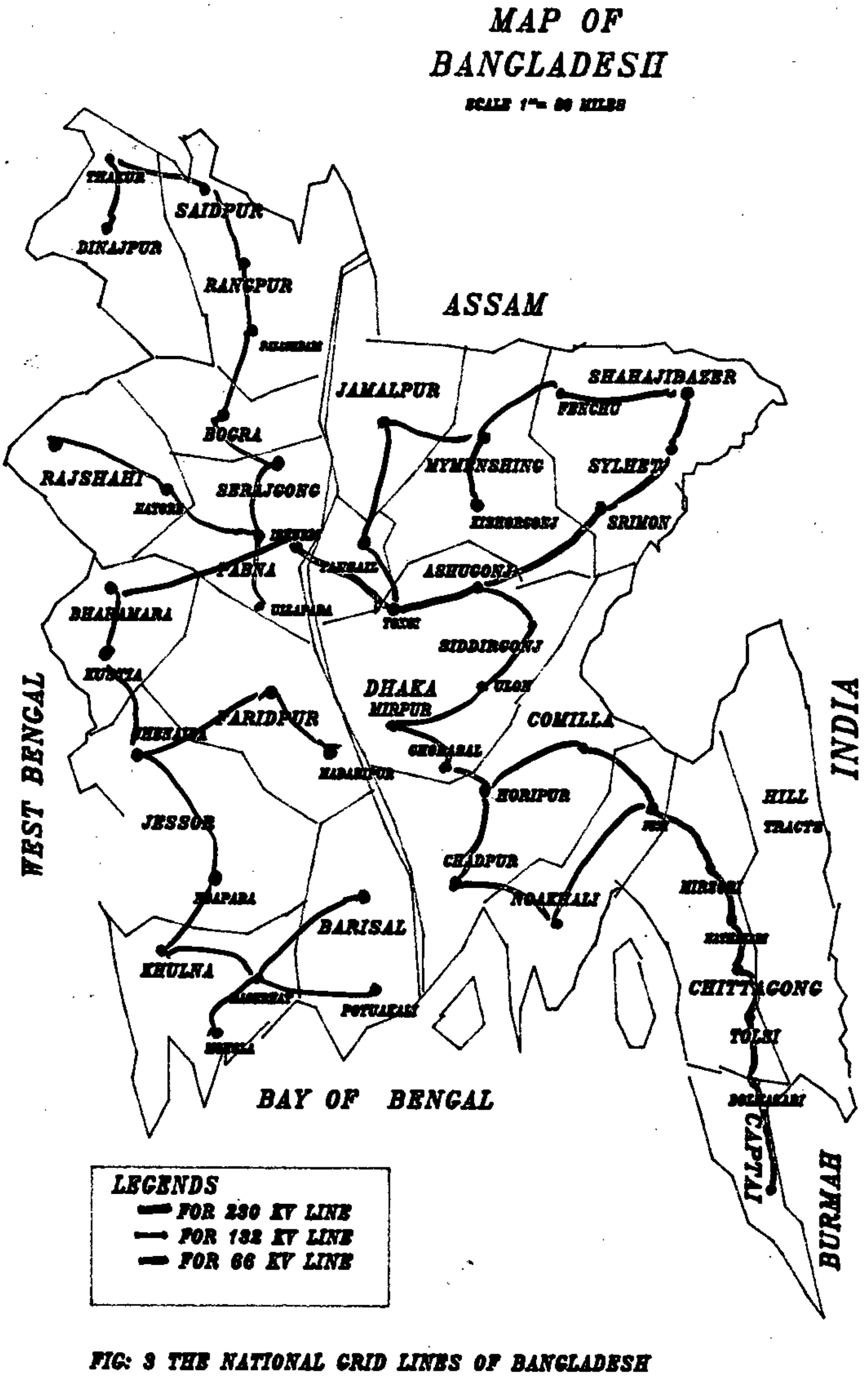


FIG: 3 THE NATIONAL GRID LINES OF BANGLADESH at different places of Bangladesh the center has a telemeter board to display the momentary power generation at different power stations and the total power transmission through different grid lines.

The center also povide frequency and voltage meters to display the voltage and frequency status of the grid system. At the time of electrical faults the operators can convey the decisions of the system expert over carrier telephony.

This is a challenging problem of PDB to ensure the most economical and efficient grid supply in context to the growing electrical demand with limited power stations of which a good number of generators are inefficient. Moreover to give the accurate decisions at the time of electrical faults in the interconnected complicated grid system has become increasely difficult. In this contest the followings are taken as the objectives of the present study:

a) The application of AI (artificial intelligence) methodologies to the power system is a new area of research. The behavior of the modern interconnected grid system has become more and more complicated, causing decision making process to be increasely difficult. This study is aimed to develop an expert system to assist the decision making and to suggest the necessary actions by applying the expert knowledge in order to ensure the national grid supply normal.

b) During the real time environment if a problem is detected, the expert system can be applied to suggest a solution to the operator in time, based on the incorporated expert knowledge.

c) The knowledge of the system experts (the chief engineer, director, deputy director) will be incorporated in the global data base working memory. So the appropriate decisions can be achieved quickly without bothering the superior experts.

d) The knowledge required to perform a task will be expressed in terms of production rules which are very close to natural language and therefore easy to understand to the operators. Each production rule represents a piece of knowledge relevant to the task. Hence it is very convenient to add or to remove the rules when more experienced is gained.[14 & 12]

1.5 A BRIEF METHODOLOGY OF PRESENT WORK:

The proposed approach to the development of an expert system consisting of the folling steps--

a) A successfull expert system relies on a high quality knowledge base. So the first step is to identification of knowledge to develop such system.

b) To procure knowledge used by the system experts of the load dispatch center. The knowledge required to solve a problem may be sophisticated, which often results in a large rule base, developed on engineering judgment.

c) Converting the production rules in to the object oriented language like PROLOG or LISP.

d) The graphical representation of the of the system problem and its solutions to make a user-friendly man-machine interaction.

Finally the proposed expert system may offer more fastly the optimum decisions both by the text and in graphics during the time of electrical problems to the system operators and will be a valuable tool in operating the national grid system in LDC of PDB.

CHAPTER-2

EXPERT SYSTEM SURVEY

2.1 GENERAL DISCUSSION:

The expert system area is expanding rapidly. Both Government and Industries in abroad are introducing commercial expert system in their respective fields and with in a few years a large number of U.S. companies has involved in AI or Expert system research and development. Major Universities , research insitutions and private corporations are working to bring the promise of expert systems to fulfilment. In this section survey reports of some important aspects of expert system in practical fields are discussed.

There are over one thousand expert systems reported to be in use today and their number is rapidly increasing. These expert systems have been build to solve different types of problems, but their basic activities can be grouped into different categories. Table 1 summerizes the basic categories of expert system with their activities. The categories are discussed below

a) INTERPRETATION: Expert system that perform interpretation typically use sensor data to infer situation descriptions. An example is interpreting gauge reading in a chemical process plant to infer the status of the process. Interpretation system may process many different kinds of data, for example, both vision and speech understanding system use natural input--visual images in one case, audio signal in other-- to infer features and meaning.

TABLE: 1 GENERAL CATEGORIES OF EXPERT SYSTEM ACTIVITIES

CATEGORY	PROBLEM ADDRESSED
INTERPRETATION	INFERRING SITUATION DESCRIPTIONS FROM SENSOR DATA
PREDICTION	INFERRING LIKELY CONSEQUENCES OF GIVEN SITUATIONS
DIAGNOSIS	INFERRING SYSTEMS MALFUNCTION & DISORDERNESS FROM OBSERVATION
DESIGN	CONFIGURING OBJECTS UNDER CONSTRAINTS & EXAMS
PLANNING	DESIGNINIG ACTIONS
MONITORING	COMPARING OBSERVATIONS TO OUTPUT
DEBUGGING	PRESCRIBING REMEDIES FOR MALFUNCTION
REPAIR	EXECUTING PLANS TO ADMINISTER PRESCRIBED REMEDIES
INSTRUCTION	DIAGNOSING, DEBUGGING, AND REPAIR STUDENT BEHAVIOR
CONTROL	GOVERNING OVERALL SYSTEM BEHAVIOR

Chemical interpretation system use X-ray diffraction data or mass spectral to infer the structure of compounds. Geological interpretation system use dipmeter logs to determine subsurface geological structure. Medical interpretation systems use measurements from patient monitoring systems (e.g. heart rate, blood pressure) to diagnose and treat illness.

b) **PREDICTION:** Expert systems that perform prediction infer the likely consequences of given situations. Examples are predicting the damage to crops from some type of insect, estimating global oil demand from the current geo-political world situation, and predicting where armed conflict will next occur based on intelligence reports.

c) **DIAGNOSIS:** Expert systems that perform diagnosis use situations, behavior characteristics, or knowledge about component design to infer probable causes of system malfunctions. Examples are determining the causes of diseases from symptoms observed in patients, locating faults in electrical circuits, and finding defective components in the coolant systems of nuclear reactors.

Diagnosis systems may interact with the user to help find the faults and then suggest courses of action to correct them. A good number of diagnosis system have been developed in medicine problem area, however, many diagnosis systems are now being built for engineering and computer systems.

d) **DESIGN:** Expert systems that perform design develop configurations of objects based on a set of problems constraints. Examples are gene-cloning, designing integrated circuits layouts, and creating complex organic molecules. The design systems can save much unnecessary search by creating plans for producing the wanted configuration and evaluating them in the context of the problem constraints. The two most popular application areas design systems seem to be molecular biology and micro-electronics.

e) **MONITORING:** Expert systems that perform monitoring compare actual systems behavior to expected behavior. Examples are monitoring instrument readings in a nuclear reactor to detect accident conditions and assisting patients in a intensive care unit by analyzing data from the ICU monitoring equipment.

f) **REPAIR:** Expert systems that perform repair follow a plan to administer some prescribed remedy. An example is tuning a mass spectometer, i.e. setting the instrument's operation controls to achieve optimum sensitivity consistent with correct peak ratio and shapes. Very few repair systems have been developed, partially because the act of executing an actual repair on a real-world object adds an extra dimension of complexity to the problems.

g) **INSTRUCTION:** Expert systems that performs instruction diaganose, debug and repair student behavior. Examples are teaching students troubleshoot electrical circuits, instructing Navy personnel in the operation of a steam propulsion plant, and educating medical students in the area of antimicrobial therapy section. Instruction systems develop a model of what the student knows and how the knowledge is applied to solve problems. They diagnose and debug the student deficiencies by analyzing the model and devising plans for correcting the deficiencies.

h) CONTROL: Expert systems that perform control adaptively govern overall system behavior. Examples are managing the manufacturing and distribution of computer systems and controlling the treatment of patients in an intensive care unit. control systems must include a monitoring component to track behavior over time, but they also may require components to perform any or all of the other types of tasks just discussed.

2.2 APPLICATION AREAS OF EXPERT SYSTEM:

Now-a-days expert systems are working more or less every sphere of problem domains, and a good number of expert systems are under research. Of these areas, the medical domain seems the most popular. More expert systems have been developed for medicine than for any other single problem area, although chemistry is a close second and closing fast. Table 2 shows the major application areas of expert system and are briefly discussed below.

a) COMPUTER SYSTEM: Expert system work in computer systems is typified by XCON, one of the first and most successful systems of this type. Initiated by Digital Equipment Corporation and Carnegie Mellon University in the late 1970s as a research project, XCON evolved into a commercial system for configuring computers. Current expert systems work in computer systems includes fault diagnosis, computer configuration, and manufacturing control.

b) **ELECTRONICS:** Expert system work in electronics is dominated by research and development efforts involving faults diagnosis and integrated circuit design. ACE, developed by Bell Laboratories in early 1980s, typifies fault diagnosis system in this area. It is being used by AT&T to locate and identify trouble spots in telephone line. Current expert system work in electronic also includes the development of instructional system for electrical trouble-shooting and digital circuit design.

c) **ENGINEERING:** Expert systems working in engineering is typified by DELTA, a fault diagnosis system developed by General Electric in the mid-1980s. General Electric plans to use DELTA on a commercial basis to help maintenance personnel find malfunction in diesel locomotives.

d) **CHEMISTRY:** Expert system work in chemistry started with DENRDAL, an innovative research project begun at Stanford University in the mid-1960s and dedicated to developing AI methods for determining the topological structure of organic compounds. Current expert system work in chemistry includes inferring molecular structure, synthesizing organic molecules and planning experiments in molecular biology.

TABLE-2	APPLICATION AREAS FOR EXPERT SYSTEM
COMPUTER SYSTEM	AGRICULTURE
ELECTRONICS	MANUFACTURING
ENGINEERING	MEDICINE
CHEMISTRY	MILITARY SCIENCE
GEOLOGY	SPACE TECHNOLOGY

e) GEOLOGY: Expert system work in geology began with PROSPECTOR, a system developed by Standford Research Institute in the mid 1970s. PROSPERTOR was designed to help geologists locate ore deposits and accurately predicted the existence of multimillion dollar molybdenum deposit. Current expert system work in geology includes well log analysis and fault diagnosis related to drilling operations.

f) MEDICINE: Expert system work in medicine began with MYCIN, one of the earliest and best known expert systems, Developed at Stanford University in the mid-1970s. MYCIN helps a physician diagnose and treat infectious blood diseases and is now being used for research and medical teaching. Current expert system work in medicine includes interpretation of medical test data, disease diagnosis, disease treatment, and instruction in medical diagnosis and management techniques.

g) MILITARY SCIENCE: Expert system work in military science has focused on interpretation, prediction and planning. One of the first military expert systems was HASP/SIAP, developed jointly by Stanford University and System Control Technology in the early 1970s. This system identifies ship types by interpreting data from hydrophone arrays that monitor regions of the ocean. Current expert system work in the military includes interpretation of sensor data, prediction of combat results and tactical planning.[2 & 14]

2.3 EXAMPLES OF EXPERT SYSTEM:

The survey of expert system examples will offer a very broad view of what expert system do and what kinds of problems they solve. In this section some represantative examples of expert system and thier specific activities are highlited in great deal.

2.3.1 EXPERT SYSTEMS IN COMPUTER ENGINEERING:

Some selected expert systems in computer system is shown in figure 4 and thier brief description given below:-

DART assist in diagnosing faults in computer hardware systems using information about the design of the device being diagnosed. The system works directly from information about the intended structure and expected behavior of the device to help find the design flaws in newly created devices. The system has been applied to simple computer circuits and the teleprocessing facility of the IBM 4331.

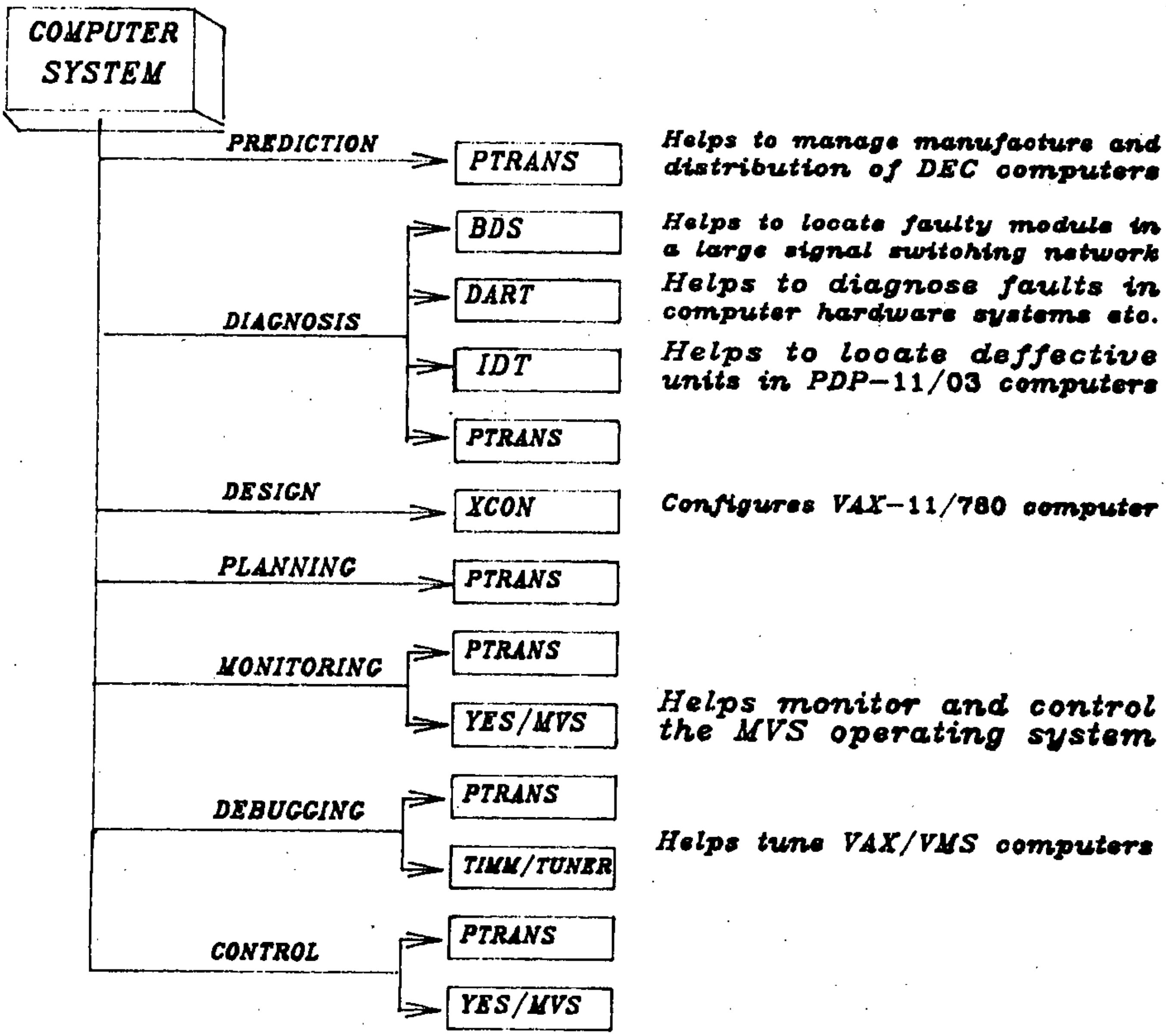

COMPUTER SYSTEM
PREDICTION
PTRANS
Helps to manage manufacture and distribution of DEC computers
DIAGNOSIS
BDS
Helps to locate faulty module in a large signal switching network
DART
Helps to diagnose faults in computer hardware systems etc.
IDT
Helps to locate deffective units in PDP-11/03 computers
PTRANS
DESIGN
XCON
Configures VAX-11/780 computer
PLANNING
PTRANS
MONITORING
PTRANS
YES/MVS
Helps monitor and control the MVS operating system
DEBUGGING
PTRANS
TIMM/TUNER
Helps tune VAX/VMS computers
CONTROL
PTRANS
YES/MVS

FIG: 4 THE SELECTED EXPERT SYSTEMS IN COMPUTER SYSTEM

DART uses a device-independent interface procedure that is similar to a type of resolution theorem proving, where the system attempts to generate a proof related to the cause of the device's malfunction. The system is implemented in MRS and was developed at Stanford university.

IDT helps a technician to locate the field replaceable units that should be replaced to fix faults in PDP 11/03 computers. The system uses knowledge about the unit under test, such as the functions of its components and their relation to one another, to select and excute diagnostic tests and interpret the results. the system is rule based using forward chaining and is implemented in LISP and OPS5.

TIMM/TUNER assists in tuning VAX/VMS computer systems in order to reduce performance problems that arise in a constant changing computer environment. It interacts with the system manager, asking a series of questions, that lead to a recommended action, such as adjusting system parameters or user authorized value, redistributing or reducing user demand, changing user software design, or purchasing new hardware. The system uses a rule-based representation scheme created within TIMM, a commercial system for automatic knowledge acquisition. TIMM/TUNER was developed by General Research Corporation and reached the stage of a commercial system.

YES/MVS helps computers operator monitor and control the

TABLE: 3 EXPERT SYSTEMS IN COMPUTER SCIENCE

AND THEIR APPLICATION AREAS

EXPERT SYSTEM	APPLICATION AREAS
CRIB	COMPUTER SYSTEM
DRAT	DIAGNOSING FAULTS IN HARDWARE
IDT	LOCATING THE REPLACEABLE UNITS
ISA	SCHEDULES CUSTOMER ORDER
MIXER	COMPUTER SYSTEM
PDS	PROCESSOR CONTROL
R1	CONFIGURES VAX11/780 SYSTEM
R1-SOLAR	COMPUTER SYSTEM
TIMM/TUNER	ASSISTS IN TUNING VAX/VMS SYSTEM
XCON	COMPUTER SYSTEM
XSEL	HELPS A SALESMAN TO SELECT COMPONENTS FOR A VAX 11/780 SYSTEM
YES/MVS	CONTROLLING MULTIPLE VIRTUAL STRORAGE

MVS (multiple virtual storage) operation system, the most widely used operating system in large mainframe IBM computers. YES/MVS system addresses six major categories of task: maintaining adequate JET (job entry system) queue space, handling network communication between computers on the same site, scheduling large batch jobs off prime shift, responding to hardware errors, monitoring sub software systems, and monitoring overall system performance. YES/MVS runs in real time, directly interpreting MVS messages and sending either commands to the operating system or recommendations to the console operator. YES/MVS is rule-based expert system with forward chaining control scheme. Table 3 summarizes the expert systems in computer science and thier application areas

2.3.2 EXPERT SYSTEM IN ELECTRONICS:

AEC identifies trouble spots in telephone network and recommends appropriate repair and rehabilitative maintenance. The system operates without human intervention, analyzing maintenance reports generated on a daily basis by CRAS, a cable repair administration computer program. Once ACE locates the faulty telephone cables, it decides whether they need preventive maintenance and selects the type of maintenance most likely to be effective. ACE then stores its recommendations in a special data base that user can access, the system make decisions by applying knowledge about center, CRAS maintenance reports, and network analysis strategies. It uses a rule-based system controlled by forward chaining and was developed by Bell Laboratory at Whippany, Newjersey.

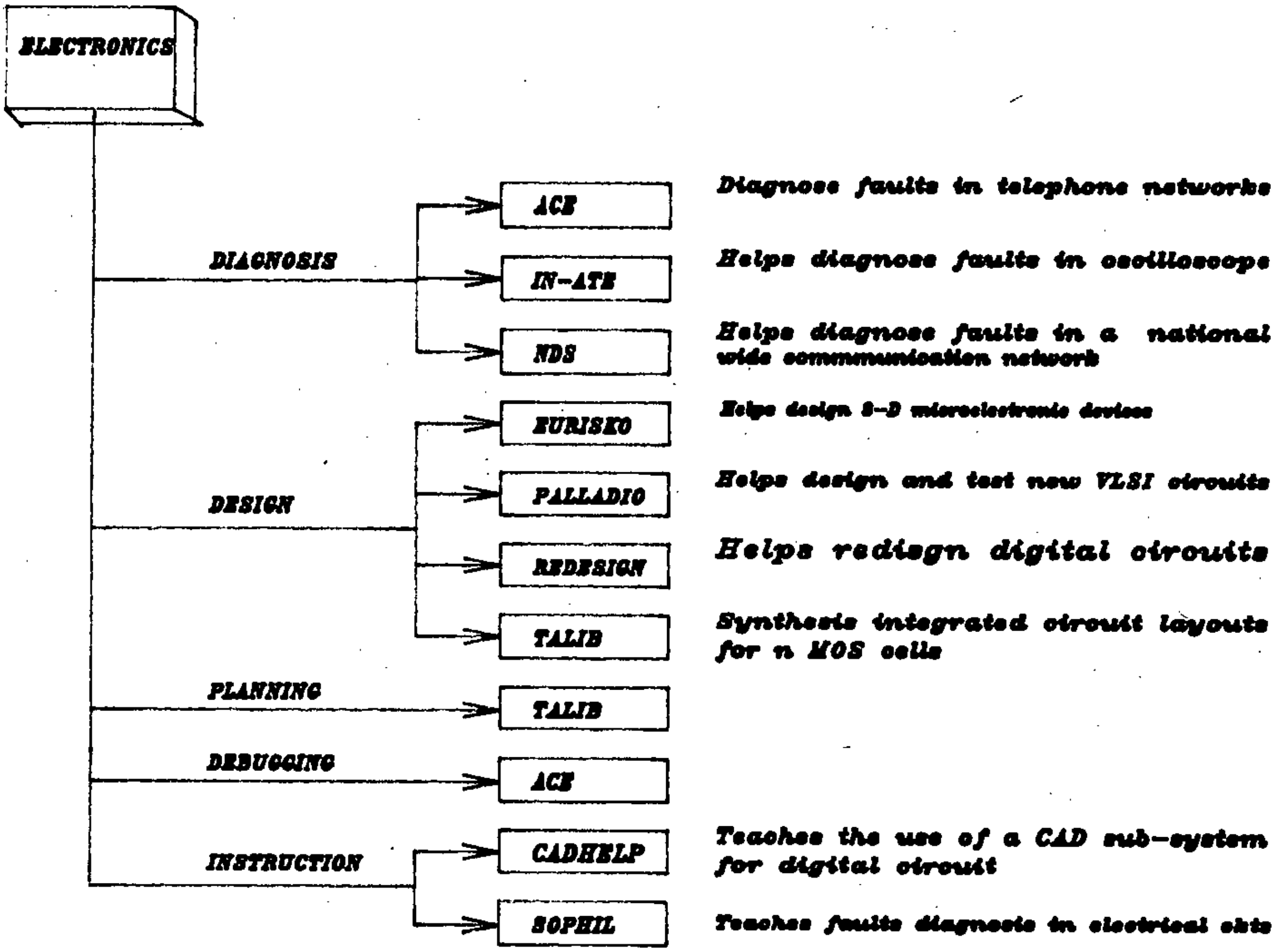

ELECTRONICS
DIAGNOSIS
ACE
Diagnose faults in telephone networks
IN-ATE
Helps diagnose faults in oscilloscope
NDS
Helps diagnose faults in a national wide commmunication network
DESIGN
EURISKO
Helps design 3-D microelectronic devices
PALLADIO
Helps design and test new VLSI circuits
REDESIGN
Helps rediagn digital circuits
TALIB
Synthesis integrated circuit layouts for n MOS cells
PLANNING
TALIB
DEBUGGING
ACE
INSTRUCTION
CADHELP
Teaches the use of a CAD sub-system for digital circuit
SOPHIL
Teaches faults diagnosis in electrical ckts

FIG 6 THE SELECTED EXPERT SYSTEMS IN ELECTRONIC DOMAINS

IN-ATE helps a technician troubleshoot a Tektronix Model 465 oscilloscope by analyzing symptoms and producing a decision tree of test points to be checked by the technician. The system applies two types of rules: those supplied by an expert diagnostician, and those generated automatically from an internal model of the oscilloscope, with a block diagram of the unit augmented with component failure rates. The system is implemented in LISP, and was developed at the Naval Research Laboratory.

EURISKO learns new heuristics and new domain-specific definitions of concepts in a problem domain. The system can learn by discovery in a number of different problem domains, including VLSI design. EURISKO has tackled the problem of inventing new kinds of three-dimensional micro-electronic devices that can be fabricated using laser recrystallization techniques and has designed new and interesting microelectronic devices. EURISKO operates by generating a device configuration, computing its input/ output behavior, assisting its functionality, and than evaluating it against other comparable devices. The system is implemented in INTERLISP and was developed at Standford University.

TALIB automatically synthesizes integrated layouts for MOS cells. the system takes as input a description of the circuit components to be laid out on the silicon water, their interconnections, and the topological and geometric requirements around the outside boundary of the circuit. From these specifications the system produce correct and compact cell layouts. TALIB creates and refines plans for laying out the circuit and than applies the plans using knowledge about subcircuit interconnection characteristics and the propagation of constraints between subcircuit. TALIB is a forward chaining, rule-based system imlpemented in OPS5. It was developed at Carnegie-Mellon University, as shown in figure 5.

2.3.3 EXPERT SYSTEMS IN ENGINEERING:

REACTOR assists reactor operators in the diagnosis and treatment of nuclear reactor accidents by monitoring instrument readings, such as feed-water flow and containment radiation level, looking for deviations from normal operating conditions. When system detects a deviation, it evaluates the situations and recommend appropriate actions, using knowledge about the reactor configuration and the functional relations of its components together with the knowledge about the expected behavior of the reactor under known accident conditions. REACTOR is implemented in LISP as a rule-based system that uses both forward and backward chaining and was developed by EG&G Idaho.

DELTA helps maintenance personnel to identify and correct malfunction in diesel electric locomotives by applying diagnostic stategies for locomotive maintenance. The system can lead the user through an entire repair procedure, presenting computer-aided drawings of parts and subsystems, repair sequences in the form of videodisc movies, and specific repair instructions ones the malfunction is identified. DELTA is rule-based system

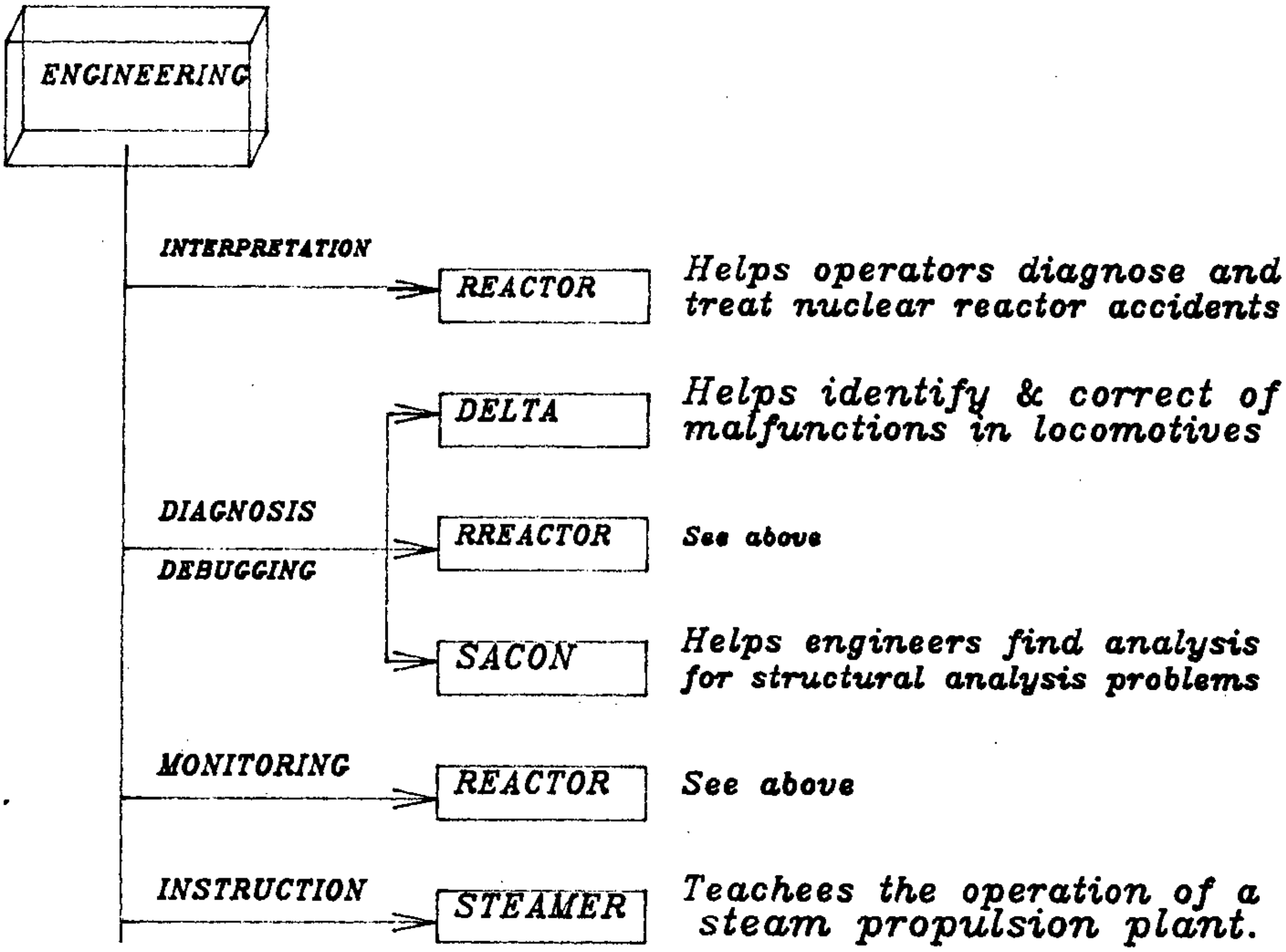

ENGINEERING
INTERPRETATION
REACTOR
Helps operators diagnose and treat nuclear reactor accidents
DELTA
Helps identify & correct of malfunctions in locomotives
DIAGNOSIS
DEBUGGING
RREACTOR
See above
SACON
Helps engineers find analysis for structural analysis problems
MONITORING
REACTOR
See above
INSTRUCTION
STEAMER
Teachees the operation of a steam propulsion plant.

FIG: 6 THE SELECTED EXPERT SYSTEMS IN ENGINEERING

developed in LISP.

SACON helps the engineers to determine and to analysis the strategies for particular structural problems. The engineer can then implement this strategy with MARC, a program that uses finite-element analysis methods to simulate the mechanical behavior of objects. SACON identifies the analysis class of the problem and recommends specific features of the MARC program to activate when performing the analysis. SACON uses knowledge about stress and deflections of a structure under different loading conditions to determine the appropriate strategy. Structure that can be analyzed include aircraft wings, reactor pressure vessels, rocket motor casings, bridges. SACON is a backward chaining, rule-based system implemented in EMYCIN. It was developed at Stanford University.

STEAMER instructs Navy propulsion engineering students in the operation of a steamer propulsion plant for a 1078-class frigate. The system can monitor the student executing the boiler light-off procedure for the plant, acknowledging appropriate student actions and correcting inappropriate ones. The system works by tying a mathematical simulation of the propulsion plant to a sophisticated graphical interface program that displays animated color diagrams of plant subsystem. The student can manipulate simulated components, such as valves, switches and pumps, and observe the effects on plant parameters, such as change in pressures, temperatures, and flows. Figure 6 illustrate the selected expert systems in engineering.

2.3.4 EXPERT SYSTEMS IN CHEMISTRY:

CRYSALIS infers the three dimensional structure of a protein from an electron density map(EDM). The system interprets X-ray diffraction data composed of position and intensity diffracted waves to infer this atomic structure. The system uses knowledge about protein composition and X-ray crystallography and heuristics for analyzing EDMs to generate and to test hypotheses about plausible protein structure. CRYSALIES uses a blackboard architecture, containing independent knowledge sources that build and test a multilevel hypotheses structure. The system is implemented in LISP & was developed at Stanford University.[2]

DENDRAL infers the molecular structure of unknown compounds from mass spectral and nuclear magnatic response data. The system uses a special algorithm developed by J. Ledenberg to systematically enumerate all possible molecular structure, it uses chemical expertise to prune this list of possibilities to a manageable size. Knowledge in DENDRAL is represented as procedural code for the molecular structure generator and as rule for the data driven component and evaluator. The system is implemented in INTERLISP and was developed at Stanford University.

MOLGEN assists the geneticist in planning gene-cloning experiments in molecular genetics. These experiments involve

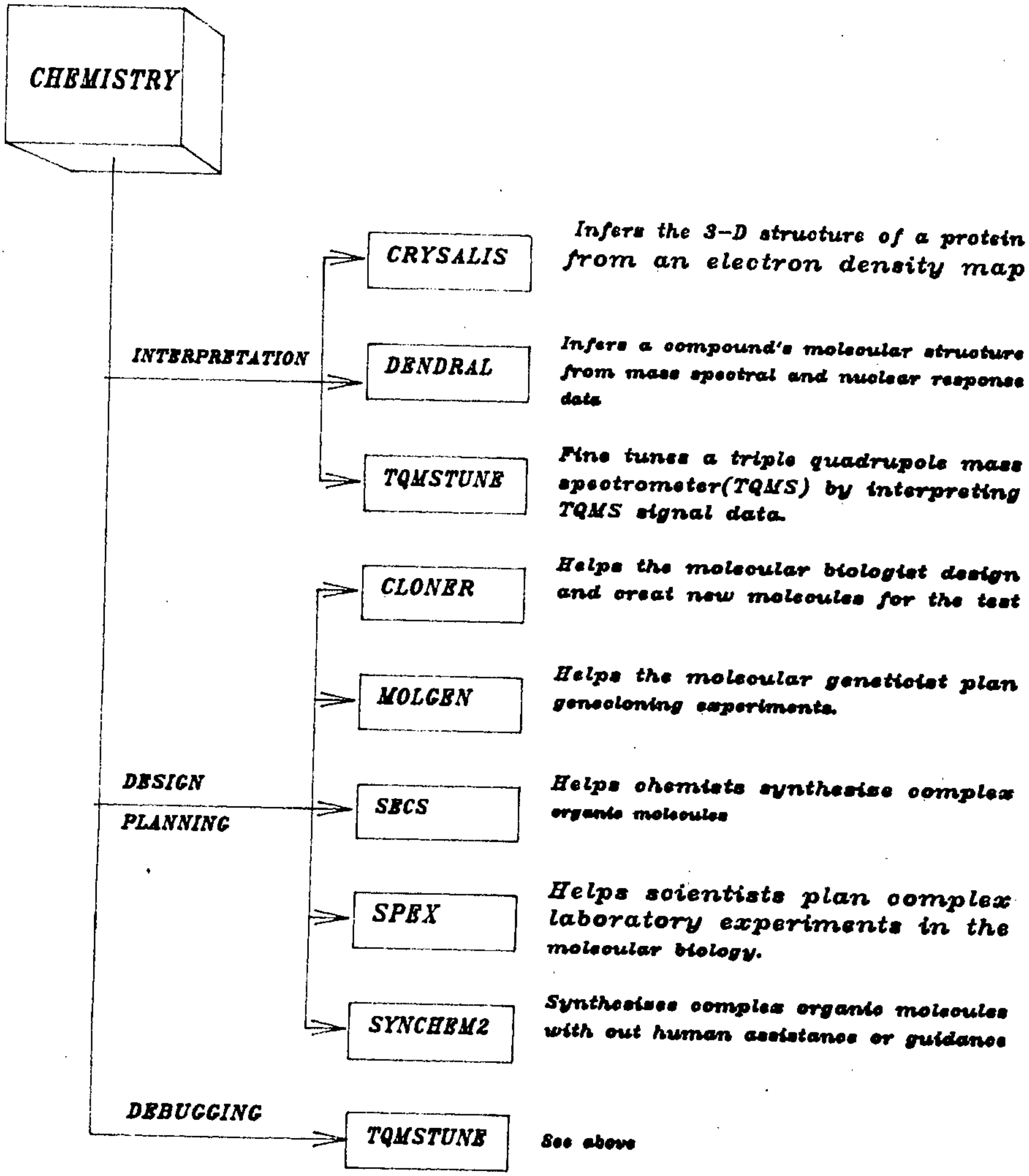

CHEMISTRY
INTERPRETATION
CRYSALIS
Infers the 3-D structure of a protein from an electron density map
DENDRAL
Infers a compound's molecular structure from mass spectral and nuclear response data
TQMSTUNE
Fine tunes a triple quadrupole mass spectrometer(TQMS) by interpreting TQMS signal data.
DESIGN
PLANNING
CLONER
Helps the molecular biologist design and creat new molecules for the test
MOLGEN
Helps the molecular geneticist plan genecloning experiments.
SECS
Helps chemists synthesise complex organic molecules
SPEX
Helps scientists plan complex laboratory experiments in the molecular biology.
SYNCHEM2
Synthesises complex organic molecules with out human assistance or guidance
DEBUGGING
TQMSTUNE
See above

FIG: 7 THE SELECTED EXPERT SYSTEMS IN CHEMISTRY splicing a gene coding for a desired protein product into bacteria so that the bacteria will manufacture it. The system uses knowledge genetics and the user's goal to create an abstract plan and then refines it to a set of specific laboratory steps. MOLGEN uses an object-oriented and frame-based representation and control scheme, and it is implemented in LISP and UNITS. It was developed at Stanford University. MOLGEN is primarily a vehicle for testing approaches for reasoning about design, rather than an operational expert system for molecular genetics.

SEQ helps molecular biologists perform several types of nucleotide sequence analysis. The system can store, retrieve and analyze nucleic acid sequences and it can provide a statistical analysis of structural homologies symmetries. SEQ's searching routines can be customized by manipulating a set of default parameters, for example, the biologist may vary the weights for penalties and size of gap results during a Needleman-wunch alignment. SEQ is implemented in LISP.[2]

SYNCHEM synthesizes complex organic molecules without requiring user interaction. The system uses knowledge about chemical reactions to create a plan for developing the target molecule from a set of given starting molecules. The system works backward, beginning with the target molecule and ties to determine with reactions could produce it and what materials would be required. The system is implemented in PL/1 and was developed at State University of New York, as shown in figure 7.

2.3.5 THE EXPERT SYSTEMS IN GEOLOGY:

Some representative expert systems extensively used in geological fields are illustrated in figure 8 and brief descriptions are given below

DIPMETER ADVISOR infers subsurface geological structure by interpreting dipmeter logs, measurements of the conductivity of rock in and around a borehole as related to depth below the surface. The system uses knowledge about dipmeter patterns and geology to recognize features in the dipmeter data and relate them to underground geological structure. The system provides the user with a menu-driven graphical interface incorporating smooth scrolling of log data. The system uses a rule based knowledge representation scheme controlled by forward chaining. It is implemented in INTER1ISP-D and operates on the Xerox 1100 series workstations. The system was developed by Schlumberger-Doll Research and reached the stage of a research prototype.

HYDRO helps a hydologist use HSPF, a computer program that simulates the physical processes by which precipitation is distributed throughout a watershed. The system assists in describing watershed chateristics to HSPF in the form of numerical parameters. The system estimates these parameters using knowledge about soil type, land use, vegetation, geology and their effect on the specific parameter in question. The system is patterned after PROSPECTOR, it uses combination rule-based and semantic net formalism to encode its knowledge.[2]

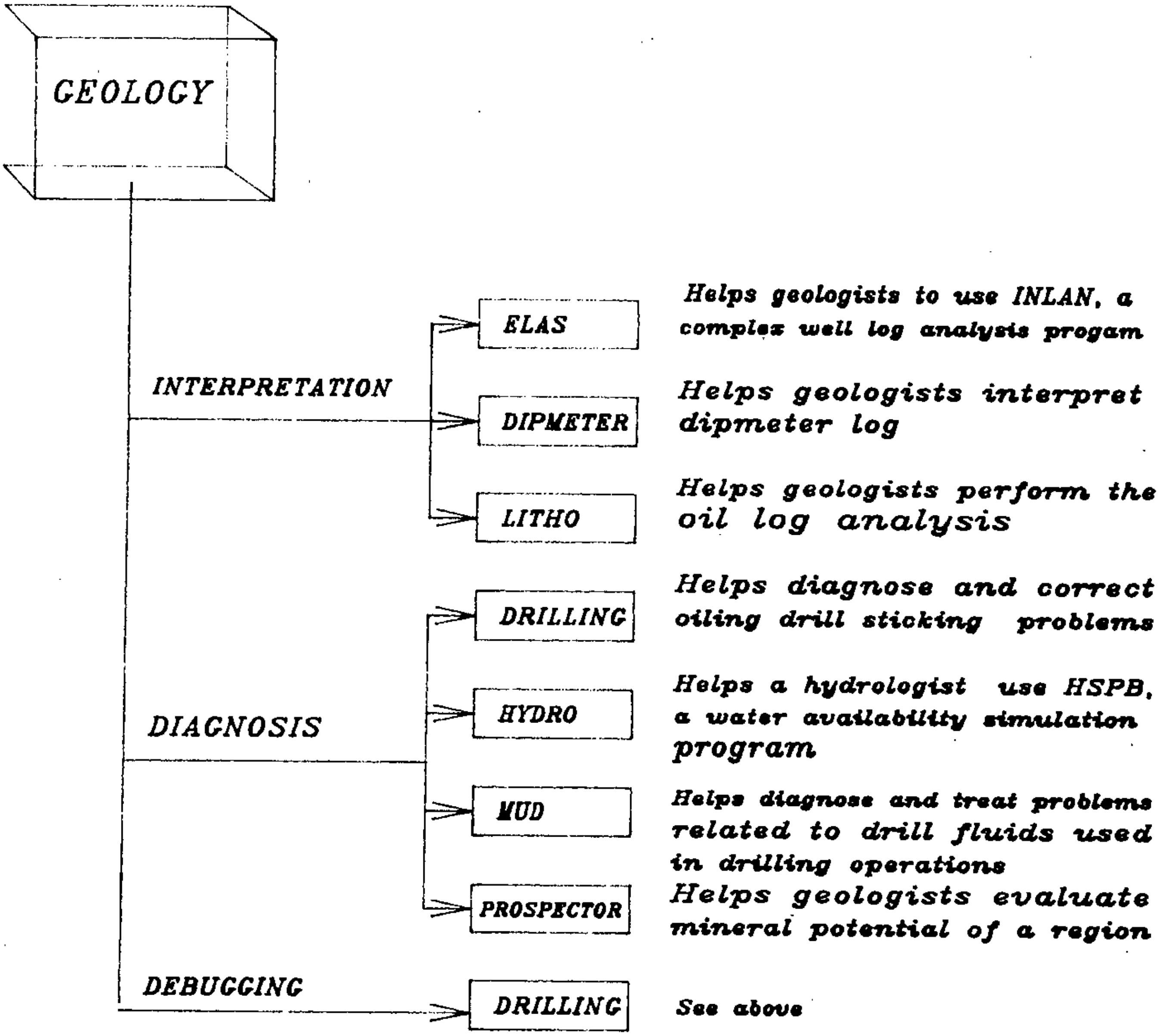

GEOLOGY
INTERPRETATION
ELAS
Helps geologists to use INLAN, a complex well log analysis progam
DIPMETER
Helps geologists interpret dipmeter log
LITHO
Helps geologists perform the oil log analysis
DIAGNOSIS
DRILLING
Helps diagnose and correct oiling drill sticking problems
HYDRO
Helps a hydrologist use HSPB, a water availability simulation program
MUD
Helps diagnose and treat problems related to drill fluids used in drilling operations
PROSPECTOR
Helps geologists evaluate mineral potential of a region
DEBUGGING
DRILLING
See above

FIG: 8 THE SELECTED EXPERT SYSTEMS IN GEOLOGY

2.3.6 EXPERT SYSTEM IN MEDICINE:

ABEL assists the clinician in diagnosing acid-base and electrolyte disorders in patients by applying about the diseases and the symptoms they produce. The system uses a causal model of the patient's possible diseases to the order queries to the clinician and guide the diagnostic process. The system was developed at MIT.

ANGY assists physician in diagnosing the narrowing of coronary vessels by identifying and isolating coronary vessels in angiograms. The system uses knowledge of cardiac anatomy and physiology to interpret the result, recognize relevant structures and eliminate irrelevant structure or artifacts caused by noise. ANGY was developed at the University of Pennsylvania.

ANNA assists physicians in administering digitalis to patients with heart problem, such as arrhythmia and congestive heart failure. The system uses patient symptoms and history to determine the appropriate dosage regimen, including the amount of digitais to administer and the rate at which it should be taken. Once the system prescribes an initial dosage, it monitors the patient's response to the drug adjusting the dosage appropriately when the demonstrated response fails to match the expected response. ANNA is implemented in LISP & It was developed at MIT.

2.3.7 EXPERT SYSTEMS IN MILITARY SCIENCE:

ADEPT aids battlefield situation assessment analysts by providing tactical interpretations of intelligence sensor reports. The system uses these reports to generate a disply combat locations on the battlefield. Military knowledge and expertise are encoded as rules conserning how and why enemy forces operate and the tactical significance of situation interpretations. The system is able to explain the reasoning behind its battlefield assessments. The expert reasoning component of ADEPT is implemented in ROSIE and a Chromatics CGC 7900 color graphics system is used to disply maps and military symbology. The system was developed at TRW .

BATTLE provides weapon allocation recommendations to military commanders in combat situations. The system improves the performance of the U.S. Marine Corps Marine Integrated Fire and Air Support by providing timely recommendations for the allocation of a set of weapons to a set of targets. To address the critical time aspect of real battle situations, the system uses a best-first strategy during consultations, considering first those propositions (battle conditions) likely to have the most cost-effective influence on higher-level propositions. BATTLE was developed at the Naval Research Laboratory in Washington, D.C. and reached the stage of a demonstration prototype, some representative expert systems of military science are shown in figure 9.

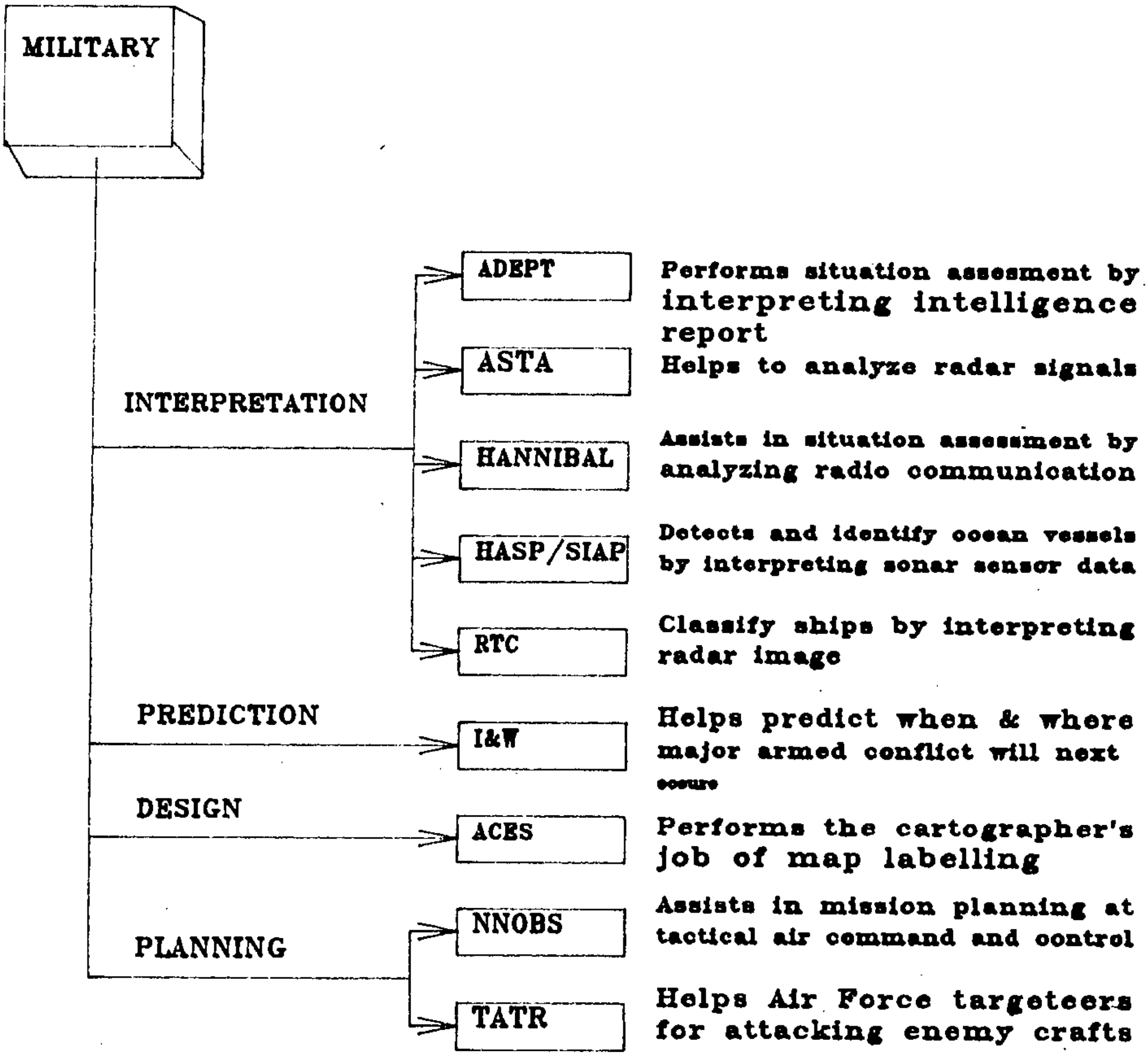

MILITARY
INTERPRETATION
ADEPT
Performs situation assesment by interpreting intelligence report
ASTA
Helps to analyze radar signals
HANNIBAL
Assists in situation assessment by analyzing radio communication
HASP/SIAP
Detects and identify ocean vessels by interpreting sonar sensor data
RTC
Classify ships by interpreting radar image
PREDICTION
I&W
Helps predict when & where major armed conflict will next occure
DESIGN
ACES
Performs the cartographer's job of map labelling
PLANNING
NNOBS
Assists in mission planning at tactical air command and control
TATR
Helps Air Force targeteers for attacking enemy crafts

FIG: 9 THE SELECTED EXPERT SYSTEMS IN MILITARY SCIENCE

2.4 ORGANIZATIONS ENGAGED IN EXPERT SYSTEM WORK:

Many of the advances in AI technology were spurred by research efforts at Universities, usually through doctoral dissertations. However some of the research organizations and companies has good contributions on various commercial aspects of expert system ranging from natural language understanding to knowledge engineering. Table 4 summarizes the list of some representative Universities, research organizations and companies and their present researches in these fields.[2]

TABLE: 4 EXPERT SYSTEM WORK AT SELECTED ORGANIZATIONS.

Name	Domains	Systems & Tools
1.Stanford University Palo Alto, CA.	Medicine, Chemistry, Computer system, Engineering, Electronics, Management science etc.	BLU BOX, PUFF, CRYSTALL, MRS, DENTRAL, SACON, SPEX.
2.Carnegie-Mellon Uni. Pittsburg, PA.	Manufacturing, Military, Process control etc.	TALIB, DAA, KBS PTRANS, XSEL.
3.MIT Cambridge, MA.	Medicine, Mathematics, Law,	NELT, MACSYMA, HODGKINS, PIP.
4.Univ. of Ilinois Urbana, IL.	Agriculture, Law, Medicine etc.	PLANT/S, BABY, ADVISE.
5.Xerox Palo Alto Research Center. (PARC) Palo Alto, CA	Data base management, Electronics.	PALLADIO, LOOPS, SMALLTALK-80.
6.The Rand Corps. Stanta MOnica, CA.	Military, Law, Crisis Managements.	ROSS, TATR, RITA, SPILLS.

TABLE: 4 (Continued)

Name	Domains	System & Tools
7.Ford Aerospac AI Labbotatiry, Houston.	Space program	RBMS, RICE, RPMS
8.Advance Informaion & Decision Systems(AI&DS) View, CA.	Military	AMUIDS, ASTA, RTC, RUBRIC. Mountain
9.Boeing Computer Serv Bellevue, WA.	Robotics,	ETS AI Center,
10.Artificial Intelligence Corporations. 100 Fifth Ave, Waltham, MA.	Natural language processing.	INTELLECT

2.5 HIGH PERFORMANCE EXPERT SYSTEMS IN MARKET PLACE:

Though there is a good number of expert systems in commertial environments, However some of them have achieved reputed fame in their fields and performing very useful work both in research & business vantage point. Table 5 summarizes a few of the more important expert systems in this category.

TABLE: 5 HIGH PERFORMANCE EXPERT SYSTEMS IN MARKET PLACE

Type	Name	Applications	Developer
	SYNCHEM2	Synthesizes complex organic molecules with out the help of a chemist.	State Univ.of New York at Stony Brook,
RESEARCH	DENDRAL	Identifies molecular struct. from mass spectral data.	Stanford Univ. Stanford, CA.
	MACSYMA	Solves algebraic simplific ations and integral problems.	MIT. Cambridge, MA.
	ACE	Provides trouble shooting reports and analysis for telephone cable maintenance.	AT & T Bell Laboratory, Whippany, NJ.
	DELTA	Helps diagnose and repair diesel electric locomotives	General Elec company. Schenectady NY.
	SPE	Diagnoses inflammatory conditions by interpreting scanning densitometer data.	Helena Lab. Beaumont, TX and Rutger University. New Brunswick NJ.
	XCON	Configures VAX-11/780 computer systems.	Digital Equip. Corporations. Hudson, MA & Carnegie-Mell Univ. Pitters.
	YES/MVS	Helps computer operator monitor the MVS O/S	IBM, Yorktown, Heights, NY.

CHAPTER–3

EXPERT SYSTEM DESIGN

3.1 CHARACTERISTICS OF AN EXPERT SYSTEM:

An expert system is defined as a computer program that has the following properties, as shown in figure 10, and each of these characteristics are discussed below:

a) EXPERTISE: An expert system must perform well, that is, achieve the same levels of performance in the domain of interest that human expert can achieve. But simply producing good solutions is not enough. Real expert not only produce good solutions but often find them quickly, while novices tend to take much longer to find the same solutions.

Thus an expert system must be skillful--apply its knowledge to produce solutions both efficiently and effectively, using the shortcuts or tricks that human expert use to eliminate wasteful or unnecessary calculations.

b) SYMBOLIC REASONING: To solve a problem, an expert system manipulates the rules and facts rather than performing standard mathematical computations. This is not to say that expert systems don't do math, rather the emphasis is on manipulating symbols. The consequence of this approach is that knowledge representation--the choice, form, and interpretation of the symbols used--becomes very important.

Also, expert system can take a problem stated in some

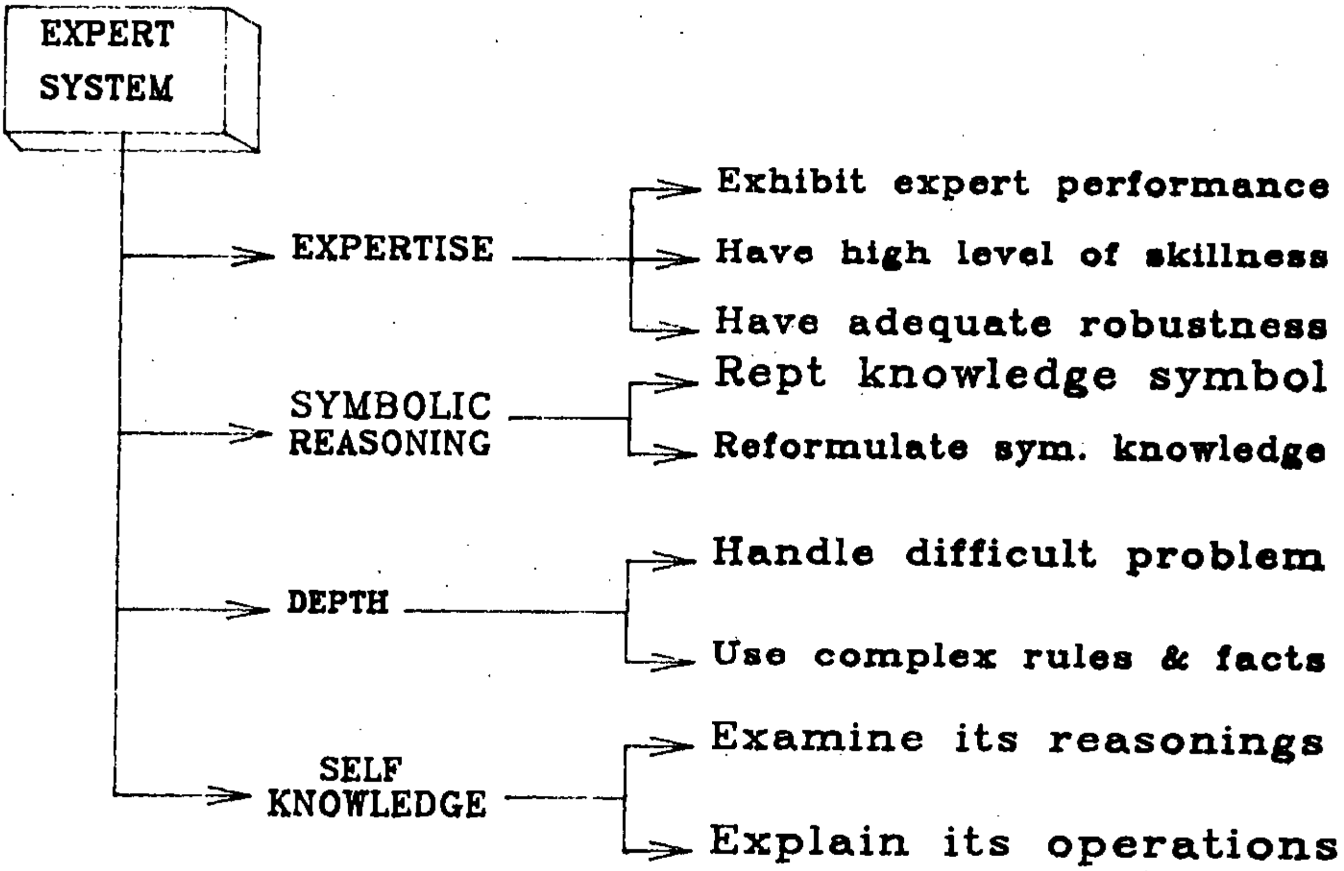


FIG: 10 THE CHARACTERISTICS OF AN EXPERT SYSTEM arbitrary manner and convert it to a form that lends itself to a fast or efficient solution. This problem reformulation capability is something expert system need to make their skill level closer to that of human experts.

c) **DEPTH:** An expert system has depth, that is it operates effectively in a narrow domain containing difficult, challenging problems. Thus the rules in an expert system are necessarily complicated, either through their individual complexity or their sheer number.

Expert system typically work in real-world problem domains, the problem solver applies actual data to a practical problem and produces solutions that are useful in some cost-effective way.

d) **SELF KNOWLEDGE:** An expert system has knowledge that lets it reason about its own operation plus a structure that simplifies this reasoning process. For example, if an expert system is organized as sets of rules, than it can easily look at the inference chains it produces to reach a conclusion. This knowledge the system has about how it reasons is called metaknowledge, that is--knowledge about knowledge. Most current expert system have what is called an explanation facility. This is knowledge for explaining how the system arrive at this answers. Most of these explanations involve displaying the inference chains and explaining the relations behind each rule used in the chain. The ability to examine their reasoning process and explain their operation is one one of the most innovative and important qualities of expert system.[2]

Self knowledge is important in an expert system because:

* User tend to have more faith in the results, more confidence on the system.
* System development is faster since the system is easier to debug.
* The assumptions underlying the system's operation are made explicit rather than being implicit.
* It's easier to predict and test the effect of a change on the system operation. [4]

3.2 FACTORS TO BE CONSIDERED IN DESIGNING EXPERT SYSTEM:

The key factors to be considered to develop an expert system are the nature, complexity and scope of the problem to be solved. figure 11 illustrate these factors and their relationship to characteristics that make a problem appropriate for expert system work and are discussed below:

a) NATURE: One of the important factor for expert system work is that the problem must have certain intrinsic qualities. It must be a problem that can be solved quite naturally by manipulating symbols and symbol structures. The ability to perform symbolic reasoning is one thing that sets expert systems apart from conventional programs.

Most real world problems require symbolic reasoning, exceptions are those that have tractable mathematical solutions. Thus mathematically- oriented problem, such as solving

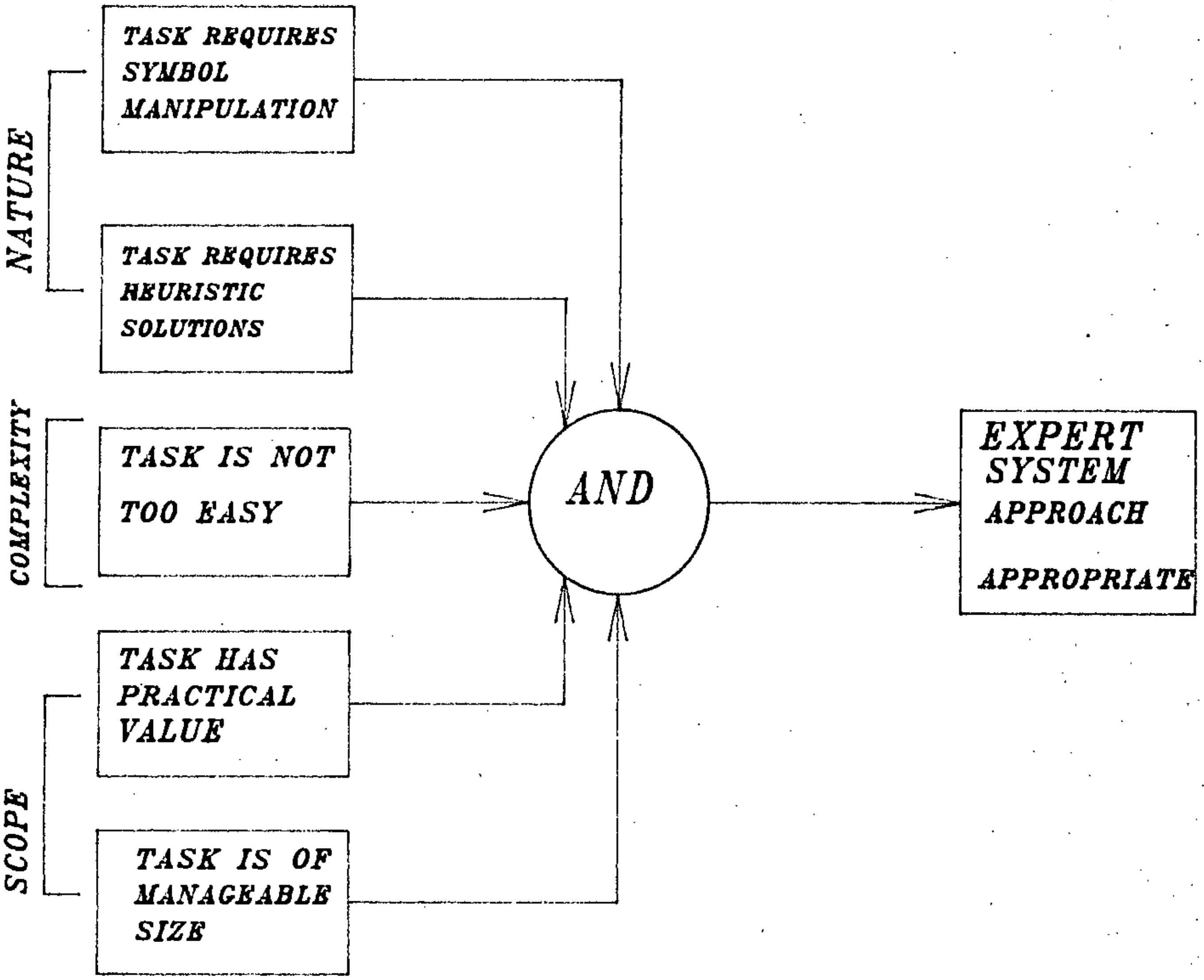

NATURE
TASK REQUIRES SYMBOL MANIPULATION
TASK REQUIRES HEURISTIC SOLUTIONS
COMPLEXITY
TASK IS NOT TOO EASY
SCOPE
TASK HAS PRACTICAL VALUE
TASK IS OF MANAGEABLE SIZE
AND
EXPERT SYSTEM APPROACH APPROPRIATE

FIG: 11 THE FACTORS TO BE CONSIDER IN DESIGNING AN EXPERT SYSTEM

differential equations with numerical analysis techniques, are usually not appropriate for expert system development.

Most problems that are appropriate for expert system work are heuristic in nature, that is, they require the use of rules of thumb to achieve acceptable solutions.

b) COMPLEXITY: The second factor that is to be considered in designing the expert system is that the research work must not be too easy. It should be a formidable, serious problem in a domain in which it takes a human years of study or practice to achieve the status of an expert.

c) SCOPE: Finally the problem should have the proper scope. It should be sufficiently narrow to make the problem manageable and sufficiently broad to ensure that the problem has some practical interest.

3.3 STRUCTURE OF AN EXPERT SYSTEM:

It is convenient to divide the development of an expert system into three main modules as illustrated in figure 12.

a) A Knowledge base,

b) An inference engine,

c) A user interface.

a) KNOWLEDGE BASE: The knowledge base is the data or knowledge

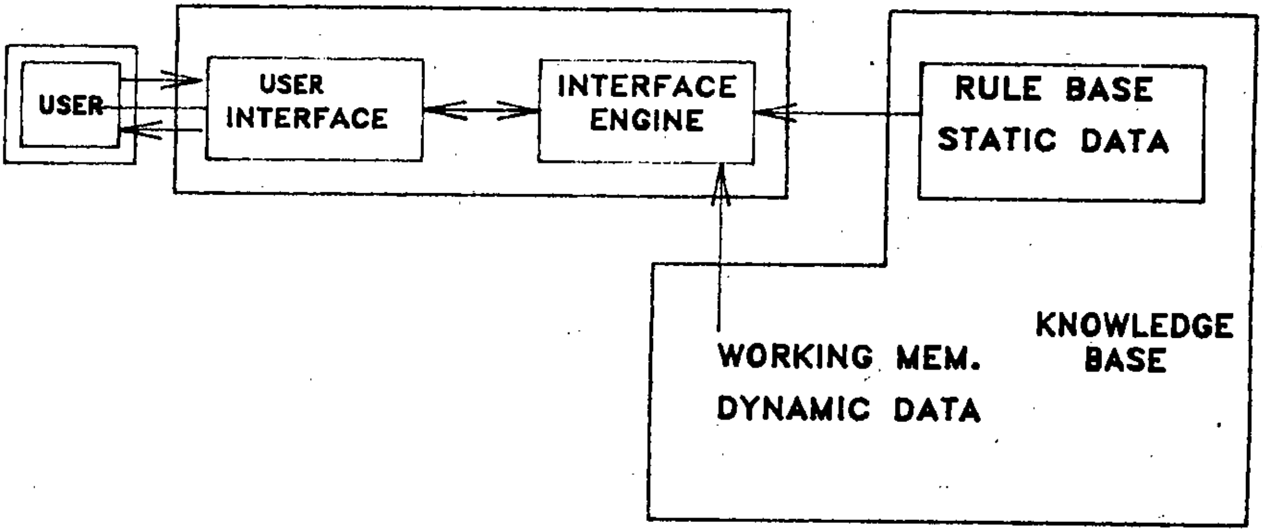


FIG:12 THE STRUCTURE OF AN EXPERT SYSTEM used to make decisions. The knowledge base contains rules and facts about the domains and also methods, heuristics and ideas for solving problem in this domain.

The knowledge base consists of two parts, the working memory and the rule base. The rule base consists of facts and rules that are complied as the part of the problem, as it corresponds to prolog static data base.

The second part of the knowledge base is working memory. It consists of the facts relative to a particular consultation program which corresponds to prolog dynamic data base. At the beginning of a consultation the working memory or dynamic data base is empty. As the consultation progresses it add facts to the working memory.

b) THE INFERENCE ENGINE: The inference engine contains an interpreter that decides how to apply the rules to infer new knowledge and a scheduler that decides the order in which the rules should be applied.

The inference engine has two functions inference and control. Inference is the basic formal reasoning process. It involves matching and unification. It uses facts that are known to derive new facts by means of rules.

The control function determines the order in which the rules are tested and what happens when a rule is succeded or failed. The inference engine takes those facts as true which are known from the rule base (static data base) and working memory (dynamic data base).

The inference engine then test the rules in the data base through the process of unification. When a rule succeds, the rule is said to fire and the conclusion of the rule is displayed or added in working memory.

c) A USER INTERFACE: The user interface causes for smooth communication between the user and the system, also providing the user with an insight into the problem solving process carried out by the inference engine.

It is convenient to view the inference engine and the user interface as one module, usually called an expert system shell or a shell for brevity.

3.4 KNOWLEDGE ENGINEERING:

Knowledge Engineering is the process of extracting knowledge about a particular problem area from an expert (or a group of experts) and putting it into an objective form for the data base.

The heart of an expert system is the powerful corpus of knowledge that accumulate during system building. The accumulation and codification of knowledge is one of the most important aspect of expert system.[12]

The two stages of knowledge engineering are

Knowledge Acquisition (or elicitation) : The process of acquiring knowledge from the expert as shown in figure 13.

Knowledge Representation : The process of formulating the knowledge in a form suitable for expression in an AI language as LISP or PROLOG.

3.4.1 BASE OF KNOWLEDGE:

There are three different bases of knowledge: Scientific laws, experience, and models. Acquiring knowledge in a scientific subject is much easier than any other field and it matter little whether we get these laws from books or from expert.

The majority of expert system applications are on accumulated wealth of experience, not scientific formulation. The greatest difficulty when collecting data in these field is that the expert may not be able to verbalize the knowledge that he uses.[1]

In the scientific fields, Knowledge acquisition is relatively easy because of the existence of scientific laws. But unfortunately vey few knowledge engineer intetested to extract knowledge . from scientific fields.

The important sources of knowledges are as follows

a) Literature e,g Text books reports databases etc.

b) Expert e,g from personal experiences,

c) Examples and case studies and

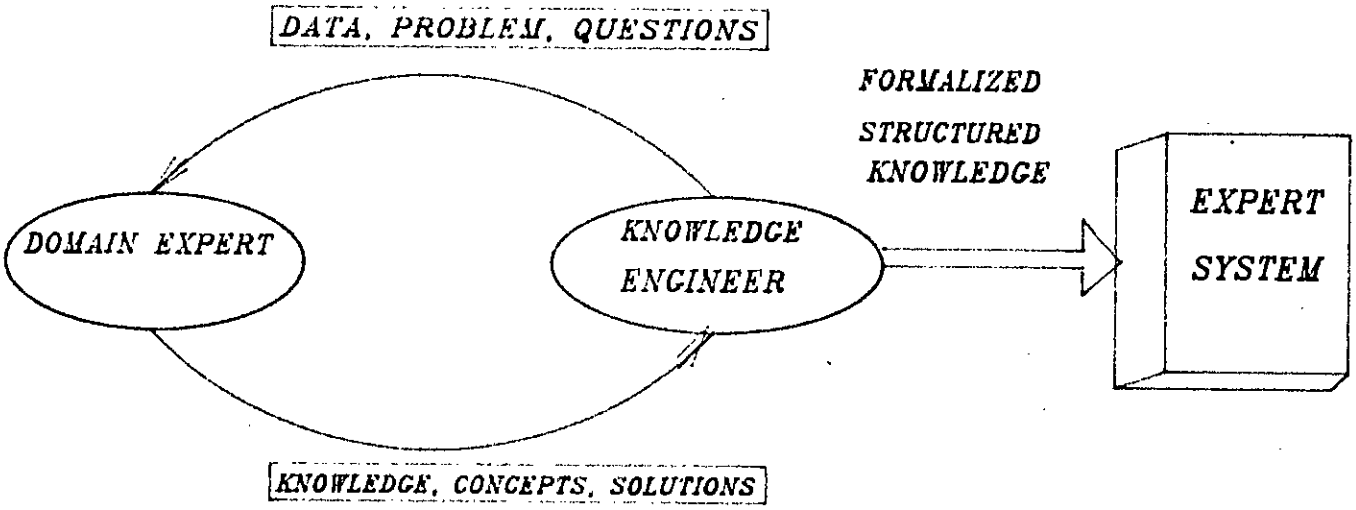


FIG: 13 THE KNOWLEDGE ACQUISITION TECHNIQUE

d) Some knowledge engineer prefer to become a "fly on the wall" and use a video camera to record expert doing his every day work over a suitable period of time. The resultant video tapes are then examined at a latter date and the relevant knowledge and expertise extracted. Figure 14 shows the sources of knowledges.

3.5 KNOWLEDGE REPRESENTATION:

A representation has been defined as " a set of syntactic and semantic conventions that make it possible to describe things"-- (WINSTON-1984). In the field of expert systems, there is no perfect methods of knowledge representation, however now-a-days some systematic ways has been developed of codifying what an expert knows about some domain. The main criterion used for repersentating knowledge are logical adequacy, heuristic power and notational convenience.

Logical adequacy means that the formalism should be capable of expressing the knowledge that one wishes to represent. Heuristic power means that as well as having an expressive language will posses a well defined syntax and semantics. There must be some means for using representations so constructed and interpreted to solve problems. Most expert system applications require the encoding of substantial amount of knowledge and this task requires that the conventions of the representation language should be as uncomplicated as possible. [8]

In view of above discussion the two fundamental methods are

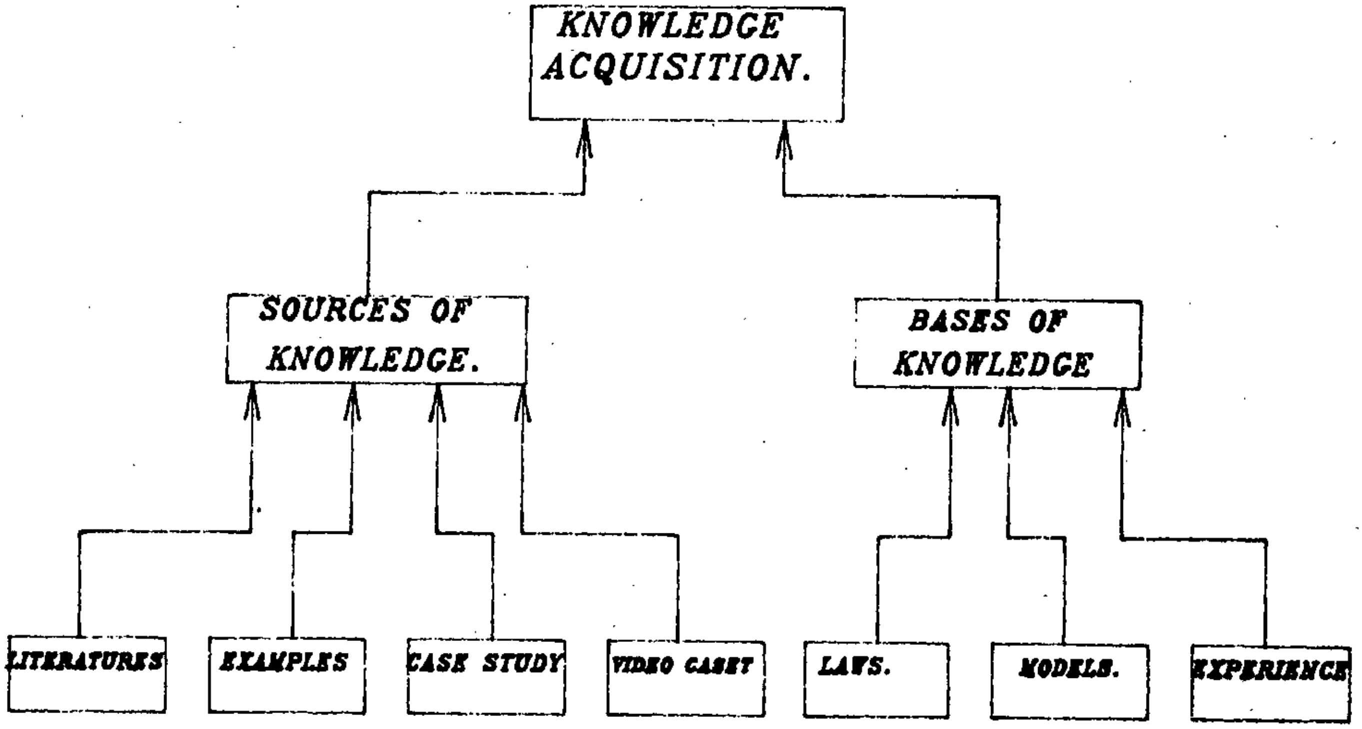


FIG: 14 THE SOURCES OF KNOWLEDGES

widely accepted by the knowledge engineer.

PROCEDURAL: Representation as program, most of the early representation schemes were predominantly procedural and have the outstanding advantageous of being highly efficient.

DECLARATIVE: Representation as data and is therefore, less heavily encoded and more understable, as illustrated in figure 15

3.5.1 KNOWLEDGE REPRESENTATION USING RULES:

In expert systems jargon the term rule has a much narrower meaning than it does in ordinary language. It refers to the most popular type of knowledge representation technique, the rule-based representation.

Rules provide a formal way of representing recommendations, directives, or strategies; they are often appropriate when the domain knowledge results from empirical associations developed through years of experience solving problems in an area. Rules are expressed as IF-THEN statements, as shown below.

i) If demand > generation,

Then load shad occur.

ii) If faults occur at station

Then fault = gen.

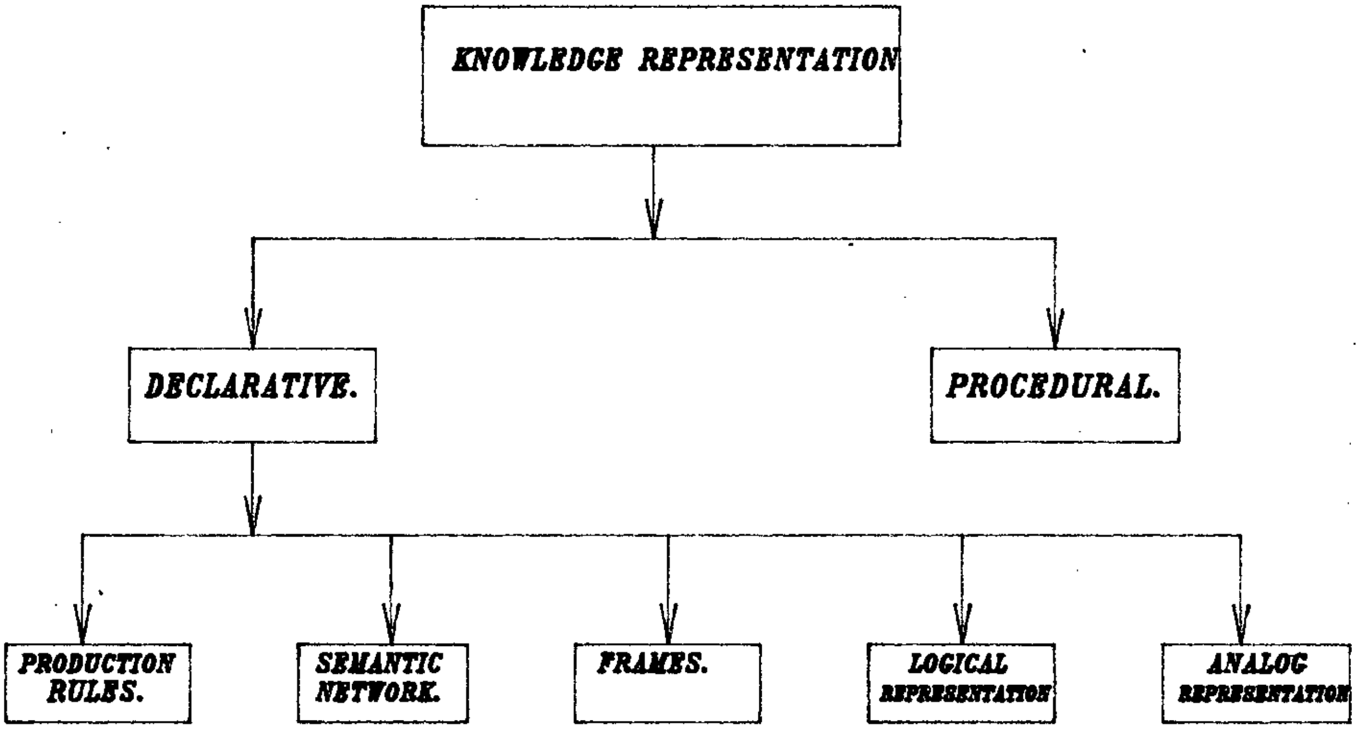


FIG:15 KINDS OF KNOWLEDGE REPRESENTATION

In a rule-based expert system, the domain knowledge is represented as sets of rules that are checked against a collection of facts or knowledge about the current situation. When the IF portion of a rule is satisfied by the facts the action specified by the THEN portion is performed. When this happens the rule is said to fire or execute.

3.5.2 KNOWLEDGE REPRESENTATION USING SEMANTIC NETS:

The term semantic net is used to describe a knowledge representation method based on network structure. Semantic nets were originally developed for use as psychological models of human memory but are now a standard representation method for AI and expert system.

A semantic net consists of points called nodes connected by links called ARCS, describing the relation between the nodes. The nodes in a semantic net stand for objects, concepts or events. Arcs can be defined in a variety of ways, depending on the kind of knowledge being represented. Common arcs used for representing hierarchics includes isa and haspart as shown figure 16.

The semantic net representation is useful because it provides a standard way analyzing the meaning of sentences. Also it points out the similarities in the meanings of sentence that are closely related but have different structure.

3.5.3 KNOWLEDGE REPRESENTATION USING FRAMES:

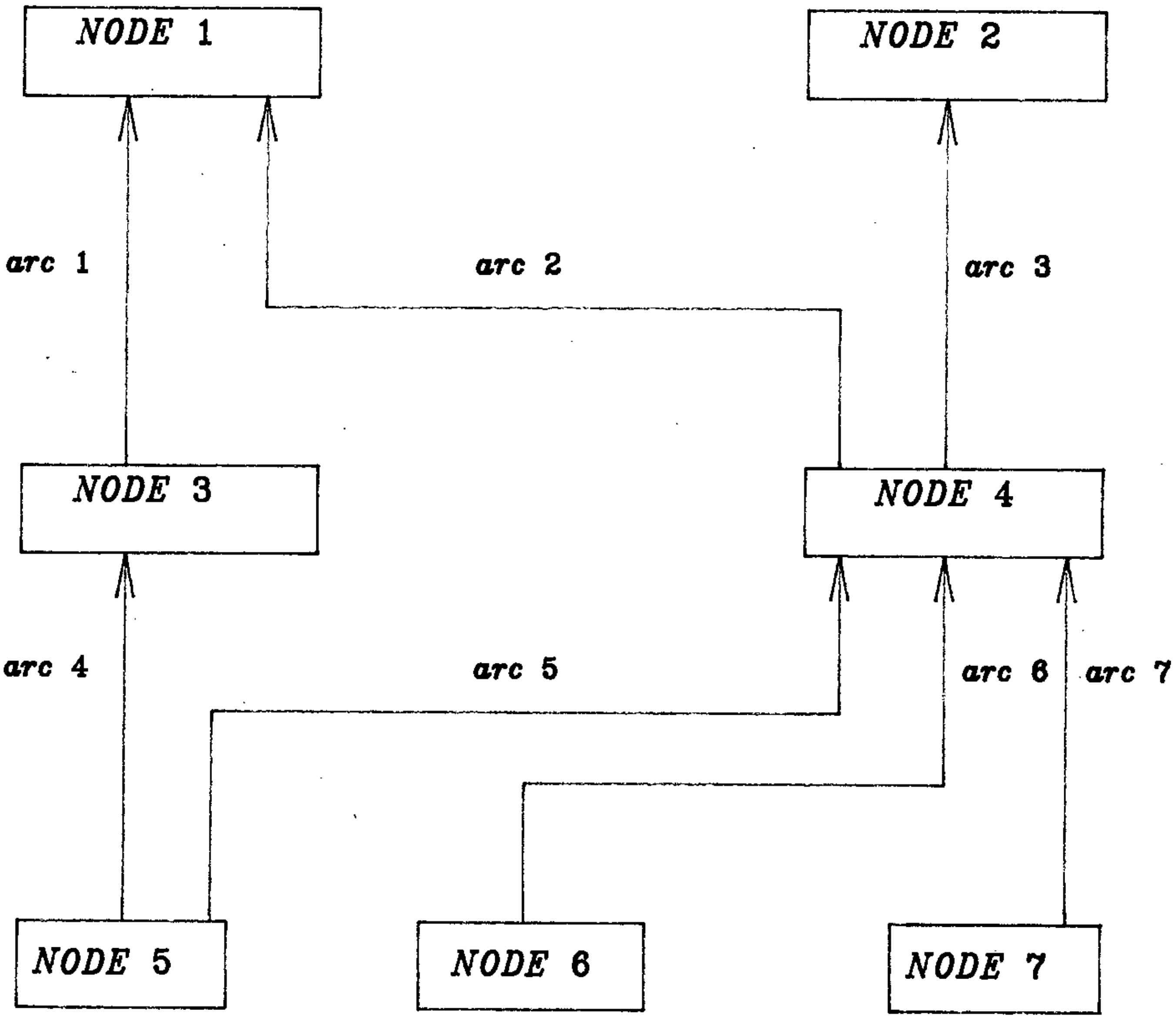


FIG: 16 THE KNOWLEDGE REPRESENTATION USING SEMENTIC NETS

In the field of Artificial intelligence, the term frame refers to a special way of representing common concepts and situations. A frame is organized much like a semantic net. A frame is a network of nodes and relations organized in a hierarchy. Where the topmost nodes represent general concepts and the lower nodes represents more specific instances of those concepts..

So far this looks just like a semantic net. But in a frame system the concepts at each node is defined by a collection of attributes (e,g name , color, size)and the values of those attributes (Smith, red, small) are called slots. Each slots can have procedures attached to it which are executed when the information in the slot is changed.

Frame system is useful for problem domains where exceptions about the form and content of the data play an important role in problem solving, such as interpreting visual scenes or understanding speech.

CHAPTER-4

BUILDING AN EXPERT SYSTEM FOR L.D.C.

4.1 GENERAL DISCUSSION:

Before 1982, the power distribution system of PDB was independent and uneven, i,e, the power stations of eastern region fulfill the demand of eastern part of the country and the western regional demand was fulfilled by the stations of western part.

To make the unit power production cost low, and to ensure the power supply even and reliable through out the whole country, the idea of modern common grid system was suggested and the load dispatch center (LDC) was founded with this responsibility.

Though a numerous advantages were achieved by the proposed system, but this improved interconnected grid system has become more and more complicated for fault analysis. In an electrical distribution system faults may occur at any moments but due to interconnectness of the national grid, a minor fault may cause a great problem and even may cause total grid failure. So the analysis of the impacts of the faults are now very complicated issue. The different types of faults which are very common in LDC are shown in figure 17.

An annual progress presentation of LDC reported that on 5th july of 1986, due to short circuit faults of ASHUGANJ sub-station, tripped the line (1356) between ASHUGANJ to GHORASHAL supply, which by turn tripped the supply between GHORASHAL to SIKALBAHA and ultimately national grid supply was failed for 32 hours, but the origin of the fault was very minor.

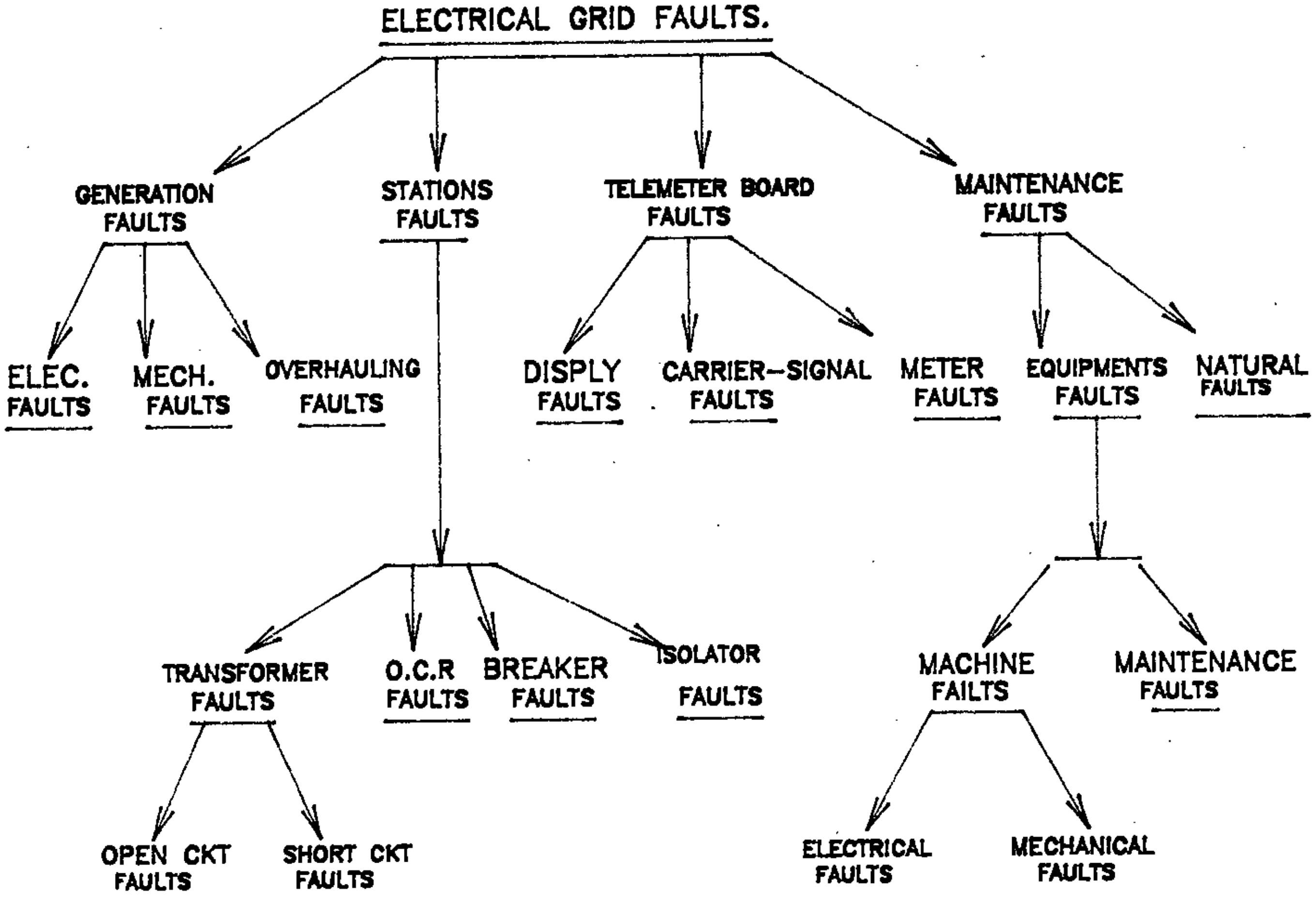


FIG: 17 WORKING THROUGH THE FAULTS SPACE

Obviously this is a growing problem of LDC to ensure the most economical and reliable supply in context to the rapidly increasing electrical demand with the limited generation units, having a complicated analysis of faults considering its effect through out the national wide supply. A real Expert system incorporating the knowledges of expert can overcome this difficulties.

4.2 POSSIBILITIES TO DEVELOP AN EXPERT SYSTEM FOR LDC:

figure 18 summarizes the problem domain characteristics required to make expert system development possible and are described below:-

a) GENUINE EXPERT EXISTS: One of the most important requirement to develop an expert system is that genuine expert exists in that field. There are people generally acknowledged to have an extremely high level of expertise in the problem area, they are significantly better than novices at solving problems in that domain.

Without a source of extensive powerful knowledge to draw on, the development effort will fail to produce a truly skillful program. At LDC the genuine expert exists who is concerned with the system for the long thirty five years.

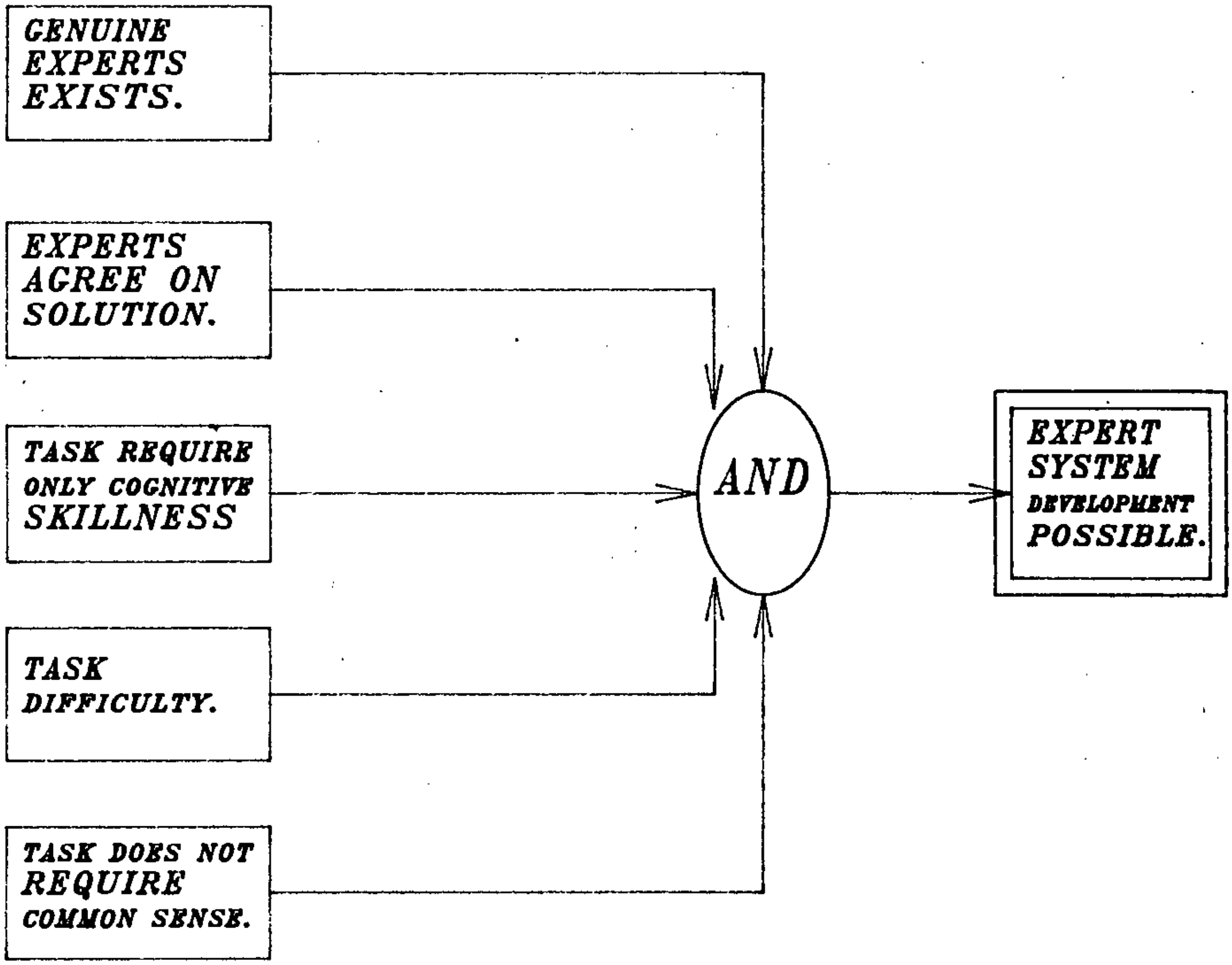

GENUINE EXPERTS EXISTS.
EXPERTS AGREE ON SOLUTION.
TASK REQUIRE ONLY COGNITIVE SKILLNESS
TASK DIFFICULTY.
TASK DOES NOT REQUIRE COMMON SENSE.
AND
EXPERT SYSTEM DEVELOPMENT POSSIBLE.

FIG: 18 THE REQUIREMENTS FOR DEVELOPMENT AN EXPERT SYSTEM AT LDC

b) EXPERT AGREE ON SOLUTIONS: The experts in the problem domain generally agree about the choice and accuracy of solutions, otherwise the validating the expert system's performance would be a near impossible task. In the electrical disturbance environment of national grid, after considering some factors there is a unique decision whether load shading will occur or not.

c) TASK REQUIRES ONLY COGNITIVE SKILLS:

The other requirements for expert system development deal with the characteristics of the problem that the expert system will solve - the task it will perform. The task must require cognitive not physical skills.

If it consists of physical manipulations that only be learned through practice, the expert system will not work. In the national electrical grid system the decisions to overcome faults is based on thinking and knowledge. There is no scope of physical labour, rather than the role of past experiences and knowledges.

d) TASK DIFFICULTY: The task difficulty also relates in certain ways to how will the experts understand the problem domain - that is the degree to which problem solving knowledge is precise and well structured.

If the task is so new or so poorly understood that it requires basic research to find solutions the knowledge engineering will not work. It will also not work if the task require a significant amount of common sense.

In the national electrical grid the faults are well defined and easy to understand the problem domain. Its solutions are devoid of common sense reasoning.

4.3 JUSTIFICATIONS TO DEVELOP AN EXPERT SYSTEM FOR LDC

Just because it's possibility to develop an expert system for a perticular task does not mean that it is desirable to do so. There are many ways to justify an expert system development effort as illustrated in the figure 19.

a) SCARCITY OF HUMAN EXPERTISE: Expert system development is justified when human experts are unavailable or unable to do the job. Sometimes human experts are scarce, very much in demand-and thus expensive. Also the problem is compounded, i,e when a company needs simillliar expert at many different physical locations.

The load despatch center also suffer with scarcity of expertise as because there is only one expert for the domain, Moreover the load despatch center remain in operation for twenty four hours. If the faults occur at mid_night, then it become very difficult to make contact with the system expert & when he goes out of the station the problem become more acute, as because at any moment faults can occur.

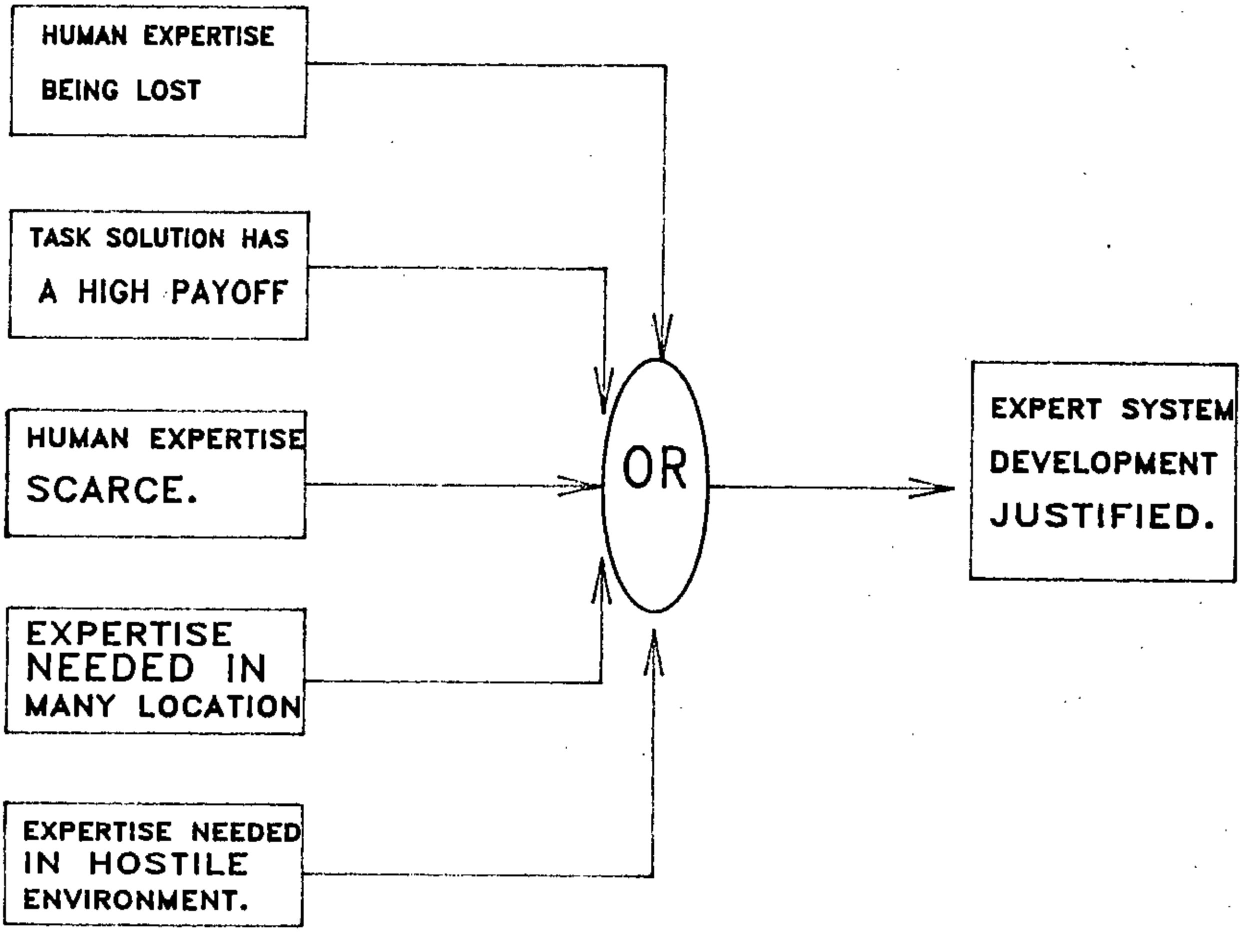

HUMAN EXPERTISE BEING LOST
TASK SOLUTION HAS A HIGH PAYOFF
HUMAN EXPERTISE SCARCE.
EXPERTISE NEEDED IN MANY LOCATION
EXPERTISE NEEDED IN HOSTILE ENVIRONMENT.
OR
EXPERT SYSTEM DEVELOPMENT JUSTIFIED.

FIG: 19 THE JUSTIFICATIONS FOR DEVELOPMENT OF AN EXPERT SYSTEM AT LDC

b) HUMAN EXPERTISE BEING LOST: Expert systems are justified when significant expertise is being lost to an organization through personal changes like retirement, transfer etc. As these are common features to LDC so a real expert system can minimize or eliminate this problem.

c) COMPLEXITY OF TASK: An enterprise can justify expert system development when the task solution not too easy. It should be a formidable, serious problem in a domain in which it takes a human years of study or practice to achieve the status of an expert.

The behavior of interconnected electrical grid system has become more and more complicated, causing decision making process to be increasely difficult. Moreover without the details knowledge of grid system, and past experiences one can not suggest the appropriate solution at the time of faults.

d) DECISION MAKING SLOW: Another important factor to build an expert system when the decision or prediction process is very slow. At L D C if any faults arises in telemeter board the system operators then contact over telephone with the domain expert and then the expert come or give decisions over telephone, which is carried out by the operators. This is quite lengthy process for electrical faults. An expert system in front of operators can eliminate this delay.

e) **EXPERT NEEDED IN UNFRIENDLY ENVIRONMENT:** Expert system development is justified when the expert decision making must take place in an unfriendly environment such as nuclear power plant, space station etc.

At the time of major faults the senior government officials make now and then telephones, sometimes rushes at L D C and order to ensure power within limited houres. Such situations causes a hostile environment and system expert is quite unable to take unbiased decision.

4.4 THE KNOWLEDGE ACQUISITION PROCESS AT LDC:

a) **KNOWLEDGE EXTRACTING SOURCES AT LDC:** Extracting the knowledge by externalizing it and thus making it available for inspection and manipulation is an important task for building the knowledge base. In order to do so the knowledge engineer first identify the major sources of knowledge for the proposed system. Some of the important sources of knowledge for LDC are illustrated in figure 20.

a) Knowledge for the expert system of L D C mainly collected through direct interaction with the system experts -- Chief engineer, transmission zone and also from the deputy director, operation.

b) The second source of knowledge for L D C was case studies, During the course of this work, different faults

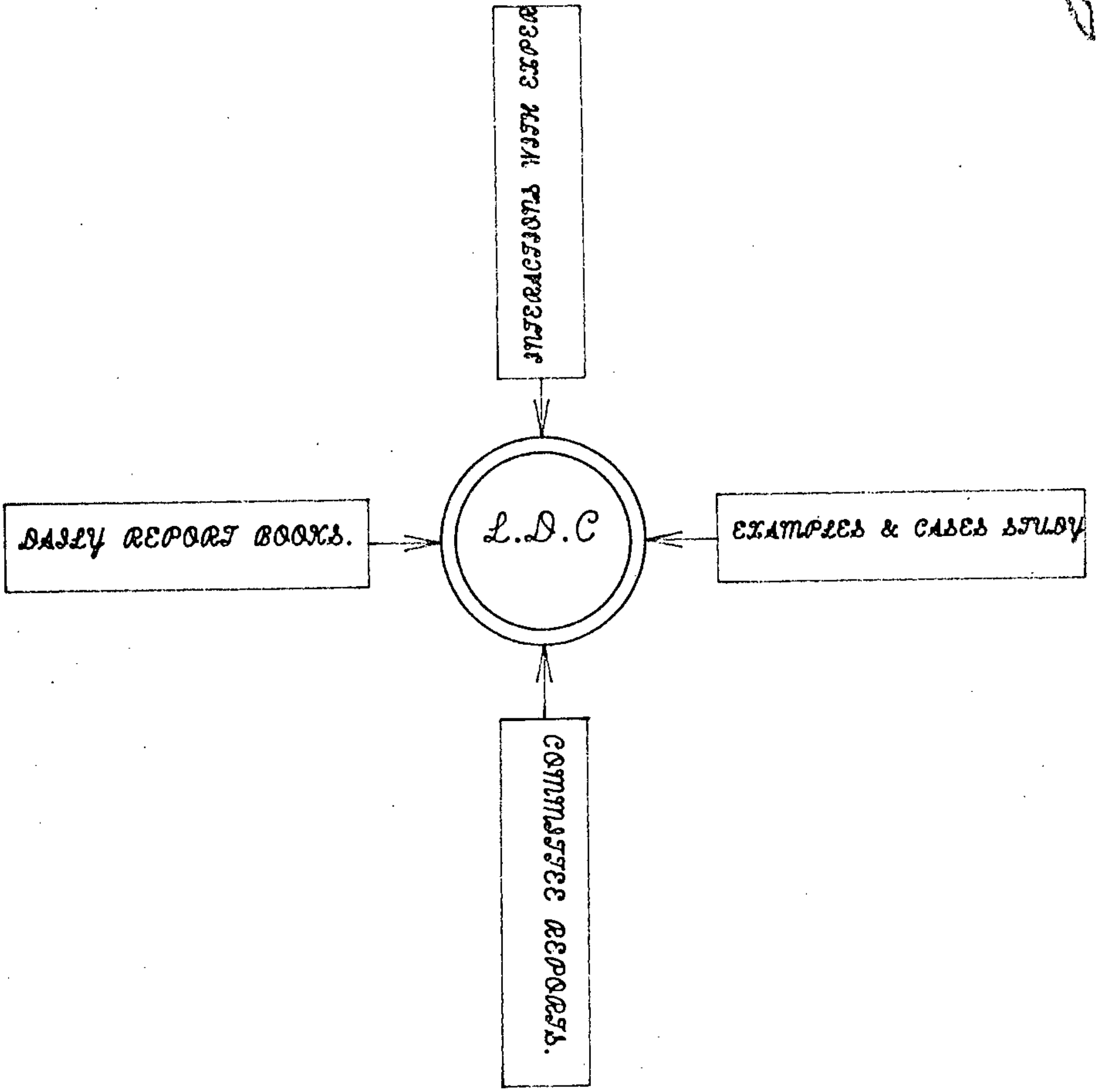


FIG: 20 THE SOURCES OF KNOWLEDGE FOR LDC occurred in the center, what necessary actions the system expert taken to make the grid supply normal were important aspects of knowledge.

c) The third source of knowledge for the said expert system was the daily roster book. In that book the shift engineers write down all the phenomenon that happen during his working hour.

If any disturbance occur in the National Electrical Grid, they note it in the daily roster book along with some important suggestions of the superior experts. These roster books were major sources of knowledge for the expert system.

d) Another important source of knowledge for the topic was the report studies. At the time of major electrical faults causing great failure of National grid, the Govt. has set up a committee to investigate the faults. The reports of the committees are preserved in L D C library. By studying these reports some knowledges were accumulated.

b) INFORMATIONS OF GRID SYSTEM: The national electrical grid is divided into two part one is the eastern grid and another is the western grid. Under eastern grid there are thirteen (13) power stations having thirty (30) generation units of which Ashuganj, Ghorasal and Kaptai are treated as base load stations.

In the western part there are seventeen power stations consisting twenty one (21) generators, most of these units are old and with less production capacity. The average per unit power production cost of eastern grid is about .25 TK. whereas for western grid it is 4.30 TK.

The western part of the grid system have thirty (30) important sub-stations and the western region have twenty six (26) sub-stations located at the different places of the country. The power line between Ashuganj to Tongi, Tongi to Ishurdi, Ashuganj to Ghorasal and from Siddergan j to Comilla sub-stations has 230 KV HTL (High tension line). The rest of the power lines of the grid system are 132 KV only having 66 KV power lines between Ishurdi to Pabna and Rajshahi sub-stations.

Asian Development Bank (ADB) has an active development program to convert all the 132 KV lines into 230 KV lines and also the single line power distribution system (e.g. Mymenshing to Kishorganj & Jamalpure, Sidderganj to Comilla etc) to convert them into double line system.

The eastern-western grid connector (Tongi to Ishurdi having line no.1298 & 1299) serves an important role for the interconnected grid system, on an average 250 MW to 300 MW power perday transfer from eastern region (low cost) to western region in order to make the national grid supply economical and reliable.

A single line diagram of national electrical grid is enclosed in appendix E. Some important system informations are summarized in table 6. (Demand & Interruption report of 14-6-1989)

TABLE: 6 NATIONAL ELECTRICAL GRID SYSTEM'S INFORMATIONS

Items	Unit production.
1. Highest system generation (for day & night)	1218.70 MW
2. Lowest system production	604.50 MW
3. Load factor	67.246 %
4. Total demand in Eastern grid	770.10 MW
5. Total demand in Western grid	297.10 MW
6. Total demand in greater Dhaka	426.90 MW
7. Total demand in greater Chittagoig	180.00 MW
8. Total power imported from Eastern grid to Western grid.	225.57 MW

c) KNOWLEDGE EXTRACTING TECHNIQUES AT LDC: "EXPERT" it appears, have a tendency to state their conclusions and the reasoning behind them in general term that are too broad for effective machine analysis. It is advantages to have the machine work at a more basic level, dealing with clearly defined pieces of basic information that it can build into more complex judgments. In contrast, the expert seldom operates at a basic level. He makes complex judgment rapidly, without laboriously re-examining and restating each step in his reasoning process. The pieces of basic knowledge are assumed and are combined so quickly that it is difficult for him to describe the process. When he examines a problem, he can not easily articulate each step and may even be unaware of the individual steps taken to reach a solutions. The more competent domain expert become the less able they are to describe the knowledge they use to solve the problem. [4]

This aspects of Expert's nature is also natural for LDC, in fact it is called the paradox of expertise. In order to avoid this problem the knowledge acquisition techniques for LDC are divided into smaller steps for the close intimation of the system expert which are summarized as follows and also shown in figure 21.

1) PROBLEM DISCUSSION: The first approach to the domain expert expert was to acquire informations about the distribution system of national electrical grid lines, the organizations of load dispatch center and also different types of faults, so that overall idea about grid system can be procured. The analysis of the system expert regarding power station_faults, generation_faults, maintenance_faults were very much helpfull. And also the present status of the generators & its unit fuel cost were the valuable tools for co-ordenating the knowledge.

In the next session of the work a set of representative problems were informally discussed with the expert, keeping interest was to determine how the expert organized his knowledge and came to a conclusions and the reasonings behind each conclusions.

An electrical demand report of different sub-stations [Appen-B], a report on total generations both eastern and western grids are enclosed [Appen-C] for necessary informations in knowledge

TABLE: 7 PER UNIT FUEL COST OF EASTERN GRID POWER STATIONS OF P.D.B

POWER STATIONS	UNIT NO	FUEL	COST (P.U) IN TK.
KAPTAI HYDRO	ALL	—	00.00
GHORASAL STEAM	3 & 4	GAS	00.30
ASHUGANJ STEAM	3,4 & 5	GAS	00.30
ASHUGANJ STEAM	1 & 2	GAS	00.31
GHORASAL STEAM	1 & 2	GAS	00.34
CHITTAGONG STEAM	60 MW	GAS	00.34
CHITTAGONG BARGE	1 & 2	GAS	00.40
HORIPUR GAS	1,2 & 3	GAS	00.40
SYLHET GAS	20 MW	GAS	00.43
ASHUGANJ	GT-1	GAS	00.46
SIDDHIRGANJ STEAM	30 MW	GAS	00.46
SHAJIBAZAR GAS	ALL	GAS	00.54
CHITTAGONG GAS	TG-7	GAS	00.71

** On courtesy of DIRECTOR L.D.C, June,1988.

TABLE: 8 PER UNIT FUEL COST OF WESTERN GRID POWER STATIONS OF P D B.

POWER STATIONS	UNIT NO.	FUEL	COST(P.U) IN TK
KHULNA STEAM	110 MW	FO	01.73
SAIDPUR DIESEL	--	FO/LDO	01.78
KHULNA STEAM	60 MW	FO	01.9I
THAKURGAON DIESEL	--	LDO	02.00
KHULNA DIESEL	--	HSD	02.15
SERAJGONJ DIESEL	--	HSD	02.15
BOGRA DIESEL	--	HSD	02.21
RAJSHAHI DIESEL	--	HSD	02.25
BARISEL DIESEL	--	HSD	02.32
KHULNA BARGE	1&2	SKO	02.32
BARISAL GAS	1&2	HSD	02.86
BHERAMARA GAS	1,2&3	HSD	02.93
SAIDPUR GAS	--	HSD	02.94
RANGPUR GAS	--	HSD	02.95
KHULNA GT-35	--	SKO	03.10
KHULNA TC-14	--	SKO	04.17
BOGRA GAS	--	HSD	05.17

On courtesy of DIRECTOR LDC, June,1988.

procurering techniques. Also the per unit fuel cost of different generators are summarized in table 7 & table 8.

2) PROBLEM ANALYSIS WITH THE EXPERT: During the second phase of this work a vivid analysis was made with the domain expert for a series of realistic problems collected from case studies, roster books and reports study etc. Special concentrations were given on-

* What kind of data does the problem require.

* What factors are to be consider to overcome the faults.

* What kinds of solutions are adequate for the problems.

The system expert gradually described all the solution process and giving as many intermediate steps as possible for each faults. The interviewing with the system expert was to determine rational behind each steps of faults. Including hypothesis being entertained, strategies being used to generate hypothesis, how a particular conclusion was reached, what were the basis of this conclusion and so on until the expert runed out of explanations.

These explanations of the system expert were valuable clusters for formulating the knowledge with IF THEN structure. A power interruption configuration on 20th May,1986 is enclosed [Appen-D] from the report books which caused the national grid failure for 48 hours.

3)SYSTEM EXAMINATION: In the next stage of knowledge acquisition process all the steps behind each faults were

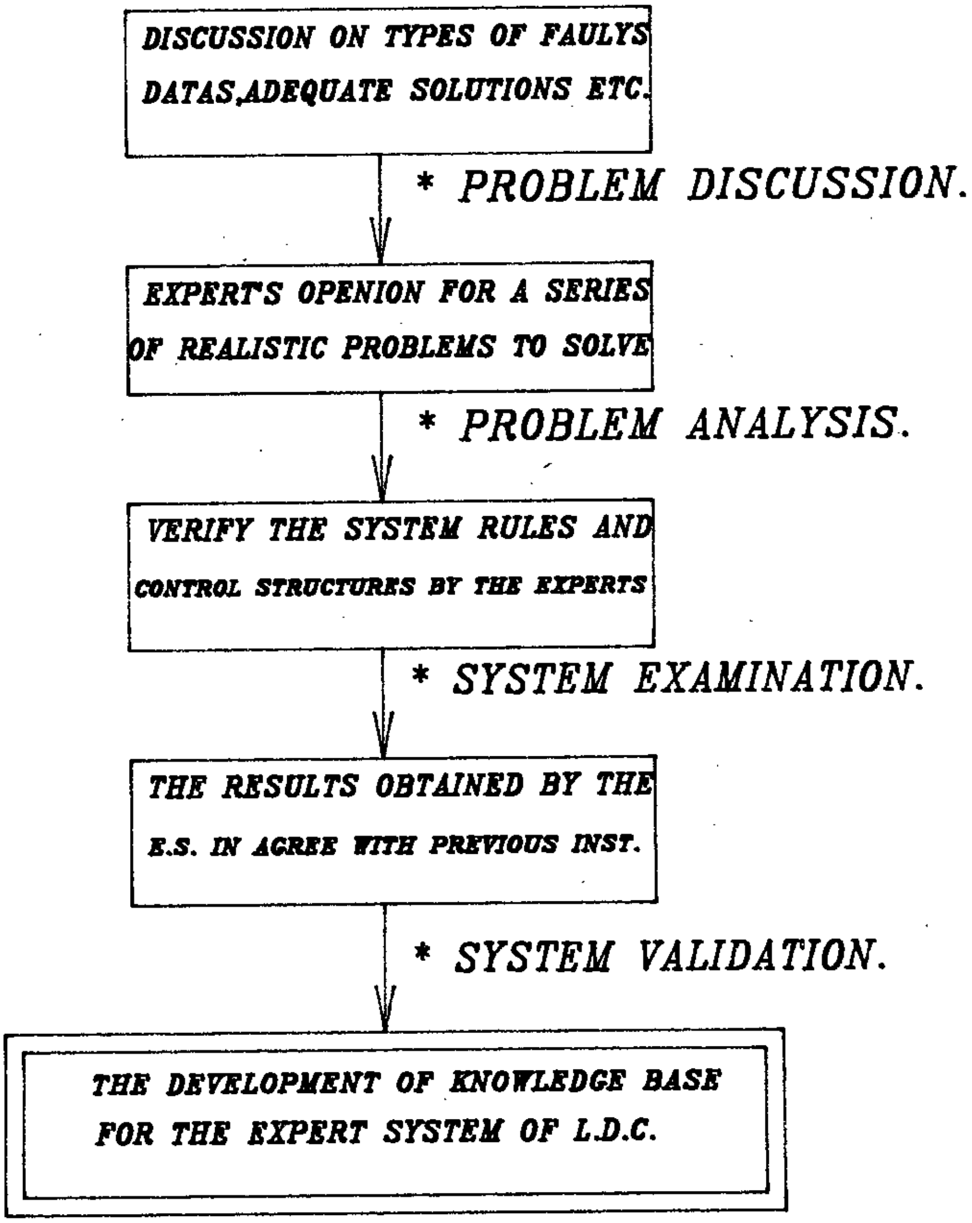

DISCUSSION ON TYPES OF FAULYS DATAS,ADEQUATE SOLUTIONS ETC.
* PROBLEM DISCUSSION.
EXPERT'S OPENION FOR A SERIES OF REALISTIC PROBLEMS TO SOLVE
* PROBLEM ANALYSIS.
VERIFY THE SYSTEM RULES AND CONTROL STRUCTURES BY THE EXPERTS
* SYSTEM EXAMINATION.
THE RESULTS OBTAINED BY THE E.S. IN AGREE WITH PREVIOUS INST.
* SYSTEM VALIDATION.
THE DEVELOPMENT OF KNOWLEDGE BASE FOR THE EXPERT SYSTEM OF L.D.C.

FIG: 21 TECHNIQUES FOR EXTRACTING KNOWLEDGES FROM DOMAIN EXPERT AT LDC written on paper using the concepts, formalisms and rules collected from the expert. Table 9 summarized these ideas. All the written cases solved by the expert were discussed again to provide quick check of the consistency and completeness of the knowledge being extracted.

. Finally the domain expert examined and reviewed each rule and evaluated the control strategies used to select the rules. This provides verifying the accuracy of each rule before the expert system is in operation.

TABLE: 9 FAULTS OCCURED & THEIR REMEDIES SUGGESTED BY EXPERT:

A) OBSERVATIONS:

date	Faults	Remedies
1.13-6-89	SIKALBAHA-MADANHAT line no. 1358 shut_down for greater Chittagong maintenance work.	1) An alter.line 1378 started up which has 300A capacity. 2)KAPTAI feed power in line 1388 through line no 1381 & 1389. No load shedding, system O.K.
2.3-5-89	At evening, system was in peak load, In the THAKURGOAN region the voltage supply	a) SAIDPUR GT started to feed power in line 1251 & 1253

TABLE 9 (Continued)

DATE	FAULTS	REMEDIES
	was under frequency.	b) KHULNA stand-by 110 mw steam lined up.
3.1-6-89	Due to storm the tower of the line no 1298 broken & power was feed through 1299 line (232), at the evening the system goes to peak demand but the generation was much below the demand.	a) All the possible peripheral generators of eastern & western grids was suggested to line up. b) 20 MW load shedding happened.
4.4-4-89	SIDDERGONJ power stations goes overhauling consequently generation become minimum.	a) Power feed from HORIPUR. b) SIKALBAHA started. c) KAPTAI also started to supply power base on the rule curve.
5.5-6-89	Line no.1346 between ASHUGONJ to KISHORGONJ shut-down due to breaker fault.	a) No alternate supply line in that region. b) Complete load shedding in Mymenshing.
6.1-4-89	Transformer 452T(5MVA) of ISHURDI sub-station was in trouble consequently line	a) Be alerted for1256 line rating. b) Partial load shedding

TABLE 9 (Continued)

DATE	FAULTS	REMEDIES
	no 661 for PABNA supply was stopped.	in ULLAHPARA. c) Partial supply from SAHJADPUR to PABNA. d) No load shed to PABNA.
7.15-5-89	Breaker no 112B of BAGERHAT sub-station was out of order	a) Increased supply from GOALPARA by the line 1220. b) Demand of MONGLA port was fulfiled. c) Partial load shedding at BAGERHAT in peak hour.
8.4-16-89	Isolator no.229S of BHERAMARA sub-station was out of order.	a) Increased supply from GOALPARA by the line 1220. b) Power demand min. no load shedding.
8.4-16-89	Isolator no 121S BHERAMARA sub-station was not functioning. Line no 1231 stopped down.	a) Load shedding FARIDPUR at peak hour. b) Partial power supply

TABLE 9 (Continued)

DATE	FAULTS	REMEDIES
		from MADARIPUR.
9.6-27-89	BHERAMARA (gas1 & gas2) was under overhauling. KHULNA barge was shut-down for shortage of water.	a) Suggested to increase power feeding at evening through E-W connector. b) At peak load it was suggested to reduce 15 MW of load at ISHURDI sub-station.
10.5-21-89	Feeder no-2 of MIRPUR sub-station was out of circuited due to faults.	a) Power feed from SIDDERGONJ sub-station by the line no-1331. b) TONGI sub-station feeded power by alter. line. c) At peak demand no load shedding recommended.
11.14-9-88	SIKALBAHA steam shut_down KHULNA barge 60 MW taken off for maintenance & CHITTAGONG steam 60 MW due to draft furnace problem, generation decreased by 20 MW.	a) Generation was much below then the national demand & 35 MW load shedding suggested for system relaibility.
12.1-4-86	In POSTOGOLA sub-station	a) N, GONJ (HORIPUR)s-

TABLE 9 (Continued)

DATE	FAULTS	REMEDIES
	(which meet 25% of greater Dhaka demand) the breaker no 203B & isolator 205B were in faults. So power could feed from TONGI through line no. 1315.	tand by generator was suggested to line up b) Power feed from HORIPURE sub-station. c) At the evening the load shedding was 1M.
13.4-4-88	GHORASAL gas(3) unit at mid-night was stoped as per decision of station chief.	a) From LDC the stand by unit was recommanded to start up. b) No load shedding for the system.
14.3-3-88	For ISHURDI 66KV sub-station in the section of RUPPUR, the grounding & transformer were in trouble and finally it tripped.	a) Power supplied in RUPPUR by extra feeder no X-256. b) The feeder has limited capacity and was unable to supply 30% of total demand. c) Less important areas were disconnected.
15.2-4-88	Readings of the telemeter board were not correct and light blinking was weak.	a) XEN, Telemeter board suggested that power supply faults.

FROM ROSTER BOOKS: TABLE 9 (CONTINUED)

date	Faults	Remedies
16.1-2-87	At 5.10 a.m SAIDPUR 12 MW transformer bursted & cooliing oil vented out also at the same time SAIDPUR gas turbine tripped down.	a) Immediately power fed through 1356 line. b) Recommended to start SAIDPUR diesel plant for local demand.
17.3-1-87	line 1298 (Eastern-Western grid connector) stopped down to convert 230 KV line and only line no 1299 was in action. The demand of western grid was not fulfilled.	a) Suggested to line up all peripheral generators of western grid. b) At the system peak 40 MW load shedding occured. Of which 25 MW for ISHURDI & 15 MW for KHULNA.
18.14-4-88	SIKALBAHA steam shut-down, KHULNA barge 60 MW goes for maintenance & CHITTAGONG steam 60 MW due to draft furnace problem, generation fall-down to 20 MW.	a) Generation was much below the national demand & 35 MW load shedding was suggested for system relaibility.
19.1-2-86	In POSTOGOLA sub-station (which meet 25% of greater Dhaka demand) the breaker	a) N, GONJ (HARIPUR) stand by generator was suggested to line up.

DATE	FAULTS	REMEDIES
	no 203B & isolator 205B were in faults. So power could be feed from TONGI through line no. 1315.	b) Power feed from HARIPUR sub-station. c) At the evening the load shedding was 1 MW.
20.30-12-87	SHAHAZIBAZAR gas-2 generator´s turbine plates were broken and the unit was shut down as per order of station chief engineer.	a)Increased power feed from ASHUGONJ to SHAHJIBAZER substation through line no 1324. b) At peak demand partial load shedding to SRIMONGAL.
21.7-1-88	BARISAL gas1 was stopped due to shortage of fuel also the total western generation was much below the demand.	a) Regional demand of BARISAL was fulfilled from GOALPARA via BAGERHAT sub-station. b) POTUAKHALI was loaded at peak hour.
22.9-10-87	It was decided to disconnect line no. 1389 for maintenance operation, because some of the equipments connected with this line was causing mal-functioning.	a) Power to supply at BOROAWLIA by alternate line 1388 via SIKALBAH. b) No load shedding for the system. c) Maintenance operation both for near and far end was suggested.

DATE	FAULTS	REMEDIES
23.2-8-87	Line of 1243 (NATORE-BOGRA) tripped for some electrical faults.	a) No alternate power supply line for BOGRA. b) BOGRA gas (5MW) & diesel (20 mw) were started for regional demand. c) Partial load shedding for PALASHBARI.
24.4-16-89	Isolator no 121S BHERAMARA sub-station was not functioning. Line no 1231 disconnected.	a) FARIDPUR load reduced by 4MW at peak hour. b) Partial power supplied from MADARIPUR.
25.6-27-89	BHERAMARA (gas1 & gas2) was under overhauling. KHULNA barge was shut-down for shortage of water.	a) Suggested to increase power supply at evening through E-W connector. b) At peak load it was suggested to reduced load by 15 MW at ISHURDI sub-station.
26.5-12-88	KABIRPUR sub-station feeder no-2 could not supply power and connected breaker 403B was out of order.	a) TONGI-HORIPURE line load was kept under observation, so that it would not tripped. b) Suggested to loadshed TANGAIL & HORIPUR.

TABLE 9 (Continued)

DATE	FAULTS	REMEDIES
27.2-10-86	Both the two genetators of KAPTAI were overhauling, poer supplied in DOLHAZARI & COX'S BAZER by line no 1358 but that line also tripped due to unknown reasons.	a) Power supplied from TULSHI. b) Regional demand could not be supplied. c) Shut-down the line 1361 (COXE'S BAZAR).
28.7-12-87	Due to overhauling the generation at SIDDERGONJ is minimum. So from HORIPUR the power was transmitted, but the single line no 1352 was tripped again and again due to over current.	a) Power transmitted from SIKALBAHA. b) From KAPTAI if water level high.
29.2-8-87	Line of 1443 (FARIDPUR-MADA) tripped for some electrical faults.	a) No alternate power supply line for MADARIPUR load shedding for MADARIPUR.
30.9-9-87	Due to storm electrical tower of the line no 1203 between JOYPURHAT and BHERAMARA was fall down.	a) Breaker, Isolator 112S, 305B of BAGERHAT end disconnected. b) Consequently breaker, Isolator, 425B,306 of JOYPURHAT end disconnected. c) Total load shedding at JOYPURHAT for 26Hr.

4.5 KNOWLEDGE REPRESENTATION PROCESS AT LDC:

With the over all analysis of national grid system and the knowledges of the system expert relating to faults analysis, it was conveniently accepted that all the procured knowledges of the LDC can be best represented by RULES system representation techniques. The most important reason behind the acceptance of this system of representation because of its flexibility. The present status of the generators and their generation capability changes frequently due to mechanical and electrical problems, consequently the pieces of knowledge procured on the basis of these generators also changes. Only the RULE system representation techniques provides the facility to change a rule or to add the rule by changing the static data base of the program. The following important rules developed for some representative faults are described below-

GENERATION FAULTS

RULE NO-1

PROBLEM: THE GHORASAL STEAM (GT4) GOES MAINTENANCE AND ASHUGANJ GAS (GT-1) TRIPS FOR ELECTRIAL FAULTS.

ACTIONS: a) If water level high, start all machines of KAPTAI
b) Start generators of estern region,
c) Start the machines of western regions,
d) Note the max. demand & total generation.

DECISIONS:IF DEMAND > GENERATION THEN REDUCE LOAD SHEED ELSE GRID O.K.

Figure No: 22

RULE NO-2

PROBLEM: KHULNA BARGE (1&2) STOP FOR MAINTENANCE, CHITTAGONG STEAM (60 MW) HAS DRAFT FURNANCE FAULT, SIKALBAHA (1 & 2) WAS

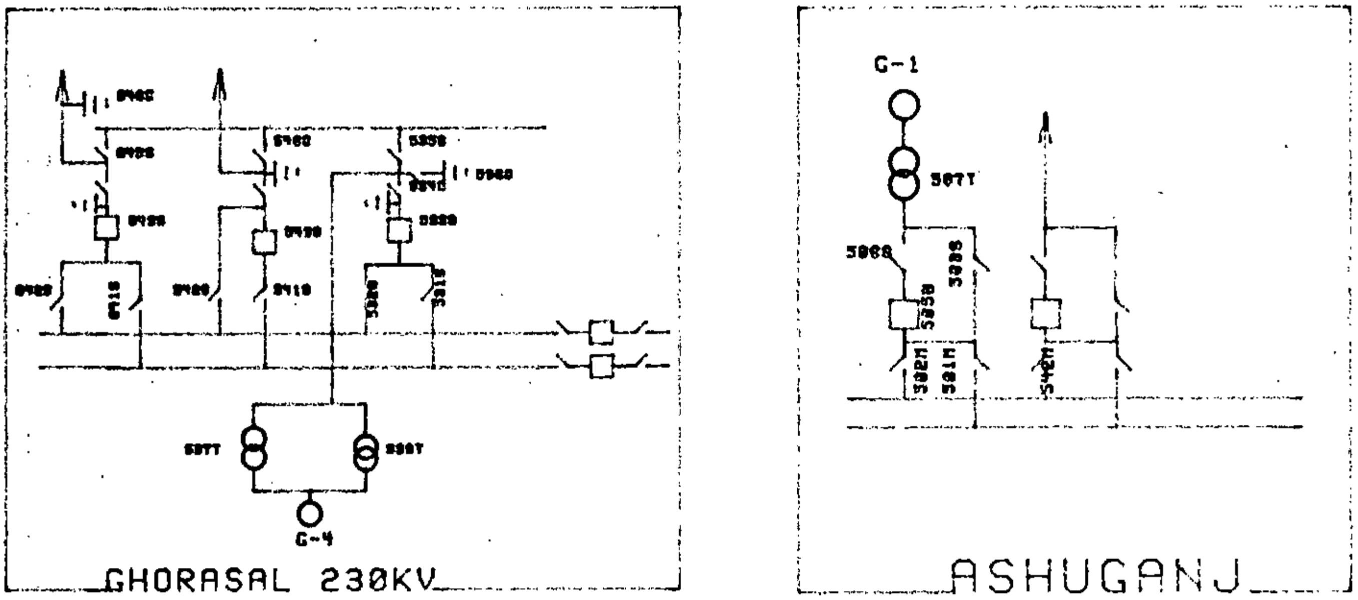


FIG: 22 THE GENERATION FAULTS FOR RULE NO.-1

OFF FOR MECHANICAL FAULTS.

ACTIONS: a) Whether all the base stations are in operation,
b) Start "KAPTAI" generators, if rule curve allowes,
c) Start the perepherial generators of estern grid,
d) Start in sequence the generators of western grid.

DECISIONS:IF GENERATION > DEMAND THEN SYSTEM IS BALANCED, ELSE LOAD SHED ACCORDING TO PRIORITY.

RULE NO-3

PROBLEM:"KAPTAI" WATER LEVEL LOW, "KHULNA" STEAM (110 MW) GOES FOR MAINTENANCE, "CHITTAGONG" BARGE (1) STOPPED FOR SALANITY.

ACTIONS: a) "ASHUGONG" and "GHORASAL" all machines started as principal generators
b) Power stations in estern grid started up,
c) Power stations in western grid started up,
d) Calculate total demand and generation.

DECISIONS: IF GENERATION > DEMAND, NATIONAL GRID O.K, ELSE SWITCH OFF LOADS SHEED ACCORDING TO PRIORITY.

Figure No: 23

RULE NO-4

PROBLEM: "ASHUGONG" STEAM (2) TRIPS FOR ELECTRICAL FAULTS, "GHORASAL" GAS (2) GOES TO MAINTENANCE.

ACTIONS: a) If water level high, start "KAPTAI" generators,
b) All the machines of "ASHUGONG" and "GHORASAL" will be used to supply the base load,
c) Start rest of generators of estern grid,
d) Start rest of the generators of western grid.

DECISIONS: IF GENERATION > DEMAND, THEN GRID IS NORMAL, ELSE REDUCE LOAD.

RULE NO-5

PROBLEM: "HORIPUR" GAS (20 MW) TRIPS, "CHITTAGONG" GAS (TG7)

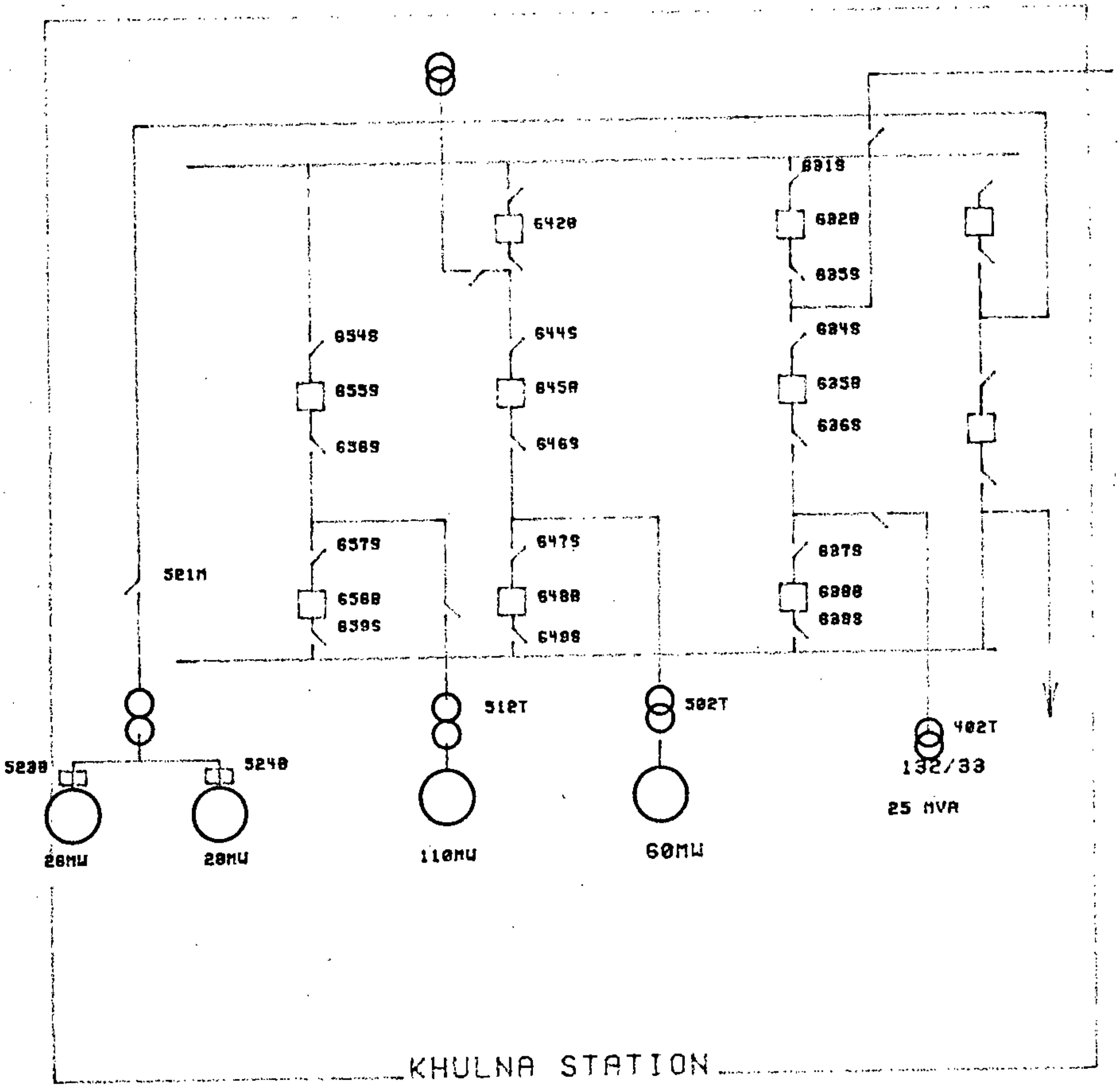


FIG: 23 THE GENERATION FAULTS FOR RULE NO.-3

IN MAINTENANCE, "KAPTAI" WATER LEVEL MUCH BELOW RULE CURVE.

ACTIONS: a) Start the GTs of base stations,

b) line up Estern peripherial GTs,
c) Run up Western peripherial GTs.

DECISIONS: IF GENERATION > DEMAND, SYSTEM O.K. ELSE SWITCH CERTAIN LOADS.

Figure No: 24

SUB-STATION FAULTS.

RULE No-6

PROBLEM: BREAKER NO 112B AT NATORE SUB-STATION TRIPS FOR FAULTS SO LINE NO 1242 GOES OFF.

ACTIONS: a) Be alert about ratings of parallel line no 1243,
b) Start up SAIDPUR GT1 and GT2,
c) Start up RANGPUR GT1,
d) Start up BOGRA GT1.

DECISIONS: IF DEMAND > GENERATION, LOAD SHEDDING "THAKURGOAN" AND "PALASHBARI". ELSE SYSTEM NORMAL.

Figure No: 25

RULE NO-7

PROBLEM: BREAKER NO115B OF "ASHUGONG" SUB-STATION TRIPS, CONSEQUENTLY LINE NO 1323 ALSO GOES OFF.

ACTIONS: a) Be carefull of alternate line No 1324,
b) "SHAHJIBAZER" GT1 start up,
c) "SYLHET" GT1 line up,
d) Note down maximam demand in that region,
e) Total regional generation.

DECISIONS: IF DEMAND > GENERATION, THEN SWITCH OUT "SRIMONGAL" ELSE SYSTEM O.K.

RULE No-8

PROBLEM: LINE NO 1358 GOES FOR MAINTENANCE DUE TO FAULT IN

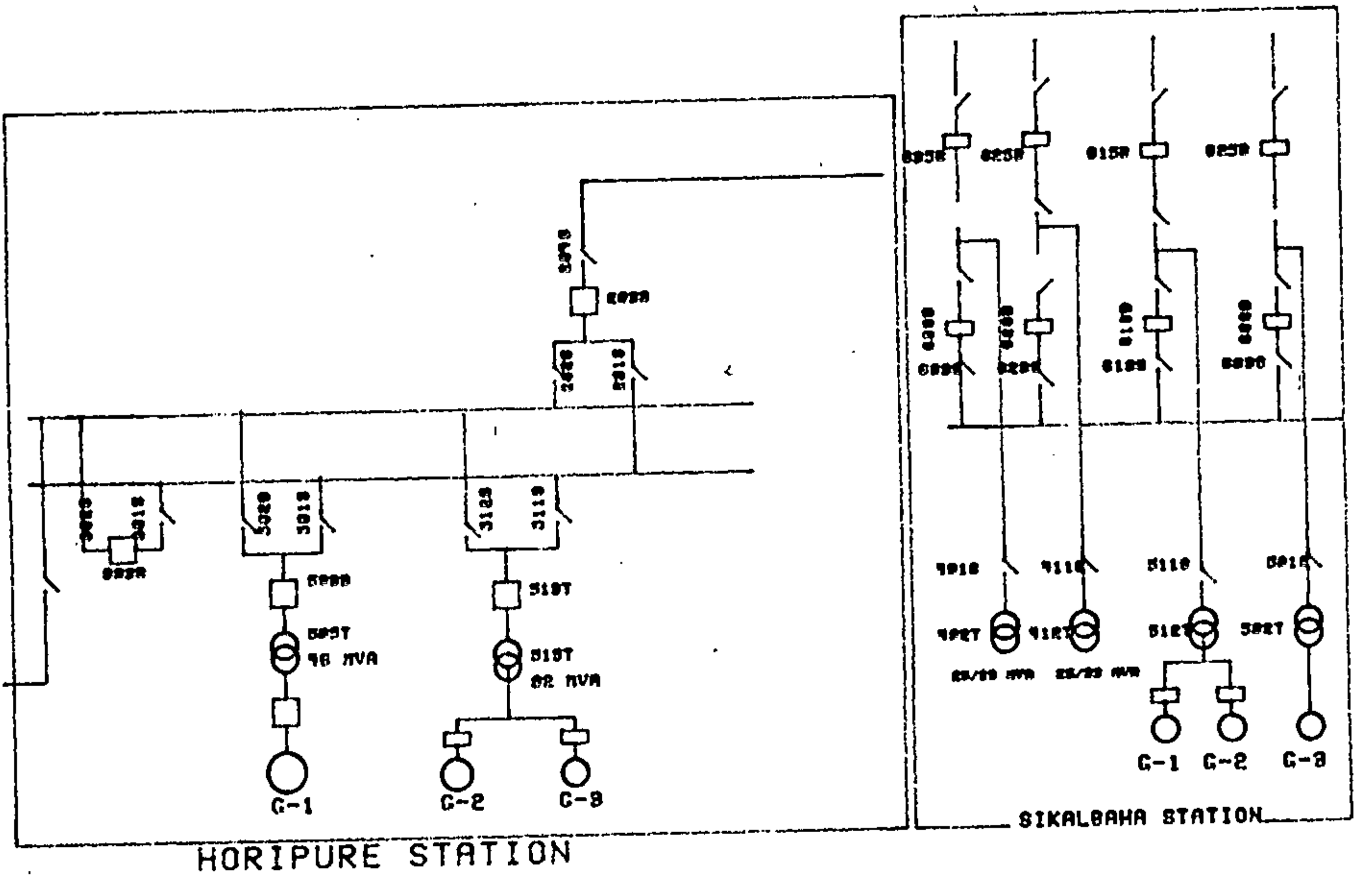


FIG:24 STATION FAULT FOR RULE NO.-5

BREAKER NO 625B AT "SIKALBAHA".

ACTIONS: a) Parallel lines ratings of 1388 and 1389 must be minimum,
b) Power supplied from "KULSHI" sub-station through line no 1379,

c) Power supplied from "HALISHAHAR" sub-station through line no 1378.

DECISIONS: IF SUPPLY > DEMAND, SYSTEM O.K. ELSE SWITCH OUT "BAROAWLIA" LOAD.

Figure No:26

RULE NO-9

PROBLEM: IN "BHERAMARA" SUB-STATION CIRCUIT BREAKER NO 605B TRIPS DUE TO OVER CURRENT AND LINE NO 1231 SHUT DOWN.

ACTIONS: a) Be carefull about rating of line no 1237,
b) "BARISAL" GT1 & GT2 start on,
c) if demand not fullfil.

DECISIONS: SWITCH OUT "PATUAKALI" LOAD, ELSE SYSTEM O.K.

RULE NO-10

PROBLEM: "IRSHURDI" 66KV SUB-STATION INFORMED THAT 133B HAVE MECHANICAL FAULTS AND 661 LINE SHUT DOWN.

ACTIONS: a) Loading of the lines 660 & 1265 are to be minimum,
b) Power feed from "RAJSHAHI" sub-stations,
c) Power supplied from "SHAHJADPUR" sub-station.

DECISIONS: IF DEMAND > SUPPLY, THEN SWITCH OUT "SIRAJGONG" AND "ULLAHPARA" LOAD, ELSE SYSTEM NORMAL.

Figure No:27

RULE NO-11

PROBLEM: AT "GOALPARA" SUB-STATION BREAKER NO 608B AND LINE 1220 TO BE SWITCHED OUT FOR MAINTENANCE.

ACTIONS: a) Loading of the line 1221 be minimum,
b) "BARISHAL" GT1 & GT2 start on.

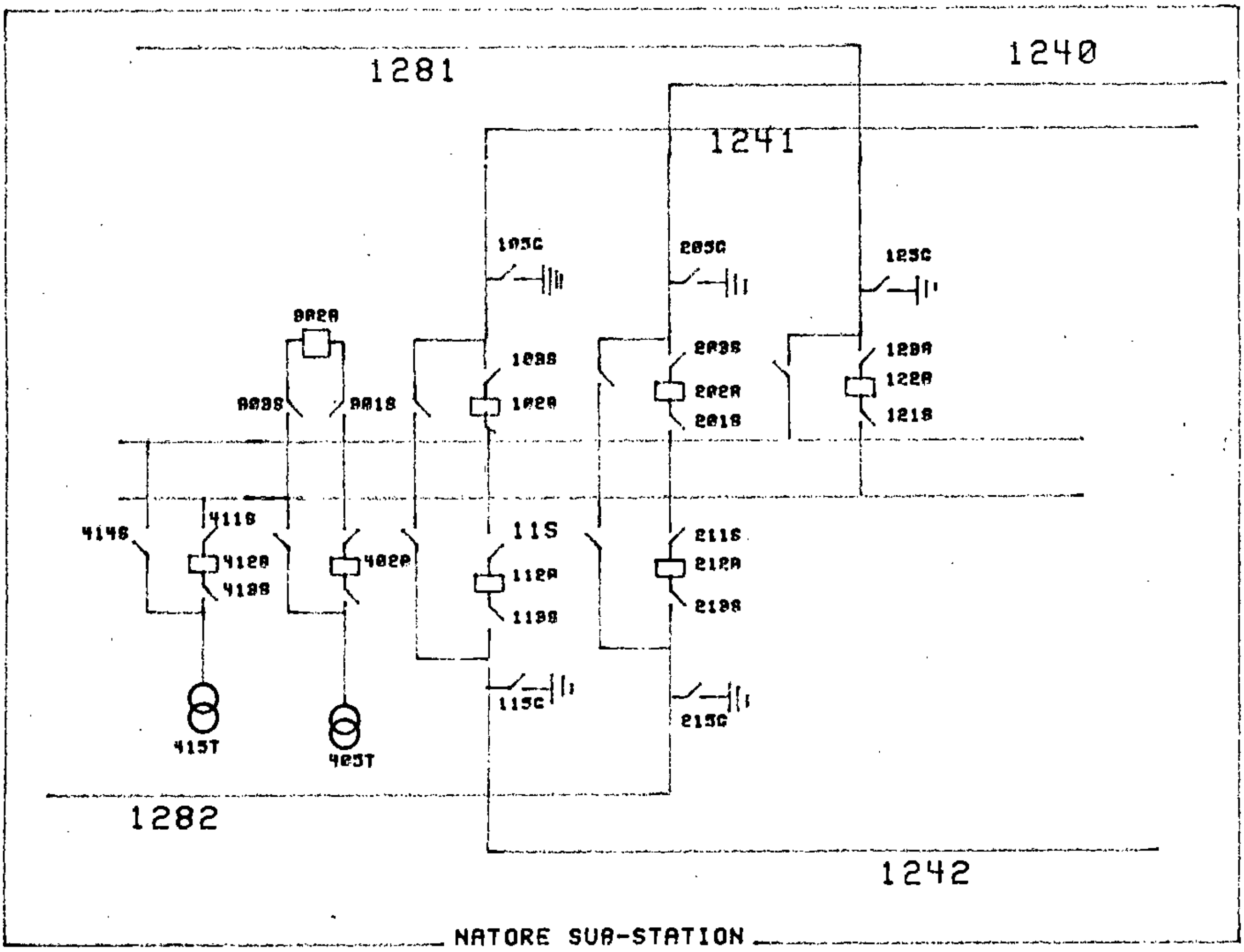


FIG: 25 FAULTS FOR NATORE SUB-STATION OF RULE NO.-6

DECISIONS: IF GENERATION > DEMAND, THEN SYSTEM O.K, ELSE SWITCH OUT "PATUAKHALI".

RULE NO-12

PROBLEM: CIRCUIT BRKEAKER (NO 212B) CREATED PROBLEM AT KHULNA SUB-STATION.

ACTIONS: a) Check rating of line no 1203,
b) Power directed from "BHERAMARA" sub-station through line no 1203

DECISIONS: IF DEMAND>SUPPLY, LOAD SHEDDING IN "JESSOR" ELSE OK.

Figure No:28

RULE NO-13:

PROBLEM: FOR SOME ELECTRICAL FAULTS AT NOAPARA SUB-STATION CB NO 605B SHUT DOWN.

ACTIONS: a) No tripping of line no 1206 due to overcurrent,
b) Power feed from "BOTTAIL" sub-station,

DECESIONS: IF DEMAND > SUPPLY LOAD SHEDDING IN "JHENIDAH", ELSE SYSTEM O.K.

MAINTENANCE FAULTS

RULE NO-14

PROBLEM:" HARIPOR - HATHAZARI" LINE (NO 1353) SWITCHED OUT FOR GREATER CHITTAGONG WORKS.

ACTIONS: a. power can be supplid through 1552 no. line.
b. kaplai can Supply through 1384 No. lines if water level high
c. What is the impact on the National grid.

DECISION: a) Brcaker no 113B, and Isolator 114S disconnected at HORIPUR.
b) Breaker no -102B, and Isolator 1011S also disconnceted from system at HATHAZARI.

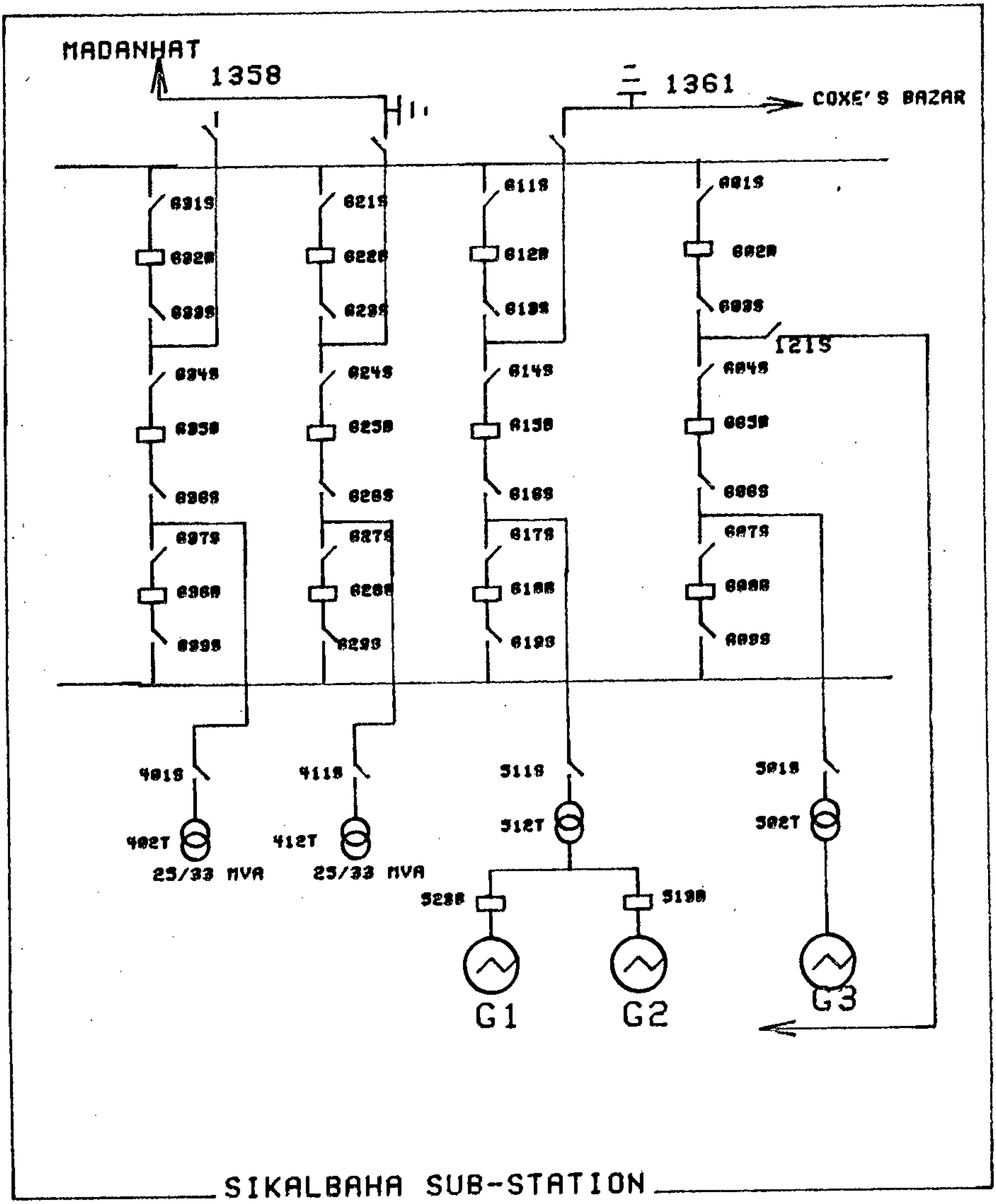


FIG:26 SUB-STATION MECHANICAL FAULT FOR RULE NO-8 disconneeted from system at HATHAZARI,

c) Grounding No 115G in HORIPUR and Grounding also 105G at HATHAZARI.

RULE NO 15

PROBLEM: "ISHURDI-PABNA" LINE. NO. 662 SHUT-DOWN FOR MAINTENANCE WORK:

ACTIONS: a) Power Can be Supplied through SAIDPUR,
b) BOGRA power station started,
c) What is the impact on the national grid whether demand > supply not.

DECISIONS: a) Breaker no 133B and Isolator 134S disconneeted form ISHURDI end,
b) Breaker no.102B and Isolator 103S disconnected from PABNA end,
c) Grounding at 125 G in ISDHURDI and Grounding at 109G in PABNA.

RULE NO-16.

PROBLEM:GREATER DHAKA CE ASKED FOR 1313 LINE MAINTENANCE "SIDDIRGONJ TO MIRPUR."

ACTIONS: A) Power Supplied form TONGI through 1311 line,
b) Whether demand > Supply, if so reduce load at Mirpur Sub-station.
c) Circuit Breaker no 941B and Isolator 141S disconnceted form SIDDIRGONJ end.

DECISIONS: a) Breaker no 113B and Isolator no 114S disconneeted form MIRPUR end,
b) Ensure Grounding No-116G at MIRPUR and Grounding 142G at SIDDIRGONJ.

RULE NO-17

PROBLEM: FOR TOWER FAULT LINE NO. 1220 DISCONNECTED FOR MAINTENANCE..

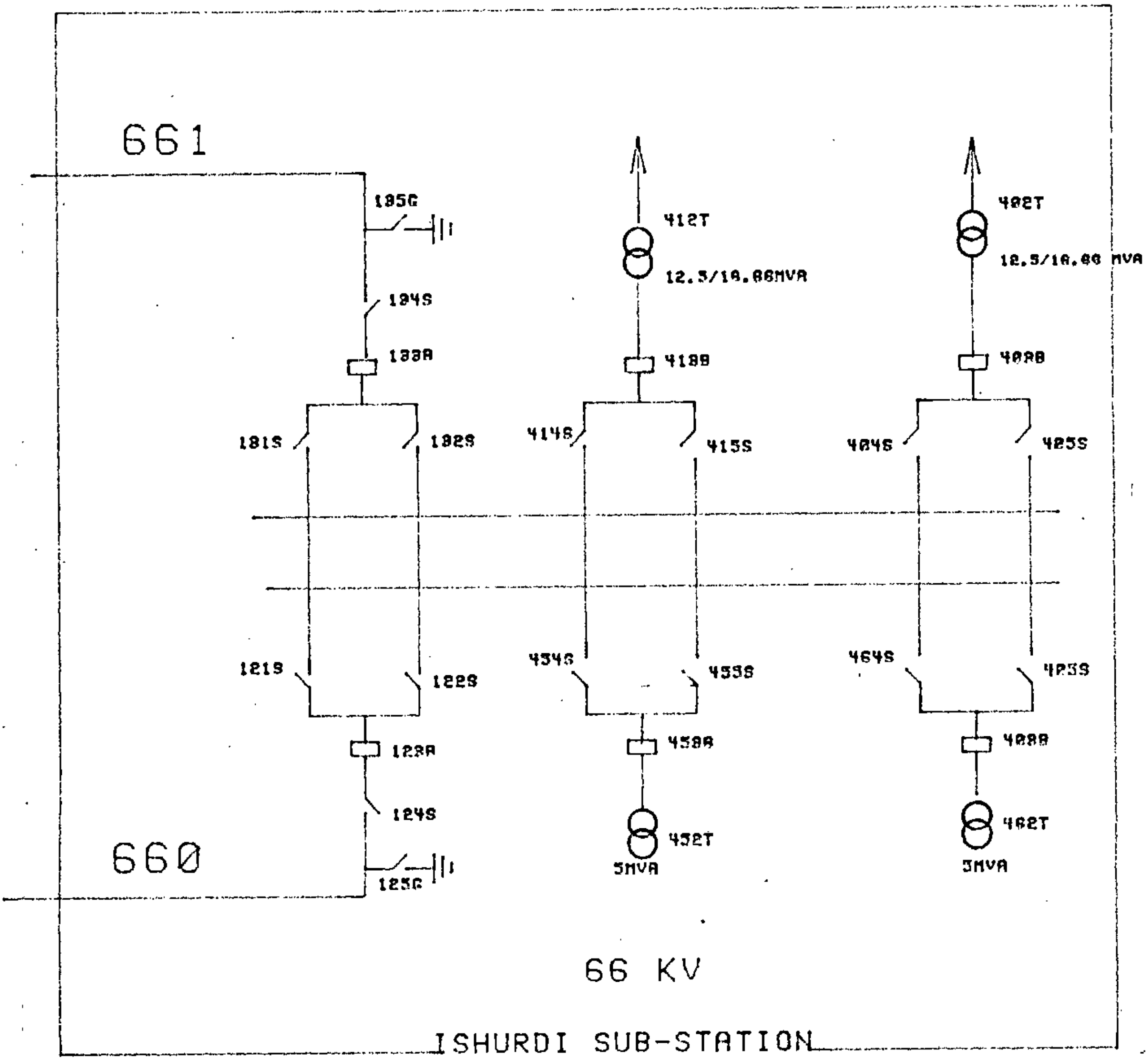


FIG: 27 THE BREAKER FAULT FOR ISHURDI SUB-STATION OF RULE NO.-10

ACTIONS: a) Power Supplied from BARISHAL through line no-1221,
b) whether demand > Supply, if So load shedding in BAGERHAT.

DECISIONS: a) Breaker no 608B and Breaker no 605B and Isolator 607S and 606S disconnceted from GOALPARA end,
b) Breaker no 102B and Isolator 103S and 101S disconneeted from BAGERHAT end,
c) Put Grounding no.112G at GOALPARA and also Grounding no.105G at BAGERHAT.

RULE NO-18

PROBLEM: LINE NO, 1245 "BOGRA---POLASHBARI" DAMAGED BY STORM TO BE REPAIRED.

ACTIONS: a. Power feed through 1245 lines,
b. If demand > Supply, switch out POLASHBARI.

DECISIONS: a) Circuit Breaker no. 213B Isolator 214S disconnected from the BOGRA end,
b) Breaker no,209B and Isolator 204S disconneeted from POLASHBARI end,
c) Ensure Grounding no.215G at BOGRA and Grounding no 205G at POLASHBARI.

RULE NO-19

PROBLEM: SAIDPUR 12 MVA TRANSFORMER'S FUSE BLOWN OUT DUE TO STORM. (Saidpur diesel plant goes into maintenance)

ACTIONS: a) Power feed through line no 1248,
b) Start up the RANGPUR Gas GT,
c) If demand > Supply, reduce load,

le
DECISIONS: a) Disconncet Breaker no 203B and Isolator no 209S at SAIDPUR end,
le
b) Disconncet Breaker no 213B and Isolator 219S at RANGPUR end,
c) Grounding no 205G at POLASHBARI and Grounding no-

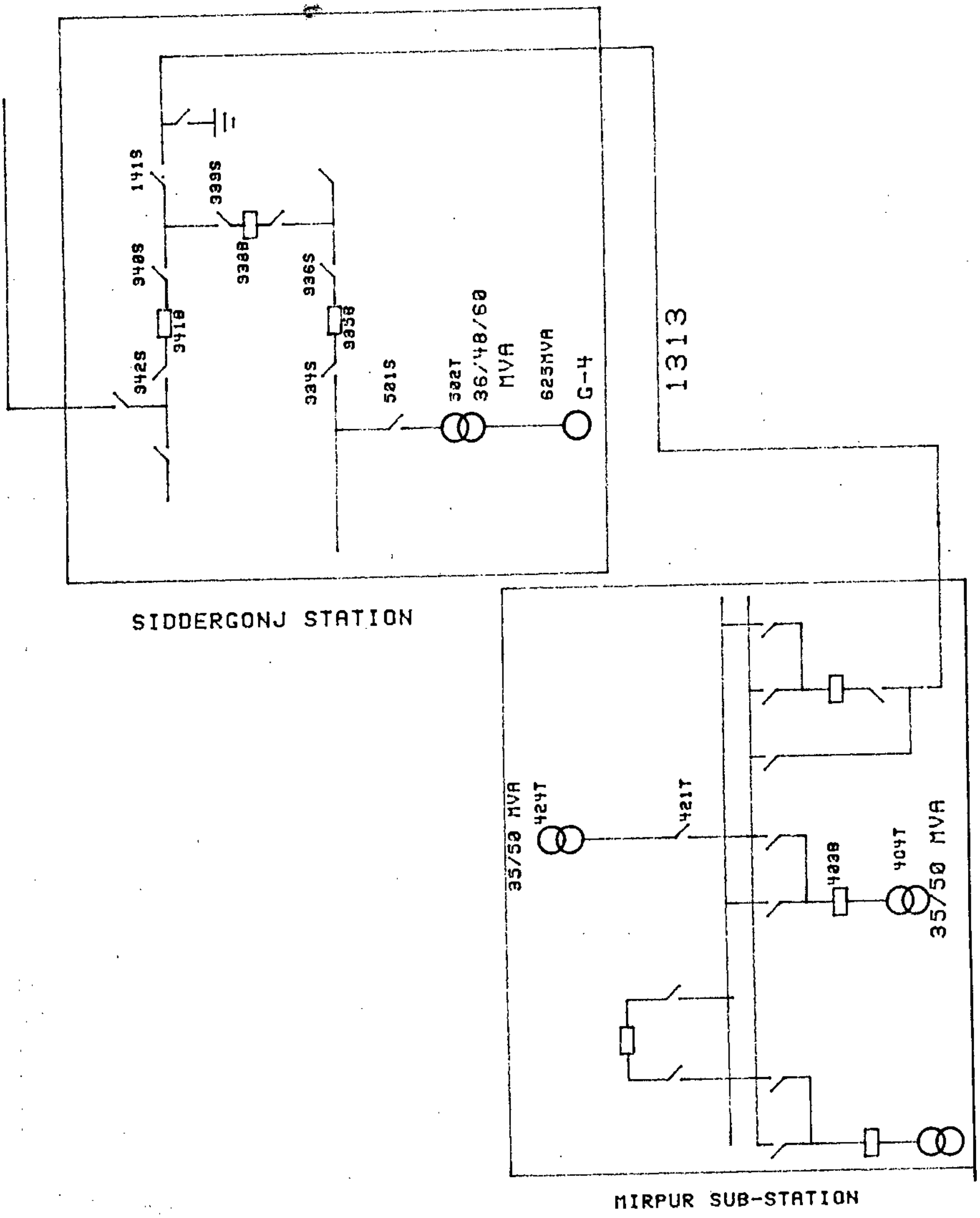


FIG: 28 MAINTENANCE FAULTS FOR RULE NO.-16

215G at RANGPUR.

RULE NO-20

PROBLEM: FOR MAINTENANCE WORK OF GREATER CTG " MADANHAT--SIKALBAHA" LINE NO 1358 TO BE DISCONNECTED.

ACTIONS: a) Power Supplied through Line no 1378,
b) KAPTAI can be started,
c) Demand > Supply, if So DOHAZARI feeder switch out.

CHAPTER–5

EXPERT SYSTEM

PROGRAM DEVELOPMENT FOR L.D.C.

5.1 JUSTIFICATIONS FOR USING PROLOG AS PROGRAMMING TOOLS:

On the basis of the special features of PROLOG a brief description of reasons using PROLOG as programming language is given below-

a) PROLOG is a fifth-generation language that takes programming into a new dimension. Because of its natural logical approach, both people new to programming and professional programmers can build powerful applications- such as expert system, natural language interface etc.

b) PROLOG is a declarative language, this means that, given necessary facts and rules, it can use deductive reasoning to solve programming problems. This type of declarative language, characterized by the fact that no definite algorithm or facts are explicitly defined in the program, rather it provide a set of rules and facts on the inference mechanism of PROLOG to find a path with in the problem domain and to find a solution to a particular problem.

c) PROLOG'S power is in its ability to infer facts from other facts, the description in a prolog program is used to specify the desired relation between input data and the output will be generated from that input.

Since at the Load Despatch Center the different types of faults can be defined in the form of rules. So the descriptive type of language like PROLOG is more suitable than procedural languages.

d) Execution of PROLOG program is controlled automatically. When PROLOG program is executed, the system tries to find all possible sets of values that satisfy the given goal. That's why to reach a conclusion after traversing different nodes of the faults PROLOG offer more suitable solution at LDC.

e) Procedural languages distinguish between a program and the data it uses. The procedure and control structures of a program are defined by the programmer. The program manipulates the data according to the defined procedure. But developing programs with PROLOG is dramatically different, A PROLOG program is a collection of data or facts and relationships combining these facts. In other words the data base is an integral part of program.

The knowledge extracted from the daily Roster Books and from previous history at L.D.C are collected in the form of heuristic. By using these heuristic PROLOG can easily reach a goal.

f) One of the outstanding feature of turbo prolog is that it hold dynamic database, during program execution, the facts obtained from the user during consultation session that apply only to the current consultations are stored in the dynamic data base. Facts can be added using asserta or assertz predicates and removed with the retract predicates.

The expert system developed has a consultation module. The informations gathered from consultation are important to reach a goal, also the informations that are necessary changes extracted consultation . So it is obvious to have a dynamic data base as tMe PROLOG has.

g) Turbo PROLOG has a very short simple syntax, the turbo PROLOG typically uses 10 times fewer programming lines when solving a program than pascal. Among other things turbo PROLOG has a built in pattern recognisation facility and as well as simple and efficient way of handling recursive structure.

PROLOG'S special features as mentioned above and its proximity with logical reasoning facilitates codification of a program and makes theoretically possible "intelligent computer" in a limited sense.[6 & 9].

5.2 PROGRAM EXECUTION METHODS FOR THE EXPERT SYSTEM AT LDC:

There are different types of program execution methods on PROLOG to reach a goal. It is relatively straight forward to develop a skeleton of expert system, if we restrict ourselves to the basic functions lying at the heart of such system. But to develop a powerful expert system we have to consider three fundamental elements: Knowledge consisting of conditional statements, data obtained from the problem at the hand, the process of inference.

5.2.1 METHODS OF PROGRAM EXECUTION:

On the basis of this concepts two major strategies that have been used in executing a program for the expert system at L.D.C.

a) Unification.

b) Backtracking.

a) UNIFICATION:A Unifier of two terms is a substitution making the terms identical. If two terms have a unifier, we say they unify in all otherwise a failure is used to occurs. In fact when a goal or respective sub-goal is invoked, PROLOG tries to prove the goal by matching the said goal with the facts or rules in the program. The algorithm used for this matching is called "Unification".

In PROLOG, predicates unify with each other if

* They have the same relational name (predicates).

* They have the same number and same type of arguments.

* All arguments unify with each other.

From the user point of view the rules for unification are

* A free variable will unify with any terms that satisfies the conditions of unifications. After unification the variable will be bound to the value of the term.

* A constant may unify with itself for any variable. If the constant is unified with a free variables, the free variables will be bound to the value of the terms.

* A free variable to unify with another free variable, after unification, the two variables will act as one. If one of them is bound to a value the other one will be bound to the same value.

As an example let us consider the expert system program. If we enter the goal Breaker (X, '12MW'), then the following unification will take place

```
                    breaker (115,'12MW').
                    breaker (X , '12MW').
```

Therefore, the answer to the initial goal will be X=115.

b) **BACKTRACKING:** The backward chaining technique used in different places of the program to make the expert system more faster. The process of matching the goal or sub-goal to a fact or head of a rule when one fact or a rule fails is known as backtracking. It is very important concept in execution of program for expert system at L.D.C. Let us consider the following conceptual program-

```
Load shedding if
            problem 1,
            check_kaptai,
            check_generator.

problem 1, if
            gen_fault,
            breaker_faults,
            station_faults.
```

```
problem 1 if
                 line_faults,
                 maint_faults.

check_Kaptai if
                 water_level,
                 add_hydro.

check_Kaptai if
                 mech_faults,
                 goes_maint,
                 add_hydro.
```

In the above program in order to prove loadshade we have to prove problem1, check_Kaptai, and check_generator. However, in order to prove problem1 we have to prove either gen_faults, or breaker faults or stations faults. Alternately the second predicates line faults or maintenance faults and so on.

The execution of prolog program takes place with the left most path being search first. It is therefore a case of depth-first search. This means that to prove load shade we first try to prove problem 1. If the first rule of problem1 (i,e gen_faults,breaker faults etc.) is satisfied. It will keep a mark or pointer here and then proceed to prove next goal check_Kaptai.

If the first rule for check_Kaptai fails, backtacking occur and the next rule (check_Kaptai:- mech_faults, goes_maintenance, add_hydro) is tested , if this rule also fails then program backtracks to the previous rule problem1, where it earlier place the pointer. The next rule for problem1 (problem1: line faults, maint_faults) is to resatisfy the rule problem1. The whole process is known as "depth first search with backward chaining".

5.2.2 CONTROL TOOLS USED FOR PROGRAM EXECUTION:

The program developed for the expert system executes by the process of backtracking and unification to search through its problem space to find a solution to a particular problem or to satisfy a goal using the stated facts and rules in the data base.

This gives the program its declarative nature. However it is often necessary to introduce some procedural nature in the program specially to control the search space which will otherwise grow combinationally. This is required to run the program efficiently, to reduce runtime and to use minimum storage.

To accumulate the above advantages the following control predicates used in the expert system program.

* Recursion

* Cut

* Fail

* Repeat

RECURSION: Recursion is a technique in which something is defined in terms of itself. Recursion is a technique of using a clause to

a) DEFINING THE PROBLEM: It was very trivial to consider first in program writing is to find out upto what extent the result by the program is expected to do. This includes defining the set of input, the set of outputs and if possible to determine some of the the operations required by the program.

The first step in designing the system is to clearly define what types of problem the expert system will solve and to what extent the system will provide recommendation for solving the fault. The expert system deals with specific problem of national grids and its optimum actions to overcome the faults.

b) GATHERING KNOWLEDGE ABOUT THE PROBLEM DOMAIN: The accumulation and codification of knowledge is one of the most important aspect for an expert system. To make a program intelligent, we must provide it with lot of high quality specific knowledge about some problem area. The knowledge is accumulated for this system

1) From domain experts.
2) Reports of the previous faults.
3) From shift engineers.
4) From daily roaster books. etc.

c) CHOOSING A DATA STRUCTURE AND KNOWLEDGE REPRESENTATION:

The biggest problem in designing an expert system is the proper interaction between the knowledge engineer and the domain expert. The knowledge engineer observes, talks, with human expert in a objective form.

invoke a copy of itself. For example , we use the recursion clause check_generator in the expert system program of LDC as

```
check_generator (S,D,_):-
                    S>D, !.
check_generator (S,D,_):-
                    generator (S,Y,Z,N),
                    SS=S+Z,
                    NN=N+1,
                    check_generator (SS,Y,Z,NN).
```

The above recursion rule used to add the generators in the line unless the supply is greater than demand. The general form of a recursive clause are as follows

```
clauses to terminate the recursive loop.
head of the recursive clause:-
body with in the recursive clause,
the recursive clause causing it to invoke itself
body outside the recursive loop.
```

Special care was taken with recursive clause so that the clause does not enter in an infinite loop. Care should be taken in writing the terminal clause so that the recursive terminates each line.

CUT '!': Cut is a means to stop backtracking. The 'cut' control predicate is implemented in the program as an exclamation point (!). The cut tool always succeds and once it succeds it acts as a fence, effectively blocking any backtracking beyond the cut. As an illustration we consider a segment program of the expert system

```
                    add_hydro (G,D_):-
                              G>D, !.
                    add_hydro (G,D,_):-
                              hydro_generator (N,G,X,M),
                              NN=N+1,
                              GG=G+M,
                              add_hydro (NN,GG,X,M).
```

Here by recursive procedure the program will line on a generator and subsequently check the updated total generation. At the moment when the generation will be greater than the demand the cut (!) will stop further backtracking otherwise the program will enter in a infinite loop.

Cut is always used to reduce the search space to increase runtime. To reduce storage requirements and to increase overall program efficiency.

FAIL: The fail predicate is built-in-predicate which forces a clause to fail. It is important when we want to test all the clauses of a particular type. In the expert system the predicate rule (X,Y) uses to fail predicates. This has been made so that the program tests all the possible rules to find out all possible faults that may occur for the particular types of faults.

The Cut-Fail can be used to provide negation. For example, to write a predicate which succeds if a certain other predicates say fault (X) fails. This can be done by using the not predicate

```
not ( fault (X) ).
```

However, it is our experience that the use of the not predicate in this version of the compiler (1.0) causes the program to behave erratically. The cut-fail combination to initiate the not predicate as shown below

```
The_fault (X):-
          fault (X),!,fail.
The_fault (_).
```

REPEAT: In order to make the expert system more efficient and friendly and protectable in nature, the repeat predicate is used in the program. This predicate is one such predicate which is evidently recursive in nature. It causes a tree structure of infinite branches as shown figure 32.

The 'repeat' predicate is defined as

```
repeat.
      repeat:-
            repeat.
```

The repeat predicate is extensively used in the expert system to make it user oriented. This is shown below

```
end:-
```

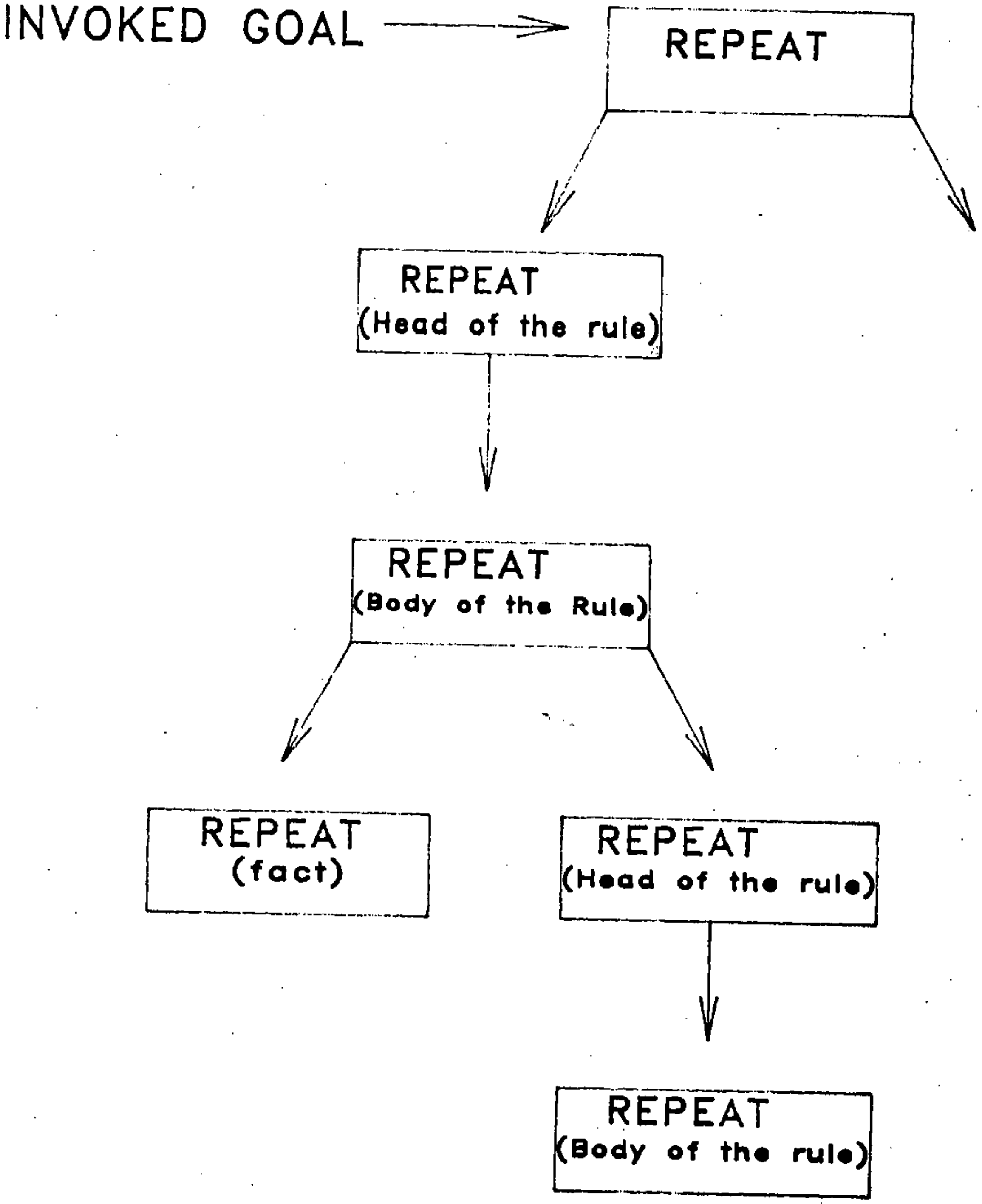


FIG: 29 THE REPEAT PERIODIC ILLUSTRATIONS AND ITS INFINITE BRANCHES

```
  write ("Is the water level above the rule curve")
  reply (X),
  X='y',
   .
   .
  end.
reply(X):-
      repeat,
      readchar (X),
      answer (X).!,
      answer ('y'),
      answer ('n'),
```

The effect of the above program segment is that when the computer asks the user whether he wants to know about the water condition of Captai, pressing any key other than y or n will have no effect. Pressing 'n' will terminate the program. Pressing 'y' will cause the program to continue.[11 & 9]

5.3 STEPS IN THE DEVELOPMENT OF EXPERT SYSTEM PROGRAM:

An expert is a person who, because of training and experience, is able to do things that the rest of us cannot, experts are not only proficient but also smooth and efficient in the actions thy take. Efficient programming techniques is an important factor to make the expert system's nature as expert mentioned above. The following stages are used as a guideline to develop the expert system program.

All the knowledge accumulated from system experts are stored in a series of if then rules which are typically of the form

IF<CONDITION>, THEN <ACTION>.

The reasoning process of the expert system involves matching and unification. The system assumes that there is a problem and works towards solving the problem by 'triggering rules' if their activation pattern matches with the data element in the dynamic data base.

d) CHOOSING AN ALGORITHM: Although PROLOG is declarative, we may need to choose an algorithm or more accurately a decision tree to be used to search through the problem space. PROLOG by itself uses depth-first search technique. The algorithms used are details discussed in the next article.

e) DEVELOPING THE PROGRAM: The final step is to write the code of the program in PROLOG. To do so we have to first convert the gathered knowledge into PROLOG rules and facts. The program is divided into two main sections. The first section which is the heart of the expert system contains question asking, grid faults, production rules & explanation module.

The second part of the program use tortle graphics facility of turbo PROLOG to display the location of the faults. Finally it require to consider efficiency, storage requirement, runtime, etc. of the program.

5.4 PROGRAM MODULES OF THE EXPERT SYSTEM AT LDC:

The heart of an expert system is its corpus of knowledge. How is that knowledge organized is the prime factor to support the decision making and to reach a goal. The strategy followed here is of modular programming in developing the LDC expert system. The system is organized into four distinct modules. The figure 30 shows the different program modules of the expert system.

a) Inference engine

b) Questioning module

c) Knowledge base module

d) Fault location module

a) INFERENCE ENGINE OR RULE ACQUISITION PROGRAM:

The inference engine, as its name implies provides the motive power to the system. Its functions are: to determine what data it needs to solve the problem at hand, to get this data via the support software, to lodge it in the data base, to employ the contents of the knowledge base to draw inferences and to record these as well in database. It exercises these functions repeatedly, until it can do, need to do or no more.

Like any other expert system the inference engine is the core of the system. The user enters into a consultation with the expert system whereby the system gathers informations about the

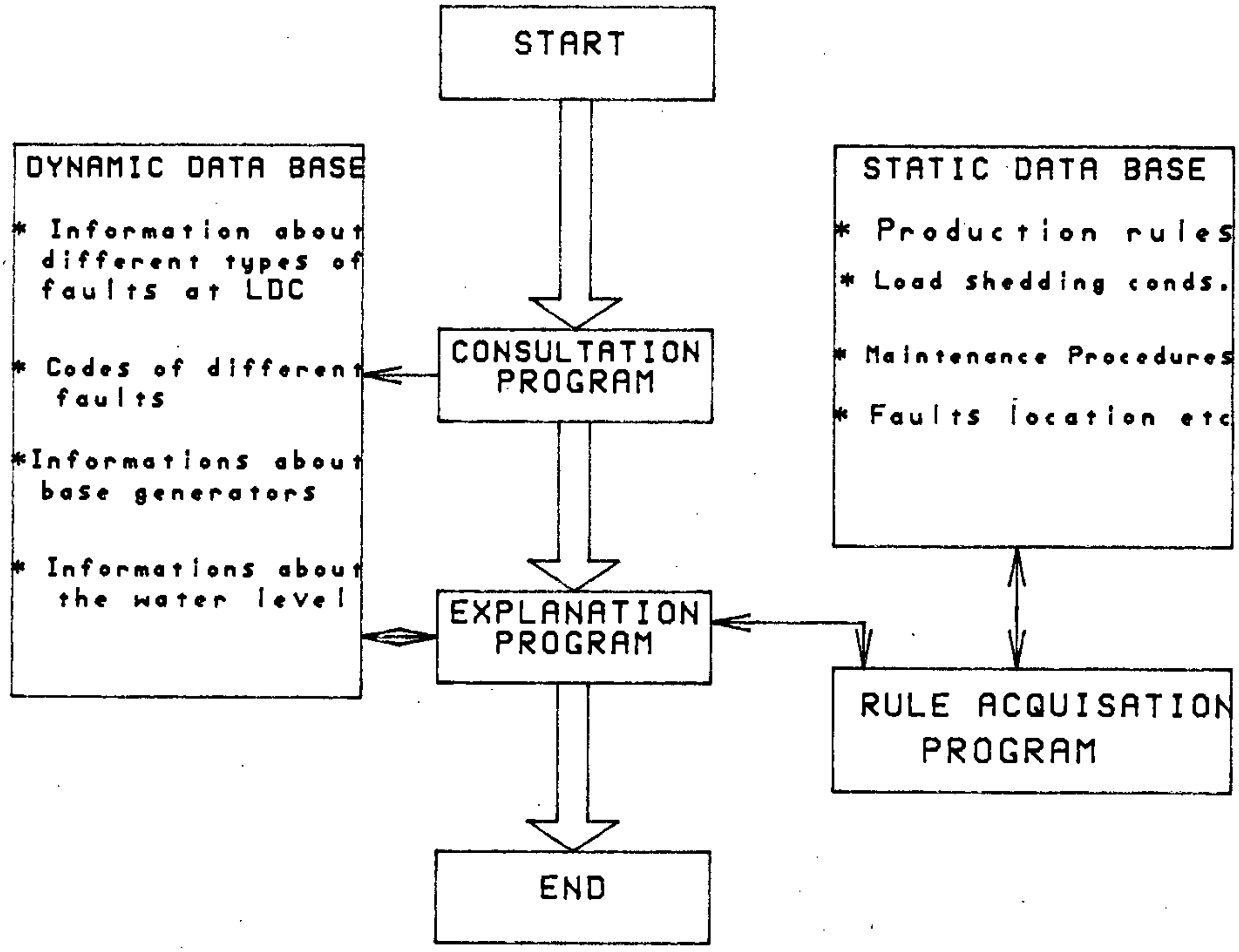


FIG: 30 THE ORGANISATION OF THE LDC EXPERT SYSTEM PROGRAM MODULE

All the knowledge accumulated from system experts are stored in a series of if then rules which are typically of the form

IF<CONDITION>, THEN <ACTION>.

The reasoning process of the expert system involves matching and unification. The system assumes that there is a problem and works towards solving the problem by 'triggering rules' if their activation pattern matches with the data element in the dynamic data base.

d) CHOOSING AN ALGORITHM: Although PROLOG is declarative, we may need to choose an algorithm or more accurately a decision tree to be used to search through the problem space. PROLOG by itself uses depth-first search technique. The algorithms used are details discussed in the next article.

e) DEVELOPING THE PROGRAM: The final step is to write the code of the program in PROLOG. To do so we have to first convert the gathered knowledge into PROLOG rules and facts. The program is divided into two main sections. The first section which is the heart of the expert system contains question asking, grid faults, production rules & explanation module.

The second part of the program use turtle graphics facility of turbo PROLOG to display the location of the faults. Finally it require to consider efficiency, storage requirement, runtime, etc. of the program.

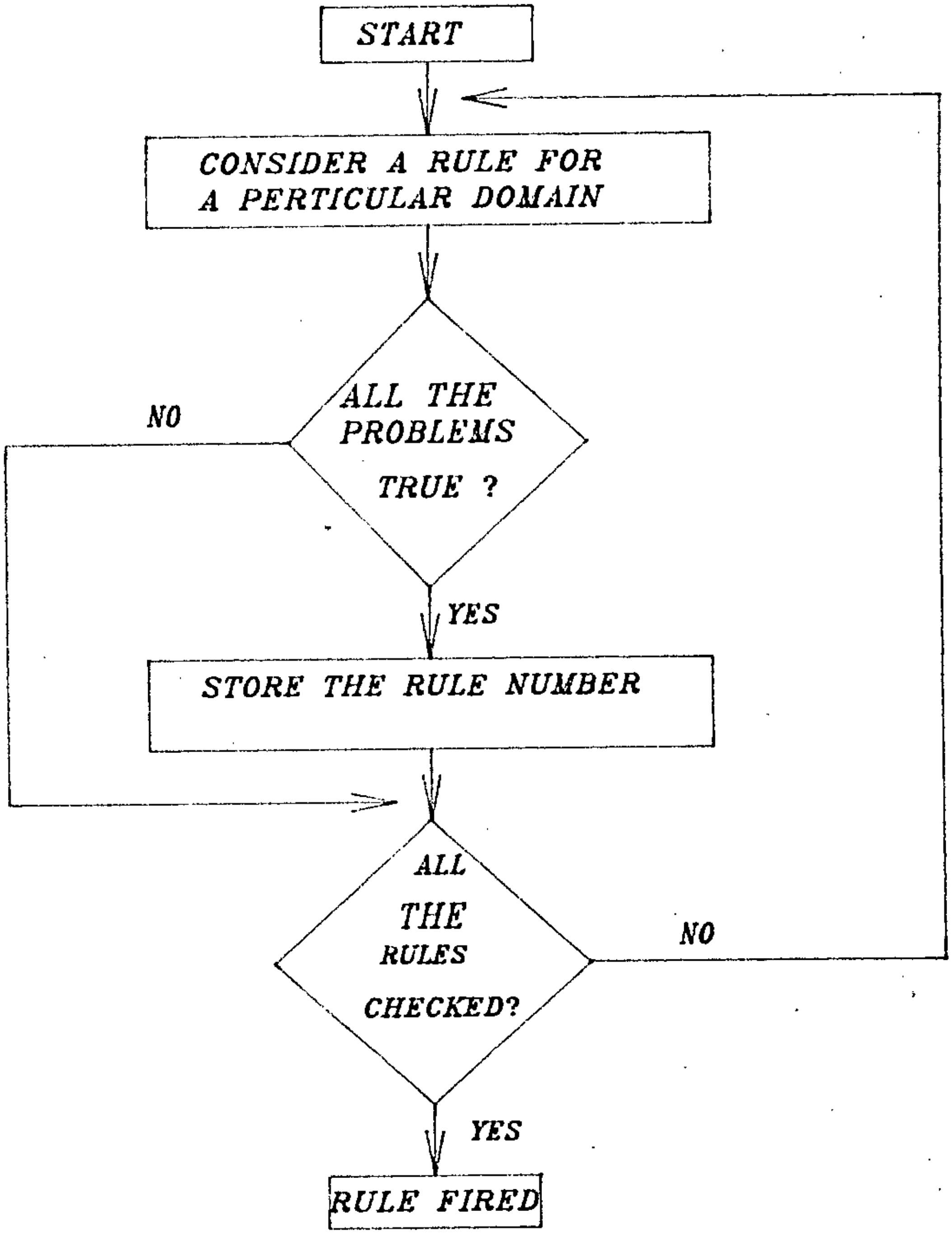


FIG: 34 THE RULE FIRING MECHANISM OF EXPERT SYSTEM PROGRAM heads, one is for the temporary informations called dynamic data base another for permanent informations called static data base. The temporary informations are accumulated during consultation and can be altered during program execution.

The permanent informations includes the generators status of different power stations, its MW capacity, electrical and mechanical conditions etc. Also the informations of different power lines, the conditions of the equipments connected with the power lines. As well as the sub-station positions of national electrical grid system.

d) FAULT LOCATION MODULE:

The "Fault location module" uses the line and turtle graphics facility of turbo PROLOG to locate the faulty generator, breaker and lines of the national electrical grid. This module is helpful to see the location, place and number of faulty component so that the operator can convey more specific instructions. This module is meant to be used on a monitor having CGA or EGA adapter card.

5.5 DECISION TREES FOR DIFFERENT FAULTS AT LDC:

a) DECISION TREES FOR GENERATION FAULTS:

This logical approach assist the programmer to select which generator should make on at the time of faults. The generator selection depends upon different factors

1) Per unit cost of production

2) Place of fault occur

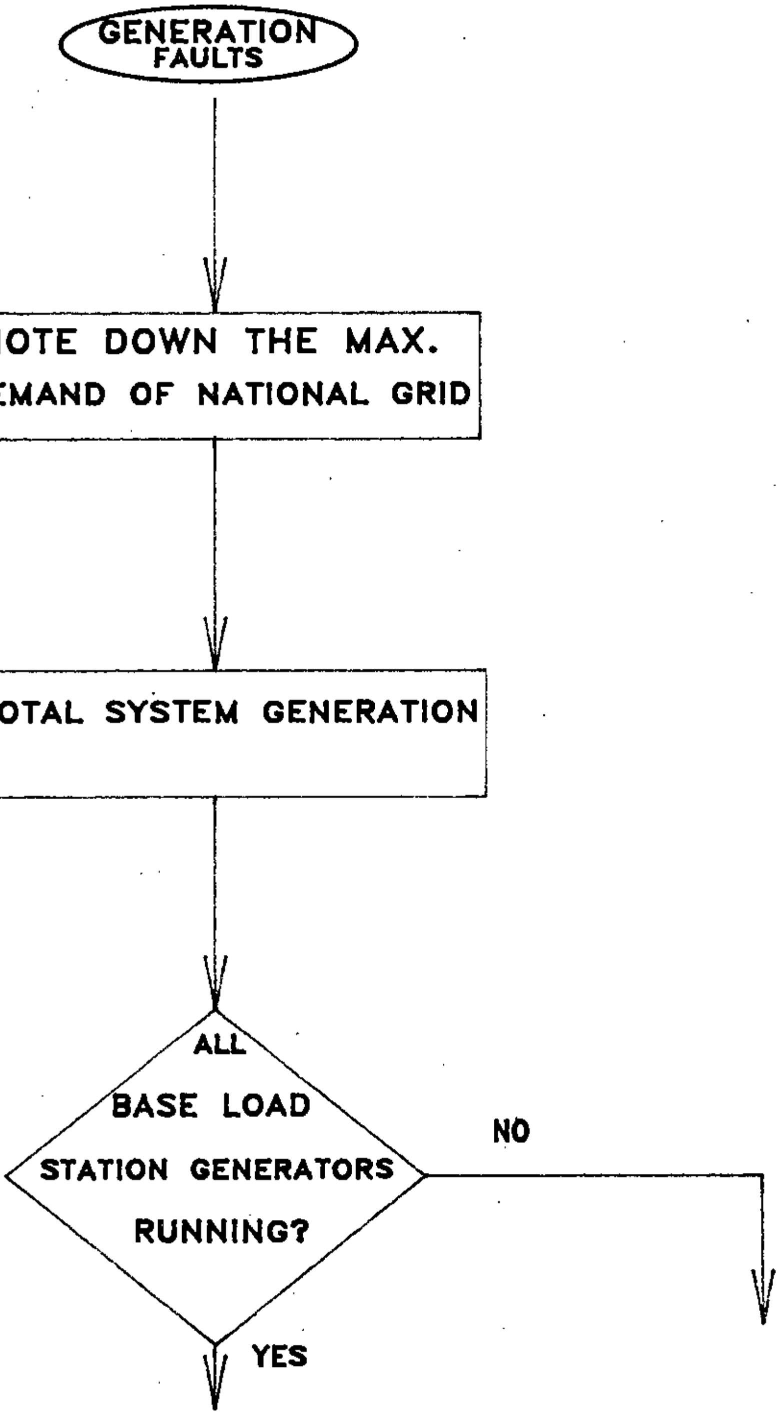
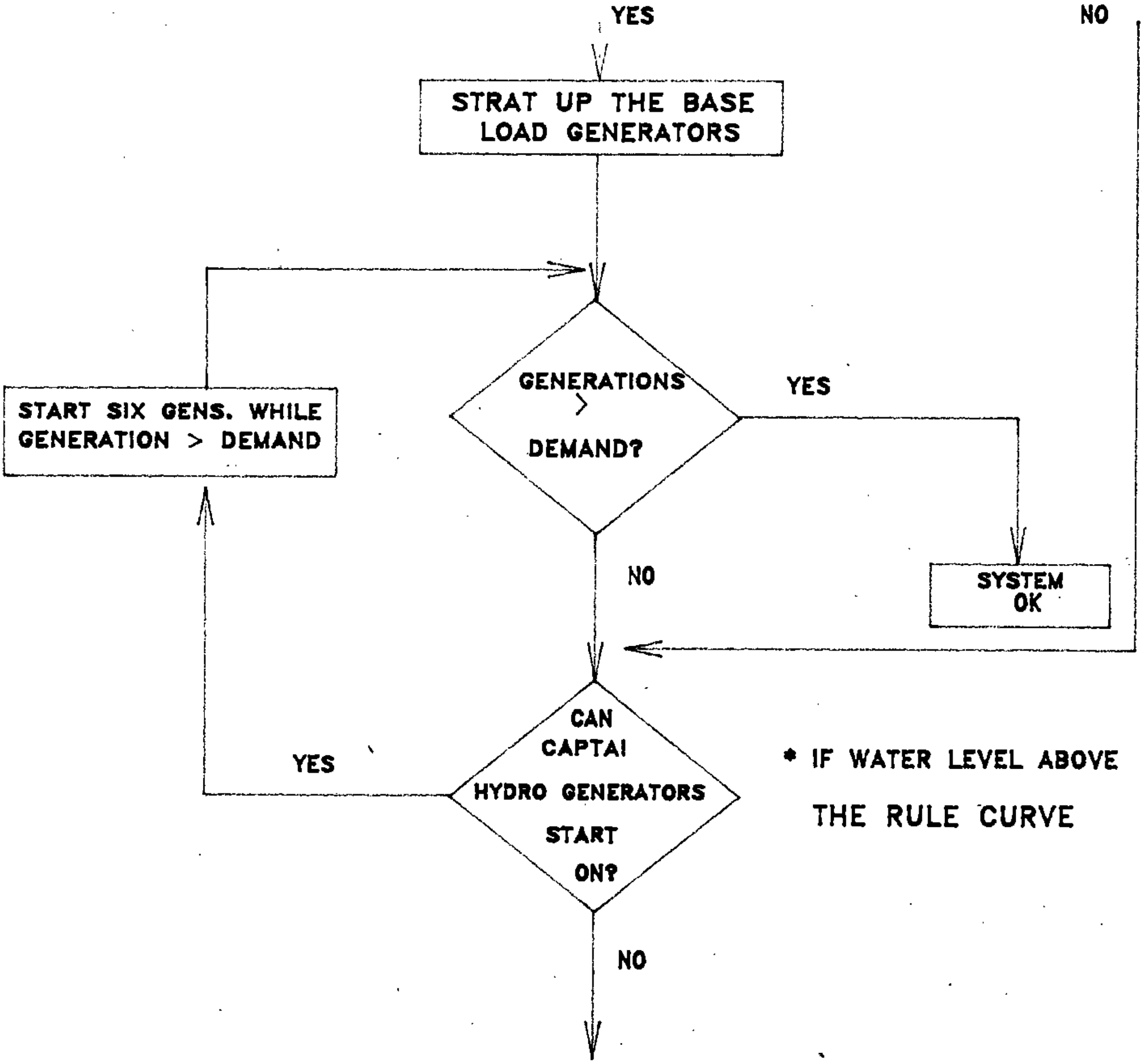
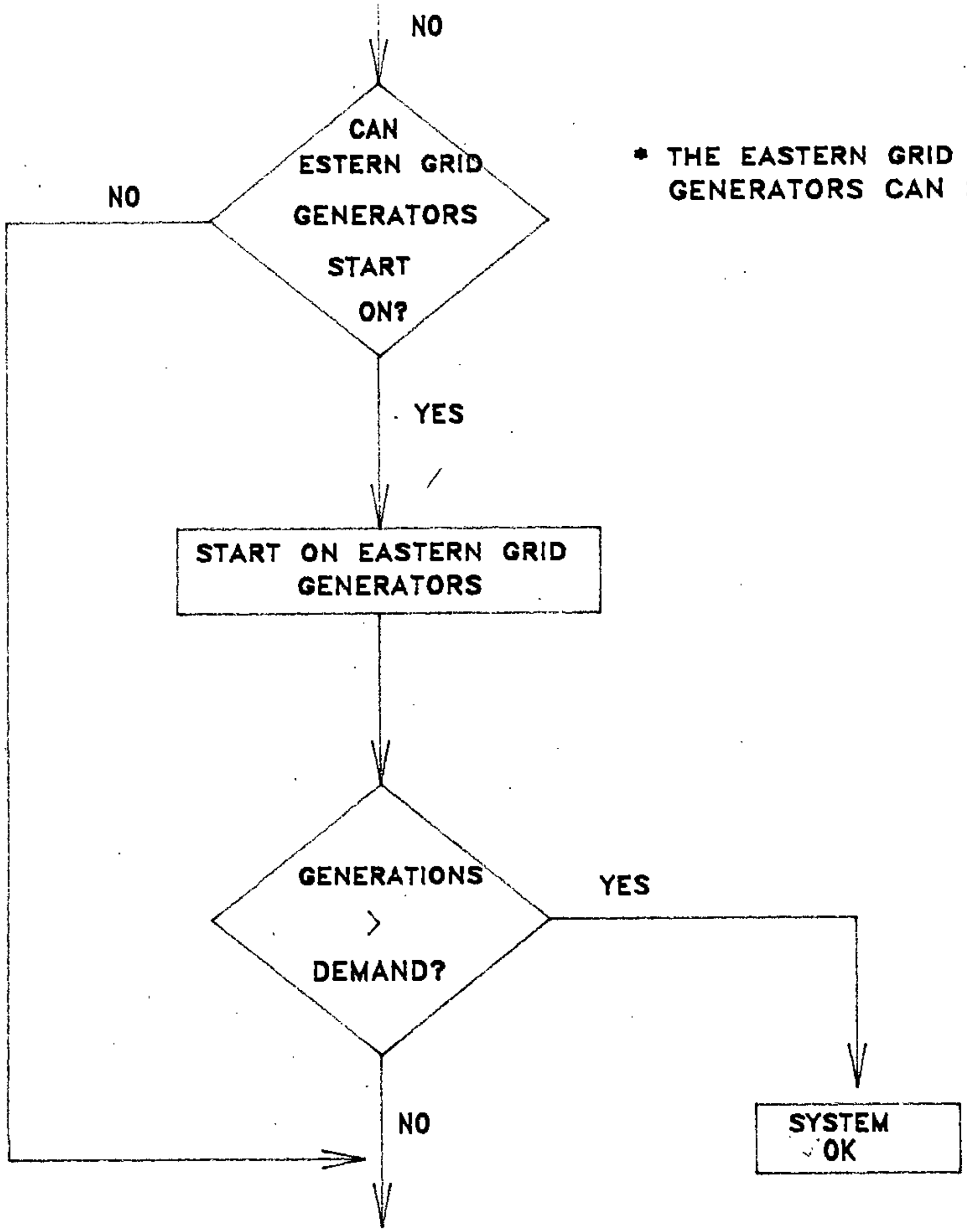

GENERATION FAULTS
NOTE DOWN THE MAX. DEMAND OF NATIONAL GRID
TOTAL SYSTEM GENERATION
ALL BASE LOAD STATION GENERATORS RUNNING?
NO
YES

FIG: 32 THE DECISION TREE FOR GENERATION FAULTS


YES
NO
STRAT UP THE BASE LOAD GENERATORS
GENERATIONS > DEMAND?
YES
START SIX GENS. WHILE GENERATION > DEMAND
NO
SYSTEM OK
CAN CAPTAI HYDRO GENERATORS START ON?
YES
NO

* IF WATER LEVEL ABOVE THE RULE CURVE

FIG: 32 (CONTINUED)


NO
CAN ESTERN GRID GENERATORS START ON?
NO
YES
START ON EASTERN GRID GENERATORS
GENERATIONS > DEMAND?
YES
NO
SYSTEM OK

* THE EASTERN GRID PHERIPHERIAL GENERATORS CAN STARTED ON?

FIG: 32 (CONTINUED)

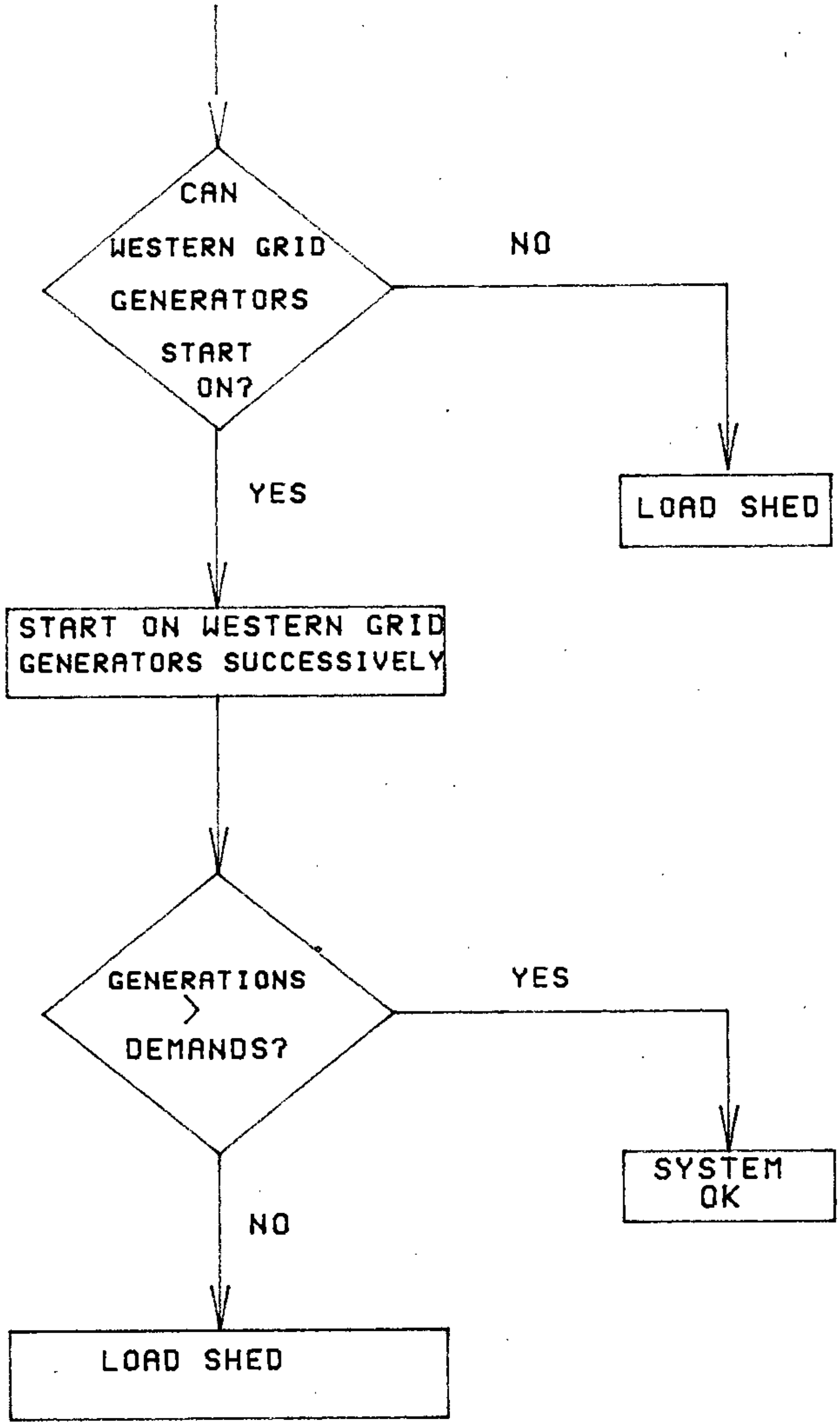
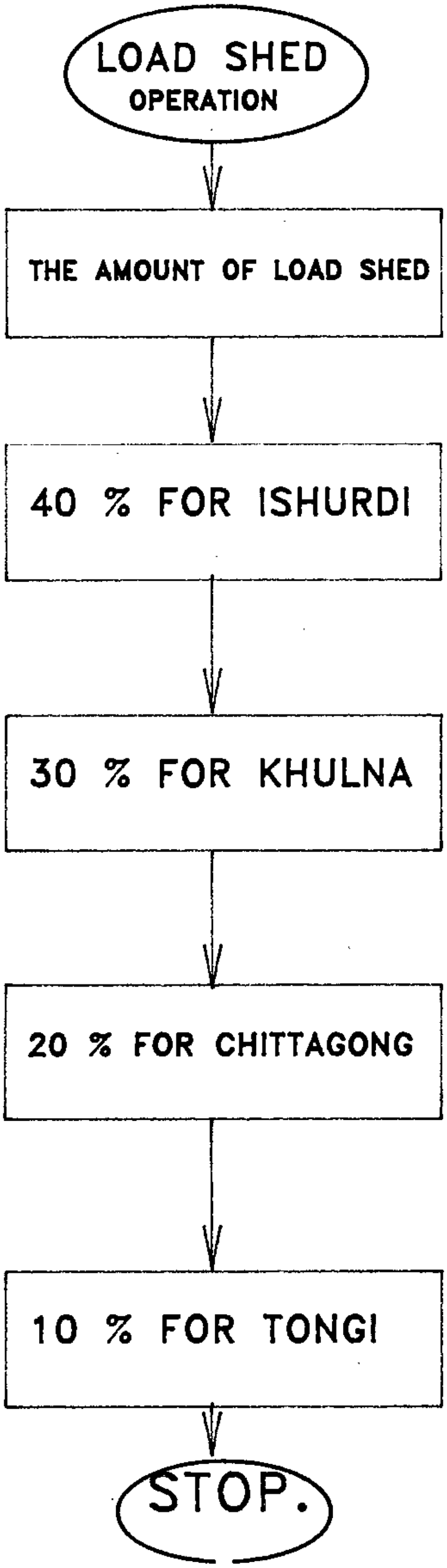

CAN WESTERN GRID GENERATORS START ON?
NO
LOAD SHED
YES
START ON WESTERN GRID GENERATORS SUCCESSIVELY
GENERATIONS > DEMANDS?
YES
SYSTEM OK
NO
LOAD SHED

FIG: 92 (CONTINUED)



THIS PRIORITY OF LOAD SHED AND ITS AMMOUNT DECIDED BY THE BOARD OF MINISTERS

3) Amount of demand

4) Electrical & Mechanical positions of gens

On the basis of the expert´s idea and the above factors the decision tree will first proceed to check the condition of the Kaptai generators then broadly Eastern grid generators and lastly the Western grid generators. The approach will proceed according to the priority of the generators as selected in the data base.

For each and every generator on it goes subroutine to the demand conditions of national grid. At the moment when the generation is greater then demand the tree gives suggestion with blinking text. Otherwise the tree goes to load shedding routing.

The priority of load shedding also selected by the authority concerned. The approach is to also proceeds according to that very priority. The 40% of total load shedding for Ishurdi i,e, for the Northern part of the country which is comparatively less important in both industrial & administrative view point. Then 30% for Khulna region which is more important in industrial aspect, 20% for Chittagong and lastly the least load shedding for Tongi. Tongi is the distribution head of greater Dhaka. Figure 32 shows the decision tree for generation faults.

b) DECISION TREE FOR MAINTENANCE & BREAKER FAULTS:

For the different reasons, some times the lines of the national grid goes to maintenance. The natural Calamity and the malfunction of the equipment connected with the power line are

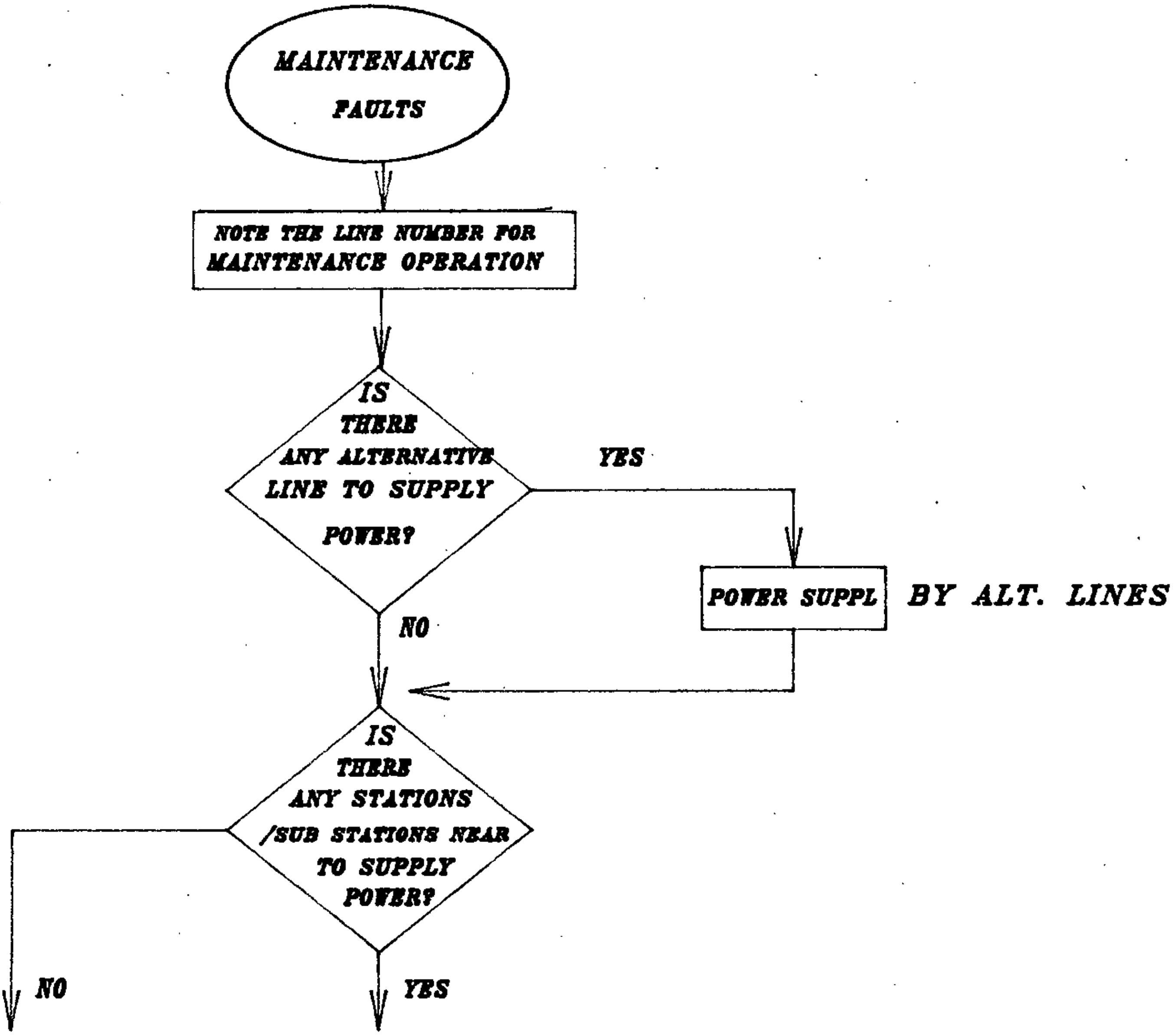

MAINTENANCE FAULTS
NOTE THE LINE NUMBER FOR MAINTENANCE OPERATION
IS THERE ANY ALTERNATIVE LINE TO SUPPLY POWER?
YES
POWER SUPPL BY ALT. LINES
NO
IS THERE ANY STATIONS /SUB STATIONS NEAR TO SUPPLY POWER?
NO
YES

FIG: 33 THE DECISION TREE FOR MAINTENANCE FAULTS

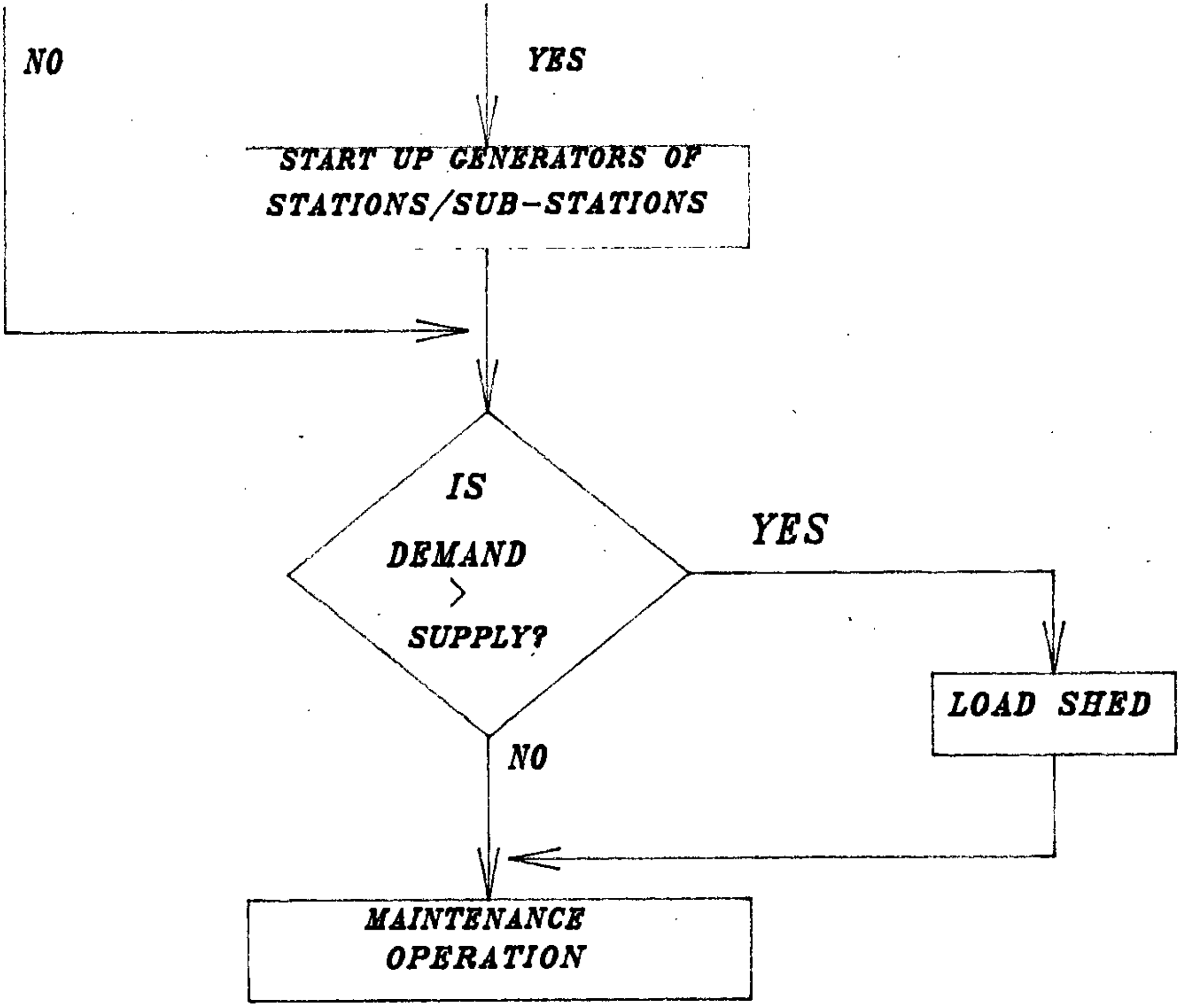
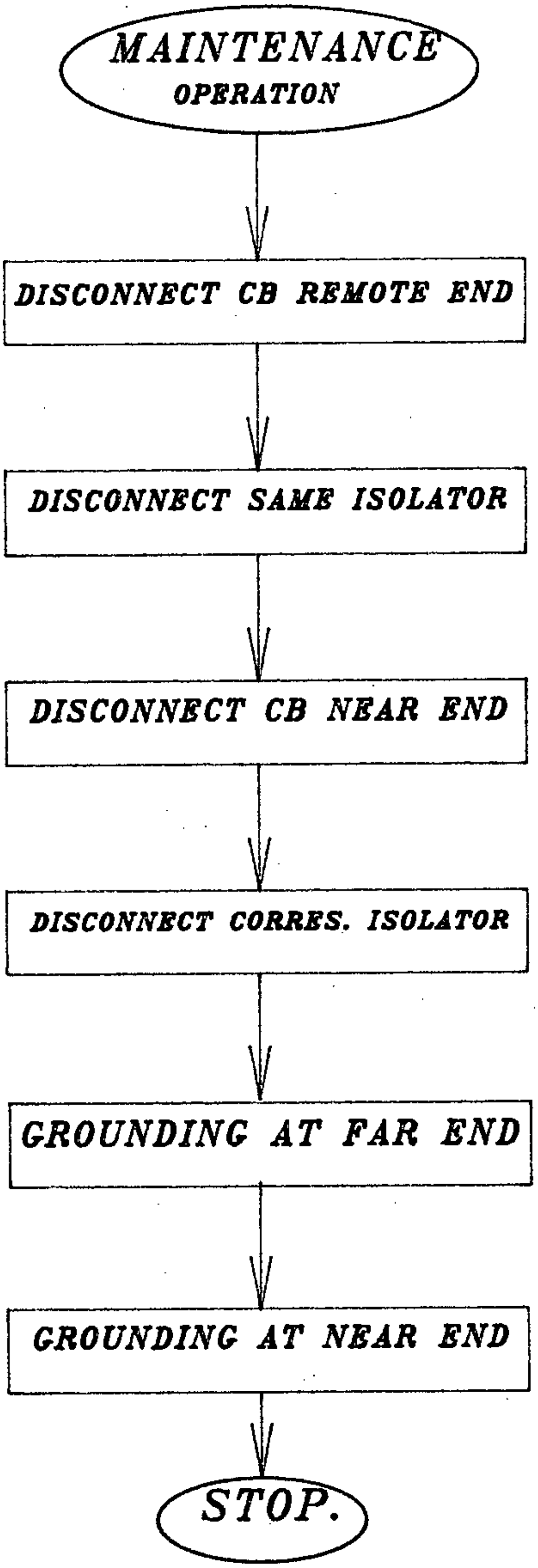


FIG: 33 (CONTINUED)


MAINTENANCE OPERATION
DISCONNECT CB REMOTE END
DISCONNECT SAME ISOLATOR
DISCONNECT CB NEAR END
DISCONNECT CORRES. ISOLATOR
GROUNDING AT FAR END
GROUNDING AT NEAR END
STOP.

mainly responsible for maintenance operation. The logical approach before going to maintenance operation first is to consider alternative lines to face the regional power demand, obviously the rating of the alternate line must not exceed the limit.

Then the decision tree goes for searching, that is, is there any possibility to feed power near by stations or sub-stations. Then it goes to a subroutine to line up a generator if possible. Aftér adding the sub-stations or stations contribution it check whether this additional supplies greater than regional demand, if so then the lines concerned goes maintenance operation without load shedding otherwise maintenance operation happens with load shedding.

For the maintenance operation the approach first gives necessary instruction of remote sub-station with the specific number of the breaker, Isolator, grounding point, then the same for the closer sub-station. Figire 33 & 34 summarizes the abuve actions of maintenance and sub-stations faults.

c) MODULE FOR TRANSFORMER & TELEMETER BOARD FAULTS:

These two modules are identical in nature in programming viewpoint, we uses the technique of almost the diagonestic types of expert system. In this programming module it first asks some questions with the yes/no options, by the response of the user it create a dynamic database. By the process of unification the informations stored are matched with the static data base and goes to a conclusion with the name of the faults and reasons behind it.

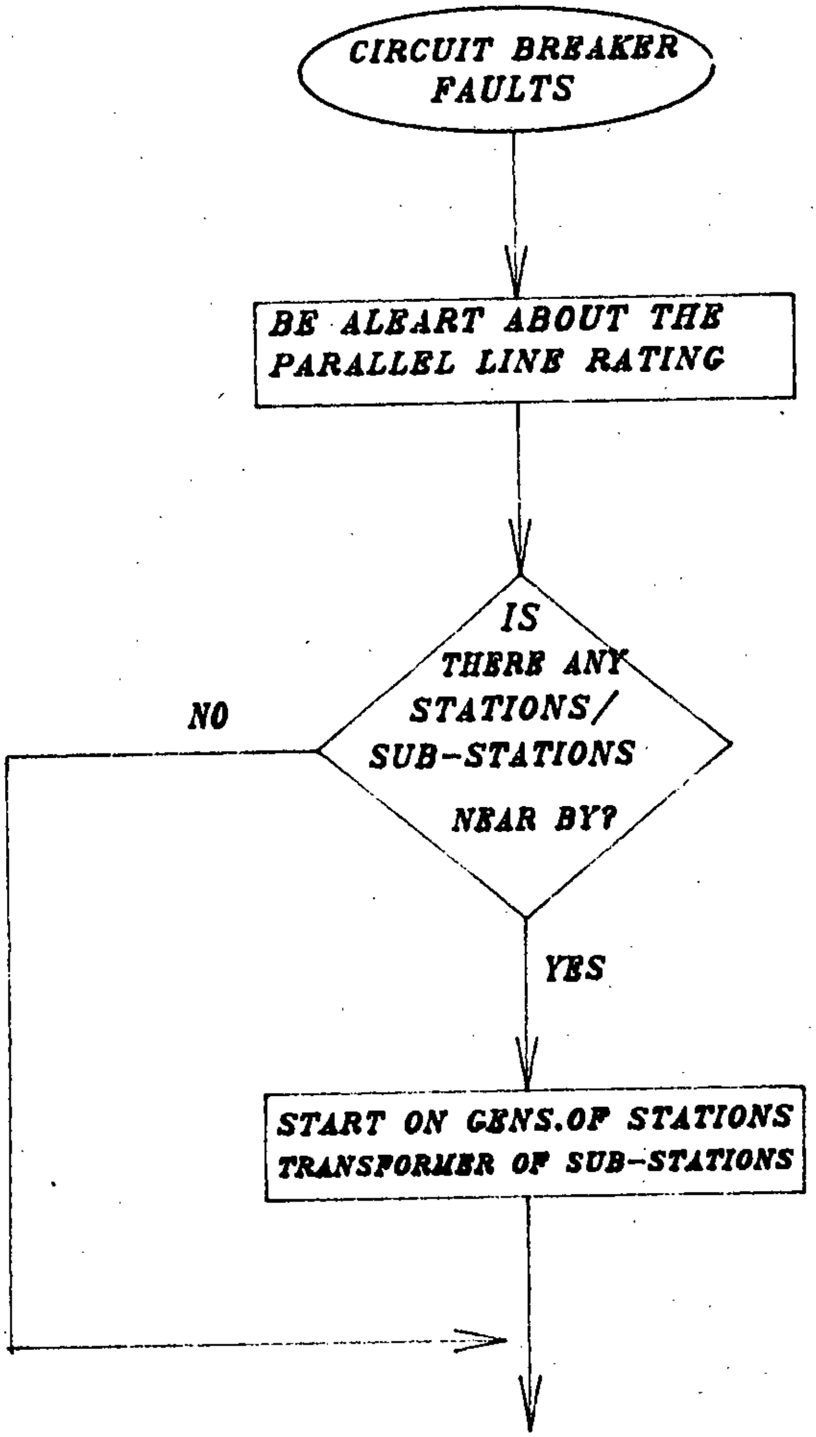

CIRCUIT BREAKER FAULTS
BE ALEART ABOUT THE PARALLEL LINE RATING
IS THERE ANY STATIONS/ SUB-STATIONS NEAR BY?
NO
YES
START ON GENS.OF STATIONS TRANSFORMER OF SUB-STATIONS

FIG:34 THE DECISION TREE FOR SUB-STSTION FAULTS

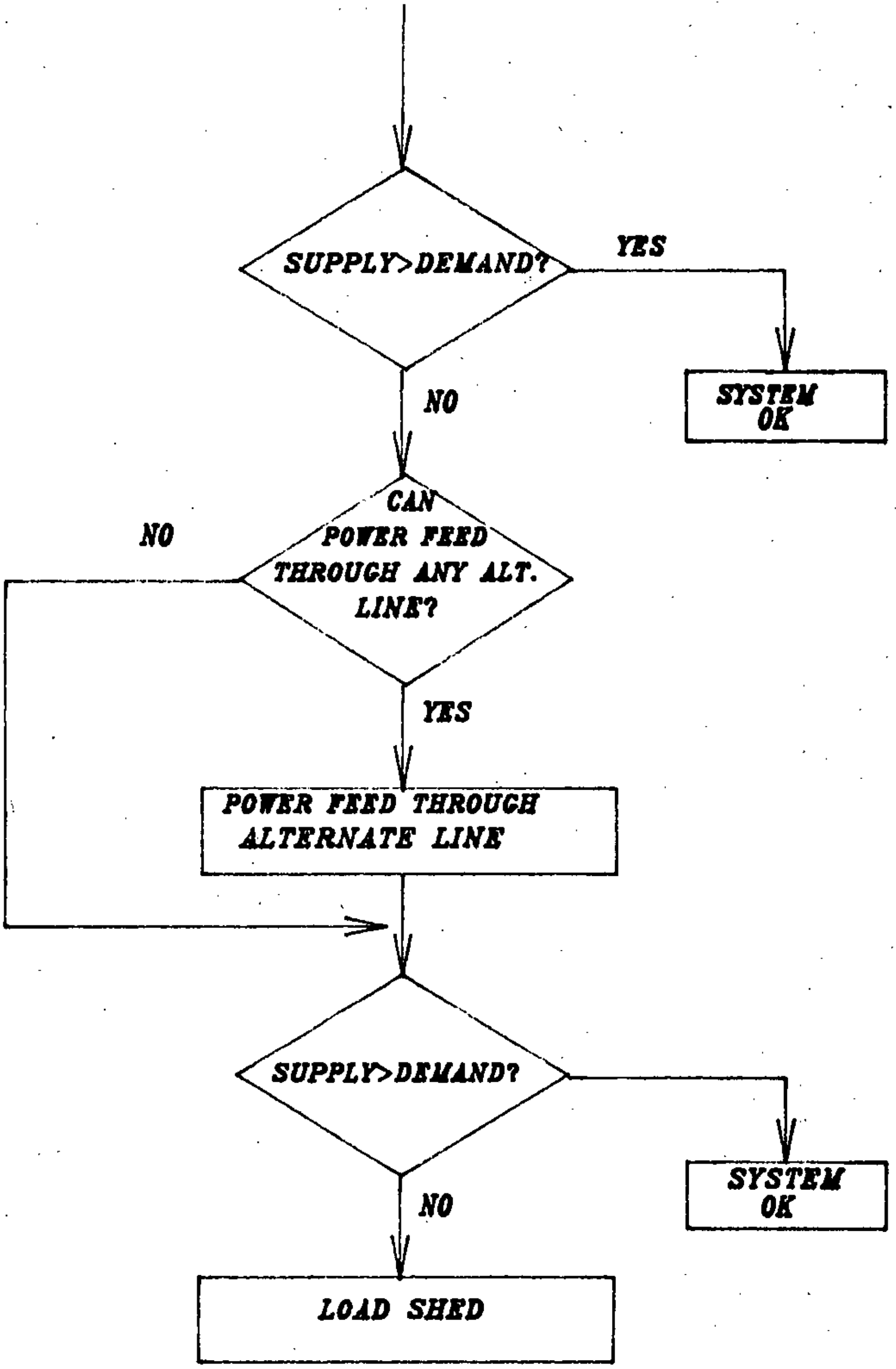
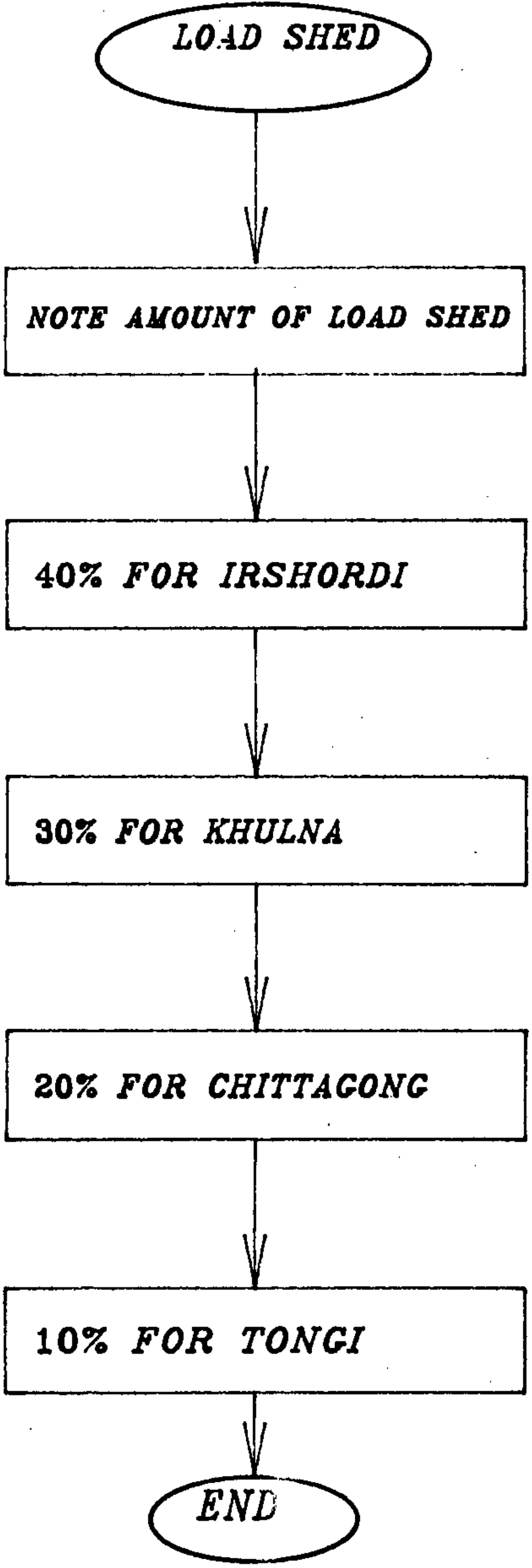

SUPPLY>DEMAND?
YES
SYSTEM OK
NO
CAN POWER FEED THROUGH ANY ALT. LINE?
NO
YES
POWER FEED THROUGH ALTERNATE LINE
SUPPLY>DEMAND?
SYSTEM OK
NO
LOAD SHED

FIG: 34 (CONTINUED)


LOAD SHED
NOTE AMOUNT OF LOAD SHED
40% FOR IRSHORDI
30% FOR KHULNA
20% FOR CHITTAGONG
10% FOR TONGI
END

CHAPTER—6

DISCUSSION OF RESULTS AND CONCLUSIONS

6.1 DISCUSSION OF RESULTS:

a) The Expert system developed for the central load dispatch center with this auspicious hope that it will be an useful tools for the national electrical grid supply specially to take an expertise and most accurate decision at the time of electrical crisis or faults so that the grid supply is reliable and most economical. The occurrence of faults in the interconnected grid system is very common phenomenon but when the faults occurs, consequently by the consultating with this expert system it will prompt a textual proficient suggestion by applying heuristic rules of knowledge base just like an expert gives efficient and smooth suggestions by dint of his experience and training. But an additional outstanding feature of the developed system is that it can exhibits the location of the faults along with the single line diagram, so that the operator can convey specific suggestions to the concern sub-stations or stations. Moreover unlike human expert the artificial expert system never faded or seriously affected due to over use.

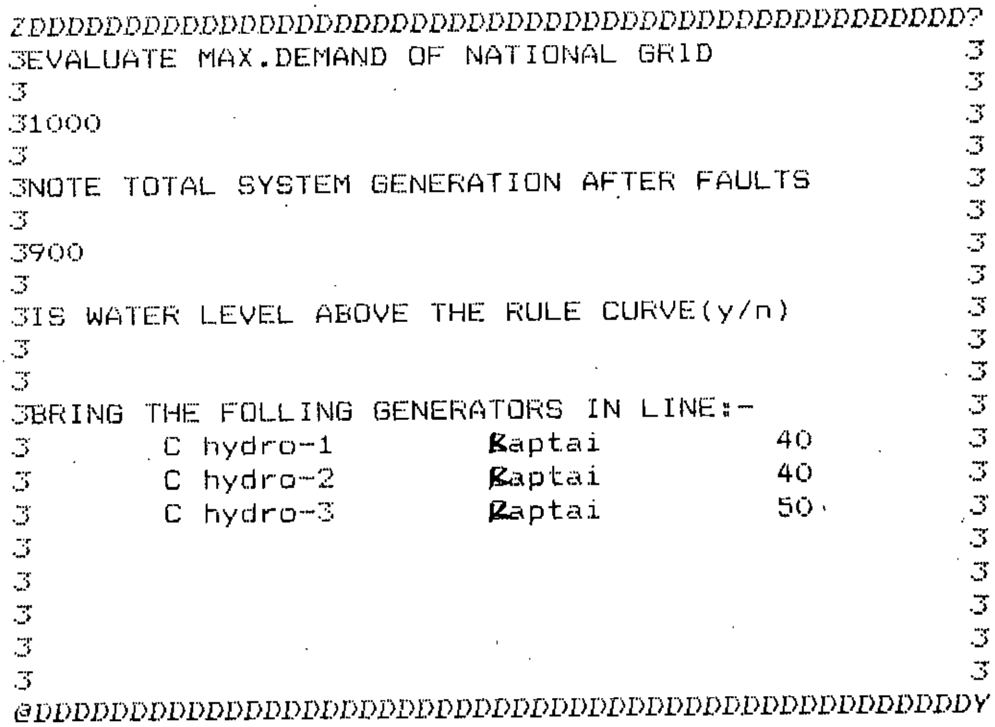
b) To make the system more friendly specially for the new operators, the expert system have the option to show the single line diagram of national electrical grid of both the eastern and western region, so that a perfect idea about the different stations and sub-stations, their location in the telemeter board as well as the generator, breaker and isolator numbers etc. With out this idea the operator can not convey proper instructions to the faulty regions. So the proposed expert system is also a helpful training kit for the users.

c) The most useful feature of the developed expert system is it can offer a high level expertise in problem solving. This expertise can represent the best thinking of the top experts in this field and leading to the fault solutions most accurate and efficient .

d) An important by product of the developed expert system is that the accumulation and codification of the knowledge that uses for the reliable supply of national electrical grid. So long there were no such systematic approach in this field. Moreover the knowledge can be increased by adding heuristic rules obtained from the domain expert. this corpus of knowledge that defines the proficiency of an expert system can also provide an additional feature as an institutional memory. This compilation of knowledge will be a consensus of high level option and a permanent record of the best strategies and methods used by the system staff.

6.2 RESULT VERIFICATION:

The textual suggestions given by the developed expert system for a faults are most accurate than that of the necessary steps taken by the system expert at the time of crisis. The power of searching and analyzing status, grid lines data which are essential factors for fault analysis



```
ZDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD?
3EVALUATE MAX.DEMAND OF NATIONAL GRID                        3
3                                                            3
31000                                                        3
3                                                            3
3NOTE TOTAL SYSTEM GENERATION AFTER FAULTS                   3
3                                                            3
3900                                                         3
3                                                            3
3IS WATER LEVEL ABOVE THE RULE CURVE(y/n)                    3
3                                                            3
3                                                            3
3BRING THE FOLLING GENERATORS IN LINE:-                      3
3       C hydro-1          Kaptai          40                3
3       C hydro-2          Kaptai          40                3
3       C hydro-3          Kaptai          50                3
3                                                            3
3                                                            3
3                                                            3
3                                                            3
3                                                            3
@DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDY
```

FIG: 35 A DISPLAY OF EXPERT SYSTEM RESULT.

can made more exactly and firstly from the given informations. As for an example, from the roster book a fault occur on the 25th March, 1988. The fault was that the ASHUGONJ'S two base generators were overhauling condition. The system expert suggested to line up the hydro generators of KAPTAI. The same suggestion were given by the expert system also shown in figure 35.

6.3 LIMITATIONS OF THE PRESENT WORK:

Building an expert system requires a major investment of time, money, energy and faith. If the problem is appropriate and if adequate resources are committed the process will be accelerated more rapidly. But even the smoothest development effort will have rough spots. Some of this unavoidable limitations of the developed expert system are discussed below:

a) The domain experts are not familiar with computers and doubts that an approach using computers will be of any use for the LDC. That's why they can not spare much time for the project moreover there was a whispered in the center that if a computerized method introduced in the LDC many persons will lose the jobs. It was a great pitfall in extracting high quality knowledge from the system experts.

b) Self knowledge means that the expert system can analyze its own operations and thus understand and explain how and why it reaches particular conclusions. The expert system developed in PROLOG doesn't provide built-in-explanation facilities only having the auxiliary display window to show the chains of rules to reach a conclusions. After the system performs well attempts were made to add mechanisms for allowing the system to explain and justify its operations, but it was difficult to develop and also lest understood.

c) Choosing the correct tools for building the expert system is one of the most difficult decision to make in expert system development. It was decided to use PROLOG but ultimately it was found that the tools was inappropriate for some specific task of grid system. PROLOG compiler ver 1.1 has limited codes to compile and has very narrow graphical facilities. Unfortunately this increased the unaceptable length of the program.

6.4 SCOPE OF FUTURE WORK:

The expert system developed as a part of the thesis is only a skeleton structure of what could be in the future. By no means it can be considered as a self-sustained commercial valuable software package rather it is a research prototype package. However there is a wide scope for expansion of the system. Hope that this research work will lead a guide line to future expert system design projects undertaken in the undergraduate/ graduate level in our University. A brief outline of possible scopes of expansion of the existing expert system is enumerated below:

a) In terms of sophistication and utility, the development stage of an expert system can classify into four dimensions, firstly begins as a demonstration prototype that is a small, demonstration program that handles a portion of the problem that will eventually be addressed. A typical rule-based demonstration prototype might contain 5 to 20 rules and performs adequately one or two test cases. The second or most current expert systems have evolved to the stage of research prototype. These system are medium to large size program that have been revised through testing on real problems in the user community. They are moderately reliable, contain smooth friendly interface and address the needs of the end users. A typical rule-based research/field prototype contain 100 to 500 rules, performs very well on many tests and take two to three years to develop. Our work has reached upto this stage. In future with some further work the developed system can be turned into production and commercial model by adding some field tests making the system more fast, efficient, reliable in the user environment.

b) Computer hardware advances have made possible to develop an integrated expert system called the intelligent system that is the expert system embedded in microprocessor chip to form an integrated hardware/software package. One example this is the EEG analysis system, an expert system embedded in Motorola MC6801 single chip eight bit microcomputer and designed to interpret electroencephalograms recorded for renal patients. Likely an intelligent system can be developed for assisting and correcting faults of LDC by introducing concept of single piece equipments.

c) A natural language interface could be developed which will permit the user to communicate with the expert system during the consultation process using as human language.

APPENDIX-A

EXPERT SYSTEM PROGRAM.

```
/********EXPERT SYSTEM PROGRAM START******/

code=4024


domains

        faults , indication = symbol
        query = string
        reply = char

database

        off_generator ( string, string )
        gen_dem(integer,integer)

        dem_rat (integer,integer)
        xline (integer )
        supp_dem (integer,integer,integer)

        xpositive(indication)
        xnegative(indication)

predicates

        go
        new_gen_dem(integer,integer)
        next_generator (integer,integer,integer)
        check_Captai(integer,integer)
        generator (integer,string,string,integer)
        add_hydro (integer,integer,integer)
        hydro_generator (integer,string,string,integer)
        clear_datas.

        breaker (integer,integer,integer)
        problem (string,integer)
        parallel (integer,integer)
        rating (integer,integer)
        check_rating (integer,integer,integer,integer)
        start_generator (integer,integer,integer,integer)
        regional_generator (integer,string,integer,integer)
        check_generator(integer,integer,integer)
        new_supp_dem(integer,integer,integer)

        check_operation (integer,integer)
        power_line (integer,integer,integer,string,integer)
        disconnect_operation (string,integer)
        operation(string,integer)
        power_feed (integer)

        clear_facts
        graph
        form_block (integer,integer,integer,integer,integer)
        block1 (integer,integer,integer,integer)
        gr(integer)
        gen(integer,integer,integer)
        tran(integer,integer,integer)
        lin(integer,integer,integer)
        start1
        start2
        start3
        start4
        go1

        start
        choice(char)
        problem1(faults)
        indication(indication)
        positive(query,indication)
        clear_fact
        remember(indication,reply)
        ask(query,indication,reply)

        western
        star
        boundary
        trin(integer,integer,integer,integer,integer)
        blok(integer,integer,integer,integer,integer,integer)
        form_trin(integer,integer,integer,integer,integer)
        arrow(integer,integer,integer)
        star1
        star2
        thaku
        golp
        barisal
        faridpur
        bager
        jhen
        inter

        graphs
        form_block (integer,integer,integer,integer,integer)
        block1 (integer,integer,integer,integer)*/
        draw1
        draw2
        draw3
        triangle

         load1
         load2
         load3
         clock(integer,integer)
         block(integer,integer)
         milco(integer,integer)
         ground(integer,integer)


goal
        go.
```

```
clauses


/*      PROGRAM  TO  CREAT  MENU   */


go:-
        makewindow(1,0,0,"",0,0,25,80),
        makewindow(1,87,106,"EXPERT SYSTEM",3,15,15,50),
        write("\t\t    FOR LDC\n\t\t    *******"),
        nl,nl,nl,nl,nl,write("\t WHAT TYPE OF PROBLEM ?")nl,nl,
        nl,nl,write(" PRESS THE SPACE BAR FOR THE OPTION MENU"),
        readchar(_),
        makewindow(2,71,106,"OPTION   MENU",7,39,18,41),
        cursor(3,0),
        write("******************************************"),nl,
        write("  GENERATOR PROBLEM - TYPE 'g'"),nl,nl,
        write("  MAINTENANCE PROBLEM - TYPE 'm'"),nl,nl,
        write("  STATION PROBLEM - TYPE 's'"),nl,nl,
        write("  TELIMETER BOARD PROBLEM - TYPE 't'"),nl,nl,
        write("******************************************"),
        nl,nl,write("TYPE THE FIRST CHARACTER OF YOUR CHOICE"),
        readchar(X),
        choice(X),
        start.


choice(X):-

        X = 'g', !,
        makewindow(3,71,106,"GENERATOR  PROBLEM", 12,0,13,32),
        cursor(0,6),
        write("==================="),
        nl,write(" 01 - CAPTAI FAULTS "),nl,nl,
        write(" 02 - W.G.M. FAULTS "),nl,nl,
        write("    Enter A Number "),
        makewindow(3,71,106,"",21,23,3,6),
        cursor(0,1),
        write(" 03 - E.G. SUB-STATION FAULTS"),nl,nl,nl,nl,
        readchar(_).


choice(X):-
        X = 'm',!,clearwindow,
        makewindow(3,71,106,"MAINTENANCE  PROBLEM", 5,20,18,50),
        nl,nl,write(" 04 - TURBINE MAINTENANCE "),nl,nl,
        write(" 05 - GENERATOR MAINTENANCE"),nl,nl,
        write("    Enter A Number "),
        makewindow(3,71,106,"",21,23,3,6),
        cursor(0,1),
        write(" 06 - MOTOR MAINTENANCE"),nl,nl,
        readchar(_).


choice(X):-
        X = 's', !, clearwindow,
        makewindow(3,71,106,"STATION  PROBLEM", 5,20,18,50),
        nl,nl,write(" 07 - TRANSFORMER FAULTS "),nl,nl,
        write(" 08 - GROUND FAULTS "),nl,nl,
        write(" 09 - E.G. SUB-STATION FAULTS"),nl,nl,
        write("    Enter A Number "),
        makewindow(3,71,106,"",21,23,3,6),
        cursor(0,1),
        readchar(_).


choice(X):-

        X = 't', !, clearwindow,
        makewindow(3,71,106,"TELIMETER BOARD PROBLEM", 5,20,18,50),
        nl,nl,write(" 07 - TRANSFORMER FAULTS "),nl,nl,
        write(" 08 - GROUND FAULTS "),nl,nl,
        write(" 09 - E.G. SUB-STATION FAULTS"),nl,nl,
        write("    Enter A Number "),
        makewindow(3,71,106,"",21,23,3,6),
        cursor(0,1),
        readchar(_).


choice(_):-

        makewindow(3,71,106,"",5,20,18,50),nl,nl,nl,
        write("SORRY,YOU HAVE PRESSED THE WRONG KEY"),nl,nl,
        write("PRESS THE SPACE BAR TO START AGAIN"),nl,
        write("------------------------------------"),
        readchar(_),go.

/* PROGRAM TO CREATE EXPLANATION MENU    */

clauses
        go:-
        makewindow(1,23,2,"OPTION MENU",3,3,8,50),
        write("* Breaker Fault; Code: 01."),nl,nl,
        write("* Generetion Fault; Code: 02."),nl,nl,
        write("Enter your option code(01/02) : "),
        readint(X),clearwindow,removewindow,checki(X).

        check1(X):-  X=01,
        makewindow(2,141,2,"BREAKER FAULT",3,3,8,35),
        write("No Breaker Fault, SYstem OK."),nl,nl,nl,
        write("Press SPACE bar to return."),readchar(_),
        clearwindow,removewindow,!.

        checki(X):-   X=02,
        makewindow(3,23,2,"GENERETION FAULT",3,3,5,60),
        write("Is  the water level above rule curve(y/n)?.."),
        readchar(Ch),clearwindow,removewindow,check2(Ch).

        check1(X):-  X<>01,
        makewindow(2,23,2,"",3,3,4,50),
        write("Your Code is (01/02); Try again..."),readint(Y),
        clearwindow,removewindow,checki(Y).
```

```
          check2(Ch):-  Ch='n',
          makewindow(2,141,2,"GENERETION FAULT",3,3,8,55),
          write("Try to open Gus Turbine, OK."),nl,nl,nl,
          write("*** Please,Press SPACE bar to return. ***"),readchar(_),
          clearwindow,removewindow,!.

          check2(Ch):-  Ch='y',
          makewindow(2,23,2,"KAPTAI",3,3,8,55),
          write("Is the base Generetor at KAPTAI running?"),nl,nl,
          write("Press (y/n)......."),readchar(Reply),
          clearwindow,removewindow,check(Reply).

          check2(Ch):-  Ch<>'y',
          makewindow(2,23,2,"",3,3,7,50),
          write("You Pressed a wrong Key."),nl,
          write("Please press (y/n); Thanks..."),readchar(C),
          clearwindow,removewindow,check2(C).

          check(Reply):-   Reply='n',
          makewindow(2,141,2,"KAPTAI",3,3,8,55),
          write("Start Base Generetors, OK."),nl,nl,nl,
          write("*** Please,Press SPACE bar to return. ***"),readchar(_),
          clearwindow,removewindow,!.

          check(Reply):-   Reply='y',
          makewindow(2,141,2,"",3,3,8,55),
          write("SYSTEM, OK."),nl,
          write("You may start extra Generetors."),nl,nl,nl,
          write("*** Please,Press SPACE bar to return. ***"),readchar(_),
          clearwindow,removewindow,!.

          check(Reply):-   Reply<>'n',
          makewindow(2,23,2,"",3,3,7,50),
          write("You pressed a wrong Key."),nl,
          write("Please press (y/n); Thanks..."),readchar(Rep),
          clearwindow,removewindow,check(Rep).


          /*      EASTERN GRID START    */


    start:-


          makewindow(1,31,2,"OPTION MENU",10,10,10,40),
          write("Do you want to see the eastern gride ?"),nl,
          write("Then press the code 01"),nl,
          readint(Reply),
          Reply= 01,
          graphics(2,1,2),
          penpos(0,0,0),
          line(0,0,31999,0,7),
          line(31999,0,31999,31999,7),
          line(31999,31999,0,31999,7),
          line(0,31999,0,16000,7),
          line(0,0,0,12000,7),
          cursor(0,30),write("WELCOME TO "),
          cursor(1,28),write("EASTERN GRIDE"),
          start1,
          start2,
          start3,
          start4,
          go1.


start1:-

          cursor(1,2),write("Kabirpur"),
          lin(2500,2250,800),
          tran(1,2250,1600),
          line(2200,800,12000,800,7),
          line(2800,3000,2800,6000,7),
          line(2800,3000,2200,3000,7),
          line(2800,6000,2200,6000,7),
          cursor(1,15),
          write("Tangail"),
          lin(3000,2400,5800),
          tran(2,2400,7000),
          cursor(1,65),write("Shahjibazar"),
          lin(10000,2400,21500),
          tran(1,2400,22800),
          tran(3,2400,26800),
          gen(2,4400,27200),
          lin(2500,4000,28800),
          tran(1,4000,30300),
          gen(1,5700,30700),
          line(1500,20000,6500,20000,7),
          line(1500,20000,1500,22000,7),
          line(1500,22000,2200,22000,7),
          line(700,19000,6500,19000,7),
          line(700,19000,700,23000,7),
          line(700,23000,2200,23000,7),
          cursor(4,15),write("Ghorasal"),

          lin(9000,6500,2000),
          tran(2,6500,5400),
          gen(2,8000,5800),
          tran(2,6500,9000),
          cursor(4,35),write("Ashuganj"),
          lin(11000,6500,11400),
          tran(5,6500,12200),
          gen(5,8000,12600),
          line(6000,10500,6000,12000,7),
          line(6000,10500,6500,10500,7),
          line(6000,12000,6500,12000,7),
          line(5000,10000,5000,12500,7),
          line(5000,12500,6500,12500,7),
          line(5000,10000,6500,10000,7),
          tran(2,6500,20000).
```

```
start2:-

        cursor(5,62),write("Sreemongal"),
        lin(5000,8000,23000),
        tran(1,8000,25000),
        line(8000,23700,2400,23700,7),
        line(8000,27000,21800,27000,7),
        cursor(8,8),write("Tongi"),
        lin(4000,12000,500),
        tran(2,12000,4000),
        line(12000,2500,6500,2500,7),
        line(12000,2900,6500,2900,7),
        cursor(8,36),write("33 KV"),
        lin(3500,11500,13500),
        tran(2,11500,14300),line(11500,13500,12000,13500,7),
        line(14700,15400,12500,15400,7),
        cursor(7,45),write("Khishorganj"),
        lin(4500,10000,17500),
        tran(1,10000,19500),
        line(6500,18000,19000,18000,7),
        cursor(11,57),write("Fenchuganj"),
        lin(3000,15500,24000),
        tran(1,15500,25000).


start3:-

        cursor(11,8),write("Ullon"),
        lin(3000,15500,2800),tran(2,15500,3000),
        line(15000,5700,15500,5700,7),
        line(15000,5700,15000,6700,7),

        line(15500,2700,12000,2700,7),
        cursor(9,23),write("Sidderganj"),lin(9000,13000,7300),
        tran(1,13000,9000),gen(1,15000,9400),
        tran(2,13000,10500),
        lin(3300,14700,10000),gen(3,15200,10800),
        line(6500,8000,24500,8000,7),
        line(6500,7800,24500,7800,7),
        lin(2500,14700,14000),gen(1,15200,15400),
        line(15000,13000,15000,14700,7),
        line(14700,13000,15000,13000,7),
        line(14700,14700,15000,14700,7),
        cursor(13,60),write("Sylhet"),tran(1,18300,25000),
        lin(3000,18300,24000),
        cursor(16,59),write("Chattak"),lin(3000,21800,24000),
        tran(1,21800,26000),
        cursor(10,46),write("Mymensingh"),
        lin(4500,14500,17500),tran(2,14500,18500),
        cursor(13,46),write("Jamalpur"),
        lin(3500,18000,18000),tran(1,18000,18500),
        cursor(14,2),write("Mirpur"),lin(2500,19000,150),
        tran(2,19000,1000),
        cursor(13,21),write("Comilla"),lin(4000,18000,7800),
        tran(2,18000,8500),line(18000,11000,20500,11000,7),
          cursor(16,21),write("Feni"),lin(2000,22000,7800),
          tran(1,22000,8500).


start4:-

          cursor(15,29),write("Chandpur"),lin(3000,20500,11000),
          tran(2,20500,11500),
          cursor(16,5),write("postagola"),
          lin(3800,21800,3000),tran(2,21800,4400),
          line(21800,6700,15000,6700,7),
          line(22000,200,12500,200,7),
          line(22000,200,22000,3000,7),
          line(12500,200,12500,2500,7),
          line(12500,2500,12000,2500,7),
          cursor(18,12),write("Madanhat"),lin(6000,24500,4500),
          tran(2,24500,7400),line(24500,5300,27500,5300,7),
          line(23500,10000,24500,10000,7),
          line(23500,10000,23500,21000,7),
          line(23500,21000,25000,21000,7),
          line(23800,10500,24500,10500,7),
          line(23800,10500,23800,12200,7),
          line(23800,12800,23800,20500,7),
          line(23800,20500,25000,20500,7),
          line(23800,12200,25000,12200,7),

          line(23800,12800,25000,12800,7),
          cursor(19,3),write("Kulshi"),lin(3500,25500,2000),
          tran(2,25500,2500),line(25500,4000,28500,4000,7),
          cursor(20,15),write("Halishahar"),
          lin(4500,27500,4500),
          tran(2,27500,5000),
          line(27500,9000,29500,9000,7),
          line(29500,9000,29500,9200,7),
          line(29500,9200,29000,9200,7),
          cursor(21,29),write("Sikalbahar"),
          lin(4800,29000,9200),tran(3,29000,11000),
          gen(1,30500,11400),
          lin(2500,29000,4500),tran(2,29000,5000),
          gen(2,30500,5400),
          line(29000,7000,30000,7000,7),
          line(29000,10000,24500,10000,7),
          line(29000,9500,25700,9500,7),
          line(25700,9500,25700,10000,7),
          cursor(21,1),write("Baroaulia"),
          lin(2500,28500,1500),tran(2,28500,2000),
          cursor(19,36),write("Chandraghona"),
          lin(2500,25000,12000),tran(2,25000,12500),
          cursor(18,54),write("Kaptai"),lin(5000,25000,20000),
          tran(3,25000,20500),gen(3,26500,21000),
          cursor(22,54),write("Dohazari"),
          lin(2500,29200,20000),tran(1,29200,21000),
          line(29700,20500,29700,13500,7),
          line(29700,20500,29200,20500,7),
          line(29200,13500,29700,13500,7),
          line(27000,26000,31999,26000,7),
        line(27000,26000,27000,31999,7),
        cursor(22,68),write("GENERATOR"),gen(1,28000,27000),
        cursor(23,68),write("TRANSFORMER"),tran(1,29500,26500),
        readchar(_).


gen(0,_,_):-!.

gen(N,X,Y):-

        NN=N-1,
        penpos(X,Y,0),
        forward(700),right(90),forward(700),right(90),
        forward(700),right(90),forward(700),
        X1=X,Y1=Y-350,
        penpos(X1,Y1,0),back(500),
        XX=X+0,YY=Y+1000,
        gen(NN,XX,YY).


tran(0,_,_):-!.

tran(M,G,H):-

        MM=M-1,
        penpos(G,H,30),
        forward(700),right(120),forward(700),right(120),forward(700),
        J=G+400,
        penpos(J,H,30),
        forward(700),right(120),forward(700),right(120),forward(700),
        R1=J+700,C1=H,R2=J+1000,C2=H,
        line(R1,C1,R2,C2,7),
        GG=G+0,
        HH=H+1000,
        tran(MM,GG,HH).


lin(L,F,K):-

        penpos(F,K,90),
        forward(L).


/*              WESTERN GRID START      */


clauses

        western:-

        write("do you want to display the western grid?(y/n)"),
        readchar(Reply),
        Reply='y',
        star.
```

```
star:-

        graphics(4,0,4),
        boundary,
        start1,
        start2,
        inter,
        thaku,
        golp,
        barisal,
        faridpur,
        bager,
        jhen.

boundary:-

        line(1,1,1,31999,9),
        line(1,1,31999,1,9),
        line(31999,1,31999,31999,9),
        line(1,31999,31999,31999,9),
        cursor(0,25),
        write("WELCOME TO WESTERN GRID"),
        trin(2,28500,27500,0,600),
        cursor(22,70),
        write("= Trans"),
        blok(1,29700,27000,90,0,500),
        cursor(23,70),
        write("= Gen").

star1:-

        cursor(2,20),
        write("pabna"),
        cursor(2,65),
        write("ULLAPARA"),
        line(4500,7000,4500,11000,9),
        line(4500,25000,4500,29000,9),
        line(4500,7500,6000,7500,9),
        line(4500,9000,6000,9000,9),
        line(4500,10500,6000,10500,9),
        line(4500,25500,6000,25500,9),
        line(4500,27000,6000,27000,9),
        line(4500,28500,7000,28500,9),
        line(6000,10500,6000,25500,9),
        line(6000,7500,6000,5000,9),
        line(7000,28000,7000,29000,9),
        line(7000,28500,8000,28500,9),
        trin(2,8000,28500,0,1000),
        line(9000,28500,9500,28500,9),
        arrow(10000,28500,90),
        line(9500,27000,9500,29500,9),
        line(9500,27500,10000,27500,9),
        trin(2,5500,27000,0,600),
        line(6000,27000,6500,27000,9),
        arrow(6500,27000,90),
        line(9500,28500,10000,28500,9),
        blok(1,10000,27000,90,0,1000),
        trin(2,6000,9000,0,1000),
        line(7000,9000,8000,9000,9),
        arrow(8000,9000,90).

star2:-

            /*   RAJSHAHI        */

        line(6000,5000,10000,5000,9),
        line(10000,3500,10000,6500,9),
        line(10000,700,10000,2000,9),
        line(10000,1300,11000,1300,9),
        trin(2,11000,1300,0,1000),
        line(12000,1300,13000,1300,9),
        line(10000,1300,8000,1300,9),
        line(10000,4000,8000,4000,9),
        line(8000,1300,8000,4000,9),
        cursor(5,3),
        write("Rajshai"),
        cursor(7,16),
        write("Ishurdi"),
        line(10000,4000,11000,4000,9),
        line(10000,6000,11000,6000,9),
        form_trin(2,2,11000,4000,0,2000,1000),
        line(12000,4000,28000,4000,9),
        line(12000,6000,28000,6000,9),
        line(13000,3000,13000,23000,9).

inter:-

        /*interconector*/
        line(13000,22000,14700,22000,9),
        line(13000,22300,14200,22300,9),
        line(14700,22000,14700,25000,9),
        line(14200,22300,14200,25000,9),
        cursor(9,50),
        write("Ishurdi"),
        cursor(11,60),
        write("to eastern grid"),
        line(13000,500,13000,2500,9),
        line(13000,700,14000,700,9),
        arrow(14000,700,90),
        line(13000,2000,14000,2000,9),
        line(13000,3200,14000,3200,9),
        line(13000,19000,16000,19000,9),
        line(13000,20000,16000,20000,9),
        line(16000,18000,16000,22000,9),
        line(16000,21500,29000,21500,9),
        line(16000,22000,21000,22000,9),
        line(21000,21500,21000,24000,9),
        trin(2,21500,23500,0,1000),
        line(22500,23500,23500,23500,9),
        line(22500,20000,22500,22500,9),
        trin(2,23000,20500,0,1000),
        line(24000,20500,24500,20500,9),
arrow(24500,20500,90),
line(25000,19000,25000,22500,9),
form_trin(2,2,25500,19000,0,1500,1000),
line(26500,19000,28500,19000,9),
arrow(23500,23500,90),

line(26500,20500,26700,20500,9),
line(26700,20000,26700,21500,9),
trin(2,27200,20700,0,1000),
line(28200,20700,28500,20700,9),
line(28500,17500,28500,20700,9),
line(28500,17500,29000,17500,9),
arrow(29000,17500,90),
line(28500,18500,28700,18500,9),
line(28500,19200,28700,19200,9),
line(28500,19900,28700,19900,9),
blok(3,28700,18250,90,700,500),
line(29000,21500,29000,26000,9),


trin(2,29300,22000,0,600),
line(29900,22000,31000,22000,9),
arrow(31000,22000,90),
form_trin(2,2,16500,18500,0,1200,1000),
line(17500,18500,18000,18500,9),
line(17500,19700,18000,19700,9),
line(18000,18000,18000,20000,9),
trin(2,18500,19000,0,1000),
line(19500,19000,20200,19000,9),
line(20200,18000,20200,20000,9),
line(20200,18000,21000,18000,9),
arrow(21000,18000,90),
line(20200,18700,20700,18700,9),
line(20200,20000,20700,20000,9),
blok(2,20700,18200,90,1300,1000),
blok(1,14000,1500,90,0,1000),
trin(2,14000,3200,0,1000),
line(17000,500,17000,12000,9),
line(17000,700,18000,700,9),
line(17000,2700,18000,2700,9),
line(17000,4700,18000,4700,9),
line(17000,10000,18000,10000,9),
form_trin(3,2,18000,700,0,2000,1000),
trin(2,18000,10000,0,1000),
line(19000,700,20000,700,9),
line(19000,2700,19500,2700,9),
line(19000,4700,19500,4700,9),
line(19000,10000,19500,10000,9),
blok(3,19500,200,90,2000,1000),
line(19500,9000,19500,11000,9),
line(19500,10500,20500,10500,9),
trin(2,20500,10500,0,1000),
line(21500,10500,22000,10500,9),
line(22000,10000,22000,11000,9),
line(22000,10500,22500,10500,9),
arrow(22500,10500,90),
        line(21000,2500,21000,4000,9),
        line(21000,3000,21500,3000,9),
        trin(2,21500,3000,0,1000),
        cursor(12,16),
        write("Bheramara").

thaku:-

        cursor(14,70),
        write("Thakurgaon"),
        cursor(12,40),
        write("Bogra"),
        cursor(17,42),
        write("Rangpur"),

        cursor(15,52),
        write("Palasbari"),
        cursor(19,55),
        write("Saidpur"),
        line(20000,26000,20000,29000,9),
        line(20000,26000,29000,26000,9),
        trin(2,20400,27500,0,600),
        line(21000,27500,21300,27500,9),
        line(21300,26500,21300,28000,9),
        trin(2,21700,27500,0,600),
        line(22300,27500,22700,27500,9),
        line(22700,26500,22700,28500,9),
        line(22700,26500,24000,26500,9),
        arrow(24000,26500,90),
        line(22700,27500,23200,27500,9),
        blok(1,23200,27250,90,0,500).

golp:-

        cursor(21,1),
        write("Goalpara"),
        line(28000,3500,28000,10000,9),
        form_trin(6,2,28400,4000,0,1000,500),
        line(28900,4000,28200,4000,9),
        line(28900,5000,29200,5000,9),
        line(28900,6000,29200,6000,9),
        line(28900,7000,29500,7000,9),
        line(28900,8000,29500,8000,9),
        line(28900,9000,30500,9000,9),
        blok(2,29500,6750,90,1000,500),
        line(30500,8500,30500,10000,9),
        line(30500,8800,31000,8800,9),
        line(30500,9800,31000,9800,9),
        blok(2,31000,8550,90,1000,500),
        line(29200,3500,29200,6300,9),
        form_trin(3,2,29500,4000,0,1000,400),
        line(30000,4000,30300,4000,9),
        line(30000,5000,30300,5000,9),
        line(30000,6000,30700,6000,9),
        blok(2,30300,3700,90,1000,600),
        line(30700,5500,30700,6800,9),
        line(30700,6000,31000,6000,9),
        line(30700,6500,31500,6500,9),
        arrow(31500,6500,90),
        blok(1,31000,5750,90,0,500).

barisal:-

        cursor(14,32),
        write("Barisal"),
        line(20000,12000,20000,17000,9),
        form_trin(3,2,20500,13000,0,1000,600),
        line(21000,13000,21500,13000,9),
        line(21000,14000,21500,14000,9),
        line(21000,15000,21500,15000,9),
        line(21500,12500,21500,16000,9),
        line(21500,13000,21800,13000,9),
        blok(1,21800,12750,90,0,500),
        trin(2,22000,15500,0,600),
        line(22500,15500,22800,15500,9),
        line(22800,14000,22800,16000,9),
        line(22800,14500,23200,14500,9),
        line(22800,15500,23600,15500,9),
        arrow(23600,15500,90),
        blok(1,23200,14250,90,0,500).

faridpur:-

        line(17000,7000,20000,7000,9),
        line(20000,6500,20000,8000,9),
        line(20000,6800,24000,6800,9),
        trin(2,20500,7500,0,600),
        line(24000,6500,24000,8000,9),
        trin(2,24500,7000,0,600),
        line(25000,7000,25500,7000,9),
        arrow(25500,7000,90),
        line(24000,7500,25500,7500,9),
        line(25500,7500,25500,12000,9),
        line(25500,12000,20000,12000,9),
        cursor(18,20),
        write("Madaripur").

bager:-

        cursor(20,32),
        write("Bagerhat"),
        line(27000,9000,27000,12500,9),
        line(27000,9300,26700,9300,9),
        line(26700,9300,26700,8500,9),
        line(26700,8500,28000,8500,9),
        line(27000,11500,29000,11500,9),
        trin(2,27500,12000,0,600),
        line(28000,12000,28500,12000,9),
        arrow(28500,12000,90),
        line(29000,11000,29000,12000,9),
        trin(2,29500,11500,0,600),
        line(30000,11500,30500,11500,9),
        arrow(30500,11500,90),
        line(27000,12200,20000,12200,9).

jhen:-

        line(23000,4000,23000,2000,9),
        form_trin(2,2,23500,2500,0,1000,600),
        line(24000,2500,24300,2500,9),
        line(24000,3500,24300,3500,9),
        readchar(_).
        form_trin(0,_,_,_,_,_,_):-!.
        form_trin(Nc,N,R,C,A,S,L):-
        trin(N,R,C,A,L),
        CC=C+S,
        NNc=Nc-1,
        form_trin(NNc,N,R,CC,A,S,L).


trin(0,_,_,_,_):-!.

trin(N,R,C,A,L):-

        penpos(R,C,A),
        left(30),forward(L),
        right(120),forward(L),
        right(120),forward(L),
        NN=N-1,
        RR=R-L/2,
        trin(NN,RR,C,A,L).

blok(0,_,_,_,_,_):-!.

blok(N,R,C,A,S,L):-

        penpos(R,C,A),
        forward(L),right(90),
        forward(L),right(90),
        forward(L),right(90),
        forward(L),
        NN=N-1,
        CC=C+S,
        blok(NN,R,CC,A,S,L).

arrow(R,C,A):-

        penpos(R,C,A),
        left(60),forward(300),
        penpos(R,C,A),
        left(120),forward(300).


go1:-
        write ("\n which type of problem?"),nl,
        readln (P),nl,
        write ("which category?"),nl,
        readint (C),nl,
        problem (P,C).

problem ("gen",01):-

        write ("Evaluate the max. demand of nation grid."),nl,
        readint (D),nl,
        write ("Note down the total system generation after faults."),nl
,
        readint (G),nl,
        asserta(gen_dem(G,D)),
        check_Captai(G,D).

problem("gen",01):-nl,nl,

        write(" ACTION: LOAD SHEDDING!!"),nl,
        field_attr (16,1,24,135),
        gen_dem (G,D),
        H=D-G,
        write ("Amount of load shead =",H ),nl,
        I=H*0.4,
        write ("load shead first ISHURDI=",I,"MW"),nl,
        K=H*0.3,
        write ("load shead second KHULNA=",K,"MW"),nl,
        C=H*0.2,
        write ("load shead third CHITTAGONG=",C,"MW"),nl,
        T=H*0.1,
        write ("load shead lastly TONGI=",T,"MW"),nl,
        clear_datas.

problem ("st",02):-

        write ("WHICH BREAKER IS FAULTY?"),nl,
        readint (B),nl,
        write ("WHAT IS THE TOTAL SUUPPLY AFTER FAULTS"),nl,
        readint(S),
        write ("WHAT IS THE PRESENT DEMAND?"),nl,
        readint (D),
        breaker (B,L,AC),
        parallel (L,PL),
        rating (PL,R),
        asserta (supp_dem(S,D,AC)),
        check_rating (S,D,R,AC),!.

problem(st,02):-

        write ("SYSTEM GOES LOAD SHEAD"),nl,
        field_attr (16,0,25,135),
        supp_dem(S,D,AC),
        H=D-S,
        write ("AMOUNT OF LOAD SHEAD=",H,"MW"),nl,
        graph.

problem (mt,03) :-

        write ("which line goes to be maintenance"),nl,
        readint (L),nl,
        assertz (xline(L)),
        write ("what is the max. regional demand"),nl,
        readint (D),nl,
        write ("can power supplied through alt.line (y/n)?"),nl,
        readchar (Reply),nl,
        Reply ='y',
        write ("put the rating of alt.line "),nl,
        readint (R),nl,
        asserta (dem_rat (D,R)),
        power_feed (_).

problem(t,04):-

        makewindow(1,23,7,"Telemeter Board",2,5,10,60),
        write(" This is expert system Telemeter Faults"),nl,nl,
        write( " The expert system will ask  about faults."),nl,
        write("In response you press key 'y' or 'n'."),nl,nl,nl,
        write("                PLEASE PRESS THE SPACE BAR."),
        sound(5,5000),
        readchar(_),
        removewindow,
        clearwindow,
        problem1(Faults),!,nl,nl,nl,nl,nl,nl,
        write("**** THE  PROBABLE  FAULTS IS ****  ",Faults),nl,
        clear_fact,
        sound(10,200).


/*             PROGRAM FOR TELIMETER BOARD PROBLEM  */


problem(t,04):-

        sound(1,6000),nl,nl,nl,nl,nl,
        write("*** SORRY , NO KNOWN PROBLEM ****"),nl,
        clear_fact,
        sound(10,6000).

positive(_,Indication):-

        xpositive(Indication),!.

positive(Query,Indication):-

        not(xnegative(Indication)),
        ask(Query,Indication,Reply),
        Reply='y'.

ask(Query,Indication,Reply):-

        write(Query),
        readchar(Reply),
        write(Reply),nl,
        remember(Indication,Reply).

remember(Indication,'y'):-
```

```
        asserta(xpositive(Indication)).

remember(Indication,'n'):-
        asserta(xnegative(Indication)).

clear_fact :-
        retract(xpositive(_)),fail.

clear_fact :-
        retract(xnegative(_)),fail.
        clear_fact.

indication(faulty_display):-
        beep,
        positive("Faulty display  TB (y/n) ?",faulty_display).

indication(weak_blinking_display):-
        beep,
        positive("Weak_blink_displayTB ?",weak_blinking_disply).

indication(dark_line_display):-
        beep,
        positive("Dark_line_ of TB (y/n) ?",dark_line_display).

indication(partial_or_total_darkness) :-
        beep,
        positive("Par/totolly dark TB ?",partial_total_darkness).

indication(beep_sound):-
        beep,
        positive("Beep sound of TB (y/n) ?",beep_sound).

indication(no_meter_reading):-
        beep,
        positive("No meter reading  TB (y/n) ?",no_meter_reading).

indication(voltmeter_no_reading):-
        beep,
        positive("No voltmeter  TB (y/n)?",voltmeter_no_reading).

indication(varmeter_no_reading):-
        beep,
        positive("No varmeter TB (y/n) ?",varmeter_no_readimg).

indication(frequencymeter_no_reading):-
        beep,
        positive(" No freqmeterTB (y/n)?",frequencymeter_no_readimg).

indication(faulty_reading):-
        beep,
        positive("False meter  TB (y/n) ?",faulty_reading).

indication(totally_darkness):-
        beep,
        positive("Totally dark of TB (y/n) ?",totally_darkness).
```

```
indication(displaying_faults):-
        beep,
        positive("No display in TB (y/n) ?",displaying_faults).

indication(instrument_fault):-
        beep,
        positive("No reading of meters TB (y/n)?",instrument_fault).

indication(fuse_off):-
        beep,
        positive("Fuse of TB OFF (y/n) ?",fuse_off).

problem1(carrier_signal_weak):-

        indication(faulty_display),
        indication(weak_blinking_display),
        indication(dark_line_display).

problem1(display_circuit_faults):-

        indication(partial_or_total_darkness),
        indication(beep_sound),
        indication(no_meter_reading).

problem1(instrument_faults):-

        indication(voltmeter_no_reading),
        indication(varmeter_no_reading),
        indication(frequencymeter_no_reading),
        indication(faulty_reading).

problem1(power_supply_faults):-

        indication(totally_darkness),
        indication(displaying_faults),
        indication(instrument_fault).

problem1(fuse_faults):-
        indication(fuse_off).

check_Captai(G,D):-

        write ("Is water level above the rule curve?"),nl,
        readchar (Reply),
        Reply ='y',
        add_hydro (G,D,N),!.

check_Captai(_,_):-

        gen_dem(G,D),
        next_generator(G,D,N).

new_gen_dem(G,D):-

        retract(gen_dem(_,_)),
        asserta(gen_dem(G,D)),!.
```

```
add_hydro (G,D,_):-
        G>D,!.

add_hydro(G,D,N):-

        hydro_generator (N,X,Y,Z),!,
        NN=N+1,
        GG=G+Z,
        write ("        ",X,"          ",Y,"          ",Z),nl,
        new_gen_dem(G,D),
        add_hydro (GG,D,NN).

next_generator (G,D,N):-

        G>D,!.

next_generator (G,D,N):-

        generator (N,X,Y,Z),!,
        NN= N+1,
        write ("        ",X,"          ",Y,"          ",Z),nl,
        G1=G+Z,
        new_gen_dem(G1,D),
        next_generator (G1,D,NN).

clear_datas:-
        retract (gen_dem (_,_)),fail.
        clear_datas.

        hydro_generator (12,"C hdro-1","Captai",40).
        hydro_generator (13,"C hydro-2","Captai",40).
        hydro_generator (14,"C hydro-3","Captai",50).
        hydro_generator (15,"C hydro-4","captai",50).
        hydro_generator (16,"C hydro-5","Captai",50).

        generator (1,"A steam-1", "Ashugong", 64).
        generator (2,"A steam-2", "Ashugong",65 ).
        generator (3,"A steam-3", "Ashugong",150).
        generator (4,"A stem-4", "Ashugong",150).
        generator (5,"A stem-5", "Ashugong",150).
        generator (6,"A gas-1", "Ashugong",50).
        generator (7,"A gas-2",  "Ashugong",30).
        generator (8,"G steam-1","Ghorashal",55).
        generator (9,"G steam-2","Ghorashal",65).
        generator (10,"G steam-3","GHorashal",210).
        generator (11,"G steam-4","Ghorashal",220).


/*          GRAPHICS  FOR  GENERATION  FAULT      */


graph:-

        graphics (2,0,6),
        pencolour (3),
        pendown,
        draw1,
        draw2,
        draw3.

draw1:-

        cursor (1,35),
        write ("KHULNA SUB-STATION"),
        cursor (2,35),
        write ("------------------"),
        line (3000,3000,3000,12000,7),
        line (3000,25000,3000,30000,7),
        line (6000,3000,6000,7000,7),
        line (6000,19000,6000,30000,7),
        line (8000,7000,8000,7500,7),
        line (8000,7500,7500,8000,7),
        line (8000,8000,8000,9000,7),
        line (7750,9000,8250,9000,7),
        line (7850,9250,8150,9250,7),
        line (7950,9500,8050,9500,7),
        line (8000,12000,8000,12500,7),
        line (8000,12500,7500,13000,7),
        line (8000,13000,8000,14000,7),
        line (7750,14000,8250,14000,7),
        line (7850,14250,8150,14250,7),

        line (7950,14500,8050,14500,7),
        line (8000,19000,8000,19500,7),
        line (8000,19500,7500,20000,7),
        line (8000,20000,8000,21000,7),
        line (7750,21000,8250,21000,7),
        line (7850,21250,8150,21250,7),
        line (7950,21500,8050,21500,7),
        line (8000,25000,8000,25500,7),
        line (8000,25500,7500,26000,7),
        line (8000,26000,8000,27000,7),
        line (7750,27000,8250,27000,7),
        line (7850,27250,8150,27250,7),
        line (7950,27500,8050,27500,7),
        line (6000,7000,11500,7000,7),
        line (3000,12000,11500,12000,7),
        line (6000,19000,11500,19000,7),
        line (3000,25000,11500,25000,7),
        line (10500,7000,10500,10000,7),
        line (10500,12000,10500,15000,7),
        line (10500,19000,10500,22000,7),
        line (10500,25000,10500,28000,7),
        line (10500,10000,11500,10000,7),
        line (12500,10000,14000,10000,7),
        line (15000,10000,16500,10000,7),
        line (17500,10000,21500,10000,7),
        line (22500,10000,24000,10000,7),
        line (25000,10000,26000,10000,7),
        line (27000,10000,29000,10000,7),
        line (28000,10000,28000,7000,7),
        line (12500,7000,21500,7000,7),
        line (22500,7000,28000,7000,7),
        line (12500,10000,11500,10500,7),
        line (16500,10000,17500,10500,7),
        line (22500,10000,21500,10500,7),
        line (26000,10000,27000,10500,7),
        line (12500,7000,11500,6500,7),
        line (21500,7000,22500,6500,7),
        line (18500,3000,18500,30000,7),
        line (20500,3000,20500,17000,7),
        line (10500,15000,11500,15000,7),
        line (12500,15000,14000,15000,7),
        line (15000,15000,16500,15000,7),
        line (17500,15000,21500,15000,7),
        line (22500,15000,24000,15000,7),
        line (25000,15000,26000,15000,7),
        line (27000,15000,29000,15000,7),
        line (28000,15000,28000,12000,7),
        line (12500,12000,21500,12000,7),
        line (22500,12000,28000,12000,7).


draw2:-
        line (12500,15000,11500,15500,7),
        line (16500,15000,17500,15500,7),
        line (22500,15000,21500,15500,7),
        line (26000,15000,27000,15500,7),
        line (12500,12000,11500,11500,7),
        line (21500,12000,22500,11500,7),
        line (12500,19000,14000,19000,7),
        line (15000,19000,16500,19000,7),
        line (17500,19000,18500,19000,7),
        line (12500,19000,11500,19500,7),
        line (16500,19000,17500,19500,7),
        line (10500,22000,11500,22000,7),
        line (12500,22000,20500,22000,7),
        line (20500,17500,20500,19500,7),
        line (20500,20000,20500,30000,7),
        line (20500,17500,20000,17000,7),
        line (20500,20000,20000,19500,7),
        line (12500,22000,11500,22500,7),
        line (20500,18000,21500,18000,7),
        line (22500,18000,28000,18000,7),
        line (28000,18000,28000,21000,7),
        line (18500,21000,21500,21000,7),
        line (22500,21000,24000,21000,7),
        line (25000,21000,28000,21000,7),
        line (22500,21000,21500,21500,7),
        line (22500,18000,21500,18500,7),
        line (12500,25000,11500,25500,7),
        line (12500,25000,14000,25000,7),
        line (15000,25000,16500,25000,7),
        line (16500,25000,17500,25500,7),
        line (17500,25000,21500,25000,7),
        line (22500,25000,21500,25500,7),
        line (22500,25000,24000,25000,7),
line (25000,25000,26000,25000,7),
line (26000,25000,27000,25500,7),
line (27000,25000,29000,25000,7),
line (28000,25000,28000,28000,7),
line (28000,28000,22500,28000,7),
line (21500,28000,22500,28500,7),
line (21500,28000,12500,28000,7),
line (12500,28000,11500,28500,7),
line (11500,28000,10500,28000,7),
form_block (2,14000,9500,1000,1000,5000),
form_block (2,14000,18500,1000,1000,6000),
form_block (2,24000,9500,1000,1000,5000),
form_block (2,24000,20500,1000,1000,4000),
cursor(5,2),
write("Goalpara"),
cursor(5,72),

write("Noapara"),
cursor(1,7),
write ("1213"),
cursor (3,7),
write ("1214"),
cursor (1,71),
write ("1210"),
cursor(3,71),
write("1212"),
cursor (9,12),
write ("104S"),
cursor(17,12),
write ("404S"),
cursor (7,18),
write ("105G"),
cursor (9,20),
write ("103S"),
cursor (11,19),
write ("102B"),
cursor (13,20),
write ("101S"),
cursor (17,20),
write ("401S"),
cursor (19,19),
write ("402B"),
cursor (20,20),
write ("403S"),
cursor (7,31),
write ("205G"),
cursor (9,31),
write ("204S"),
cursor (17,31),
write ("414S"),
cursor (9,39),
write ("203S"),
cursor (11,40),
write ("202B"),
cursor (13,39),
write ("201S"),
          cursor (17,39),
          write ("411S"),
          cursor (19,40),
          write ("412B"),
          cursor (20,39),
          write ("413S").

draw3:-

          cursor (17,47),
          write ("903S"),
          cursor (17,54),
          write ("901S"),
          cursor (19,55),
          write ("902B"),
          cursor (15,38),
          write ("905S"),
          cursor (15,45),
          write ("904S"),
          cursor (7,48),
          write ("115G"),
          cursor (9,49),
          write ("113S"),
          cursor (11,50),
          write ("112B"),
          cursor (13,49),
          write ("111S"),
          cursor (9,57),
          write ("114S"),
          cursor (7,63),
          write ("215G"),
          cursor (9,64),
          write ("213S"),
          cursor (11,65),
          write ("212B"),
          cursor (13,64),
          write ("211S"),
          cursor (9,72),
          write ("214S"),
          cursor (17,64),
          write ("421S"),
          cursor (19,65),
          write ("422B"),
          cursor (20,64),
          write ("423S"),
          cursor (17,72),
          write ("424S"),

          form_block(2,20350,6850,300,300,5000),
          form_block(2,20350,17850,300,300,4000),
          form_block(1,20350,27850,300,300,50),
          form_block(2,18350,9850,300,300,5000),
          form_block(2,18350,18850,300,300,2000),
          form_block(1,18350,24850,300,300,40),
          form_block (1,50,50,31940,31900,100),
          penpos (29000,10000,1),
        triangle,
        penpos (29500,10000,1),
        triangle,
        cursor (23,27),
        write ("405T"),
        penpos (29000,15000,1),

        triangle,
        penpos (29500,15000,1),
        triangle,
        cursor (23,40),
        write ("415T"),
        penpos (29000,25000,1),
        triangle,
        penpos (29500,25000,1),
        triangle,
        cursor (23,65),
        write ("425T"),
        cursor (24,21),
        write ("48/64 MVA"),
        cursor (24,34),
        write ("48/64 MVA"),
        cursor (24,59),
        write ("48/64 MVA"),
        readchar (_).

triangle:-

        right(30),
        forward (1000),
        left(120),
        forward (1000),
        left(120),
        forward(1000).

form_block (0,_,_,_,_,_):- !.

form_block (N,S,E,H,L,M):-

        block1 (S,E,H,L),
        SS=S,
        EE=E+M,
        NN=N-1,
        form_block (NN,SS,EE,L,H,M).

block1 (S,E,H,L):-
        penpos (S,E,0),
        forward(H),
        left (90),
        forward(L),
        left (90),
        forward(H),
        left (90),
        forward(L).
```

```
/*                  PROGRAM FOR STATION(BREAKER) FAULTS      */


check_rating (S,D,R,AC):-
        write ("CAN POWER FEED BY ALT.LINE (y/n)"),nl,
        readchar(Reply),
        Reply='y',
        supp_dem(S,D,AC),
        S+R=S1,
        S1>D,!,
        write("GRID SYSTEM O.K.NO LOADSHEAD"),nl,
        field_attr (16,0,28,135),
        graph.

check_rating(S,D,R,AC):-

        supp_dem(S,D,AC),
        check_generator(S,D,AC).

new_supp_dem (S,D,AC):-

        retract (supp_dem(_,_,_)),
        asserta (supp_dem(S,D,AC)),!.

check_generator(S,D,AC):-

        write("CAN POWER FEED BY NEARER STATIONS (y/n)"),nl,
        readchar (Reply),
        Reply ='y',
        new_supp_dem(S,D,AC),
        start_generator(N,S,D,AC).

start_generator(N,S,D,AC):-

        S>D,!.

start_generator(N,S,D,AC):-

        regional_generator(N,X,M,AC),
        NN=N+1,
        SS=S+M,
        write ("BRING GENERATOR " ,X,   "IN LINE" ),nl,!,
        new_supp_dem(S,D,AC),
        start_generator (NN,SS,D,AC),
        graph.

graphs:-

        write (" Do you want to see graph (y/n)"),nl,
        readchar (Reply),
        Reply='y',
        write ("the graph code."),nl,
        readint (A),
        gr(A).
        breaker (203,1242,1).
        breaker (115,1320,2).

        parallel (1320,1321).
        parallel (1242,1243).

        rating (1242,5).
        rating (1243,5).
        rating (1320,6).
        rating (1321,7).

        regional_generator (1,"SG-1",2,1).
        regional_generator (2,"SG-2",1,1).
        regional_generator (3,"RG-1",2,1).
        regional_generator (4,"SY-1",10,2).
        regional_generator (5,"SY-2",3,2).
        regional_generator (6,"SY-3",4,2).


/*                PROGRAM FOR MAINTENANCE FAULTS     */


power_feed (_):-

        write ("can power feed station/sub station (y/n)?"),nl,
        readchar (Reply),nl,
        Reply ='y',
        check_operation (D,R).

check_operation (D,R):-

        dem_rat (D,R),
        xline (L),
        power_line (L,PL,R,ST,M ),
        RR=R+M,
        RR>D,!,
        write ("GRID SYSTEM O.K. NO LOAD SHEAD."),nl,
        field_attr (16,0,30,135),
        disconnect_operation(S,L),
        graphs.

check_operation (D,R):-

        dem_rat(D,R),
        xline (L),
        power_line (L,PL,R,ST,M),
        RR=R+M,
        E=D-RR,
        write ("SYSTEM GOES REGIONAL LOADSHED"),nl,
        field_attr(16,0,30,135),
        write ("AMOUNT OF LOADSHED = ",E,"MW"),nl,
        disconnect_operation(S,L),
        graph.

disconnect_operation (S,L):-
```

```
            operation (S,L),
            write ( S,L ),nl,
            clear_facts.

    clear_facts:-

            retract (dem_rat(_,_)),fail.
            clear_facts:-
            retract (xline(_)),fail.
            clear_facts.

    graphs:-

            write ("Do yuo want to see graph (y/n)"),nl,
            readchar (Reply),
            Reply='y',
            write ("The graph code."),nl,
            readint (A),
            gr(A).

            power_line (1353,1552,2,"captai_2",45).
            power_line (661, 1073,2, "saidpur_gas",20).
            power_line (1278,1234,2, "bogra_gas",30).
            operation ("disconnect 113B,114S,115G from HORIPUR &102B,101S,10
5G from HATHAZARI",1353).
            operation ("disconnect 133B,134S,135G from ISHURDI &102B,103S,10
9G from PABNA" ,661).
            operation ("disconnect 215B,216S,204G from ULLAPARA &204B,205S;2
06Gfrom BOGRA",1278).


    /*                  GRAPHIS PROGRAM FOR MAINTENANCE FAULTS    */


    gr (661):-
            graphics (4,0,2),
            pencolour (1),
            pendown,
            form_block (1,1000,1000,12000,6500,100),

            cursor (10,6),
            write ("PABNA"),
            form_block (2,3500,3000,500,500,2000),
            form_block (2,8000,3000,500,500,2000),
            line (6000,1200,6000,6000,7),
            line (5000,5250,7000,5250,7),
            line (4000,5250,4500,5250,7),
            line (4500,5250,5000,5500,7),
            line (7000,5250,5000,5250,7),
            line (8000,5250,7500,5250,7),
            line (7500,5250,7000,5500,7),
            line (8500,5250,9500,5250,7),
            line (9500,5250,10000,5500,7),
            line (10000,5250,20000,5250,7),
            line (12000,5250,12000,5750,7),
            line (12000,5750,12500,6000,7),
            line (12000,6000,12000,6500,7),
            line (11500,6500,12500,6500,7),
        line (11750,6725,12250,6725,7),
        line (20000,5250,20000,14000,7),

        cursor (14,20),
        write ("6  6  1"),
        field_attr (24,9,4,129),
        cursor (6,14),
        write ("102B"),
        cursor (8,14),
        write ("103S"),
        form_block (1,17500,14000,12000,12000,2000),
        form_block (3,23000,16000,500,500,3500),
        form_block (3,26500,16000,500,500,3500),
        cursor (23,43),

        write ("ISHURDI  66KV"),
        line (24500,14500,24500,25000,7),
        line (25000,14500,25000,25000,7),
        line (20000,14000,20000,16250,7),
        line (20000,16250,21500,16250,7),
        line (23000,16250,22500,16250,7),
        line (22500,16250,22000,16500,7),
        line (20500,16250,20500,16725,7),
        line (20500,16725,20000,17000,7),
        line (20500,17250,20500,17750,7),
        line (21000,17750,20000,17750,7),
        line (20750,18000,20250,18000,7),
        line (20525,18250,20400,18250,7),
        line (23500,16250,23725,16250,7),
        line (23750,15500,23750,17000,7),
        line (23750,15500,24000,15500,7),
        line (23750,17000,24000,17000,7),
        line (24400,17000,25400,17000,7),
        line (24400,15500,25400,15500,7),
        line (26500,16250,26000,16250,7),
        line (26000,15500,26000,17000,7),
        line (26000,15500,25725,15500,7),
        line (26000,17000,25725,17000,7),
        line (27000,16250,27500,16250,7),
        line (27500,16250,27750,16500,7),
        line (27850,16250,29500,16250,7),
        line (28500,16250,28500,16750,7),
        line (28500,16750,28250,17000,7),
        line (28500,17250,28500,17750,7),
        line (28000,17750,29000,17750,7),
        line (28250,18000,28750,18000,7),
        line (28400,18250,28600,18250,7),
        cursor (18,43),
        write ("133B"),
        cursor (14,43),
        write ("135G"),
        readchar (_).

form_block (0,_,_,_,_,_):- !.

form_block (N,S,E,H,L,M):-
```

```
        block1 (S,E,H,L),
        SS=S,
        EE=E+M,
        NN=N-1,
        form_block (NN,SS,EE,L,H,M).

block1 (S,E,H,L):-
        penpos (S,E,0),
        forward(H),
        left (90),
        forward(L),
        left (90),
        forward(H),
        left (90),
        forward(L).

/*          GRAPHICS FOR MAINTAINENCE          */


predicates

        make_block(integer,integer,integer,integer,integer)
        block1(integer,integer,integer)
        make_trangle1(integer,integer,integer,integer,integer)
        make_trangle2(integer,integer,integer,integer,integer)
        go1
        go2
        go3
goal
        go1,
        go2,
        go3.
clauses
go1:-
        graphics(4,0,5),
        pencolour (1),
        pendown,
        make_block(1,300,300,16000,10000,5000),
        line(800,7000,800,11000,7),
        line(800,11000,11000,11000,7),
        line(800,7000,2200,7000,7),
        line(1700,7000,1700,7700,7),
        line(1700,7700,1300,8200,7),
        line(1700,8200,1700,8900,7),
        line(1400,8900,2000,8900,7),
        line(1600,9100,1800,9100,7),
        line(3100,7000,2200,7400,7),
        line(3100,7000,4200,7000,7),

        make_block(1,4200,6650,700,700,500),
        cursor(2,20),write("104S"),
        cursor(3,20),write("103B"),
        line(4900,7000,6800,7000,7),
        line(5500,9000,5500,300,7),
        line(5500,5500,6100,5500,7),
        line(6700,5500,6100,5100,7),
        line(6700,5500,7900,5500,7),
        line(7900,5500,8500,5100,7),
        line(8500,5500,9500,5500,7),
        line(10100,5500,9500,5900,7),
        line(9000,5500,9000,2500,7),
        line(10100,5500,11300,5500,7),
        line(10700,5500,10700,7000,7),
        line(11300,5500,11800,5900,7),
        line(12800,5500,11800,5500,7),
        line(12800,8000,12800,300,7),
        line(10700,7000,13100,7000,7),
        line(15600,7000,15600,4700,7),
        line(15600,4700,15000,4100,7),
        line(15600,4100,15600,3400,7),
        line(15300,3400,15900,3400,7),
        line(15100,3100,15700,3100,7),
        make_block(1,13100,6650,700,700,600),
        line(13800,7000,14400,7000,7),
        line(14400,7000,15000,7400,7),
        line(15000,7000,16500,7000,7),
        line(5500,4500,4800,4500,7),
        make_block(1,4100,4150,700,700,600);
        line(4100,4500,3100,4500,7),
        make_trangle2(2,2700,4000,1000,1000,400),
        line(1700,4500,1000,4500,7),

        line(5500,2500,6100,2500,7),
        line(6700,2500,6100,2100,7),
        line(6700,2500,7900,2500,7),
        line(7300,2500,7300,4500,7),
        line(5500,4500,7300,4500,7),
        line(7300,5500,7300,7000,7),
        line(8500,2500,9500,2500,7),
        line(10000,2500,9500,2100,7),
        line(10000,2500,10500,2500,7),

        make_block(1,10500,2200,700,700,200),
        line(11200,2500,11700,2500,7),
        line(11700,2500,12100,2150,7),
        line(12100,2500,12750,2500,7),
        line(5500,300,12750,300,7).


go2:-

        line(7300,7000,5500,7000,7),
        line(11000,11000,11000,27000,7),
        line(11000,27000,16500,27000,7),
        line(16000,27000,16000,27700,7),
        line(16000,27700,15600,28300,7),
        line(16000,28300,16000,29000,7),
        line(15700,29000,16300,29000,7),
        line(15900,29300,16100,29300,7),
        cursor(11,68),write("128G"),
        cursor(13,68),write("127S"),
        line(17100,27000,16500,26600,7),
        line(17100,27000,18300,27000,7),
        line(17100,23200,18300,23200,7),
        line(18900,27000,18300,26600,7),
        line(18900,23200,18300,22800,7),
        line(18900,27000,19500,27000,7),
        line(18900,23200,19500,23200,7),
        cursor(14,68),write("126S"),
        cursor(14,52),write("526S"),
        make_block(1,19500,26650,700,700,700),
        make_block(1,19500,22850,700,700,700),
        line(20200,27000,20800,27000,7),
        line(20200,23200,20800,23200,7),
        line(20800,27000,21400,26600,7),
        line(20800,23200,21400,22800,7),
        cursor(16,68),write("124S"),
        cursor(16,52),write("524S"),
        line(21400,27000,22600,27000,7),
        line(21400,23200,22600,23200,7),
        line(22600,27000,23200,26600,7),
        line(22600,23200,23200,22800,7),

        cursor(17,68),write("122M"),
        cursor(17,52),write("522M"),
        line(23200,27000,23800,27000,7),
        line(23200,23200,23800,23200,7),
        line(23800,30000,23800,21000,7),
        cursor(15,69),write("125B"),
        cursor(15,52),write("525B"),
        line(17900,27000,17900,29400,7),
        line(17900,23200,17900,24200,7),
        line(17900,29400,19100,29400,7),
        line(17900,24200,19100,24200,7),
        line(21750,27000,21750,29400,7),
        line(21700,23200,21750,24200,7),
        line(19100,29400,19700,29800,7),
        line(19100,24200,19750,24600,7),
        cursor(15,75),write("123S"),
        cursor(15,62),write("523S"),
        line(19700,29400,22600,29400,7),
        line(19700,24200,22600,24200,7),
        line(22600,29400,23200,29800,7),
        line(22600,24200,23200,24600,7),
        cursor(17,75),write("121M"),
        cursor(17,62),write("521M"),
        line(23200,29400,24500,29400,7),
        line(23200,24200,24500,24200,7),
        line(24500,30000,24500,21000,7).

go3:-

        make_trangle1(2,15500,22700,1000,1000,400),
        line(15500,23200,15000,23200,7),
        make_trangle1(1,14000,22700,1000,1000,1000),
        cursor(10,61),write("CC-1"),
        cursor(11,61),write("(G.T)"),
        cursor(12,61),write("527T"),

        make_block(1,11500,18500,14000,13100,200),
        cursor(9,59),write("ASHUGONG"),
        cursor(6,15),write("102S"),
        cursor(7,15),write("112S"),
        cursor(0,1),write("KISHORGANJ"),
        cursor(9,9),write("111S"),
        cursor(10,20),write("113B"),
        cursor(11,20),write("114S"),
        cursor(11,12),write("115S"),
        cursor(2,1),write("10/13MVA"),
        cursor(3,4),write("403B"),
        cursor(5,1),write("401S"),
        cursor(6,7),write("402S"),
        cursor(7,1),write("903S"),

        cursor(8,1),write("202B"),
        cursor(9,1),write("901S"),
        cursor(7,42),write("1 3 4 6"),
        cursor(20,7),write("MAINTENANCE FAULT OF LINE NO 1346"),
        cursor(21,7),write("BETWEEN ASHUGANG & KISHORGONJ"),
        line(16500,23200,17000,23200,7),
        make_block(1,0,0,31000,31900,1),
        readchar(_).

        make_block(0,_,_,_,_,_):- !.
        make_block(N,F,G,H,I,J):-
        block1(F,G,H,I),
        FF=F,
        GG=G+1,
        NN=N-1,
        make_block(NN,FF,GG,I,H,J).

block1(F,G,H,I):-

        penpos(F,G,0),
        forward(H),
        left(90),
        forward(I),
        left(90),
        forward(H),
        left(90),
        forward(I).

make_trangle1(0,_,_,_,_,_):-  !.

make_trangle1(N,A,B,C,D,E):-

        penpos(A,B,90),
        forward(C),
        right(120),
        forward(C),
        right(120),
        forward(C),
        NN=N-1,
        AA=A+E,
        make_trangle1(NN,AA,B,C,D,E).

make_trangle2(0,_,_,_,_,_):- !.

make_trangle2(N,A,B,C,D,E):-

        penpos(A,B,90),
        forward(C),
        left(120),
        forward(C),
        left(120),
        forward(C),
        NN=N-1,
        AA=A+E,
        make_trangle2(NN,AA,B,C,D,E).


/*      GRAPHICS  PROGRAM  FOR BRAKER FAULTS      */


gr(203):-

        graphics (4,0,2),
        pencolour (1),
        pendown,
        form_block (1,9000,2000,12500,13000,100),
        cursor (17,14),
        write ("SAIDPUR"),
        cursor (14,29),
        write ("203B"),
        cursor (9,29),
        write ("213B"),
        cursor (17,29),
        write ("205G"),

        form_block (2,18000,7000,500,500,4000),
        form_block (2,12000,7000,500,500,4000),
        line (14000,3000,14000,14000,7),
        line (15000,3000,15000,14000,7),
        line (16000,3000,16000,14000,7),
        line (18500,11250,19000,11250,7),
        line (19000,11250,19500,11000,7),
        line (19500,11250,25000,11250,7),
        line (20500,11250,20500,11750,7),
        line (20500,11750,20000,12000,7),
        line (20500,12250,20500,12750,7),
        line (20000,12750,21000,12750,7),
        line (20250,13000,20750,13000,7),
        line (20400,13250,20525,13250,7),
        line (18000,11250,17500,11250,7),
        line (17500,10250,17500,12250,7),
        line (17500,10250,17000,10250,7),
        line (17500,12250,17000,12250,7),
        line (15000,10250,16500,10250,7),
        line (16000,12250,16500,12250,7),
line (16500,10250,17000,10500,7),
line (16500,12250,17000,12500,7),
line (12500,11250,13000,11250,7),
line (13000,10250,13000,12250,7),
line (13000,10250,13500,10250,7),
line (13000,12250,13500,12250,7),

line (14000,10250,13750,10250,7),
line (15000,12250,13750,12250,7),
line (13750,10250,13850,10500,7),
line (13750,12250,13850,12500,7),
line (12000,11250,11500,11250,7),
line (11500,11250,11000,11750,7),
line (10700,11250,8000,11250,7),
line (9725,11250,9725,10725,7),
line (9725,10725,10250,10500,7),
line (9725,10250,9725,10000,7),
line (9250,10000,10250,10000,7),
line (9550,9725,10000,9725,7),
line (9700,9500,9800,9500,7),

form_block (1,9000,18500,12500,13000,100),

cursor (17,55),
write ("RANGPURE"),
form_block (2,18000,24000,500,500,4000),
form_block (2,12000,24000,500,500,4000),

line (14000,19000,14000,30000,7),
line (15000,19000,15000,30000,7),
line (16000,19000,16000,30000,7),
line (18000,24250,17500,24250,7),
line (12000,24250,11500,24250,7),
line (18500,24250,19000,24250,7),
line (19000,24250,19500,24000,7),
line (19500,24250,25000,24250,7),
line (25000,24250,25000,27000,7),
line (20000,24250,20000,24750,7),
line (20000,24750,19500,25000,7),
line (20000,25000,20000,25500,7),
line (19500,25500,20500,25500,7),
line (19725,25750,20250,25750,7),
line (19850,26000,20100,26000,7),
line (12500,24250,13000,24250,7),
line (13000,23250,13000,25250,7),
line (13000,23250,13500,23250,7),
line (13000,25250,13500,25250,7),
line (15000,23250,13600,23250,7),
line (14000,25250,13600,25250,7),
line (17500,23250,17500,25250,7),
line (17500,23250,17000,23250,7),
line (17500,25250,17000,25250,7),
line (16000,23250,16500,23250,7),
line (15000,25250,16500,25250,7),
line (11500,24250,11000,24500,7),
line (11000,24250,5000,24250,7),
        line (5000,24250,5000,17500,7),
        line (5000,17500,25000,17500,7),
        line (25000,17500,25000,11250,7),
        cursor (2,45),
        write ("1 2 4 9"),
        line (7000,24250,7000,23750,7),
        line (7000,23750,7500,23500,7),
        line (7000,23250,7000,22750,7),
        line (6500,22750,7500,22750,7),
        line (6750,22500,7250,22500,7),
        line (6900,22250,7100,22250,7),
        readchar(_).

form_block (0,_,_,_,_,_):- !.

form_block (N,S,E,H,L,M):-

        block1 (S,E,H,L),
        SS=S,
        EE=E+M,
        NN=N-1,
        form_block (NN,SS,EE,L,H,M).

block1 (S,E,H,L):-

        penpos (S,E,0),
        forward(H),
        left (90),
        forward(L),
        left (90),
        forward(H),
        left (90),
        forward(L).

/*                      BOROAULIA  SUBSTATION    */


predicates
        draw
        block(integer,integer)
        new
        trian(integer,integer)

goal
        graphics(1,1,0),
        draw.

clauses

draw:-
        line(0,0,0,31999,2),line(0,31999,31999,31999,2),
        line(31999,31999,31999,0,2),line(31999,0,0,0,2),
        line(18000,3000,18000,27000,2),

        cursor(1,1),write("\t     B A R O A U L I A" ),
        cursor(2,2), write("\t\tsub station"),
line(7000,5000,9000,5000,2),
line(9000,5500,10000,5000,2),
line(10000,5000,11000,5000,2),

block(11000,4500),
line(12000,5000,20000,5000,2),
line(20000,5500,21000,5000,2),
line(21000,5000,28000,5000,2),
line(28000,5000,28000,20000,2),
line(7000,3000,7000,27000,7),
line(7000,9000,9000,9000,2),
line(9000,9500,10000,9000,2),
line(10000,9000,11000,9000,2),

block(11000,8500),
line(12000,9000,20000,9000,2),
line(20000,9500,21000,9000,2),
line(21000,9000,26000,9000,2),
line(25500,8800,26000,9000,2),
line(26000,9000,25500,9200,2),
line(7000,13000,9000,13000,2),
line(9000,13500,10000,13000,2),
line(10000,13000,11000,13000,2),

block(11000,12500),
line(12000,13000,20000,13000,2),
line(20000,13500,21000,13000,2),
line(21000,13000,25000,13000,2),

trian(25000,13000),
line(26100,13000,27000,13000,2),
line(7000,17000,9000,17000,2),
line(9000,17500,10000,17000,2),
line(10000,17000,11000,17000,2),

block(11000,16500),
line(12000,17000,20000,17000,2),
line(20000,17500,21000,17000,2),
line(21000,17000,25000,17000,2),

trian(25000,17000),
line(26100,17000,27000,17000,2),
line(7000,21000,9000,21000,2),
line(9000,21500,10000,21000,2),
line(10000,21000,11000,21000,2),


block(11000,20500),
line(12000,21000,20000,21000,2),
line(20000,21500,21000,21000,2),
line(21000,21000,26000,21000,2),
line(25500,20800,26000,21000,2),
line(26000,21000,25500,21200,2),
line(7000,25000,9000,25000,2),
line(9000,25500,10000,25000,2),
        line(10000,25000,11000,25000,2),

        block(11000,24500),
        line(12000,25000,20000,25000,2),
        line(20000,25500,21000,25000,2),
        line(21000,25000,26000,25000,2),
        line(25500,24800,26000,25000,2),
        line(26000,25000,25500,25200,2),
        line(18000,4000,20000,4000,2),
        line(20000,4500,21000,4000,2),
        line(21000,4000,23000,4000,2),
        line(23000,4000,23000,5000,2),
        line(18000,7000,20000,7000,2),
        line(20000,7500,21000,7000,2),
        line(21000,7000,23000,7000,2),
        line(23000,7000,23000,9000,2),
        line(18000,11000,20000,11000,2),
        line(20000,11500,21000,11000,2),
        line(21000,11000,23000,11000,2),
        line(23000,11000,23000,13000,2),
        line(18000,15000,20000,15000,2),
        line(20000,15500,21000,15000,2),
        line(21000,15000,23000,15000,2),
        line(23000,15000,23000,17000,2),
        line(18000,19000,20000,19000,2),
        line(20000,19500,21000,19000,2),
        line(21000,19000,23000,19000,2),
        line(23000,19000,23000,21000,2),
        line(18000,23000,20000,23000,2),
        line(20000,23500,21000,23000,2),
        line(21000,23000,23000,23000,2),
        line(23000,23000,23000,25000,2),
        cursor(10,7),write("102B"),
        cursor(10,17),write("402B"),
        cursor(10,27),write("212B"),new.

new:-

        cursor(23,15),write("333"),
        cursor(19,12),write("405T"),
        cursor(19,22),write("415T"),
        readchar(_).

block(X,Y):-

        YI=Y+1000,
        penpos(X,YI,0),
        forward(1000),right(90),forward(1000),
        right(90),forward(1000),right(90),
        forward(1000).

trian(U,V):-

        penpos(U,V,30),
        forward(1200),right(120),forward(1200),
        right(120),forward(1200),
UU=U+500,
VV=V+0,
penpos(UU,VV,30),
forward(1200),right(120),forward(1200),
right(120),forward(1200).


       graphics(4,1,4),
       load1,
       load2,
       load3.
 load1:-
       line(3000,10000,3000,28000,7),
       line(4000,20000,4000,26000,7),
       line(3000,28000,28000,28000,7),
       line(4000,26000,30000,26000,7),
       line(3000,10000,4500,10000,7),
       line(4000,20000,4500,20000,7),
       line(5000,10000,5500,10000,7),
       line(5000,10000,4500,9500,7),
       ground(4000,10000),
       line(5000,20000,5500,20000,7),
       line(5000,20000,4500,19500,7),
       block(5500,9500),
       block(5500,19500),
       line(6500,10000,7500,10000,7),
       line(6500,20000,7500,20000,7),
       line(9000,5000,9000,25000,7),
       line(9500,5500,9500,25000,7),
       line(10000,5000,10000,25000,7),
       line(9000,5000,10000,5000,7),
       milco(12500,10000),
       milco(12500,20000),
       clock(6500,10000),
       clock(6500,20000),
       block(12500,9500),
       block(12500,19500),
       ground(15000,10000),
       ground(15000,20000),
       line(14500,20000,15000,20000,7),
       line(14500,10000,15000,10000,7),
       line(15000,5000,15000,10000,7),
       line(14000,10000,14500,9500,7),
       line(14000,20000,14500,19500,7),
       line(13500,10000,14000,10000,7),
       line(13500,20000,14000,20000,7),
       line(16000,5000,16000,30000,7),
       line(16350,20000,16350,29000,7),
       line(16000,30000,20000,30000,7),
       line(16350,29000,20000,29000,7),
       clock(16000,5000),
       clock(16350,20000),
       ground(16850,20000),
       block(18500,7000),
       block(18800,22000),
       line(19500,7500,20000,7500,7),
        line(19700,22500,20000,22500,7),
        line(20000,7500,20500,8000,7),
        line(20000,22500,20400,23000,7),
        line(20500,7500,20800,7500,7),
        line(20400,22500,20900,22500,7),
        line(21300,7500,20800,8000,7),
        line(21400,22500,20900,23000,7),
        line(21300,7500,22000,7500,7),
        line(21400,22500,22000,22500,7).
load2:-
        block(22000,7000),
        block(22000,22000),
        milco(26000,5000),
        milco(26000,20000),
        cursor(11,2),write("PALASHBARI"),
        cursor(10,14),write("1244"),
        cursor(9,19),write("113B"),
        cursor(11,19),write("148S"),
        cursor(10,28),write("114S"),
        cursor(8,13),write("111S"),
        cursor(8,33),write("112S"),
        cursor(8,38),write("211S"),
        cursor(8,58),write("212S"),
        cursor(9,44),write("213B"),
        cursor(11,43),write("214S"),
        cursor(11,35),write("BOGRA"),
        cursor(10,53),write("215G"),
        cursor(11,66),write("1241"),
        cursor(13,66),write("1240"),
        cursor(6,13),write("162S"),
        cursor(6,33),write("101S"),
        cursor(6,38),write("202S"),
        cursor(6,58),write("201S"),
        cursor(4,19),write("103B"),
        cursor(3,30),write("105G"),
        cursor(3,19),write("104S"),
        cursor(3,43),write("204S"),
        cursor(4,54),write("203B"),
        cursor(6,66),write("1"),
        cursor(7,66),write("2"),
        cursor(8,66),write("4"),
        cursor(9,66),write("3"),
        cursor(16,71),write("ISHURDI"),
        cursor(6,71),write("1"),
        cursor(7,71),write("2"),
        cursor(8,71),write("4"),
        cursor(9,71),write("2").
   load3:-
        cursor(13,58),write("203S"),
        cursor(14,58),write("202B"),
        cursor(15,58),write("201S"),
        cursor(16,58),write("211S"),
        cursor(17,58),write("212B"),
        cursor(19,58),write("213S"),
        cursor(21,52),write("215G"),
        cursor(14,38),write("204S"),
        cursor(18,38),write("214S"),
        cursor(13,21),write("103S"),
        cursor(14,21),write("102B"),
        cursor(14,2),write("104S"),
        cursor(17,1),write("114S"),
        cursor(15,21),write("101S"),
        cursor(16,21),write("111S"),
        cursor(17,21),write("112B"),
        cursor(19,21),write("113S"),
        cursor(21,15),write("115G"),
        cursor(23,5),write("NATORE"),
        cursor(24,17),write("FIG. FOR RULE NO-1 OF"),
        cursor(24,40),write("BREAKER FAULTS."),
        line(23200,2500,19000,2500,7),
        line(23200,17500,19000,17500,7),
        ground(26000,5000),
        ground(26000,20000),
        line(26000,5000,28000,5000,7),
        line(26000,20000,30000,20000,7),
        line(28000,5000,28000,28000,7),
        line(30000,5000,30000,26000,7),
        readchar(_).
ground(A,B):-
        C=B+500,
        D=A-500,
        E=C+500,
        F=E+500,
        G=A+500,
        H=A-300,
        I=F+200,
        J=H+600,
        K=A-200,
        L=I+200,
        M=A+200,
        line(A,B,A,C,7),
        line(A,C,D,E,7),
        line(A,E,A,F,7),
        line(D,F,G,F,7),
        line(H,I,J,I,7),
        line(K,L,M,L,7).
clock(M,N):-
        S=M+1200,
        T=N-2500,
        W=N+2500,
        L=S+500,
        B=T+500,
        K=W+500,
        Z=L+500,
        O=Z+500,
        D=Z+1000,
        line(M,N,S,N,7),
        line(S,T,S,W,7),
        line(S,T,L,T,7),
        line(S,W,L,W,7),
        line(L,K,Z,W,7),
        line(Z,W,O,W,7),
        line(L,B,Z,T,7),
        line(Z,T,D,T,7).
block(X,Y):-
        Z=X+1000,
        P=Y+1000,
        line(X,Y,X,P,7),
        line(X,P,Z,P,7),
        line(Z,P,Z,Y,7),
        line(Z,Y,X,Y,7).
milco(A,B):-
        C=A-1000,
        E=B-2500,
        F=B+2500,
        G=C-500,
        H=G-500,
        I=E-500,
        R=F+500,
        U=H-1000,
        J=H-500,
        line(A,B,C,B,7),
        line(C,E,C,F,7),
        line(C,E,G,E,7),
        line(G,E,H,I,7),
        line(H,E,J,E,7),
        line(C,F,G,F,7),
        line(G,F,H,R,7),
        line(H,F,U,F,7).

   /* END OF EXPERT SYSTEM PROGRAM    */
```

APPENDIX—B

DEMAND REPORT OF SUB-STATIONS.

১৪/৬/৮২

কেন্দ্রীয় বিদ্যুৎ নিয়ন্ত্রণ কেন্দ্র

তারিখে বিদ্যুৎ চাহিদা ও ব্যাহতির সার-সংক্ষেপ

সিষ্টেমের তথ্য	দিনের বেলায় সবাধিক চাহিদাকালীন উৎপাদন	: 927.1	মেগাওয়াট	১১	টায়
	দিন-রাতের সর্বাধিক চাহিদাকালীন উৎপাদন	: 1218.70	মেগাওয়াট		টায়
	দিন-রাতের সর্বনিম্ন চাহিদাকালীন উৎপাদন	: 608.50	মেগাওয়াট		টায়
	সিষ্টেম লোড ফ্যাক্টর	: 67.246	%		—
	পূর্ব গ্রীড এলাকার সর্বাধিক চাহিদা	: 770.25	মেগাওয়াট		টায়
	পশ্চিম গ্রীড এলাকার সর্বাধিক চাহিদা	: 297.10	মেগাওয়াট		টায়
	বৃহত্তর ঢাকার সর্বাধিক চাহিদা	: 426.90	মেগাওয়াট		টায়
	বৃহত্তর চট্টগ্রামের সর্বাধিক চাহিদা	: 180.0	মেগাওয়াট		টায়
	বৃহত্তর খুলনার সর্বাধিক চাহিদা	: 68.8	মেগাওয়াট		টায়
পূব পশ্চিম আন্তঃ সংযোগ	পূর্ব গ্রীড থেকে পশ্চিম গ্রীডে সর্বাধিক বিদ্যুৎ শক্তি আমদানী	: 225.0	মেগাওয়াট		টায়
	পশ্চিম গ্রীড থেকে পূর্ব গ্রীড সর্বাধিক বিদ্যুৎ শক্তি রপ্তানী	:	মেগাওয়াট		টায়
	পূর্ব গ্রীড থেকে পশ্চিম গ্রীডে মোট বিদ্যুৎ শক্তি আমদানী	: 59,12,070,	কিলোওয়াট-ঘন্টা		
	পশ্চিম গ্রীড থেকে পূর্ব গ্রীডে মোট বিদ্যুৎ শক্তি রপ্তানী	:	কিলোওয়াট-ঘন্টা		

গুরুত্বপূর্ণ উপকেন্দ্রের সর্বাধিক চাহিদা

উপকেন্দ্র	মেগাওয়াট	সময়	উপকেন্দ্র	মেগাওয়াট	সময়	উপকেন্দ্র	মেগাওয়াট	সময়
সিদ্ধিরগঞ্জ	64		কুমিল্লা	28.5		মাদারীপুর	7.8	
উলন	61		চাঁদপুর	7.0		বরিশাল	22.1	
টঙ্গী	85		ফেনী	20.0		বাগেরহাট	7.1	
মীরপুর	63.5		মদনহাট	59.0		মঙ্গলা	9.8	
পোস্তগোলা	72.		কুলশী	56.00		ঈশ্বরদী	17.7	
ধানমণ্ডি	52.		হালিশহর	25.5		নাটোর	4.1	
কবিরপুর	3.9		বার আউঃ	16.0		বগুড়া	29.4	
ঘোড়াশাল	22		শিকলবাহা			পলাশবাড়ী	6.9	
আশুগঞ্জ	13.9		দোহাজারী	16.5		রংপুর	19.0	
কিশোরগঞ্জ	8.6		চন্দ্রঘোনা	7.5		সৈয়দপুর	4.6	
ময়মনসিংহ	21.		গোয়ালপাড়া	0.8		পূর্বসাদিপুর	6 3	
জামালপুর	7.7		খুলনাসেন্ট্রাল	65		ঠাকুরগাঁও	5.5	
টাঙ্গাইল	20.0		নওয়াপাড়া	2.6		রাজশাহী	21.5	
শাহজীবাজার	7.25		যশোহর	23		পাবনা	7.2	
শ্রীমঙ্গল	10.7		ঝিনাইদহ	13.0		শাহজাদপুর	4.2	
ফেঞ্চুগঞ্জ	8.8		বটতৈল	20.0		উল্লাপাড়া	9.9	
সিলেট	17.2		ভেড়ামারা	10.0		সিরাজগঞ্জ	7.9	
ছাতক	8.8		ফরিদপুর	1.8		কাপ্তাই	9.5	

সরবরাহে ব্যাহতি

No load Shed

APPENDIX-C

GENERATION CHART OF POWER STATIONS.

কেন্দ্রীয় বিদ্যুৎ নিয়ন্ত্রণ কেন্দ্র

বাংলাদেশ বিদ্যুৎ উন্নয়ন বোর্ড

প্রাত্যহিক সকাল ৮টার রিপোর্ট

তারিখ :
বার :

কাপ্তাই জলাধারে পানির উচ্চতা
আজ সকাল ৬টায় : ৬৮.১৭
রুল কার্ভ অনুযায়ী : ৭৭.৭৪
(গড় সমুদ্র সমতল থেকে)

তারিখে বিদ্যুৎ উৎপাদনের সার-সংক্ষেপে

বিদ্যুৎ কেন্দ্র	যন্ত্র:	বর্তমান উৎপাদন ক্ষমতা (মেগাওয়াট)	আজকের উৎপাদন ক্ষমতা (মেগাওয়াট)	সর্বাধিক চাহিদাকালে উৎপাদন (মেগাওয়াট)	মোট উৎপাদন (কিলোওয়াট ঘণ্টা)	আনুমানিক গ্যাস খরচ (মিলিয়ন ঘনফুট)	মন্তব্য
আশুগঞ্জ বাষ্প	১	৬৪.০					
	২	৬৪.০					
	৩	১৫০.০					
	৪	১৫০.০					
	৫	১৫০.০					
আশুগঞ্জ কম্বাইণ্ড সাইকেল	গ্যাস	৫০.০					
	বাষ্প	৩০.০					
আশুগঞ্জ গ্যাসটারবাইন	২	৫০.০					
ঘোড়াশাল বাষ্প	১	৫৫.০					
	২	৫৫.০					
	৩	২১০.০					
	৪	২১০.০					
কাপ্তাই পানি বিদ্যুৎ	১	৪০.০					
	২	৪০.০					
	৩	৫০.০					
	৪	৫০.০					
	৫	৫০.০					
শাহজী বাজার গ্যাস টারবাইন	ফিয়াট	৪২.০					
	সিই এম	৪৮.০					
সিদ্ধিরগঞ্জ বাষ্প		৩০.০					
		৫০.০					
হরিপুর গ্যাস টারবাইন	১	৩৩.০					
	২	৩৩.০					
	৩	৩৩.০					
চট্টগ্রাম বার্জ	১	২৮.০					
	২	২৮.০					
চট্টগ্রাম বাষ্প		৬০.০					
চট্টগ্রাম গ্যাস টারবাইন		১০০.০					
সিলেট গ্যাস টারবাইন		২০.০					
খুলনা (১১০ মেঃ ওঃ)		১১০.০					
খুলনা (৬০ মে ও)		৫৫.০					
খুলনা বার্জ	১	২৫.০					
	২	২৮.০					
খুলনা জিজি-১৪/জিটি-১৫		৮.০/৯.০					
ভেড়ামারা গ্যাস		৫১.০					
বরিশাল গ্যাস	১	২০.০					
	২	২০.০					
সৈয়দপুর গ্যাস		২০.০					
রংপুর গ্যাস		২০.০					
বগুড়া গ্যাস		৫.০					
সম্মিলিত ডিজেল		২০.০					
গ্রীডের সর্বমোট			1581.0	1218.7	24,695		

পূর্বগ্রীডের মোট উৎপাদন : 190,59,659 কি. ও: ঘ:

পশ্চিম গ্রীডের মোট উৎপাদন : 6 8 995 কি: ও: ঘ:

APPENDIX—D

POWER INTERUPTION CONFIGURATION

(ON 21ST MAY,1986)

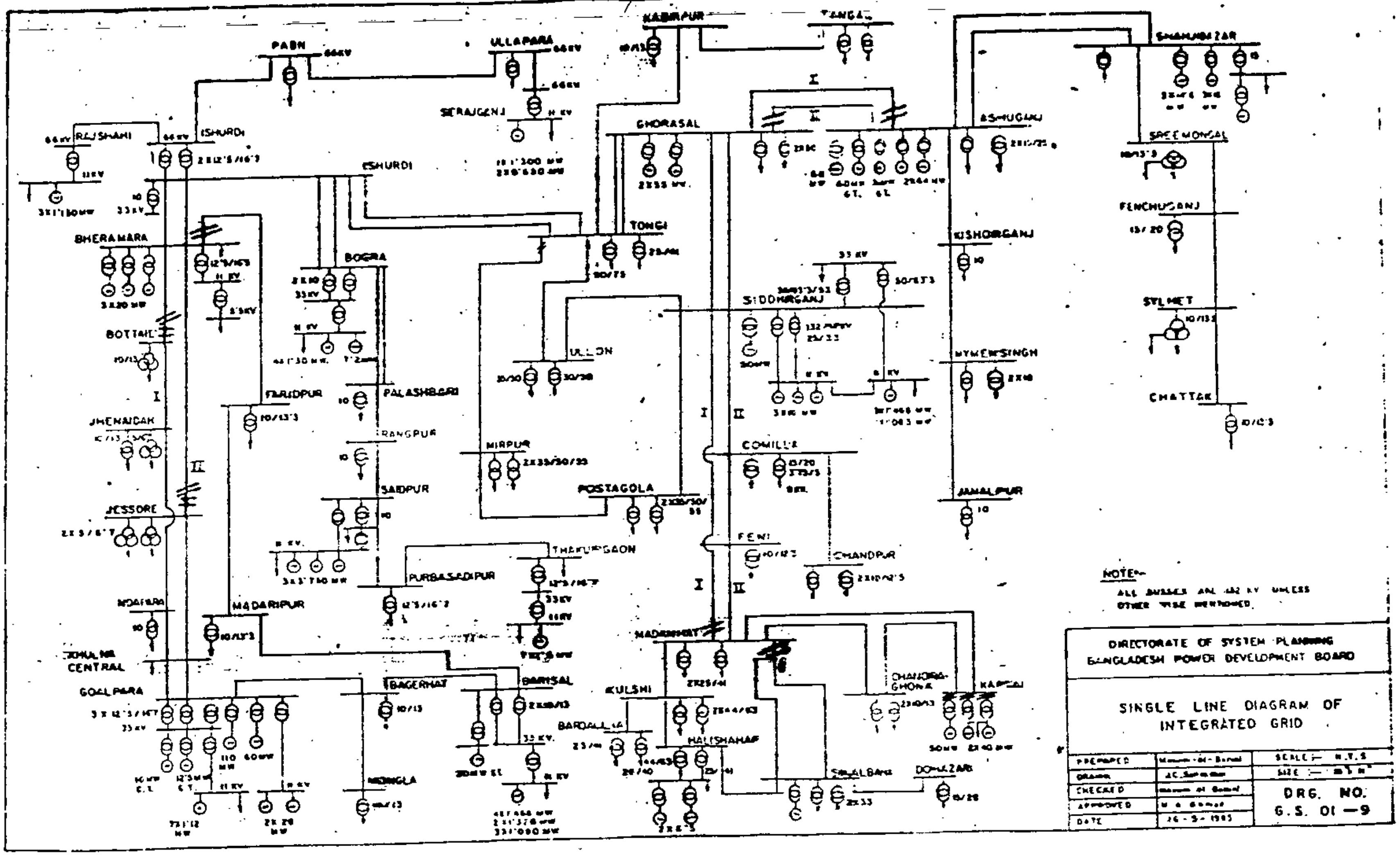
Power Inter-uption on May 31, 1986

ANNEXURE - :2

Time - 10-17 A.M.

1st Tripping

AS ON 20-5-86



NOTE:-
ALL BUSSES ARE 132 KV UNLESS OTHER WISE MENTIONED.

DIRECTORATE OF SYSTEM PLANNING
BANGLADESH POWER DEVELOPMENT BOARD

SINGLE LINE DIAGRAM OF INTEGRATED GRID

PREPARED	[illegible]	SCALE :- N.T.S
DRAWN	A.C. [illegible]	SIZE :- [illegible]
CHECKED	[illegible]	DRG. NO. G.S. 01 —9
APPROVED	[illegible]	
DATE	26-5-1985	

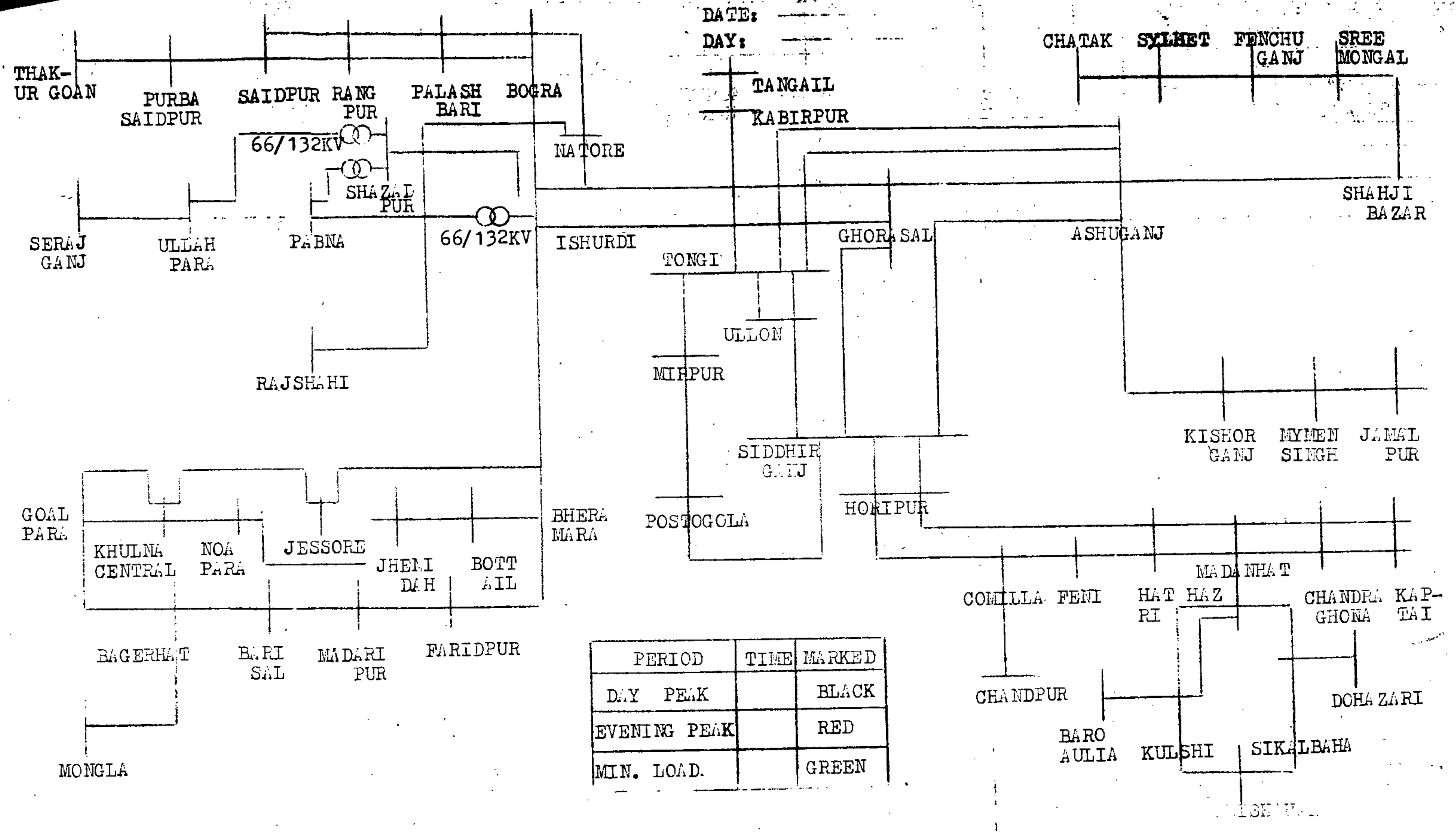
APPENDIX E

(CONSOLATED SINGLE LINE DIAGRAM OF ELECTRICAL GRID SYSTEM)

VOLTAGE (KV) DIFFERENT POINTS OF [illegible]

DATE:

DAY:



PERIOD	TIME	MARKED
DAY PEAK		BLACK
EVENING PEAK		RED
MIN. LOAD.		GREEN

REFERENCES.

[1] STEVE TORRANCE.

" The Mind and the Machine."

John willy & sons, 1978.

[2] CARL TOWNSEND.

" Introduction to Turbo Prolog."

BPB Pblications, 1987.

[3] PETTER SMITH.

"Expert System Development in Prolog and Turbo Prolog."

Halsted Press, A division of John Wiley & Sons,

New York, 1983.

[4] DONALD A. WATERMAN.

" A Guide to Expert System."

Addision-Wesly Publishing Company, California-1985.

[5] W.F.CLOCKSIN. & C.S. MELLISH.

" Programming in prolog"

Springer-Verlay Publications-1984.

[6] T.SAKAGUCHI. AND K.MATSUMOTO.

" Development of a knowledge base system for power system Restoration."

IEEE Trans. on power apparatus & systems.

Vol. pas-102, No-2, Feb, 1983, PP-320-329.

[7] PERSONAL CONSULTANT.

" Artificial Intelligence Publications."

The AI Report, vol, 1, no-12, 1986.

[8] PETER. S. SELL.

" Expert systems- A Practical Introduction."

John Willey & Sons, New York-1986.

[9] **"Design and Implementation of a Dental Diagnostic system for predontal diseases."**

Project Report, CSE Department.

I.I.T. Bomby, 1988.

[10] **" TURBO PROLOG OWNER'S HANDBOOK."**

Borland Publications, 1983.

[11] ALTY.J.L & COOMS M.J.

" Expert System Concepts & Expamles."

NCC Publications, 1984.

[12] G.NOWLAN. & MCDERMOTT.

A foundation for Knowledge acquisition.

Proceedings of the IEEE workshop on

principles of knowledge base system,

IEEE Computer Society Press,

1109 Spring Sheet,

Silver Spring Md, 1985.

[13] A.E.FELTZIN,H.GARCIA, Dr. A.I.LACAVA.

" Development of Artificial Intelligence System by the BOC Group."

The BOC Group Technology Magazine, March, 1989.

[14] PETER JACKSON.

Introduction to Expert Systems

Addision- Wesly Publications-1986

[15] W.B. GEVARTER.

" An Overview of Expert System."

NBSIR 82-2505, National Bureau of Standards.

Washington, D.C. May,1982.

[16] IVAN BRATKO.

" Prolog Programming for Artificial Intelligence."

Addesson-Wesley Publications, 1986.

[17] GL SIMONS.

" Introducing Artificial Intelligence"

NCC Publications, 1981.

[18] CHEN-CHINGLIU. & KEVIN TOMSOVIC.

" An Expert System Assisting Decision Making of Reactive Power/Voltage COntrol.

IEEE Transactions on power system,

Vol PWRS-1,No-3,Augest-1986.

[19] M.YAZDANI. & A.NARAYANA.

" Artificial intelligence in human effects."

ELLIS HORWOOD SERIES, 1985.

```
assertz (xline(L)),
write ("what is the max. regional demand"),nl,
```

page - 1- 4,
16,
20,
26
6
7