

**To Develop a Knowledge-Based System for Parallel Machine  
Operations Scheduling**

By

Mohd Razman Bin Ramli

Dissertation submitted in partial fulfillment  
of the requirement  
for the Bachelor of Engineering (Hons)  
(Mechanical Engineering)

**JUNE 2004**

**Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan**

t

QA

76.76

.E95

M697

2004

- 1) Expert systems (computer science)
- 2) Knowledge management
- 3) PNE -- Thesis

## CERTIFICATION OF APPROVAL


**To Develop Knowledge Based System for Parallel Machine  
Operation Scheduling**

By

Mohd Razman Bin Ramli

A project dissertation submitted to the  
Mechanical Engineering Department  
Universiti Teknologi PETRONAS  
in partial fulfillment of the requirements for  
the BACHELOR OF ENGINEERING (Hons)  
(MECHANICAL ENGINEERING)

Approved by,

 2/6/2004  

---

(Mrs. Ainul Akmar Mokhtar)

UNIVERSITI TEKNOLOGI PETRONAS  
TRONOH, PERAK  
JUNE 2004

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



---

MOHD RAZMAN BIN RAMLI

## ABSTRACT

As the industrialized world develops, more and more resources are becoming critical. Machines, manpower and facilities are now commonly thought of as critical resources in production and service activities. Scheduling these resources leads to increased efficiency, utilization, and ultimately, profitability. Current operation scheduling methods require complex mathematical modeling techniques that demand the substantial and extensive knowledge. Otherwise, the simple methods may not provide good results. The project is intended to engage in the issue of parallel machine operation scheduling from the knowledge based system perspective. The deal between both fields will emerge a new development of formulation and integration in artificial intelligence at area of industrial scheduling. Eventhough there are diversity techniques in manufacturing industry, the scope of the study is only limited to identical parallel operation scheduling due to time constraint towards the completion of the project. The goal of this project is to produce a working model of knowledge system for parallel machine operation scheduling. The execution of the project will be conducted in two semesters. For the First Semester, the data gathering and carrying out the associated case studies were explicitly nurtured as to aid better understanding of the project. The case studies performed were: (1) Parallel Processing – Jobs of Equal Weight, (2) Parallel Processing – Weighted Jobs, and (3) Parallel Processing – Jobs with Due Dates. In the Second Semester, the knowledge system was effectively developed. The development of the knowledge system was done in the expert interface which consists of six parts. The first step is theory familiarization, followed by user interface development, the inference engine development, dry run or testing and verification of the system. When all the steps are taken, the knowledge system can be considered as complete.

## ACKNOWLEDGEMENT

The author is vastly indebted to many people who have helped and inspired from the early beginning till the end of this Final Year Project. First and foremost, the greatest sincere gratitude goes to the supportive supervisor, Mrs. Ainul Akmar Bt Mokhtar; Lecturer Mechanical Engineering Department, UTP, who had been very helpful and encouraging throughout this project. The stimulus discussions held were greatly appreciated.

To all lecturers/tutors from Information Technology Department, especially to Mr. Jale Bin Ahmad and Mr. Justin Dinesh Devaraj, who had been very co-operative in helping and sharing their expertise in VB application - thanks for the priceless help and support.

The author would like to acknowledge the external examiner, Mr. Thayananda, Staff Engineer from Penang Seagate Industries (M) Sdn. Bhd., Mr. Hilmi Hussin and Mohd Faizari Bin Mohd Nor, Lecturers of Mechanical Engineering Department, UTP for their supports in providing the necessary information, knowledge, useful ideas, corrections and suggestions with regard to this study.

Million of thanks go to the author's helpful colleagues who had been very supportive and thoughtful in sharing their ideas and support to him.

Finally to the lovely parents, without the enormous help and tremendous support, the accomplished project will not be successful as it was.

## TABLE OF CONTENTS

CERTIFICATION OF APPROVAL . . . . .	i
CERTIFICATION OF ORIGINALITY . . . . .	ii
ABSTRACT . . . . .	iii
ACKNOWLEDGEMENT . . . . .	iv
TABLE OF CONTENTS . . . . .	v
SPECIFIC DEFINITION AND ABBREVIATIONS . . . . .	vii
LIST OF FIGURES . . . . .	viii
LIST OF TABLES . . . . .	ix
LIST OF APPENDICES . . . . .	x
<b>CHAPTER 1: INTRODUCTION . . . . .</b>	<b>1</b>
1.1 Background of Study . . . . .	1
1.2 Problem Statement . . . . .	3
1.2.1 Problem Identification . . . . .	3
1.2.2 Significant of the Project . . . . .	3
1.3 Objectives and Scopes of Study . . . . .	4
<b>CHAPTER 2: LITERATURE REVIEW/THEORY . . . . .</b>	<b>5</b>
2.1 Operation Scheduling . . . . .	5
2.1.1 Jobs . . . . .	5
2.1.2 Machines . . . . .	6
2.1.3 Measures . . . . .	6
2.1.4 Scheduling Algorithms . . . . .	7
2.2 Parallel Processing . . . . .	7
2.2.1 Minimizing Flowtime for Normal Jobs	9
2.2.2 Minimizing Flowtime for Weighted Jobs	9
2.2.3 Due Dated Jobs . . . . .	10
2.3 Knowledge Based System . . . . .	11
2.3.1 Knowledge System Terminology . . . . .	14
2.3.2 Domain Knowledge . . . . .	15

<b>CHAPTER 3:</b>	<b>METHODOLOGY/PROJECT WORK.</b>	<b>17</b>
3.1	Procedure Identification	17
3.2	Knowledge Based System Development	17
3.3	Tool for Knowledge System Development	19
3.4	System Layouts	20
3.4.1	Parallel Machine	20
<b>CHAPTER 4:</b>	<b>RESULT/DISCUSSION</b>	<b>23</b>
4.1	Case Studies	24
4.1.1	Case Study I	24
4.1.2	Case Study II	26
4.1.3	Case Study III	28
4.2	Introductory Forms	30
4.3	Objective Form	31
4.4	The Coding Structures	34
4.5	Verifications of Knowledge System	36
<b>CHAPTER 5:</b>	<b>CONCLUSION/RECOMMENDATION</b>	<b>43</b>
5.1	Conclusion	43
5.2	Recommendations	44
5.2.1	Current Project	44
5.2.1	Further Expansion	45
<b>REFERENCES</b>		<b>47</b>
<b>Appendix A</b>		<b>49</b>
<b>Appendix B</b>		<b>55</b>
<b>Appendix C</b>		<b>67</b>
<b>Appendix D</b>		<b>83</b>

## SPECIFIC DEFINITION AND ABBREVIATIONS

1	UTP	Universiti Teknologi PETRONAS
2	FYP	Final Year Project
3	KBS	Knowledge-Based System
4	VB	Microsoft Visual Basic 6.0.
5	RCT	Remaining Cumulative Time
6	RCR	Remaining Cumulative Ratio
7	RCD	Remaining Cumulative Due Dates
8	LPT	Longest Processing Time
9	SPT	Shortest Processing Time
10	EDD	Earliest Due Dates
11	WSPT	Weighted Shortest Processing Time
12	IT	Information Technology



## LIST OF FIGURES

- Figure 2.1:** Typical Parallel Processing Setup
- Figure 2.2:** Classical Characterization of Knowledge System (Stefik, 1995)
- Figure 2.3:** ‘Bubble Sort’ Coding Structure
- Figure 3.1:** Flowchart of Knowledge System Development
- Figure 3.2:** Tree Chart for Parallel Machine Interface Layout
- Figure 4.1:** Start Up Form
- Figure 4.2:** Jobs with Equal Weight Form
- Figure 4.3:** Jobs with Priorities and Ranked by Weight Form
- Figure 4.4:** Jobs with Due Dates Form
- Figure 4.5:** Result for Shortest Processing Time
- Figure 4.6:** Result for Weighted Shortest Processing Time
- Figure 4.7:** Result for Due Dated Jobs

## LIST OF TABLES

<b>Table 3.1:</b>	Steps of Knowledge Based System Development
<b>Table 3.2:</b>	The Founding Theory of the Objectives
<b>Table 4.1:</b>	Jobs Available for Case Study I
<b>Table 4.2:</b>	Sorted Jobs for Case Study I
<b>Table 4.3:</b>	Job Assignments on Each Machine for Case Study I
<b>Table 4.4:</b>	Jobs Available for Case Study II
<b>Table 4.5:</b>	Sorted Jobs for Case Study II
<b>Table 4.6:</b>	Final Assignments on Each Machine for Case Study II
<b>Table 4.7:</b>	Jobs Available for Case Study III
<b>Table 4.8:</b>	Tabulated Data of the Jobs in Case Study III
<b>Table 4.9:</b>	Final Assignments on Each Machine for Case Study III
<b>Table 4.10:</b>	Data for Jobs with Equal Weights
<b>Table 4.11:</b>	Sorted Jobs for Jobs with Equal Weights
<b>Table 4.12:</b>	Assigned Jobs for Jobs with Equal Weights to Machines
<b>Table 4.13:</b>	Data for Jobs with Priorities by Weights
<b>Table 4.14:</b>	Sorted Jobs for Jobs with Priorities by Weights
<b>Table 4.15:</b>	Assigned Jobs for Jobs with Priorities by Weights to Machines
<b>Table 4.16:</b>	Data for Jobs with Due Dates
<b>Table 4.17:</b>	Sorted Jobs for Jobs with Due Dates
<b>Table 4.18:</b>	Assigned Jobs for Jobs with Due Dates to Machines

## **LIST OF APPENDICES**

### **APPENDIX A**

- Appendix A.1:** Start Up Form
- Appendix A.2:** Operation Scheduling Form
- Appendix A.3:** Exit Message Box
- Appendix A.4:** Condition Selection Form (First Case)
- Appendix A.5:** Jobs with Equal Weight
- Appendix A.6:** Condition Selection Form (Second Case)
- Appendix A.7:** Jobs with Priorities Ranked By Weights Form
- Appendix A.8:** Condition Selection Form (Third Case)
- Appendix A.9:** Jobs with Due Dates Form

### **APPENDIX B**

- Appendix B.1:** Start Up Coding Structure
- Appendix B.2:** Operation Scheduling Coding Structure
- Appendix B.3:** Condition Selection Form Coding Structure
- Appendix B.4:** Jobs with Equal Weight Coding Structure
- Appendix B.5:** Jobs with Priorities Ranked By Weights Coding Structure
- Appendix B.6:** Jobs with Due Dates Coding Structure

### **APPENDIX C**

- Appendix C.1:** Oral Presentation Slides of Final Year Project

### **APPENDIX D**

- Appendix D.1:** Poster Engineering Design Exhibition (EDX) XIII

# CHAPTER 1

## INTRODUCTION

### 1.1 BACKGROUND OF STUDY

Scheduling consists of planning and prioritizing activities that need to be performed in an orderly sequence of operation. It is a tool that optimizes use of available resources. Scheduling leads to increased efficiency and capacity utilization, reducing time required to complete jobs and consequently increasing the profitability of an organization. Efficient scheduling of resources such as machines, labor, and material are a must in today's extremely competitive environment.

Scheduling methods are tools that allow production and other systems to run efficiently. The scheduling efficiency can be measured by various indexes. Two of the most popular are minimization of time required to complete all jobs or also known as makespan and minimization of penalty for completing jobs early or after due dates.

Generally, the process of scheduling requires both sequencing and resource allocation decisions. When there is only a single resource, as in the single machine model, the allocation for that resource is completely determined by sequencing decisions. But in the multiple processor models such as parallel machine system, it is a challenging problem to sequence the jobs for minimizing the setup times. At the same time; the sequence must meet the job due dates and the allocated weights.

In this project, there are several keywords that need to be understand before proceed to the next step. According to Oxford's Advanced Learner's Dictionary (1996), the definitions of the keywords are:

- ❖ *operation* : an act performed by machines
- ❖ *scheduling* : a programme of work to be done or planned events
- ❖ *parallel scheduling* : a number of identical machines are available, and jobs can be processed on any one of them.

- ❖ *sequencing* : ordering of jobs
- ❖ *knowledge-based system (KBS)* : an interface for user to comprehend the knowledge in a system

Most research in the area of industrial scheduling requires the support of highly involved and complex mathematical tools. Such mathematical techniques, which are sometimes intimidating, make understanding and hence acceptance of this project very difficult. Therefore, it is necessary to develop solutions that are easy to comprehend and thus have greater potential of being effectively used.

In determining the sequence for particular jobs, the arrangement is not simply rely only on their processing times. There are a lot of other criterions to consider in ascertain the best sequence for those jobs. The usage of computer technology in manufacturing industry has reflected the great reduction in time taken to establish the best solution at superior quality with limited resource. Thus, the integration between operating scheduling and intelligent computer system is the most welcome feasibility features that absolutely can bring the remarkable achievement for industrialists.

Hence, the KBS for parallel machine operations scheduling must be developed in order to discard the traditional method which is taken extravagant time to calculate manually. Besides the reduction of the makespan, the KBS also has the ability to be used in ‘what-if’ conditions and to avoid the repeating procedures for the same problem.

## **1.2 PROBLEM STATEMENT**

The project is intended to engage in the issue of parallel machine operation scheduling from the KBS perspective. The deal between both fields will emerge a new development of formulation and integration in artificial intelligence at area of industrial scheduling. This project will only embark upon the identical parallel processors instead of comprising with its conjugate, which are nonidentical parallel processors. This is due to the constraint of limited time during development and implementation of the project.

### **1.2.1 Project Identification**

Current operation scheduling methods require the support of highly involved and complex mathematical tools to produce the reasonable schedule. Such mathematical techniques, which are sometimes intimidating, make understanding and hence acceptance of this project very difficult. Through the tedious manual calculation to find the optimal schedule, the traditional operation scheduling consumes a lot of wasteful time that definitely reducing the productivity of the resources. Thus, the situation hindered industrialists to perform a rapid 'what-if' analysis on scheduling alternatives that might be feasible. Therefore, it is necessary to develop solutions that are easy to comprehend and thus have greater potential of being effectively used.

### **1.2.2 Significant of the Project**

Such as that, the development of KBS for parallel machines will provided an integrated system that aided the industrialists to make decision on the parallel operation scheduling issues. Consequently, the KBS also offers the more reliable, easier, simple handling tools and user-friendly interface during scheduling operations.

### 1.3 OBJECTIVE AND SCOPE OF STUDY

This main objective of this project is to develop a knowledge-based system for parallel operation scheduling. In simple approach, the details of the project are as follow:

- ❖ *To develop a knowledge base* – where theory of operation scheduling shall be integrated in one source
- ❖ *To develop an inference engine* – inference engine can be illustrated as the hub of the system where the engine will be a key factor of how the KBS will perform.
- ❖ *To develop a user interface* – a communication interface to establish the interaction between the user and the knowledge base.

The scope of the study will limit to the identical parallel machine only as the time constraint in the given time frame to accomplish the project. Every aspects related to the identical parallel machine operation scheduling will be elaborated precisely. Vitaly, the direction of the project must be always being monitored corresponds to the objectives stated to ensure the issues of the operation scheduling towards the project completion is achieved.

The concerned parameters of scheduling in retrieving the objective are:

- ❖ all jobs have equal importance and wish to minimize the makespan
- ❖ all jobs have priorities indicated by their weights and wish to minimize the makespan for each job group formed based on the priorities
- ❖ all jobs have due dates and penalty values and wish to minimize the total early and late penalties.

## **CHAPTER 2**

### **LITERATURE REVIEW / THEORY**

The literature review for this project was mostly conducted by doing a research on theory from established book, articles, journals and internet. Again, the author would like to emphasize that the basis of the project is to develop a Knowledge Based System for a parallel machine operation scheduling technique. As far as the objective are concerned, the scheduling theory for the project is already exist. The project is not interested to invent a new theoretical fact or hypothesis but merely to develop a knowledge system that is able to tackle the parallel machine issue in an easy, simple and less time consumption compared to the traditional method with the innovation of the information technology in the manufacturing industry.

#### **2.1 OPERATION SCHEDULING**

Before proceeding to the further stage, there are several root terms of scheduling operation that vitally to be acknowledged. There are Jobs, Machines, Measures and Scheduling Algorithms.

##### **2.1.1 Jobs**

Job is a piece of work to be done and to perform a job each of its operation must be processed in the order given by the routing. The processing of an operation requires the use of a particular stage for a given duration (processing time). Furthermore, it is assumed that the processing time for each job in this project is known. The job in real life may be dependent on each other and may be not. But in respect of this project, jobs are considered independent of each other.



### 2.1.2 Machines

Machines are the apparatus to perform the job. In manufacturing industry, there are two types of machines which are single machines and parallel machines. Accordingly, for single machine, there is only one machine available at a certain time to process a certain job and the job is processed one by one. Thus, the allocation of the job on single machine is completely determined by sequencing decisions (Sipper and Bulfin, 1998).

Unlike the single machine, the parallel machine is the several apparatus that able to process the same jobs simultaneously and always offer the possibility of makespan reduction. Therefore, the parallelism in the resource structure is very important to maximize the performance of machine. This includes a few job criteria like weight and due date that will impact the optimization of the machines.

### 2.1.3 Measures

A regular measure is a function of completion time in which the objective is to minimize the function and which the function only increases if at least one completion time in the schedule increases. Followings are the notations that are used in operation scheduling:

- ❖  $C$  = the completion time, the time when the job is considered complete.
- ❖  $F$  = the flowtime, the time where completion time minus the release time.
- ❖  $L$  = the lateness, the time where completion time minus the due dates.
- ❖  $T$  = the tardiness.
- ❖  $E$  = the earliness, the earliness of the job completed compared to its due date
- ❖  $\delta = 1$  if job is tardy
- ❖  $\delta = 0$  if job is on time or early

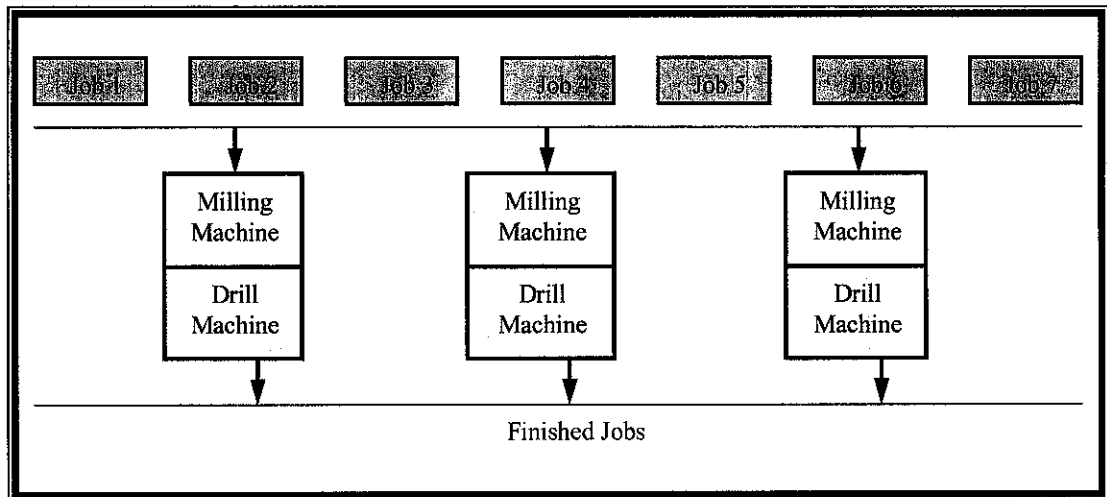
#### **2.1.4 Scheduling Algorithm**

An algorithm is a process or rule for calculation to obtain a solution to a model. There are two types of algorithm which are the exact algorithm and the heuristic algorithm. For the exact algorithm, it brings the optimal solution to every instance of the problem but for the heuristic algorithm, it reflects the solution that is almost optimal to any instance of the problem.

### **2.2 PARALLEL PROCESSING**

In single machine processing, it is assumed that there is only one facility or machine to produce all jobs. What if there are multiple facilities to provide the same services? How will the job scheduling change? How will the makespan change? The machines might be visualized as being in parallel (they do not have to be physically parallel) because they are identical to each other and are therefore performing the same services. This design is called parallel identical processors.

In the real world, there is no manufacturer that able to stand successfully without adopting parallel operation in their work. For example in early 1900s, The Ford Motor Company which invented by an American industrialist named Henry Ford, was capable to become the only major manufacturer of automobiles in Detroit (Johnston, 2000). Henry Ford was introduced the standardized interchangeable parts and parallel assembly line techniques in his plant. Before this innovation, automobiles used to be assembled by highly skilled craftsmen one by one which result the long makespan for each unit of the car and hindered the customers' satisfaction. This parallel assembly line had greatly boosted the productivity and lead to the mass production. Consequently, the sales of Ford automobiles were vastly increased at about 15 million of cars.



**Figure 2.1:** Typical Parallel Processing Setup

Figure 2.1 illustrates a good example of the configuration of typical parallel machines. There is a group of identical metal milling and drilling machines, each of which can make a number of different metal part.

In parallel processing, jobs are processed by one of several identical machines, allowing considerable reduction in makespan. The interest area needed to be figured out is the usage of identical processors to obtain the minimum makespan.

In order to determine the method of parallel machine operation scheduling, the objectives and the assumptions of the scheduling must be laid out first. For every objective that is being set will present different method of scheduling. Among the objectives of scheduling are:

- i. Minimizing Flowtime for Normal Jobs
- ii. Minimizing Flowtime for Weighted Jobs
- iii. Minimizing Earliness and Tardiness with a common Due Date

The methods of scheduling will differ from one to another depending on the objective that is tried to achieve.

### 2.2.1 Minimizing Flowtime for Normal Jobs

Under this objective, several assumptions of the problem need to be made at the initial stage. The assumptions are all jobs to be scheduled are available, and the release times are all zero, meaning that flowtime is equal to completion time. The sequence of jobs ordered from smallest to largest is the Shortest Processing Time (SPT) sequence (Sipper and Bulfin, 1998). For example, there are three jobs available namely Job 1, Job 2 and Job 3. The processing times for the jobs are 6, 7, and 3 respectively. By using the SPT method, the sequence of the job should be Job 3 – Job 1 – Job 2.

One problem with minimizing the total flowtime is that all jobs are assumed equally important or valuable, which is not always true. In fact, minimizing the flowtime is equivalent to minimizing the number of jobs in inventory. However, the value of inventory is usually more important than its size (Sipper and Bulfin, 1998). To handle this problem, weighted flowtime method is suggested as solution of this problem.

### 2.2.2 Minimizing Flowtime for Weighted Jobs

Let  $w_i$  be the weight of job  $i$ , where a larger weight means the job is more important or valuable. The total value of inventory for some schedule is:

$$\sum w_i C_i = w_1 C_1 + w_2 C_2 + \dots + w_n C_n \quad \text{where} \quad w = \text{weight}$$

$C = \text{completion time}$

Flowtime is related to customer waiting time, so if all customers or jobs are not equally important, the weighted measure is appropriate. A job with a small processing time and a large weight should be scheduled toward the front, whereas one with a large processing time and a small weight should be scheduled toward the back.

In order to do this the *ratio of processing time to weight* must be considered, and the order of the jobs should be in increasing ratio. This is usually called the Weighted Shortest Processing Time (WSPT) sequence. For example, if Jobs 1, Job 2 and Job 3 have the processing time of 6, 7 and 3 respectively while the weight are 2, 3 and 1 respectively. Now, the calculating ratio of respective job gives the result of  $6/2$ ,  $7/3$  and  $3/1$ . Job 2 has the smallest ratio and should come first while Job 1 and 3 are at the same ratio. Either job can follow after Job 2. The sequence is Job 2 — Job 1 — Job 3.

### **2.2.3 Due Dated Jobs**

If customer satisfaction is the overriding measure of performance, the due dates of the jobs must be considered. SPT sequencing does not consider due dates, such as that the schedules that are good for flowtime may be bad for due date oriented measures (Sipper and Bulfin, 1998). Among of the due dates oriented measures are maximal tardiness,  $T_{\max}$  and maximal lateness,  $L_{\max}$ . The lateness is the excess times for a job to be completed after its due date whereas the tardiness is the how slow to perform a certain job beyond the right or the expected time. The job is tardy when its completion is beyond the expected due date and the different between the completion time to the exact due date is defined as the lateness. Thus, both terms are always interrelated and the intention is always to reduce the number of tardy jobs as well as the lateness of the jobs.

To minimize the maximal tardiness, the due dates must be involved. It is reasonable to put the job with the earliest due date first, and proceed with the following. This method is called the Earliest Due Dates (EDD) sequence. For example, Job 1, Job 2 and Job 3 have the due dates of 7, 3, and 8 respectively. Therefore, the sequence of the jobs should be as Job 2 — Job 1 — Job 3.

### 2.3 KNOWLEDGE BASED SYSTEM

The main vital role subject come after parallel processing in this project is the KBS. The KBS, sometimes called as knowledge system, could deliver a few definitions depending on how it is being perceived. It offers the following seven meanings relevant to the purposes (Stefik,1995).

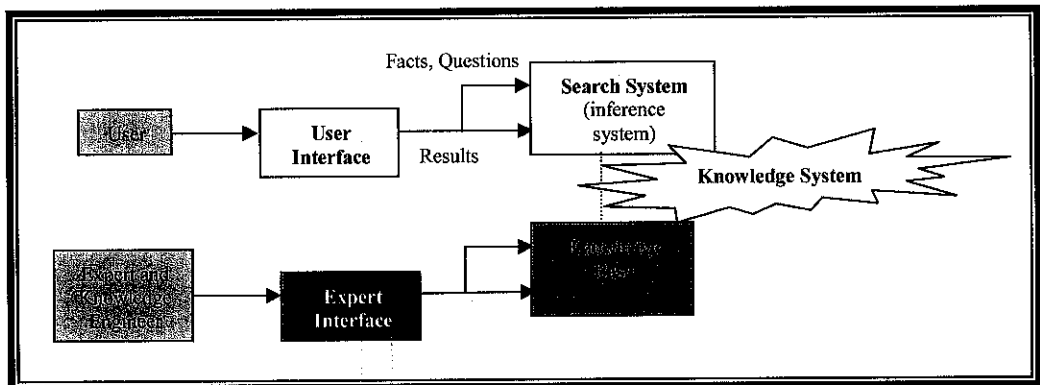
- i. A clear and certain perception of something; the act, fact or state of knowing; understanding.
- ii. Learning; all that has been perceived or grasped by the mind.
- iii. Practical experience; skill; as knowledge of seamanship.
- iv. Acquaintance or familiarity such as with a fact or place.
- v. Cognizance; recognition.
- vi. Information the body of facts accumulated by mankind.
- vii. Acquaintance with facts; range of awareness, or understanding.

All these definitions are concentrating on the intermediate rational agent that generated by knowledge to address the objectives of the project. The rational agent is any human or software that is able and logical to execute a certain task in a certain domain.

The term expert system refers to a computer system whose performance is guided by specific, expert knowledge (Stefik, 1995). It indicates that the computer has the ability same as the highest level of competency for human or even better. Thus, the expert system will reflect the reliability and consistency in the results. However, in flexibility aspects, the human experts become more preferable.

The term knowledge system refers to a computer system that represents and uses knowledge to carry out task (Stefik, 1995). Now, the interest part is more on how to manipulate and deliver out the knowledge through an integrated system which will presented to the end user.

Basically, the classical formulation of KBS consists of the knowledge base itself, two interfaces (for user and for expert), and the search system where the inference is located. These elements might be blended or overlapped as to echo the desired system.



**Figure 2.2:** Classical Characterization of Knowledge System (Stefik, 1995)

Each particular component has its own vital role in knowledge system which described as following (Stefik, 1995):

- i. *Knowledge base* – the repository for the knowledge used by the system (the rules and hints for guiding the search for solution).
- ii. *User interface* – the part of a knowledge system that interacts with the system’s primary users.
- iii. *Expert interface* – interface by which knowledge is entered into the system. The expert interface is used by a knowledge acquisition team consisting of an expert and a knowledge engineer.
- iv. *Inference engine* – the inference subsystem is the part that reasons its way to solutions of problems, with its search guided by the contents of the knowledge base. This part of a knowledge system must include provisions for setting goals, representing and recording intermediate results and managing memory and computational resources.

There is always ambiguity perceptible in using the word of data, information and knowledge. They seem to have the similar meaning and always being used interchangeably but the fact is that they definitely have slight different meanings. The words of knowledge engineering, information technology or data technology need the clarity in terms of definition to avoid misinterpretation in delivering the knowledge. The differences of these words are given as followings (Schreiber et al, 2000):

- ❖ **Data**: the uninterpreted signals that reach our senses every minute by the zillions.
- ❖ **Information**: data equipped with meaning.
- ❖ **Knowledge** : the whole body of data and information that people bring to bear to practical use in action, in order to carry out tasks and create new information.

Knowledge for certain people might be data for others. As an example an expert in computer system that dealt with binary numbers, should understand and know what the information the binary numbers bring and hence able to grasp the knowledge of the information. But a civil engineer or plain man perceives the binary numbers only as a data and cannot understand the information, hence unable to grasp the knowledge represented by the binary numbers. Thus one person's knowledge is might be another person's data (Schreiber et al, 2000)

The benefits of the development of knowledge system are the result of advancement in computer and artificial intelligence. With the progression of faster computer and artificial intelligence, more jobs can be achieved at faster rate, and at higher reliability. The top three benefits of knowledge system are (Schreiber et al, 2000):

- i. Faster decision making
- ii. Increased productivity
- iii. Increased quality of decision-making

Since the benefits of knowledge system out weight the cost to develop it, then it is worthwhile to concentrate on the development of knowledge system. This is concurrent with the strategy of achieving the higher productivity of the manufacturing process.



### 2.3.1 Knowledge System Terminology

Before going into depth of KBS, there are certain terminologies that must be beneficial to be familiarized (Schreiber et al, 2000):

- ❖ **Domain:** A domain is some area of interest. In the respect of this project, the domain of this project is single machine operation scheduling.
- ❖ **Task:** A task is a piece of work that needs to be done by an agent. In respect of this project, task that shall be performed by the knowledge system is to schedule operation for single machine.
- ❖ **Agent:** An agent is any human or software system able to execute a task in a certain domain. In respect of this project the agent is Microsoft Visual Basic software.
- ❖ **Application:** An application is the context provided by the combination of a domain and a task carried out by one or more agents. In respect of this project, the application is the whole knowledge system for single machine operation scheduling that was developed.
- ❖ **Application domain / task:** These two terms are used to refer to the domain and / or task involved in a certain application.
- ❖ **Knowledge system:** Two main components are a reasoning engine (inference) and a knowledge base. In respect of this project, the inference engine is the rules that are being used.
- ❖ **Expert system:** It can be defined that an expert system as a knowledge system that is able to execute a task that, if carried out by humans, requires expertise. In respect of this project, the knowledge system is not yet fulfilling the criteria of expert system.

### 2.3.2 Domain Knowledge

The domain knowledge will rely on the main static information and knowledge objects in an application domain. Normally there are two parts of domain knowledge, Domain schemas and knowledge bases.

- ❖ A *domain schema* is a schematic description of the domain-specific knowledge and information through a several number of type definitions (Schreiber et al, 2000).
- ❖ A *knowledge base* contains instances of the types specified in a domain schema (Schreiber et al, 2000).

#### Domain- Schema Specification

The knowledge model provides a set of modelling constructs to specify a domain schema of an application. In practice there are three main modelling constructs, which are concepts, relation and rule type (Schreiber et al, 2000).

- ❖ **Concepts:** A concept describes a set of objects or instances which occur in the application domain and which share similar characteristics. The notion of concept is similar to what is called class or object classes in other approaches.
- ❖ **Relation:** Relations between concepts are defined with the relation or binary relation construct. Relations can be used in the standard entity-relationship fashion, but can also be used for more complicated types of modelling.
- ❖ **Rule:** The dependencies of a schematic form are a sort of natural rules, indicating a logical relationship between two logical statements. The logical statements in such rules are typically expressions about an attribute value of a concept. An example of rule type is:

**If...then...else.**

## Knowledge Base

A domain schema describes domain knowledge types, such as concepts, relations and rule types. A knowledge base contains instances of those knowledge types. A knowledge base consists of two parts:

- ❖ **Uses:** The uses slot defines which type of domain knowledge instances is stored in the knowledge base.
- ❖ **Expressions:** The expressions slot contains the actual instances.

The example of the sample knowledge base is as follows (bubble sorting of Processing Time):

```
For counter2 = 0 To 8 Step 1
x = 8 - counter2
  For counter3 = 0 To x Step 1
    If intProsTime(counter3) > intProsTime(counter3 + 1) Then
      TempVar = intProsTime(counter3)
      intProsTime(counter3) = intProsTime(counter3 + 1)
      intProsTime(counter3 + 1) = TempVar

      TempVar = intJobNum(counter3)
      intJobNum(counter3) = intJobNum(counter3 + 1)
      intJobNum(counter3 + 1) = TempVar
    End If
  Next counter3
Next counter2
```

**Figure 2.3:** ‘Bubble Sort’ Coding Structure

As a whole, this project will attempt the usage of rule types of knowledge based to develop Knowledge-Based System for Parallel Machine Operation Scheduling. The founding theories of the parallel machine were discussed earlier in the first part of this chapter.

## **CHAPTER 3**

### **METHODOLOGY/PROJECT WORK**

#### **3.1 PROCEDURE IDENTIFICATION**

The author had carried out several case studies to have better understanding on the related the objectives in the project. The case studies are:

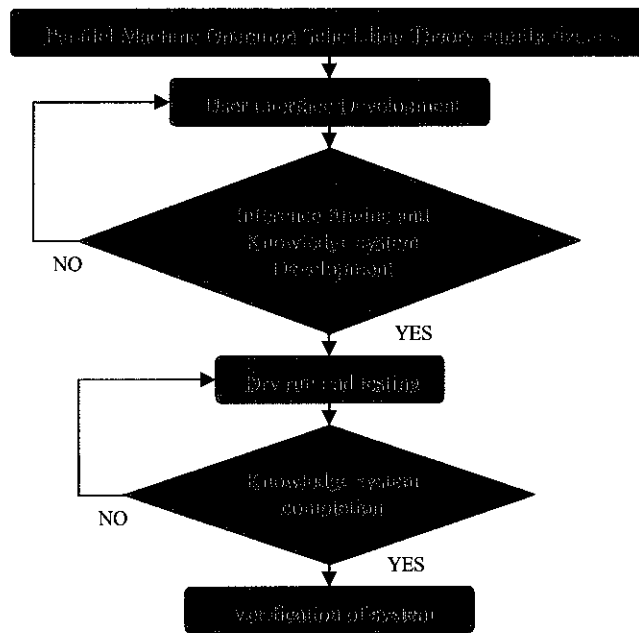
- ❖ **CASE STUDY I : Parallel Processing – Jobs of Equal Weight**
- ❖ **CASE STUDY II: Parallel Processing – Weighted Jobs**
- ❖ **CASE STUDY III: Parallel Processing – Jobs with Due Dates**

These case studies were performed precisely as it becomes the easiest approach for the author to understand the wholly about subject matter. All the steps taken are completely listed one by one as well as the calculations involved for each case that are clearly presented and simplified in the particular tables.

#### **3.2 KNOWLEDGE BASED SYSTEM DEVELOPMENT**

In simulating the KBS, the author must undergo several sequenced phases. These phases demand a conscientious planning to ensure the successful development of the KBS. The development of the knowledge system can be divided into six phases, bear in mind that the knowledge system for parallel machine operation scheduling was developed from one algorithm to another. Among the algorithms are Shortest Processing Time, Weighted Shortest Processing Time and Earliest Due Date.

Figure 3.1 is use to represent the steps that have been taken to ensure the success of the knowledge system development. The steps shown were repeatedly taken in order to develop the knowledge system, one by one theory. After that when all the completed algorithms combined together, this will complete the knowledge based system for parallel machine operation scheduling.



**Figure 3.1:** Flowchart of Knowledge System Development

During the development of the knowledge system, the whole development was done in the expert interface. When there is any failure during the development of the knowledge system, the development of the system will return back to the previous phase for modification or study. The actions that were performed during each stage are explicitly elaborated as in Table 3.1:

**Table 3.1: Steps of Knowledge Based System Development**

No	Phases	Action
1	<b>Theory Familiarization</b>	At, this phase, the theories of the parallel machine were familiarized and understood. The steps to reach the solution and sequence of jobs were studied and outlined.
2	<b>User Interface Development</b>	At this phase, the user interface will build by gathering all the application data required to produce the sequence of jobs.
3	<b>Inference engine and knowledge base development</b>	The inference engine, the knowledge base and all the rules-based will be developed at this phase. If there still insufficient information to develop KBS, return to the previous phase.
4	<b>Dry run and testing</b>	After the completion of the inference engine, user interface and knowledge base, the system will be being tested in trial mode. Any modification is done here.
5	<b>Knowledge system completion</b>	At this phase, the inference engine, user interface and knowledge base will be finalized towards the completed version. If there still any defect, the modification should be done in previous phase.
6	<b>Verification of system</b>	The verification of the system is then compared to the traditional methods.

### **3.3 TOOL FOR KNOWLEDGE SYSTEM DEVELOPMENT**

The objective of developing Knowledge-Based System for Parallel Machine Operation Scheduling presents the task that requires the usage of specific tool, more accurately a programming language. The programming language should fulfill the requirement of Knowledge Based System itself. As explained in previous chapter, the Knowledge Based System consists of four elements, which are:

- i. Knowledge base
- ii. User interface
- iii. Expert interface
- iv. Inference engine

The tool chosen to build the knowledge system should be able to fulfill the advantages/criteria as listed below:

❖ ***Developer interface:***

Ability to develop the user interface of the knowledge system and more likely referred as expert interface.

❖ ***Code form:***

The code is easy to use as its user-friendly functions and simple.

❖ ***Feasibility to modify:***

During program test run, the KBS is can be simply altered or update to meet the applicable requirement.

Hence, tool named **Microsoft Visual Basic 6.0** was selected as the medium for developing the knowledge system to convey the objectives of the project. Microsoft Visual Basic 6.0, the programming language that is globally used to build almost program/software in the market was selected due to its advantages for achieving the aim to develop the knowledge system for parallel machine operation scheduling.

### **3.4 SYSTEM LAYOUTS**

For the development of the knowledge system, it is advisable to line out the tree chart during the initial stage. This tree system is important to determine the constraints and limits of the system as well as to serve the guides to the project during the implementation.

#### **3.4.1 Parallel Machine**

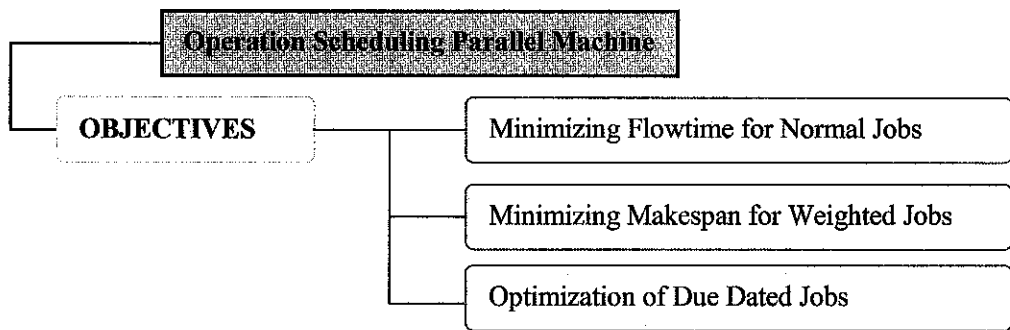
Under the title of parallel machine, the objectives of the project are well-presented. The user will then introduce to three conditions; the conditions are Jobs with Equal Weights<sup>1</sup>, Job with Priorities by Weight<sup>2</sup> and Jobs with Due Dates<sup>3</sup> as shown in Figure 3.2. From all of these options, it is intended for the user to choose the only one option. For each option, it will present the user with new set of options.

---

<sup>1</sup> Case Study I

<sup>2</sup> Case Study II

<sup>3</sup> Case Study III



**Figure 3.2:** Tree Chart for Parallel Machine Interface Layout

For every objective that the tree presents, it will use different algorithm in solving the model. The founding theory of every algorithm was presented in previous chapter. For the objectives that are being selected, user will dedicate different algorithm to solve the model. The founding theory that being used in the knowledge system is given as in Table 3.2:

**Table 3.2:** The Founding Theory of the Objectives

Objectives	Algorithm
Minimizing Flowtime	Shortest Processing Time  With the usage of this algorithm, the objective of minimizing flowtime can be achieved. This algorithm is the most basic algorithm in operation scheduling theory.
Weighted Flowtime	Weighted Shortest Processing Time  The usage of this algorithm is intended to minimize flowtime, the same objective with SPT. The difference is the inclusion of weight where each job is perceived differently in term of it importance.
Weight To Processing Time Ratio	Weight to Processing Time Ratio  This algorithm is specifically use to minimize tardiness. This algorithm can be seen as the reciprocal of WSPT algorithm
Minimizing Lateness	Earliest Due Dates  This algorithm is used to avoid maximum lateness and tardiness of a job. In order to minimize the tardiness and lateness, job with earliest due dates should be processed first and then on and on until all the jobs are done.



The methodology to develop the knowledge system is definitely differs as compared to the engineering-based experiment. Towards the completion of development of knowledge system, the allocated procedures were used repeatedly in order to achieve the successful objective of the project.

## **CHAPTER 4**

### **RESULT/DISCUSSION**

This chapter is intended to discuss the final outcome of the knowledge system development. The knowledge system was built on the foundation that has been described by the tree chart. By constantly guided by the tree chart, the project is ensured not to deflect from its objectives, not to venture out from the constraints and the tree chart also serves as the schedule of the knowledge system development schedule itself.

Here, the result of the case studies are elaborated explicitly and come with the steps involved which may illustrate the method used to the end user. This chapter also offers a lesson on how to use knowledge system. Explanations about the usage for each form are specified and the data required for respective form are listed in this chapter. As a whole, this chapter serves as the manual for the knowledge system. This chapter also presents the verification of the knowledge system, to compare and validate the sequence produced by the knowledge system and manual calculation.

The tools that was used to develop the project itself; Microsoft Visual Basic 6.0, requires in depth experience in order to develop the knowledge system. During the development, several experts on the tool were consulted in order to solve the hindrance that occurred.

## 4.1 CASE STUDIES

### 4.1.1 CASE STUDY I: Parallel Processing – Jobs of Equal Weight

Suppose there are ten jobs and three processors. Here, the desired outcome is to develop the detail schedule for each processor that will minimize the makespan. Processing times are as follows:

**Table 4.1: Jobs Available for Case Study I**

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9

In the first step, the jobs are arranged in decreasing order of their processing times<sup>4</sup>:

**Table 4.2: Sorted Jobs for Case Study I**

Job	6	3	7	5	4	10	1	2	9	8
Processing Time	3	5	5	7	8	9	10	12	12	15

The sum of the processing time is 86, and therefore with three processors the minimum makespan is  $86/3 = 29$  (the numbers are round up). This number is noted as the *remaining cumulative time* (RCT) allowed in the associated column for each machine in Table 4.3. Each time a job is assigned to a machine, it is noted in the table along with its processing time. The available cumulative time is then decreased by this value.

**Table 4.3: Job Assignments on Each Machine for Case Study I**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Processing Time	RCT=29	Job	Processing Time	RCT=29	Job	Processing Time	RCT=29
6	3	26	10	8	21	2	1	28
3	5	21	1	10	11	9	12	16
7	5	16	2	12	-1	8	15	1
5	7	9						
4	8	1						
10	9	-8						

<sup>4</sup> For the flexibility of end users, the processing time for each job is defined as 'unit', which refers to minutes/hours/days as desired by the users.

Select the first job (Job 6) from the list and assign it to the first available machine. The total available cumulative time of 29 is reduced by 3 (Job 6's processing time) which resulted to 26. The next job in the list is Job 7, with the processing time of 5. After considering Job 7, the RCT value reduced to 21 ( $26 - 5 = 21$ ). This process is repeatedly done until the RCT value is zero or in negative value.

When the negative value occurred, the preemption of the job is assumed to be allowed. Thus, the unfinished job will be proceeding to the next available machine. As an example, Job 10 has the processing time of 9 and the available RCT of MACHINE 1 is only 1 unit. After processed at MACHINE 1 for 1 unit, the job is transferred to MACHINE 2 for the execution of the balance which is another 8 units.

All these steps and conditions are applied to all jobs to all available machines until the task is completed. In MACHINE 1, the precedence of assigned jobs was 6-3-7-4-5-10 whereas for MACHINE 2, were 10-1-2. Meanwhile in MACHINE 3, the precedence of the jobs was 2-9-8. As conclusion, the total RCT value for each machine is end up with the same value which indicates that the machines work simultaneously to process the 10 jobs within 29 units of processing time. It reflects that the parallel machines effectively can reduce the makespan of the jobs.

#### 4.1.2 CASE STUDY II: Parallel Processing – Weighted Jobs

Suppose each job in the previous case study has a weight indicating its priority (the higher the weight, the higher the priority), as shown in the following table:

**Table 4.4:** Jobs Available for Case Study II

Job	1	2	3	4	5	6	7	8	9	10
Processing time	10	12	5	8	7	3	5	15	12	9
Weight	3	2	4	2	4	3	2	1	1	6

In fact, there were also three processors available. Because there is association of weight in the process, thus the ratio between the processing times to the weights will be used. Each ratio for each job is calculated and being sorted in increasing mode.

**Table 4.5:** Sorted Jobs for Case Study II

Job	6	3	10	5	7	1	4	2	9	8
Processing time	3	5	9	7	5	10	8	12	12	15
Weight	3	4	6	4	2	3	2	2	1	1
Ratio	1.00	1.25	1.50	1.75	2.50	3.34	4.00	6.00	12.00	15.00

After sorting the jobs according to their respective calculated ratio, the jobs now will be assigned to the available machines. Here the new term, remaining cumulative ratio (RCR) will be introduced. The RCR for each machine is the sum of the ratio divided by the number of available of the machines ( $48.34/3 = 16.12$ ). Notice that the actual value of the operation is 16.11333 but the figure is rounded up to 16.12 instead of 16.11. Commonly in manufacturing, when the product made is counted in decimal number, the total product is rounded into the upper bound.

**Table 4.6: Final Assignments on Each Machine for Case Study II**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12
6	1.00	15.12	2	5.22	10.9	9	-1.1	15.02
3	1.25	13.87	9	12.00	-1.1	8	15.00	0.02
10	1.50	12.37						
5	1.75	10.62						
7	2.50	8.12						
1	3.34	4.78						
4	4.00	0.78						
2	6.00	-5.22						

From the Table 4.6, it can be observed that the MACHINE 1 is executed the precedence of jobs 6-3-10-5-7-1-4-2, for MACHINE 2 the jobs were 2-9 and MACHINE 3, the assigned job were 9-8.

### 4.1.3 CASE STUDY III: Parallel Processing – Jobs with Due Dates

Consider ten jobs with the following data:

**Table 4.7: Jobs Available for Case Study III**

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9
Weight	1	1	1	2	3	2	4	6	4	3
Due Date	26	32	38	48	51	64	53	50	35	28

Assumed that there were three parallel processors and the objective is to schedule the jobs to minimize the tardiness penalty. The jobs were sorted increasingly according to their respective due dates. Then processing time for the series of the job is summed up as accumulative completion time. The tardiness job can be identified when once the time to execute the job is exceeded the due date precedence. The example of the problem is illustrated in Table 4.8. From the table, it can be observed that 7 out of 10 allocated jobs were tardy (more than “0” value in Tardiness row).

**Table 4.8: Tabulated Data to Identify Tardiness of the Jobs in Case Study III**

Job	1	10	2	9	3	4	8	5	7	6
Processing Time	10	9	12	12	5	8	15	7	5	3
Weight	1	3	1	4	1	2	6	3	4	2
Due Date	26	28	32	35	38	48	50	51	53	64
Completion Time	10	19	31	43	48	56	71	78	83	86
Tardiness	0	0	0	8	10	8	21	27	30	22

After sorting the jobs according to their respective due date which is 1-10-2-9-3-4-8-5-7-6, the jobs are now will be assigned to the available machines. Here again the new term, remaining cumulative due dates (RCD) will be introduced. The RCD for each machine is the sum of the due date values divided by the number of available of the machines ( $425/3 = 142$ ).

**Table 4.9: Final Assignments on Each Machine for Case Study III**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Due Dates	RCD=142	Job	Due Dates	RCD=142	Job	Due Dates	RCD=142
1	26	116	3	17	125	5	24	118
10	28	88	4	48	77	7	53	65
2	32	56	8	50	27	6	64	1
9	35	21	5	51	-24			
3	38	-17						

From the Table 4.9, it can be observed that MACHINE 1 is executed the precedence of jobs 1-10-2-9-3, for MACHINE 2 the jobs were 3-4-8-5 and lastly but not least for MACHINE 3, the assigned jobs were 5-7-6.



## 4.2 INTRODUCTORY FORMS

### 4.2.1 Start-Up Page



**Figure 4.1:** Start Up Form

The start up form in Figure 4.1 is intended to serve as information page for the user. This form does not serve any operation scheduling theory, it is merely used to inform the user about the knowledge system that the user going to use, the program version, and the information of the developer.

### 4.2.2 Operation Scheduling Page

The Operation Scheduling Page<sup>5</sup> form is intended as the first page that the user will come across. The user will then choose either to go on with the parallel machine or exit from the knowledge system. When the user clicks the button of "Parallel Machine", they will then be directed to Select Condition Page. If the user clicks the Exit button, a message box<sup>6</sup> will appear and ask for confirmation of either leaving or stay in the system.

---

<sup>5</sup> Please refer to Appendix A.2

<sup>6</sup> Please refer to Appendix A.3

### 4.2.3 Select Condition Page

In Select Condition<sup>7</sup> Page, the user would have THREE options to choose. It is either the *Jobs with Equal Weights* option, *Job with Priorities by Weight* option or *Jobs with Due Dates* option. Here, as clearly defined before, is intended for user to choose the related objective to the approach of problem solving. When the user clicks the selected option, the user will be directed to respective page.

## 4.3 OBJECTIVE FORM

The Select Condition form contains all the options that the knowledge system offered in term of the target the user wish to select. The user could choose any of the options presented by the Objectives form and will be directed to the respective form.

### 4.3.1 Jobs with Equal Weights

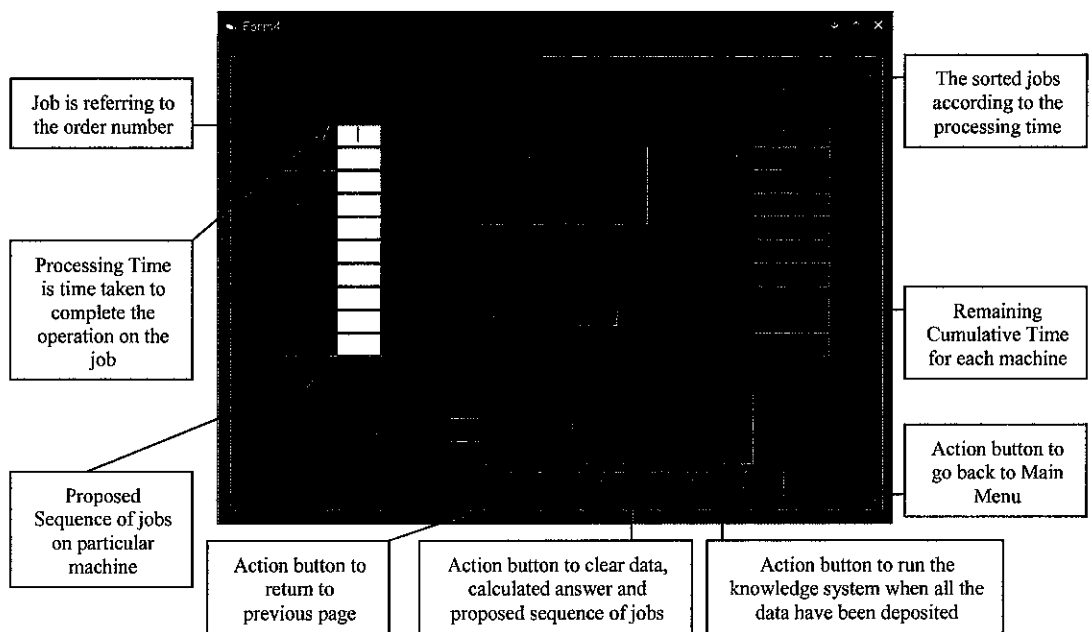
- i. In this form<sup>8</sup>, user is expected to input the processing times of their respective jobs in the text box.
- ii. When all the respective processing times have been filled, the user clicks the “Sort” button. The sequence of the jobs will then appear in the job precedence box.
- iii. The user can then clicks on the clear screen button to clear all the data for next batch of data or click back to return to previous page.

All of these buttons are the standard button that could be found throughout the usage of the knowledge system. An example of a form that contains the action buttons listed and the respective text box is shown in Figure 4.2.

---

<sup>7</sup> Please refer to Appendix A.4

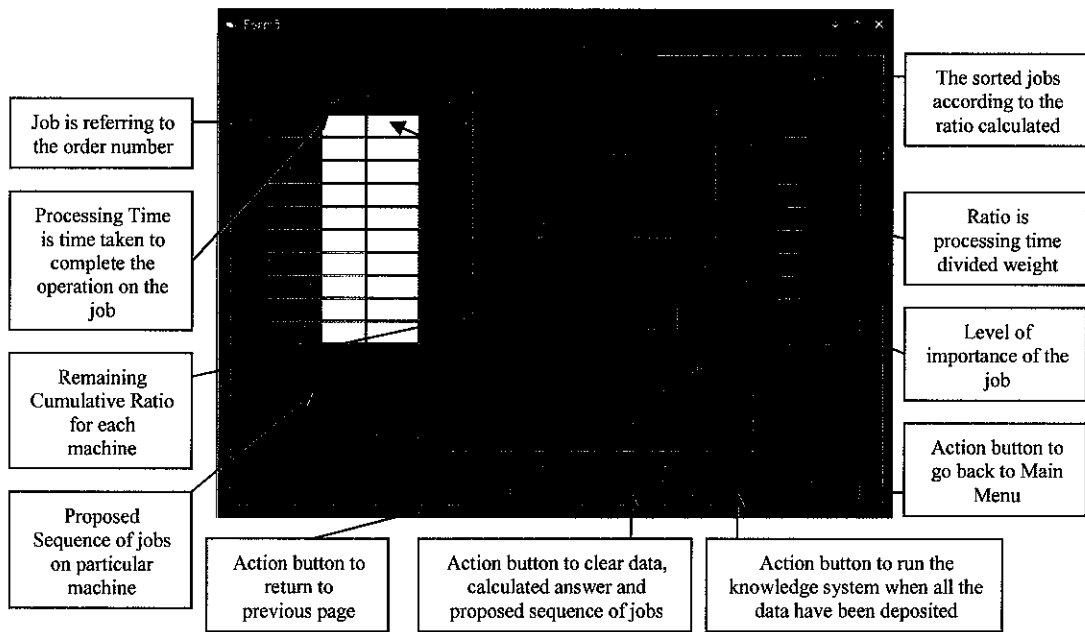
<sup>8</sup> Please refer to Appendix A.5



**Figure 4.2: Jobs with Equal Weight Form**

#### 4.3.2 Jobs with Priorities and Ranked by Weight

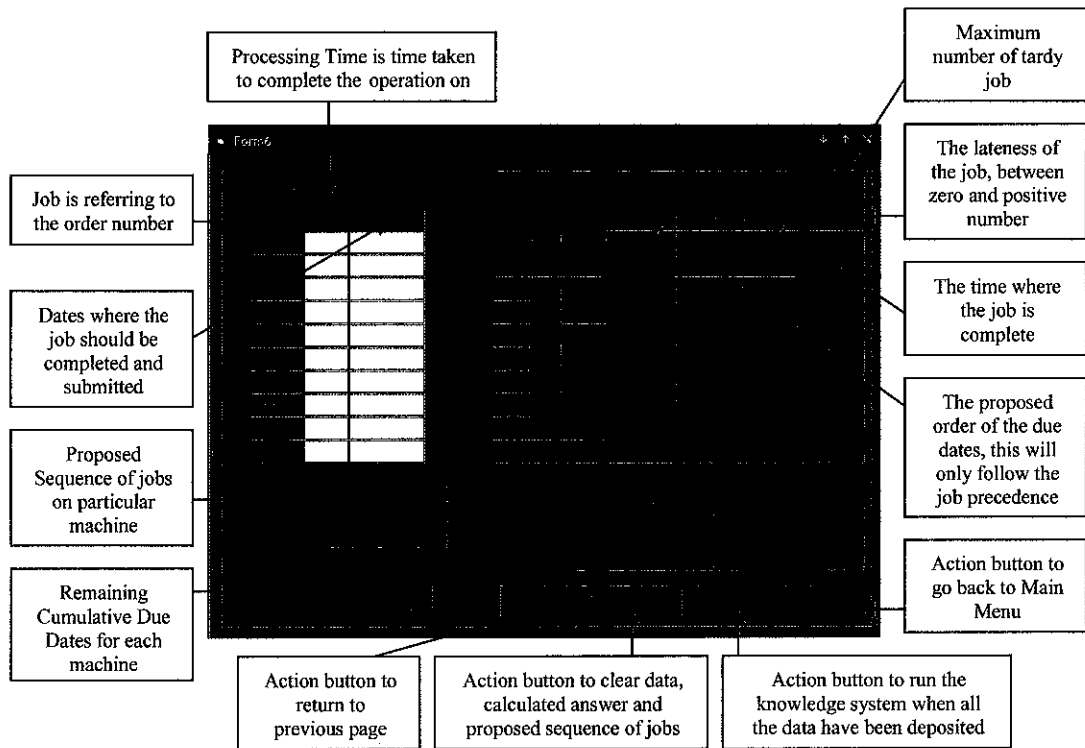
- i. In Weighted Flowtime form as shown in Figure 4.3, the user is expected to fill in the data regarding the processing time and the weight of the jobs.
- ii. When all the data have been filled in, the user clicks the "SORT!" button. The ratio box will then filled in with the ratio of processing time over weight. At the same time the knowledge system will arrange the sequence of jobs in job precedence area.



**Figure 4.3: Jobs with Priorities and Ranked by Weight Form**

### 4.3.3 Jobs with Due Dates

- i. The main objective of the form is to find the maximum tardiness. In the form as shown in Figure 4.4, the user is expected to fill in the data regarding the due dates and the processing time.
- ii. When the data has been completely fill in, the user can click the proceed button.
- iii. By clicking this proceed button; the sequence of the job is determined by using EDD method.
- iv. At the same time too, the number of jobs that are tardy will be displayed in the maximum lateness area.



**Figure 4.4:** Jobs with Due Dates Form

#### 4.4 THE CODING STRUCTURES

The well-functioned coding structure is very important to ensure the successful of the knowledge system. A few books had being referred to gain a better understanding on developing the coding (See REFERENCES).

Due to limited knowledge in programming, the author used the trial and error approach in doing the task. This required a significant long of time taken and finally, the mission was accomplished.

The coding is run and tested several times in order to avoid null algorithm in the structure. All the identifiers were named accordingly to the respected box such as *txt\_prostime1* for text box to insert processing time for Job 1. This is purposely intended as to prevent confusion later when calling the collaborated function.

The complete coding structures up to the latest one are presented in the Appendix B. These coding are being finalized as the whole structure of the program is already completed.

#### **4.4.1 Minimizing Flowtime and Shortest Processing Time**

For the coding structure of minimizing flowtime and SPT, both of them are using the same code structure to obtain the solution for the job sequence. However, the difference between the two is in the code structure of the form. Usually this will be the most significant difference from one form to another.

In both of this code structure, the theory is SPT algorithm. Such as that, processing time will play such an important role to ensure that the knowledge system is able to adhere to the rule that have been laid. In both of this form a coding structure called 'bubble sort' is used. Throughout the project, the most common coding structure that will be used is 'bubble sort' (See Figure 2.3). The differences between one from another is the data that is needed and the way to manipulate the information. This is to ensure that the job with smallest processing time will be sequenced first. This is done repeatedly until all the jobs have been sequenced properly. The 'bubble sort' is shown in Figure 4.5. After the job sequenced properly it will then displayed in its new ordered sequence.

#### **4.4.2 Weighted Flowtime and Weighted Shortest Processing Time**

As mentioned earlier, the 'bubble sort' coding structure is the most common coding structure that is used in the knowledge system. In these code structure too, for weighted flowtime and WSPT, the 'bubble sort' is used again. However the sorting is done depending on the ratio. Such as that from the information that has been supplied by the user, the ratio of processing time over ration would then be calculated. When the ratio is obtained, later then the jobs will be sequenced using 'bubble sort'. This step is done repeatedly until all the jobs have been assigned to its new sequence.

#### **4.4.3 Earliest Due Dates**

For all of these coding structures, the 'bubble sort' coding structure will arrange the job sequence depending on the due-dates. Such as that the job with earliest due-dates will be sequenced first and followed by the job with latest due-dates. The processing time however is required in this coding structure to calculate its tardiness and lateness.

#### 4.5 VERIFICATIONS OF KNOWLEDGE SYSTEM

In order to ensure the successfulness of the knowledge system, the verification of the knowledge system should be tested. This is to proof the knowledge system is fault tolerant and reliable.

This section is prepared to serve as examples of manual calculation for the Knowledge System developed. In this section, knowledge system output and manual calculation for each algorithm in the Knowledge System is presented here. For simplicity, each manual calculation in this paper shall use the same set of data except for several algorithms. The same goes with data for the knowledge system. This is to ensure fair judgment and justification for the comparison of both methods. The data was excerpt from Sipper and Bulfin, (1998) **Production: Planning, Control and Integration**, McGraw-Hill Book Co, Singapore, International Edition.

To verify the knowledge system, manual calculation and steps would be done first, the result would later then be compared to the output produced by the knowledge system. This section will visit each components of the knowledge system and compare it with the manual calculation.

#### 4.5.1 Jobs with Equal Weights

##### Manual Calculation Output (Shortest Processing Time (SPT) Method)

**Table 4.10: Data for Jobs with Equal Weights**

<b>Job</b>	1	2	3	4	5	6	7	8	9	10
<b>Processing Time</b>	10	12	5	8	7	3	5	15	12	9

*Assumption:*

- ❖ Release times are all zero.
- ❖ All jobs are the same important level.
- ❖ The processing time for each job is known.
- ❖ There are only three machines available and they are in tip top condition.
- ❖ Preemption is allowed.

The sequence of the jobs ordered from smallest to the largest is the Shortest Processing Time (SPT) sequence.

Applying SPT the sequence of the job is: **6-3-7-5-4-10-1-2-9-8**

**Table 4.11: Sorted Jobs for Jobs with Equal Weights**

<b>Job</b>	6	3	7	5	4	10	1	2	9	8
<b>Processing time</b>	3	5	5	7	8	9	10	12	12	15

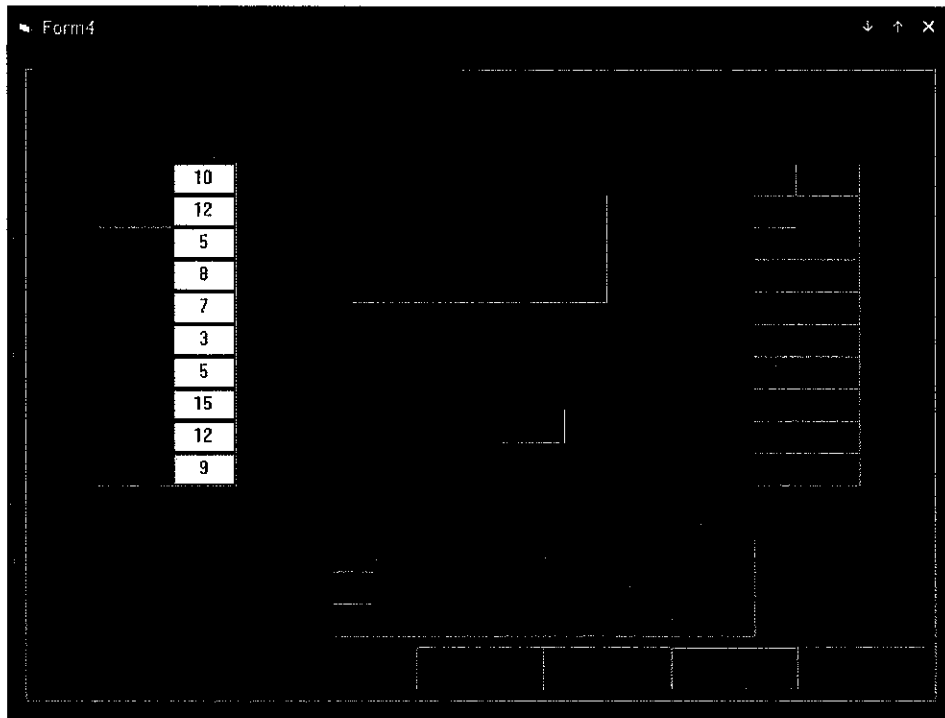
After manually assign each job to the three available machines, the outcome is:

**Table 4.12: Assigned Jobs for Jobs with Equal Weights to Machines**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Processing Time	RCT=29	Job	Processing Time	RCT=29	Job	Processing Time	RCT=29
6	3	26	10	8	21	2	1	28
3	5	21	1	10	11	9	12	16
7	5	16	2	12	-1	8	15	1
5	7	9						
4	8	1						
10	9	-8						



## Knowledge System Output



**Figure 4.5:** Result for Shortest Processing Time

Since the knowledge system produced the same sequence as the manual calculation, this component is verified.

## 4.5.2 Job with Priorities by Weight

### Manual Calculation Output (Weighted SPT (WSPT) Method)

**Table 4.13:** Data for Jobs with Priorities by Weights

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9
Weight	3	2	4	2	4	3	2	1	1	6

*Assumption:*

- ❖ Release time are all zero.
- ❖ All jobs are not of the same important level.
- ❖ The processing time and weight for each job are known.
- ❖ There are only three machines available and they are in tip top condition.
- ❖ Preemption is allowed.

A job with a small processing time and a large weight should be scheduled toward the front, whereas one with a large processing time and a small weight should be scheduled to the back. One way to do this is to look at the ratio of processing time to weight and order the jobs in increasing ratio.

**Table 4.14:** Sorted Jobs for Jobs with Priorities by Weights

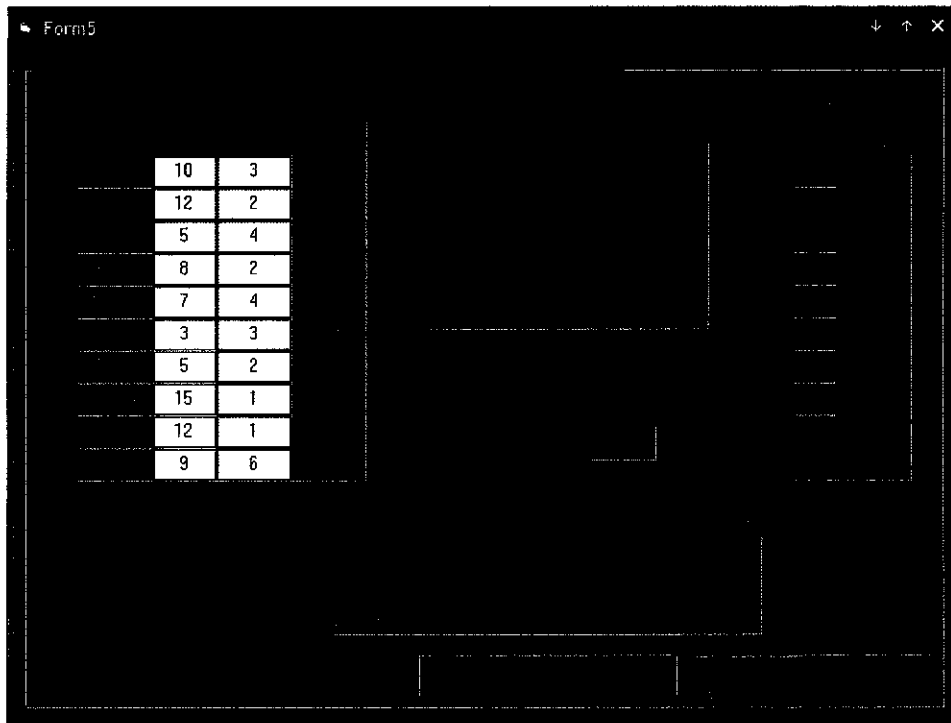
Job	6	3	10	5	7	1	4	2	9	8
Processing Time	3	5	9	7	5	10	8	12	12	15
Weight	3	4	6	4	2	3	2	2	1	1
Ratio	1.00	1.25	1.50	1.75	2.50	3.34	4.00	6.00	12.00	15.00

By using the WSPT method the sequence of the jobs are: **6-3-10-5-7-1-4-2-9-8**. After manually assign each job to the three available machines, the outcome is:

**Table 4.15: Assigned Jobs for Jobs with Priorities by Weights to Machines**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12
6	1.00	15.12	2	5.22	10.9	9	-1.1	15.02
3	1.25	13.87	9	12.00	-1.1	8	15.00	0.02
10	1.50	12.37						
5	1.75	10.62						
7	2.50	8.12						
1	3.34	4.78						
4	4.00	0.78						
2	6.00	-5.22						

**Knowledge System Output**



**Figure 4.6: Result for Weighted Shortest Processing Time**

Since the knowledge system produced the same sequence as the manual calculation, this component is verified.

### 4.5.3 Jobs with Due Dates

#### Manual Calculation Output (Earlier Due Dates (EDD) Method)

*Assumption:*

- ❖ Release time are all zero.
- ❖ All jobs are not of the same important level.
- ❖ The processing time and due date for each job are known.
- ❖ There are only three machines available and they are in tip top condition.
- ❖ Preemption is allowed.

**Table 4.16: Data for Jobs with Due Dates**

<b>Job</b>	1	2	3	4	5	6	7	8	9	10
<b>Processing Time</b>	10	12	5	8	7	3	5	15	12	9
<b>Weight</b>	1	1	1	2	3	2	4	6	4	3
<b>Due Date</b>	26	32	38	48	51	64	53	50	35	28

Here the objective is to have as little tardiness/lateness as possible. Due to that, the date oriented measures is taken in order to ensure the jobs is not tardy or late.

Applying the EDD method, the sequence of the jobs is: **1-10-2-9-3-4-8-5-7-6**

**Table 4.17: Sorted Jobs for Jobs with Due Dates**

<b>Job</b>	1	10	2	9	3	4	8	5	7	6
<b>Processing Time</b>	10	9	12	12	5	8	15	7	5	3
<b>Weight</b>	1	3	1	4	1	2	6	3	4	2
<b>Due Date</b>	26	28	32	35	38	48	50	51	53	64
<b>Completion Time</b>	10	19	31	43	48	56	71	78	83	86
<b>Tardiness</b>	0	0	0	8	10	8	21	27	30	22

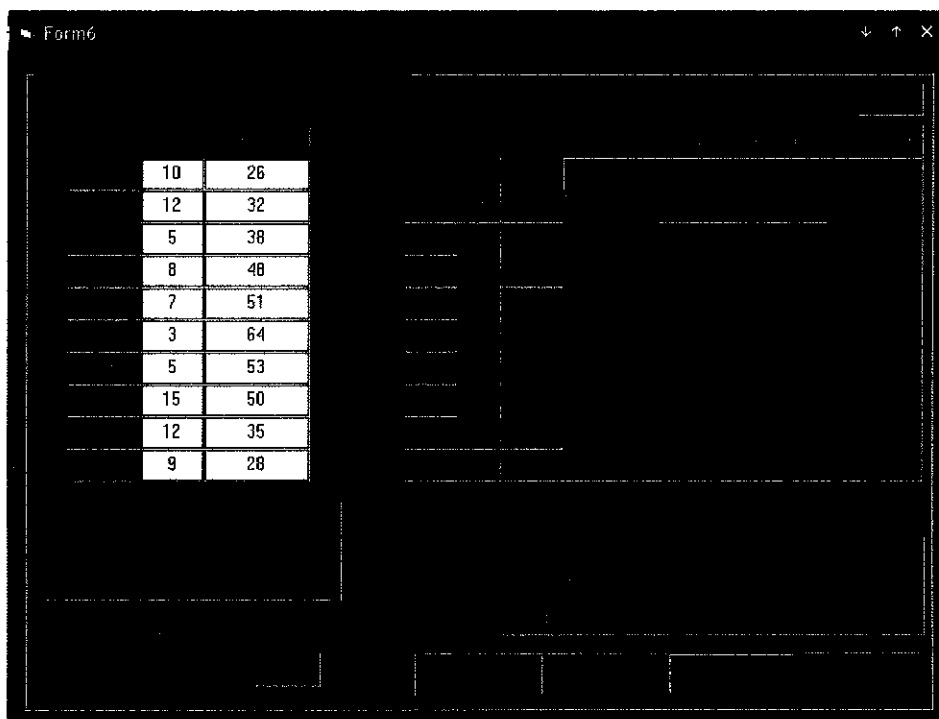
From manual calculation, we knew that there were seven jobs out of ten were tardy.

After manually assign each job to the three available machines, the outcome is:

**Table 4.18: Assigned Jobs for Jobs with Due Dates to Machines**

MACHINE 1			MACHINE 2			MACHINE 3		
Job	Due Dates	RCD=142	Job	Due Dates	RCD=142	Job	Due Dates	RCD=142
1	26	116	3	17	125	5	24	118
10	28	88	4	48	77	7	53	65
2	32	56	8	50	27	6	64	1
9	35	21	5	51	-24			
3	38	-17						

**Knowledge System Output**



**Figure 4.7: Result for Due Dated Jobs**

Since the knowledge system produced the same sequence as the manual calculation, this component is verified.

## **CHAPTER 5**

### **CONCLUSION/RECOMMENDATION**

The purpose of this chapter is to conclude the findings and discussion of the entire project and to recommend the next possible expansions regarding to the enhancement of the project content in the future.

Every project has constraints and limit that should be adhered. Simultaneously, there is minimum requirement that should be satisfied to ensure the project is well-going. Rarely, a limit or constraint could be the motivating factor in a project and it is also could be the limit that cannot be disputed.

#### **5.1 CONCLUSION**

As the industrialized world develops, more and more resources are becoming critical. Machines manpower and facilities are now commonly thought of as critical resources in production and service activities. Scheduling these resources leads to increased efficiency, utilization, and ultimately, profitability. Loosely defined, scheduling is an act of defining priorities or arranging activities to meet certain requirements, constraints, or objectives. Current operation scheduling methods require complex mathematically modeling techniques that demand the substantial and extensive knowledge. Otherwise, the simple methods may not provide good results.

The computer application in manufacturing industry has embarked upon the new dimension of operation scheduling. Thus, the integration between operating scheduling and intelligent computer system is the most welcome feasibility features that absolutely can bring the remarkable achievement for industrialists. Hence, there will be reduction in time taken (compared to manual calculation), ability to be used in “What if” situation and can avoid the repetition of same procedure in the same problem.

The integration of these two fields (IT and manufacturing), may resulting the faster rate of makespan and more reliable production. The top three benefits of knowledge system are (Schreiber et al, 2000);

- i. Faster decision making
- ii. Increased productivity
- iii. Increased quality of decision making

To conclude this project, the development of the knowledge system might be a small achievement in its area of knowledge system. However, this shall be the premature start for the vast improvement and the guide for new developer to attain the bigger size and better performances of knowledge system development.

## **5.2 RECOMMENDATIONS**

### **5.2.1 Current Project**

This knowledge system program has a great potential to be commercialized because it offers the simplicity in making decision towards the parallel operation scheduling problem. The possible customer that targeted for being the end user would be the tailor that has the dilemma in deciding which of the clothes should be sews first. Normally, the tailor has a lot of customers with different type and style of orders as well as the urgency to collect the finished orders. Thus, the decision making process become the critical part within the business. But in this case, the tailor shop should have three workers that having same abilities and skills in performing sewing and the number of clothes to be sewed in one batch is limited to ten. The author believed that after considering this program, the tailor may successful in his business as the decision making become easier and reliable.

### **5.2.2 Further Expansion**

This project is interested in developing knowledge-based system for parallel machine operation scheduling, however due to some hindrance especially in building the knowledge base part, there are still a lot of improvement that should be pursued in the future.

The improvements that are recommended for this project are:

- i. Expanding the knowledge system on the entire issues regarding parallel machines
- ii. Knowledge-Based System for Single and Parallel Machines (combined)

#### **Expanding The Knowledge System on the Entire Issues Regarding Parallel Machines**

It is recommend that the content of the project should be extended entirely covered the issues of the parallel machines. There are several issues regarding parallel machine operation scheduling that have not being tackled yet. This is due to hindrance while developing knowledge system for other parts that requires much attention. Furthermore, as the limited availability of the resources, some of the issues are unintended to be left out at present project.

Among the issues that could be developed in the future are:

- ❖ Mean Time Before Failure
- ❖ Minimize flowtime with no tardy jobs
- ❖ Dynamic scheduling
- ❖ Minimizing Set-Up times
- ❖ Parallel machine Search Methods



### **Knowledge-Based System for Single and Parallel Machines (Combined)**

As far as it is concerned, this project is the continuity from the project entitled “A Knowledge Based System for a Single Machine Operation Scheduling” which is authored by M. Zamri Che Bahar Nordin. It is suggested that both fields of operation scheduling are being combined as to offer a comprehensive valuable package to end user in problem solving.

The development of parallel machine knowledge system and its combination with single machine knowledge system will complete the knowledge system for operation scheduling. The development of knowledge system for operation scheduling is interesting subject to pursue and the product would be able to contribute to the manufacturing world in term of better decision making.

## REFERENCES

- Baker, Kenneth J. (1974), **Introduction to Scheduling and Sequencing**, John Wiley & Sons, USA.
- Wittock, R.J. (1986), **Scheduling Parallel Machines with Setups**, Research report, IBM Thomas J. Watson Research Centre, Yorktown Heights, NY.
- Kedia, S.K. (1970), **A Job Shop Scheduling Problem with Parallel Machines**, Department of Industrial Engineering, University of Michigan, USA
- McNaughton, R. (1959), **Scheduling with Deadlines and Loss Functions**, Management Science, Vol. 6, No. 1
- Sipper and Bulfin, (1998), **Production: Planning, Control and Integration**, McGraw-Hill Book Co, Singapore, International Edition.
- Tang, C.S. (1990), **Scheduling Batches on Parallel Machines with Major and Minor Setups**, European Journal of Operational Research, 46(3) 171-175.
- <http://my.fit.edu/~georgio/research/publications/gca-bb-hms-2000.pdf>
- <http://www.densis.fee.unicamp.br/~smendes/PAPERS/PPC-02.pdf>
- Schreiber, Guus et al, (2000), **Knowledge Engineering and Management**, MIT Press, Massachusetts.
- Che Bahar Nordin, M. Zamri (2002), **To Develop Knowledge Based System for Single Machine Operations Scheduling**, Final Year Project, Universiti Teknologi PETRONAS
- Bronson, Gary J. (1999), **C++ for Engineers and Scientists**, International Thomson Publishing Company, Canada.

Stefik, Mark (1995) **Introduction to Knowledge System**, Morgan Kaufmann Publishers, San Fransisco.

Zak, Diane (2001) **Visual Basic for Applications**, Thomson Learning, Massachusettes.

Zak, Diane (1999) **Programming with Visual Basic**, Thomson Learning, Massachusettes.

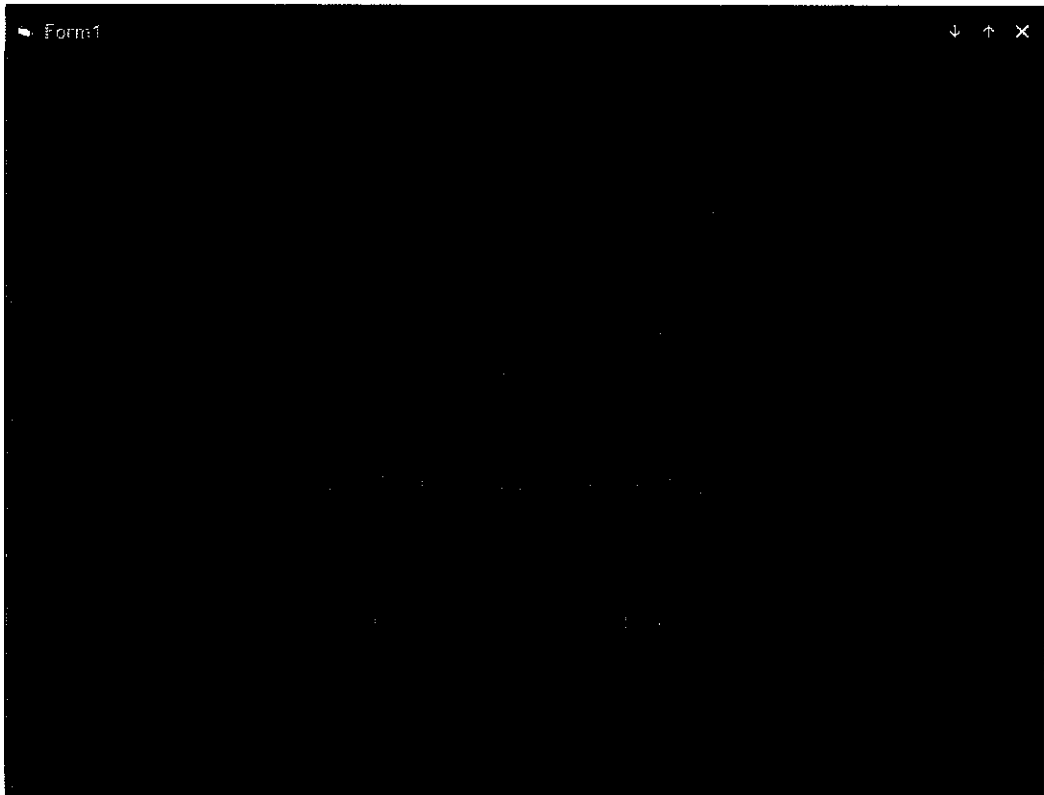
Oxfords University Press (1996) **Oxford's Advanced Learner's Dictionary**, New York

S. F Johnston, J.P Gostelow, WJ. King (2000) **Engineering & Society**, Prentice Hall.

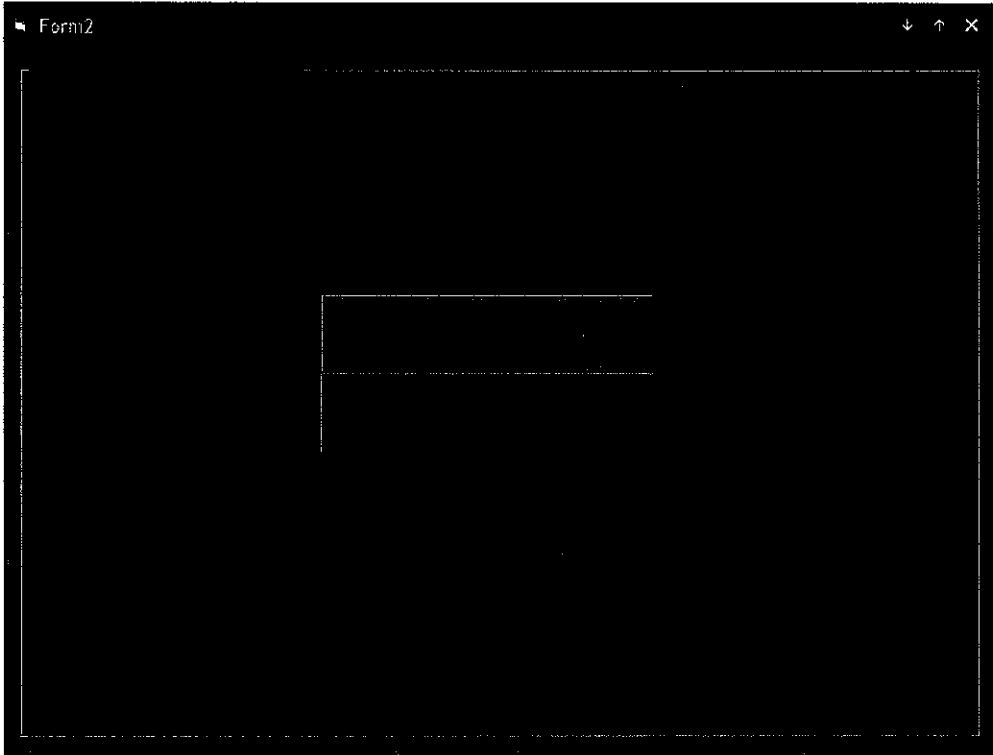
## APPENDIX A

<b>Appendix A.1:</b>	Start Up Form . . . . .	50
<b>Appendix A.2:</b>	Operation Scheduling Form . . . . .	51
<b>Appendix A.3:</b>	Exit Message Box . . . . .	51
<b>Appendix A.4:</b>	Condition Selection Form (First Case) . . . . .	52
<b>Appendix A.5:</b>	Jobs with Equal Weight . . . . .	52
<b>Appendix A.6:</b>	Condition Selection Form (Second Case) . . . . .	53
<b>Appendix A.7:</b>	Jobs with Priorities Ranked By Weights Form . . . . .	53
<b>Appendix A.8:</b>	Condition Selection Form (Third Case) . . . . .	54
<b>Appendix A.9:</b>	Jobs with Due Dates Form . . . . .	54

This section presented the system layouts as known as user interfaces built by the author in developing the knowledge base system using Microsoft Visual Basic 6.0.



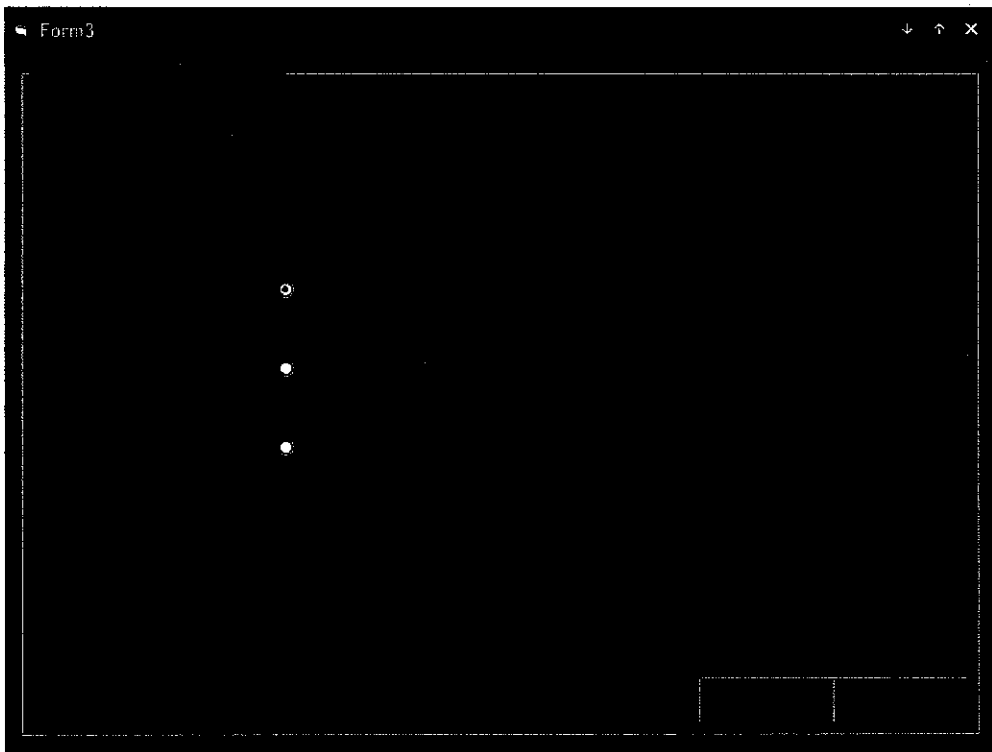
**Appendix A.1: Start Up Form**



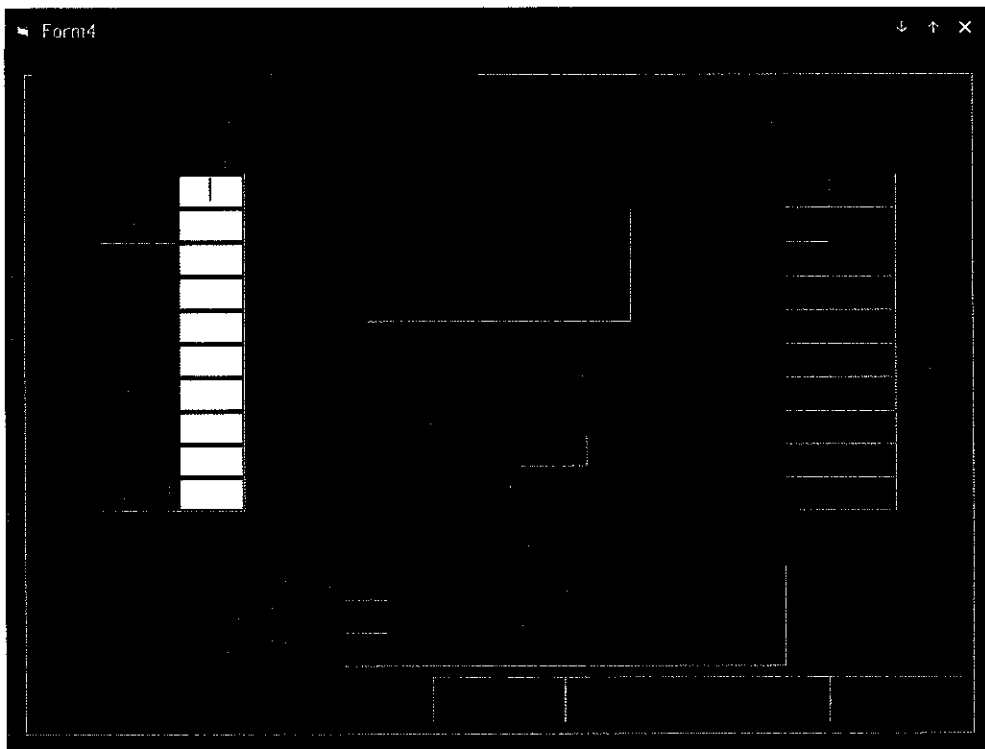
**Appendix A.2: Operation Scheduling Form**



**Appendix A.3: Exit Message Box**



**Appendix A.4: Select Condition Form (First Case)**



**Appendix A.5: Jobs with Equal Weight Form**

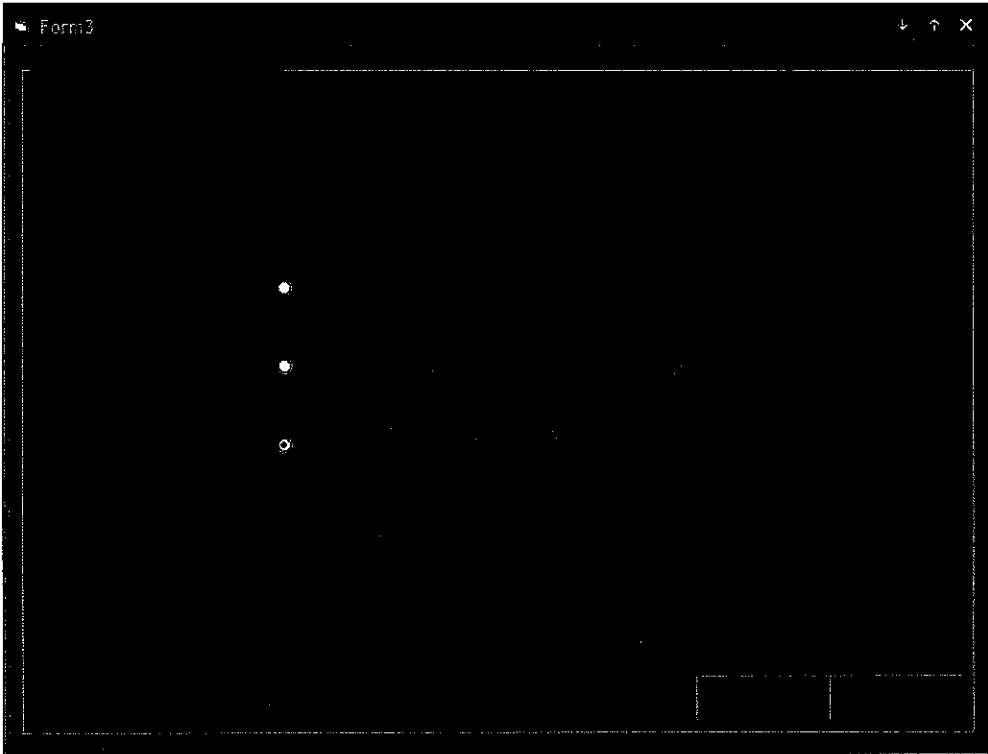
The image shows a software window titled "Form3" with a dark background. On the left side, there are three vertically aligned radio buttons. The top and bottom buttons are filled with white, while the middle one is empty. The rest of the window is mostly dark with some faint grid lines and a small rectangular area at the bottom right.

**Appendix A.6: Condition Selection Form (Second Case)**

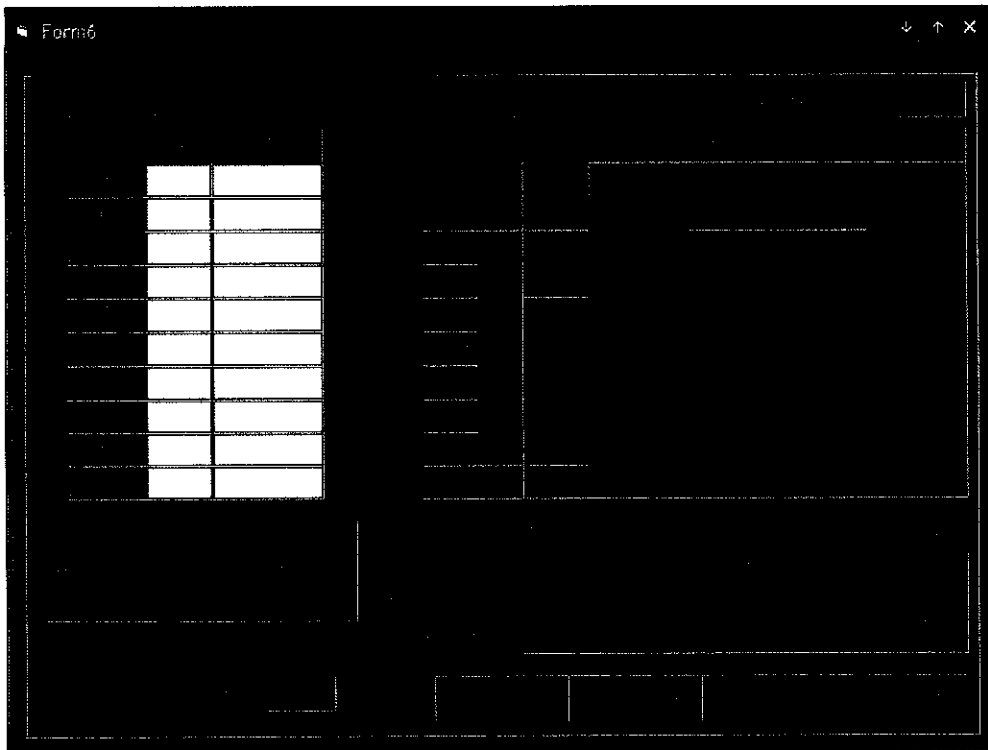
The image shows a software window titled "Form5" with a dark background. On the left side, there is a grid of 10 rows and 3 columns. The grid is white with black borders. To the right of the grid, there are several horizontal lines and a vertical line, suggesting a list or a form structure. The bottom of the window has a few faint rectangular outlines.

**Appendix A.7: Jobs with Priorities Ranked By Weights Form**





**Appendix A.8: Condition Selection Form (Third Case)**



**Appendix A.9: Jobs with Due Dates Form**

## APPENDIX B

<b>Appendix B.1:</b>	Start Up Coding Structure . . . . .	56
<b>Appendix B.2:</b>	Operation Scheduling Coding Structure . . . . .	56
<b>Appendix B.3:</b>	Condition Selection Form Coding Structure . . . . .	56
<b>Appendix B.4:</b>	Jobs with Equal Weight Coding Structure . . . . .	57
<b>Appendix B.5:</b>	Jobs with Priorities Ranked By Weights Coding Structure	60
<b>Appendix B.6:</b>	Jobs with Due Dates Coding Structure . . . . .	63

This section presented the coding structures of the program developed. The sequence of the coding is sorted accordingly to the system layout of the system.

### ***Appendix B.1: Start-Up Coding Structure***

```
Private Sub tmrStartUp_Timer()  
frmOpersched.Show  
Unload frmStartUp  
End Sub
```

### ***Appendix B.2: Operation Scheduling Coding Structure***

```
Private Sub cmdExit_Click()  
  
    Const conBTNs As Integer = vbYesNo + vbExclamation + vbDefaultButton1 +  
    vbApplicationModal  
    Dim intuserresponse As Integer  
    intuserresponse = MsgBox("Do you want to exit?", conBTNs, "Scheduling System")  
    If intuserresponse = vbYes Then  
        End  
    End If  
End Sub  
  
Private Sub cmdParallelMC_Click()  
frmSelectcond.Show  
Unload frmOpersched  
  
End Sub
```

### ***Appendix B.3: Condition Selection Form Coding***

```
Option Explicit  
  
Private Sub Back_Click()  
    Unload frmSelectcond  
    frmOpersched.Show  
End Sub  
  
Private Sub Next_Click()  
  
    If optCondition1 = True Then  
        frmCondition1a.Show  
        Unload frmSelectcond  
    End If  
  
    If optCondition2 = True Then  
        frmCondition2a.Show  
        Unload frmSelectcond  
    End If  
  
    If optCondition3 = True Then  
        frmCondition3a.Show  
        Unload frmSelectcond  
    End If  
  
End Sub
```

## *Appendix B.4          Structure Jobs with Equal Weight Coding Structure*

```
Option Explicit

Private Sub frmCondition1_Load()
    txtProsTime(0).SetFocus
End Sub

Private Sub cmdBack_Click()
    frmCondition1.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdClearScreen_Click()
Dim counter6 As Integer

    For counter6 = 0 To 9 Step 1
        txtProsTime(counter6).Text = ""
        lblJobNum(counter6).Caption = ""
        lblProsTimeAfter(counter6).Caption = ""
    Next counter6
    txtProsTime(0).SetFocus
End Sub

Private Sub cmdMainMenu_Click()
    frmCondition1.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdSORT_Click()
Dim intProsTime(0 To 9) As Integer
Dim intProsTimeAfter(0 To 9) As Integer
Dim intJobNum(0 To 9) As Integer
Dim counter As Integer
Dim counter4 As Integer
Dim counter2 As Integer
Dim counter3 As Integer
Dim X As Integer
Dim TempVar As Integer
Dim counter7 As Integer
Dim counter8 As Integer
Dim sngRCT As Single
Dim counter9 As Integer

    For counter = 0 To 9 Step 1
        intProsTimeAfter(counter) = Val(txtProsTime(counter).Text)
    Next counter

    Dim intJobSort1 As Integer, intJobSort2 As Integer
    Dim intTemp As Integer
    For intJobSort1 = 0 To 9
        For intJobSort2 = 0 To 9
            If intProsTimeAfter(intJobSort2) > intTemp Then
                End If
            End If
        Next intJobSort2
    Next intJobSort1

    For counter4 = 0 To 9 Step 1
        intJobNum(counter4) = counter4 + 1
    Next counter4

    For counter2 = 0 To 8 Step 1
        X = 8 - counter2
        For counter3 = 0 To X Step 1
            If intProsTimeAfter(counter3) > intProsTimeAfter(counter3 + 1) Then
                TempVar = intProsTimeAfter(counter3)
                intProsTimeAfter(counter3) = intProsTimeAfter(counter3 + 1)
                intProsTimeAfter(counter3 + 1) = TempVar

                TempVar = intJobNum(counter3)
                intJobNum(counter3) = intJobNum(counter3 + 1)
                intJobNum(counter3 + 1) = TempVar
            End If
        Next counter3
```

```

Next counter2
For counter7 = 0 To 9 Step 1
    lblJobNum(counter7).Caption = intJobNum(counter7)
Next counter7

For counter8 = 0 To 9 Step 1
    lblProsTimeAfter(counter8).Caption = intProsTimeAfter(counter8)
Next counter8

For counter9 = 0 To 9 Step 1
    sngRCT = sngRCT + Val(intProsTimeAfter(counter9))
    lblRCT.Caption = Round(sngRCT / 3, 0)
Next counter9

Dim intCounter As Integer, intJobVal(0 To 9) As Integer
For intCounter = 0 To 9
    intJobVal(intCounter) = lblProsTimeAfter(intCounter).Caption
Next intCounter

For intCounter = 0 To 9
    intJobNum(intCounter) = intCounter + 1
Next intCounter

Dim intTRCT As Integer, intTempRCT As Integer, intRCT As Integer
Dim strTRCT As String, dblRCT As Double
intTRCT = 0
For intCounter = 0 To 9
    intTRCT = intTRCT + intJobVal(intCounter)
Next intCounter
dblRCT = Round(intTRCT / 3, 2) 'this is RCT value
strTRCT = Str(dblRCT)
If Cint(Right(strTRCT, 1)) < 5 And Cint(Right(strTRCT, 1)) > 0 Then
    intRCT = Round(dblRCT, 0) + 1
Else
    intRCT = Round(dblRCT, 0)
End If
lblRCT.Caption = intRCT

'job sorting (value for each machine must not exceed RCT)
Dim intCRCT1 As Integer, intCRCT2 As Integer, intJob As Integer
Dim blnFirst2 As Boolean, intJobValTemp As Integer, intJobCF As Integer
Dim intJobC As Integer, blnFirst3 As Boolean, intCRCTT As Integer
Dim intJobLast As Integer, intSpaceLeft As Integer
blnFirst2 = False
intCRCTT = 0
For intCRCT1 = 0 To 3 'looping for 3 machines
    intSpaceLeft = intRCT
    For intCRCT2 = 0 To 9 'looping for 10 job slots
        '1st machine sorting
        If intCRCT1 = 0 Then
            'have space, can add
            If intSpaceLeft > 0 Then
                'space is bigger than or equal current job
                If intSpaceLeft >= intJobVal(intCRCT2) Then
                    intJob = intJob + intJobVal(intCRCT2)
                    intJobValTemp = intJobVal(intCRCT2)
                    'intJobValTemp = intJobNum(intCRCT2)
                Else
                    'space is available, but not enough
                    'carried over job value
                    intJobCF = intJobVal(intCRCT2) - intSpaceLeft
                    intJobValTemp = intSpaceLeft
                    'intJobValTemp = intJobNum(intCRCT2)
                End If
                intSpaceLeft = intSpaceLeft - intJobValTemp
                lblMachJob1(intCRCT2).Caption = intJobValTemp
                intCRCTT = intCRCT2
                blnFirst2 = True
            'space is less than current job
            Else
                intCRCT2 = intCRCT2 + 1
            End If

            '2nd machine sorting
        ElseIf intCRCT1 = 1 Then
            'have space, can add

```

```

If intSpaceLeft > 0 Then
  If blnFirst2 Then
    intJobC = intCRCTT + 1
    blnFirst2 = False
  End If
  'there is left over from machine 1
  If intJobCF > 0 Then
    intJobValTemp = intJobCF
    intJobCF = 0
    intJobC = intCRCTT - 1
    'space is bigger than or equal current job
  ElseIf intSpaceLeft >= intJobVal(intJobC) Then
    intJob = intJob + intJobVal(intJobC)
    intJobValTemp = intJobVal(intJobC)
  Else
    'space is available, but not enough
    'carried over job value
    intJobCF = intJobVal(intJobC) - intSpaceLeft
    intJobValTemp = intSpaceLeft
    blnFirst3 = True
  End If
  intSpaceLeft = intSpaceLeft - intJobValTemp
  lblMachJob2(intCRCT2).Caption = intJobValTemp
  intJobC = intJobC + 1
  intCRCTT = intJobC
  'space is less than current job
Else
  intCRCT2 = intCRCT2 + 1
End If

'3rd machine sorting
ElseIf intCRCT1 = 2 Then
  'have space, can add
  If intSpaceLeft > 0 And intJobC <= 9 Then
    If blnFirst3 Then
      intJobC = intCRCTT + 1
      blnFirst3 = False
    End If
    'there is left over from machine 1
    If intJobCF > 0 Then
      intJobValTemp = intJobCF
      intJobCF = 0
      intJobC = intCRCTT - 1
    'space is bigger than or equal current job
    ElseIf intSpaceLeft >= intJobVal(intJobC) Then
      intJob = intJob + intJobVal(intJobC)
      intJobValTemp = intJobVal(intJobC)
    Else
      'space is available, but not enough
      'carried over job value
      intJobCF = intJobVal(intJobC) - intSpaceLeft
      intJobValTemp = intSpaceLeft
    End If
    intSpaceLeft = intSpaceLeft - intJobValTemp
    lblMachJob3(intCRCT2).Caption = intJobValTemp
    intJobC = intJobC + 1
    'space is less than current job
  Else
    intCRCT2 = intCRCT2 + 1
  End If

End If
Next intCRCT2
Next intCRCT1

End Sub

```

## Appendix B.5

## Jobs with Priorities Ranked By Weights Coding Structure

```
Option Explicit

Private Sub frmCondition2_Load()
    txtProsTime(0).SetFocus
End Sub

Private Sub cmdMainMenu_Click()
    frmCondition2.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdBack_Click()
    frmCondition2.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdClearScreen_Click()
    Dim counter As Integer

    For counter = 0 To 9 Step 1
        txtProsTime(counter).Text = ""
        txtWeight(counter).Text = ""
        lblRatio(counter).Caption = ""
        lblJobNum(counter).Caption = ""
    Next counter
    txtProsTime(0).SetFocus
End Sub

Private Sub cmdSORT_Click()

    Dim counter2 As Integer
    Dim intProsTime(0 To 9) As Integer
    Dim intProsTimeAfter(0 To 9) As Integer
    Dim intWeight(0 To 9) As Integer
    Dim intJobNum(0 To 9) As Integer

    For counter2 = 0 To 9 Step 1
        intProsTimeAfter(counter2) = Val(txtProsTime(counter2).Text)
        intProsTime(counter2) = Val(txtProsTime(counter2).Text)
        intWeight(counter2) = Val(txtWeight(counter2).Text)
    Next counter2

    Dim counter3 As Integer
    Dim sngRatio(0 To 9) As Single

    For counter3 = 0 To 9 Step 1
        If FormatNumber(intProsTime(counter3)) <> 0 And FormatNumber(intWeight(counter3)) <> 0
            Then
                sngRatio(counter3) = FormatNumber(intProsTime(counter3), 2) /
                    FormatNumber(intWeight(counter3), 2)
                lblRatio(counter3).Caption = FormatNumber(sngRatio(counter3), 2)
            End If
        Next counter3

    Dim counter4 As Integer

    For counter4 = 0 To 9 Step 1
        intJobNum(counter4) = counter4 + 1
    Next counter4

    Dim Counter5 As Single
    Dim counter6 As Single
    Dim X As Integer
    Dim TempVar As Integer

    For Counter5 = 0 To 8 Step 1
        X = 8 - Counter5
        For counter6 = 0 To X Step 1
```

```

    If sngRatio(counter6) > sngRatio(counter6 + 1) Then
        TempVar = sngRatio(counter6)
        sngRatio(counter6) = sngRatio(counter6 + 1)
        sngRatio(counter6 + 1) = TempVar

        TempVar = intJobNum(counter6)
        intJobNum(counter6) = intJobNum(counter6 + 1)
        intJobNum(counter6 + 1) = TempVar
    End If
Next counter6
Next Counter5

Dim counter7 As Integer
For counter7 = 0 To 9 Step 1
    lblJobNum(counter7).Caption = intJobNum(counter7)
    lblRatioSort(counter7).Caption = FormatNumber(sngRatio(counter7), 2)
Next counter7

Dim sngRCR As Single
Dim counter8 As Integer

For counter8 = 0 To 9 Step 1
    sngRCR = sngRCR + Val(lblRatio(counter8))
    lblRCR.Caption = Round(sngRCR / 3, 2)
Next counter8

'get the value in sorted job label and put into variable
Dim intCounter As Integer, intJobVal(0 To 9) As Integer
For intCounter = 0 To 9
    'intJobVal(intCounter) = lblProsTimeAfter(intCounter).Caption
Next intCounter

Dim intTRCR As Integer, intTempRCR As Integer, intRCR As Integer
Dim strTRCR As String, dblRCR As Double
intTRCR = 0
For intCounter = 0 To 9
    intTRCR = intTRCR + intJobVal(intCounter)
Next intCounter
dblRCR = Round(intTRCR / 3, 2) 'this is RCR value
strTRCR = Str(dblRCR)
If CInt(Right(strTRCR, 1)) < 5 And CInt(Right(strTRCR, 1)) > 0 Then
    intRCR = Round(dblRCR, 0) + 1
Else
    intRCR = Round(dblRCR, 0)
End If
'lblRCR.Caption = intRCR

'job sorting (value for each machine must not exceed RCR)
Dim intCRCR1 As Integer, intCRCR2 As Integer, intJob As Integer
Dim blnFirst2 As Boolean, intJobValTemp As Integer, intJobCF As Integer
Dim intJobC As Integer, blnFirst3 As Boolean, intCRCRR As Integer
Dim intJobLast As Integer, intSpaceLeft As Integer
blnFirst2 = False
intCRCRR = 0
For intCRCR1 = 0 To 3 'looping for 3 machines
    intSpaceLeft = intRCR
    For intCRCR2 = 0 To 9 'looping for 10 job slots
        '1st machine sorting
        If intCRCR1 = 0 Then
            'have space, can add
            If intSpaceLeft > 0 Then
                'space is bigger than or equal current job
                If intSpaceLeft >= intJobVal(intCRCR2) Then
                    intJob = intJob + intJobVal(intCRCR2)
                    intJobValTemp = intJobVal(intCRCR2)
                    'intJobValTemp = intJobNum(intCRCR2)
                Else
                    'space is available, but not enough
                    'carried over job value
                    intJobCF = intJobVal(intCRCR2) - intSpaceLeft
                    intJobValTemp = intSpaceLeft
                    'intJobValTemp = intJobNum(intCRCR2)
                End If
            End If
        End If
    Next intCRCR2
Next intCRCR1

```



```

        intSpaceLeft = intSpaceLeft - intJobValTemp
        lblMachJob1(intCRCR2).Caption = intJobValTemp
        intCRCR2 = intCRCR2
        blnFirst2 = True
    'space is less than current job
Else
    intCRCR2 = intCRCR2 + 1
End If

'2nd machine sorting
ElseIf intCRCR1 = 1 Then
    'have space, can add
    If intSpaceLeft > 0 Then
        If blnFirst2 Then
            intJobC = intCRCR2 + 1
            blnFirst2 = False
        End If
        'there is left over from machine 1
        If intJobCF > 0 Then
            intJobValTemp = intJobCF
            intJobCF = 0
            intJobC = intCRCR2 - 1
        'space is bigger than or equal current job
        ElseIf intSpaceLeft >= intJobVal(intJobC) Then
            intJob = intJob + intJobVal(intJobC)
            intJobValTemp = intJobVal(intJobC)
        Else
            'space is available, but not enough
            'carried over job value
            intJobCF = intJobVal(intJobC) - intSpaceLeft
            intJobValTemp = intSpaceLeft
            blnFirst3 = True
        End If
        intSpaceLeft = intSpaceLeft - intJobValTemp
        lblMachJob2(intCRCR2).Caption = intJobValTemp
        intJobC = intJobC + 1
        intCRCR2 = intJobC
        'space is less than current job
    Else
        intCRCR2 = intCRCR2 + 1
    End If

'3rd machine sorting
ElseIf intCRCR1 = 2 Then
    'have space, can add
    If intSpaceLeft > 0 And intJobC <= 9 Then
        If blnFirst3 Then
            intJobC = intCRCR2 + 1
            blnFirst3 = False
        End If
        'there is left over from machine 1
        If intJobCF > 0 Then
            intJobValTemp = intJobCF
            intJobCF = 0
            intJobC = intCRCR2 - 1
        'space is bigger than or equal current job
        ElseIf intSpaceLeft >= intJobVal(intJobC) Then
            intJob = intJob + intJobVal(intJobC)
            intJobValTemp = intJobVal(intJobC)
        Else
            'space is available, but not enough
            'carried over job value
            intJobCF = intJobVal(intJobC) - intSpaceLeft
            intJobValTemp = intSpaceLeft
        End If
        intSpaceLeft = intSpaceLeft - intJobValTemp
        lblMachJob3(intCRCR2).Caption = intJobValTemp
        intJobC = intJobC + 1
        'space is less than current job
    Else
        intCRCR2 = intCRCR2 + 1
    End If

End If
Next intCRCR2
Next intCRCR1
End Sub

```

## **Appendix B.6      *Jobs with Due Dates Coding Structure***

```
Option Explicit

Private Sub frmCondition3_Load()
    txtProsTime(0).SetFocus
End Sub
Private Sub cmdBack_Click()
    frmCondition3.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdClearScreen_Click()
Dim counter As Integer

For counter = 0 To 9 Step 1
    txtProsTime(counter).Text = ""
    txtDueDate(counter).Text = ""
    lblJobNum(counter).Caption = ""
    lblProsTimeAfter(counter).Caption = ""
    lblDueDate(counter).Caption = ""
    lblCompTime(counter).Caption = ""
    lblTardiness(counter).Caption = ""
Next counter
txtProsTime(0).SetFocus

End Sub

Private Sub cmdMainMenu_Click()
    frmCondition3.Hide
    frmSelectcond.Show
End Sub

Private Sub cmdSORT_Click()

Dim counter2 As Integer
Dim counter3 As Integer
Dim counter4 As Integer
Dim Counter5 As Integer
Dim counter6 As Integer
Dim X As Integer
Dim TempVar As Integer
Dim intProsTime(0 To 9) As Integer
Dim intDueDate(0 To 9) As Integer
Dim intJobNum(0 To 9) As Integer

For counter2 = 0 To 9 Step 1
    intDueDate(counter2) = Val(txtDueDate(counter2))
    intProsTime(counter2) = Val(txtProsTime(counter2))
Next counter2

For Counter5 = 0 To 9 Step 1
    intJobNum(Counter5) = Counter5 + 1
Next Counter5

For counter3 = 0 To 8 Step 1
    X = 8 - counter3
    For counter4 = 0 To X Step 1
        If intDueDate(counter4) > intDueDate(counter4 + 1) Then
            TempVar = intDueDate(counter4)
            intDueDate(counter4) = intDueDate(counter4 + 1)
            intDueDate(counter4 + 1) = TempVar

            TempVar = intJobNum(counter4)
            intJobNum(counter4) = intJobNum(counter4 + 1)
            intJobNum(counter4 + 1) = TempVar

            TempVar = intProsTime(counter4)
            intProsTime(counter4) = intProsTime(counter4 + 1)
            intProsTime(counter4 + 1) = TempVar
        End If
    Next counter4
Next counter3
```

```

Dim counter7 As Integer
Dim counter8 As Integer
Dim intCompTime(0 To 9) As Integer
Dim lblProsTime(0 To 9) As Integer
Dim counter17 As Integer

For counter7 = 0 To 9 Step 1
    If counter7 > 0 Then
        intCompTime(counter7) = intProsTime(counter7) + intCompTime(counter7 - 1)
    Else
        intCompTime(counter7) = intProsTime(counter7)
    End If
Next counter7

Dim intTardiness(0 To 9) As Integer

For counter17 = 0 To 9 Step 1
    intTardiness(counter17) = intCompTime(counter17) - intDueDate(counter17)
    If intTardiness(counter17) < 0 Then
        intTardiness(counter17) = 0
    End If
Next counter17

Dim counter18 As Integer

For counter18 = 0 To 9 Step 1
    lblTardiness(counter18).Caption = intTardiness(counter18)
Next counter18

Dim counter16 As Integer

For counter16 = 0 To 9 Step 1
    lblCompTime(counter16).Caption = intCompTime(counter16)
Next counter16

Dim counter13 As Integer

For counter13 = 0 To 9 Step 1
    lblJobNum(counter13).Caption = intJobNum(counter13)
Next counter13

Dim counter14 As Integer

For counter14 = 0 To 9 Step 1
    lblProsTimeAfter(counter14).Caption = intProsTime(counter14)
Next counter14

Dim counter15 As Integer

For counter15 = 0 To 9 Step 1
    lblDueDate(counter15).Caption = intDueDate(counter15)
Next counter15

Dim counter11 As Integer
Dim intTardinessMax(0 To 9) As Integer

For counter11 = 0 To 9 Step 1
    If intTardiness(counter11) > 0 Then
        intTardinessMax(counter11) = 1
    Else
        intTardinessMax(counter11) = 0
    End If
Next counter11

Dim counter12 As Integer
Dim intTMax As Integer

For counter12 = 0 To 9 Step 1
    intTMax = intTardinessMax(counter12) + intTMax
Next counter12

Dim strTMax As String

strTMax = FormatNumber(intTMax, 0)

```

```

lblTardinessJob.Caption = strTMax
Dim sngRCD As Single
Dim counter19 As Integer

For counter19 = 0 To 9 Step 1
sngRCD = sngRCD + Val(intDueDate(counter19))
lblRCD.Caption = Round(sngRCD / 3, 0)

Next counter19

End Sub

'get the value in sorted job label and put into variable
Dim intCounter As Integer, intJobVal(0 To 9) As Integer
For intCounter = 0 To 9
'intJobVal(intCounter) = lblProsTimeAfter(intCounter).Caption
Next intCounter

Dim intTRCD As Integer, intTempRCD As Integer, intrCD As Integer
Dim strTRCD As String, dblRCD As Double
intTRCD = 0
For intCounter = 0 To 9
intTRCD = intTRCD + intJobVal(intCounter)
Next intCounter
dblRCD = Round(intTRCD / 3, 2) 'this is RCD value
strTRCD = Str(dblRCD)
If CInt(Right(strTRCD, 1)) < 5 And CInt(Right(strTRCD, 1)) > 0 Then
intrCD = Round(dblRCD, 0) + 1
Else
intrCD = Round(dblRCD, 0)
End If
'lblRCD.Caption = intrCD

'job sorting (value for each machine must not exceed RCD)
Dim intCRCD1 As Integer, intCRCD2 As Integer, intJob As Integer
Dim blnFirst2 As Boolean, intJobValTemp As Integer, intJobCF As Integer
Dim intJobC As Integer, blnFirst3 As Boolean, intCRCD2 As Integer
Dim intJobLast As Integer, intSpaceLeft As Integer
blnFirst2 = False
intCRCD2 = 0
For intCRCD1 = 0 To 3 'looping for 3 machines
intSpaceLeft = intrCD
For intCRCD2 = 0 To 9 'looping for 10 job slots
'1st machine sorting
If intCRCD1 = 0 Then
'have space, can add
If intSpaceLeft > 0 Then
'space is bigger than or equal current job
If intSpaceLeft >= intJobVal(intCRCD2) Then
intJob = intJob + intJobVal(intCRCD2)
intJobValTemp = intJobVal(intCRCD2)
'intJobValTemp = intJobNum(intCRCD2)
Else 'space is available, but not enough
'carried over job value
intJobCF = intJobVal(intCRCD2) - intSpaceLeft
intJobValTemp = intSpaceLeft
'intJobValTemp = intJobNum(intCRCD2)
End If
intSpaceLeft = intSpaceLeft - intJobValTemp
lblMachJob1(intCRCD2).Caption = intJobValTemp
intCRCD2 = intCRCD2
blnFirst2 = True
'space is less than current job
Else
intCRCD2 = intCRCD2 + 1
End If

'2nd machine sorting
ElseIf intCRCD1 = 1 Then
'have space, can add
If intSpaceLeft > 0 Then
If blnFirst2 Then
intJobC = intCRCD2 + 1
blnFirst2 = False
End If

```

```

        'there is left over from machine 1
        If intJobCF > 0 Then
            intJobValTemp = intJobCF
            intJobCF = 0
            intJobC = intCRCD2 - 1
            'space is bigger than or equal current job
        ElseIf intSpaceLeft >= intJobVal(intJobC) Then
            intJob = intJob + intJobVal(intJobC)
            intJobValTemp = intJobVal(intJobC)
        Else
            'space is available, but not enough
            'carried over job value
            intJobCF = intJobVal(intJobC) - intSpaceLeft
            intJobValTemp = intSpaceLeft
            blnFirst3 = True
        End If
        intSpaceLeft = intSpaceLeft - intJobValTemp
        lblMachJob2(intCRCD2).Caption = intJobValTemp
        intJobC = intJobC + 1
        intCRCD2 = intJobC
        'space is less than current job
    Else
        intCRCD2 = intCRCD2 + 1
    End If

    '3rd machine sorting
    ElseIf intCRCD1 = 2 Then
        'have space, can add
        If intSpaceLeft > 0 And intJobC <= 9 Then
            If blnFirst3 Then
                intJobC = intCRCD2 + 1
                blnFirst3 = False
            End If
            'there is left over from machine 1
            If intJobCF > 0 Then
                intJobValTemp = intJobCF
                intJobCF = 0
                intJobC = intCRCD2 - 1
                'space is bigger than or equal current job
            ElseIf intSpaceLeft >= intJobVal(intJobC) Then
                intJob = intJob + intJobVal(intJobC)
                intJobValTemp = intJobVal(intJobC)
            Else
                'space is available, but not enough
                'carried over job value
                intJobCF = intJobVal(intJobC) - intSpaceLeft
                intJobValTemp = intSpaceLeft
            End If
            intSpaceLeft = intSpaceLeft - intJobValTemp
            lblMachJob3(intCRCD2).Caption = intJobValTemp
            intJobC = intJobC + 1
            'space is less than current job
        Else
            intCRCD2 = intCRCD2 + 1
        End If

    End If
Next intCRCD2
Next intCRCD1

End Sub

```

## **APPENDIX C**

<b>Appendix C.1:</b>	Oral Presentation Slides of Final Year Project	68
----------------------	--	----

This section presented the presentation slides that summarize the overall contents of this project. This handout had being presented during Oral Presentation of FYP.

**WELCOME**

{20th May 2004}

**Knowledge Based System  
For Parallel Machine  
Operation Scheduling**

**Mohd Razman Bin Ramli**  
0000 1167

**Mrs Ainul Akmar Bt Mokhtar**

# CONTENTS

## INTRODUCTION

- English Language
- Computer
- High-Speed
- Standard Methods

## METHODOLOGY

- Knowledge-Based System Development
- Tools of Knowledge-Based System

## RESULTS

- 
- 
- 

## RECOMMENDATION





## PROBLEM STATEMENT

❖ To engage in the issue of parallel machine operation scheduling from the knowledge based system perspective.

❖ Current operation scheduling:  
- highly involved the complex mathematical tools  
- high time consumption (↓ productivity)

❖ KBS will provided an integrated system that aided the industrialists to make decision on the parallel operation scheduling issues.

❖ KBS offers the more reliable, easier, simple handling tools and user-friendly interface

## OBJECTIVES

To develop a Knowledge-Based System for parallel machine operation scheduling



*To develop a knowledge base*  
– theory of operation scheduling shall be integrated in one source

*To develop an inference engine*

– inference engine can be illustrated as the hub of the system where the engine will be a key factor of how the KBS will perform.

*To develop a user interface*

– a communication interface to establish the interaction between the user and the knowledge base.



## TERMINOLOGY

*Operation* : an act performed by machines

*Scheduling* : a programme of work to be done or planned events

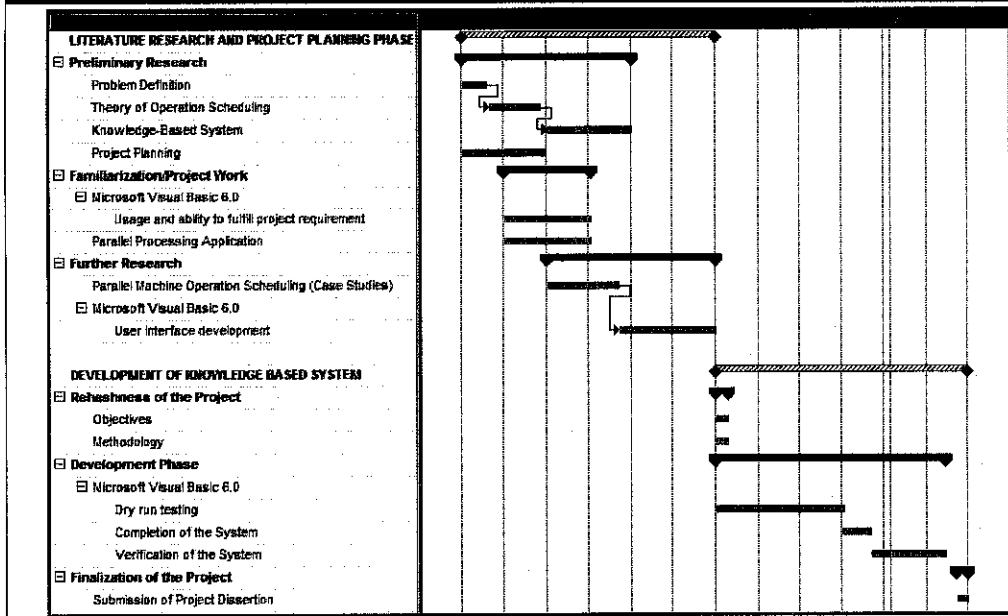
*Parallel Machines* : a number of identical machines are available, and jobs can be processed on any one of them.

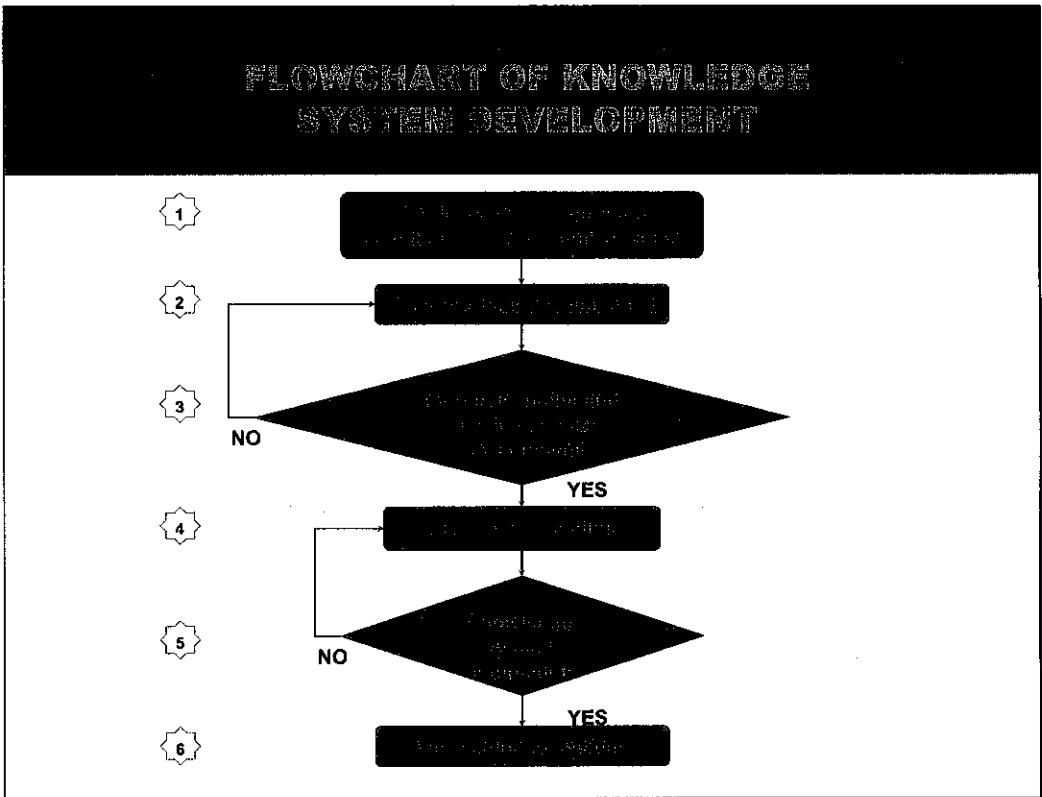
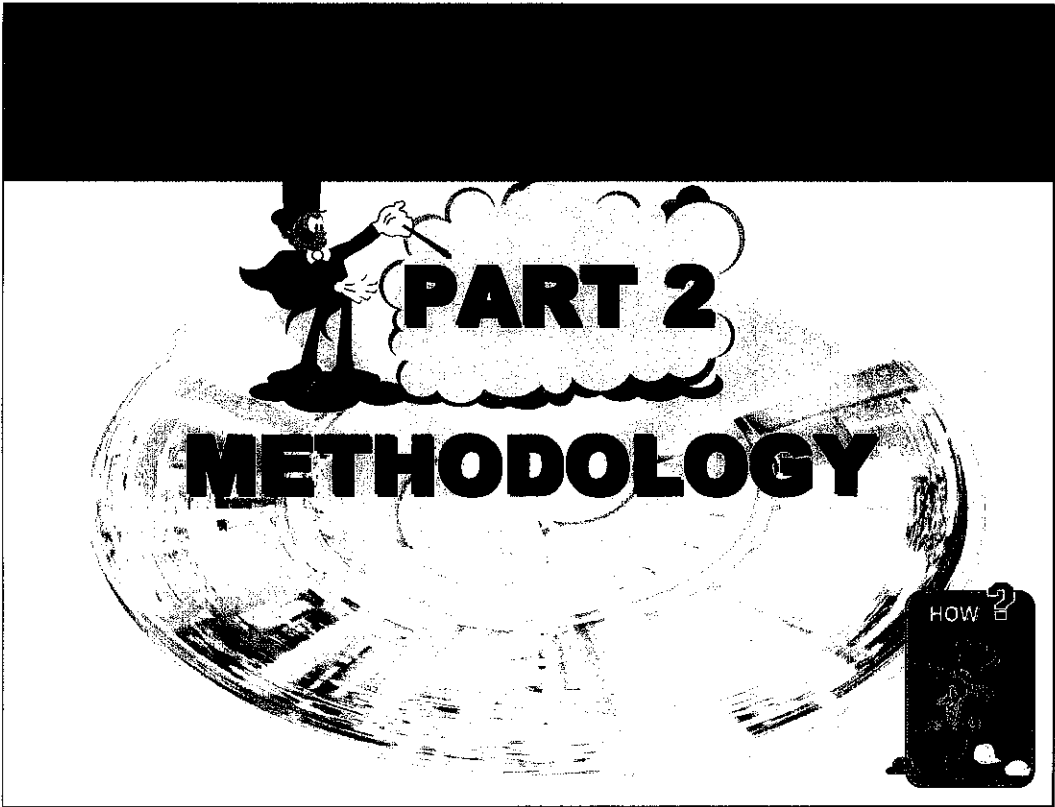
*Sequencing* : ordering of jobs

*Knowledge Based System* : an interface for user to comprehend the knowledge in a system



## PROJECT MILESTONE





TOOL FOR KNOWLEDGE SYSTEM  
DEVELOPMENT

**MICROSOFT  
VISUAL BASIC 6.0**

**Developer interface:**

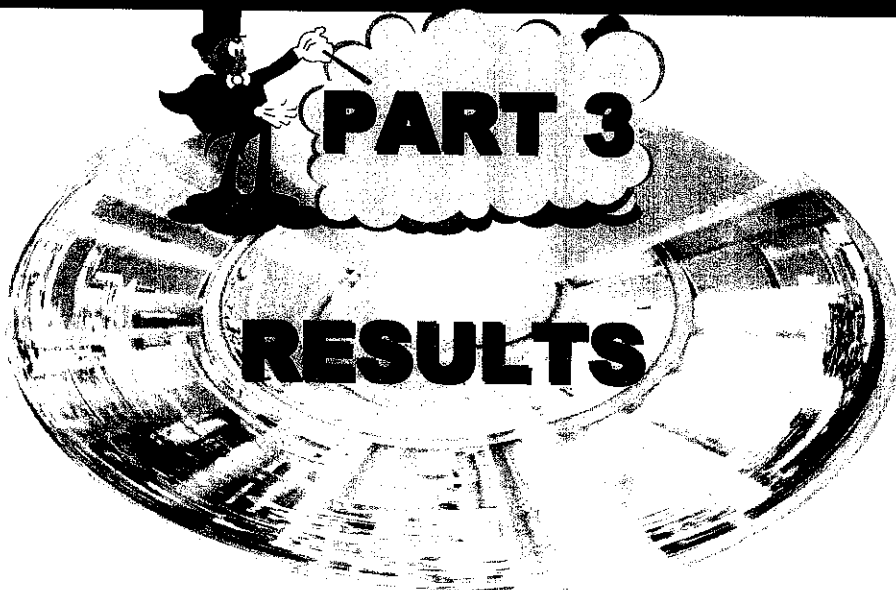
Ability to develop the user interface of the knowledge system and more likely referred as expert interface

**Feasibility to modify**

the program without the need of a programmer. It is a very simple to use and it is very easy to modify.

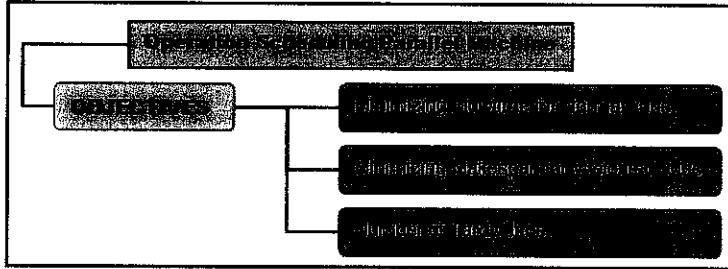
**Code form:**

The code is easy to use as its user-friendly functions and simple.



# CASE STUDIES

## CASE 34 Parallel Processing With Due Dates



**Assumption:** [Redacted]

[Redacted]

[Redacted]

# CASE STUDIES

**Raw Data:**

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9

**Sorted Data:**

Job	6	3	7	5	4	10	1	2	9	8
Processing Time	3	5	5	7	8	9	10	12	12	15

**Remaining Cumulative Time (RCT)**

$$: \frac{\text{Sum of Processing Time}}{\text{No. of Available Machine}} = \frac{86}{3} = 28.6667 \approx 29$$

**Result:**

Job	Processing Time	RCT=29	Job	Processing Time	RCT=29	Job	Processing Time	RCT=29
6	3	26	10	8	21	2	1	28
3	5	21	1	10	11	9	12	16
7	5	16	2	12	-1	8	15	1
5	7	9						
4	8	1						
10	9	-8						

# CASE STUDIES

**Raw Data:**

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9
Weight	3	2	4	2	4	3	2	1	1	6

**Sorted Data:**

Job	6	3	10	5	7	1	4	2	9	8
Processing Time	3	5	9	7	5	10	8	12	12	15
Weight	3	4	6	4	2	3	2	2	1	1
Ratio	1.00	1.25	1.50	1.75	2.50	3.34	4.00	6.00	12.00	15.00

**Remaining Cumulative Ratio (RCR)**

$$\frac{\text{Sum of Processing Ratio}}{\text{No. of Available Machine}} = \frac{48.34}{3} = 16.1133 \approx 16.12$$

**Result:**

Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12	Job	Ratio	RCR=16.12
6	1.00	15.12	2	5.22	10.9	9	-1.1	15.02
3	1.25	13.87	9	12.00	-1.1	8	15.00	0.02
10	1.50	12.37						
5	1.75	10.62						
7	2.50	8.12						
1	3.34	4.78						
4	4.00	0.78						
2	6.00	-5.22						

# CASE STUDIES

## CADP or Priority Processing With Due Dates

**Raw Data:**

Job	1	2	3	4	5	6	7	8	9	10
Processing Time	10	12	5	8	7	3	5	15	12	9
Weight	1	1	1	2	3	2	4	6	4	3
Due Date	26	32	38	48	51	64	53	50	35	28

**Sorted Data:**

Job	1	10	2	9	3	4	8	5	7	6
Processing Time	10	9	12	12	5	8	15	7	5	3
Weight	1	3	1	4	1	2	6	3	4	2
Due Date	26	28	32	35	38	48	50	51	53	64
Cumulative Time	10	19	31	43	48	56	71	78	83	86
Penalties	0	0	0	8	10	8	21	27	30	22

**Remaining Cumulative Due Dates (RCD)**

$$\frac{\text{Sum of Processing Due Dates}}{\text{No. of Available Machine}} = \frac{425}{3} = 141.6667 \approx 142$$

# CASE STUDIES

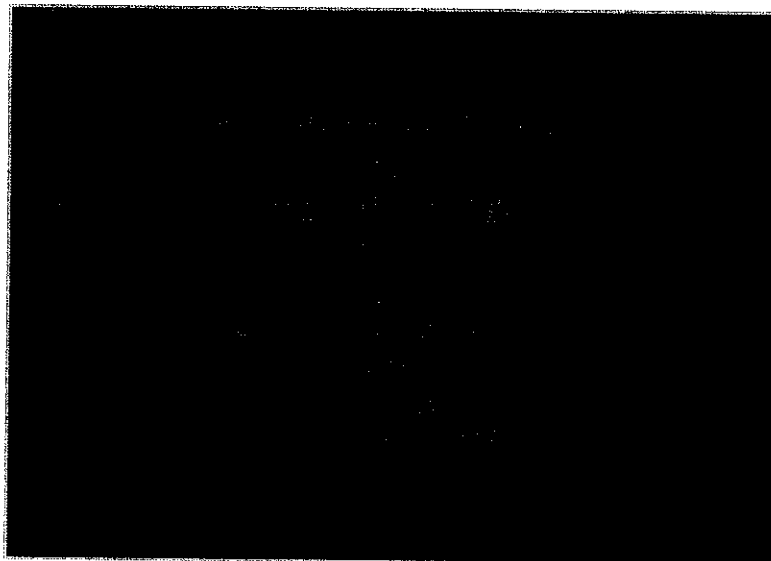
## CASE 3: Parallel Processing With Due Dates (Cont.)

**Result:**

Job	Due Dates	RCD-142	Job	Due Dates	RCD-142	Job	Due Dates	RCD-142
1	26	116	3	17	125	5	24	118
10	28	88	4	48	77	7	53	65
2	32	56	8	50	27	6	64	1
9	35	21	5	51	-24			
3	38	-17						

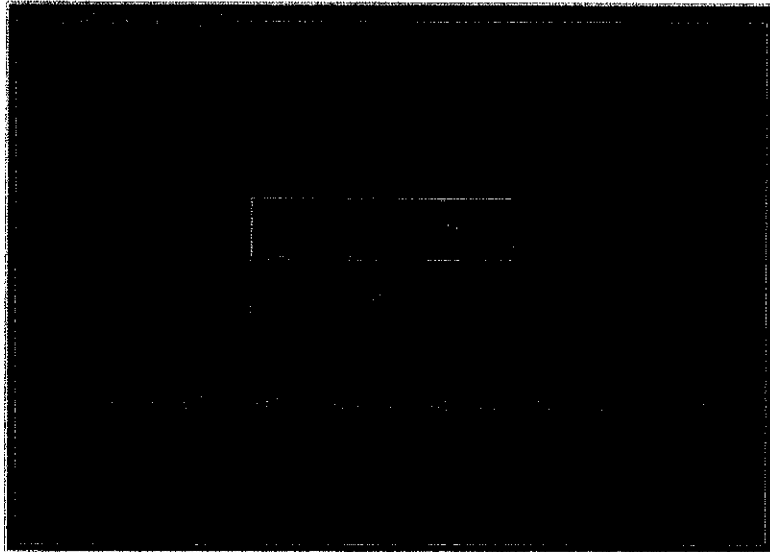
**No. of Tardy Job: 7**

# SYSTEM LAYOUT



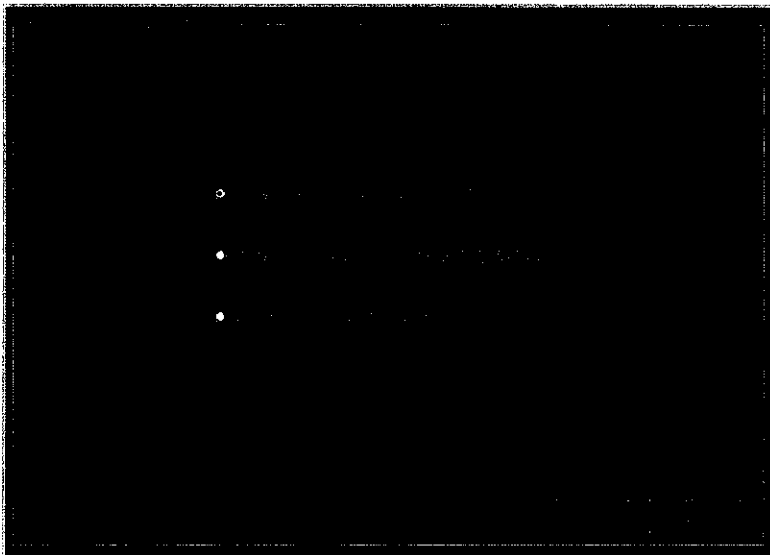
# SYSTEM LAYOUT

## Line and Scheduling Page



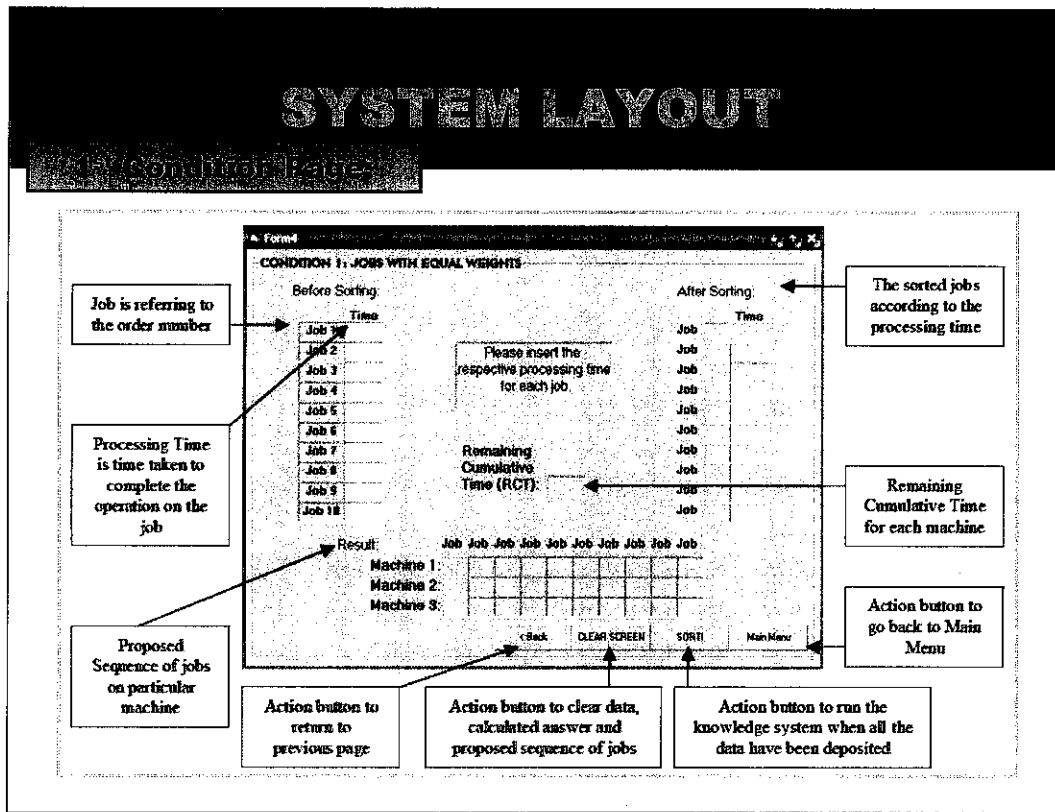
# SYSTEM LAYOUT

## Selection Page

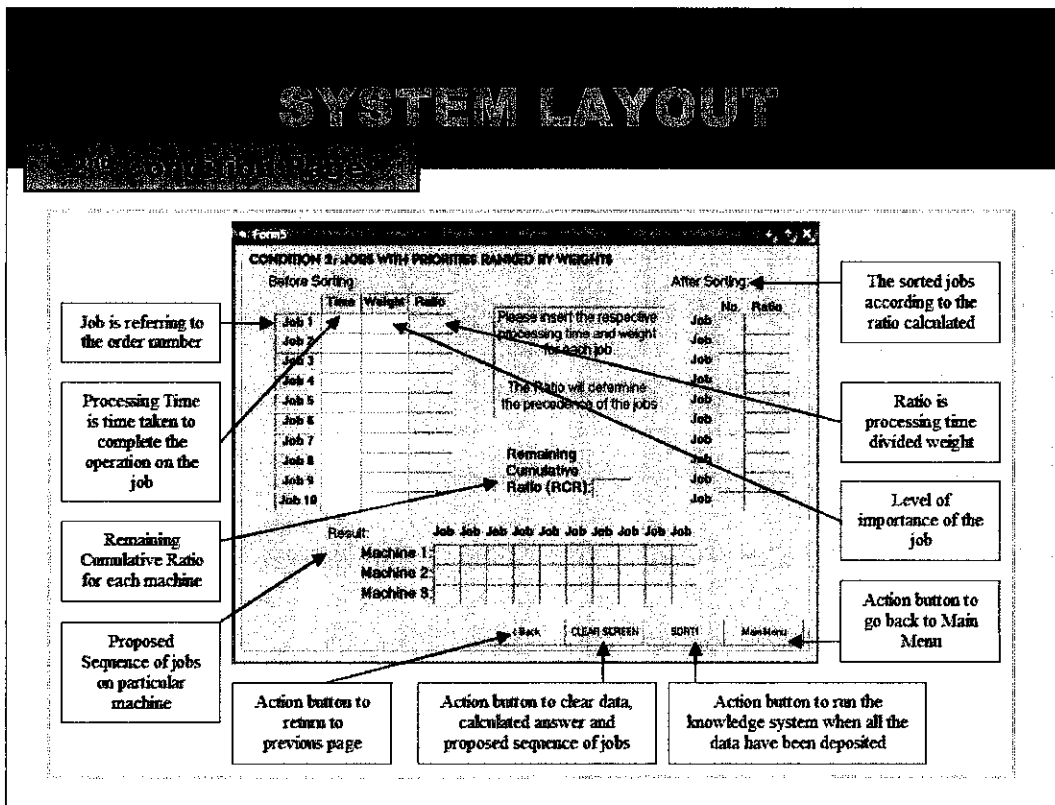




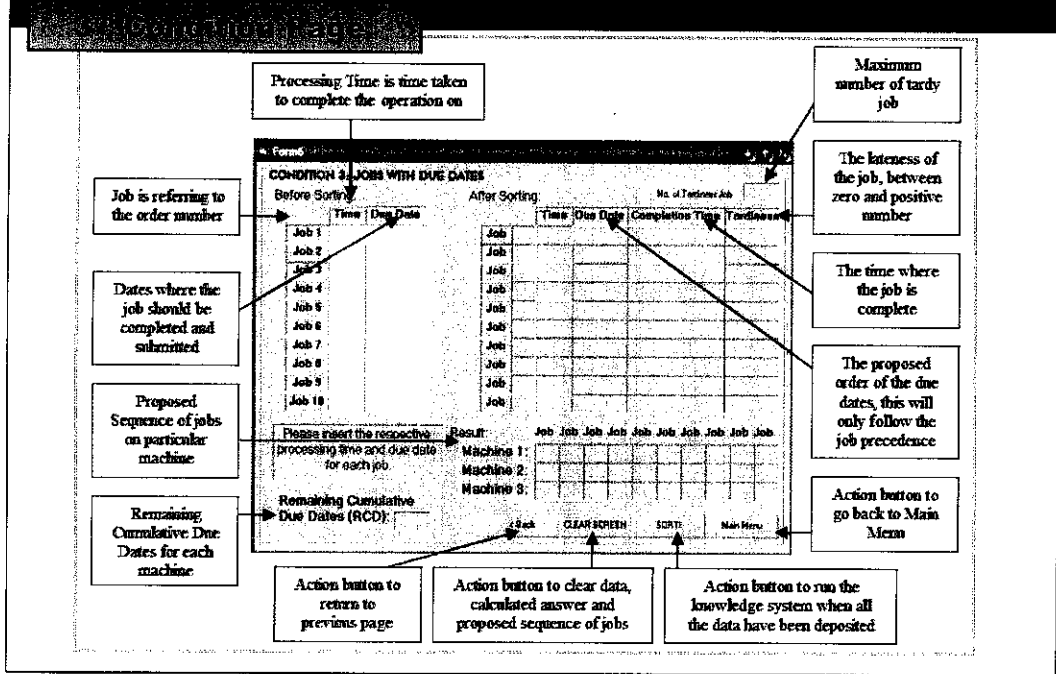
# SYSTEM LAYOUT



# SYSTEM LAYOUT



# SYSTEM LAYOUT



# VERIFICATION OF THE SYSTEM

[Redacted]

**Job Precedence: 6-7-3-5-4-10-1-9-2-8**

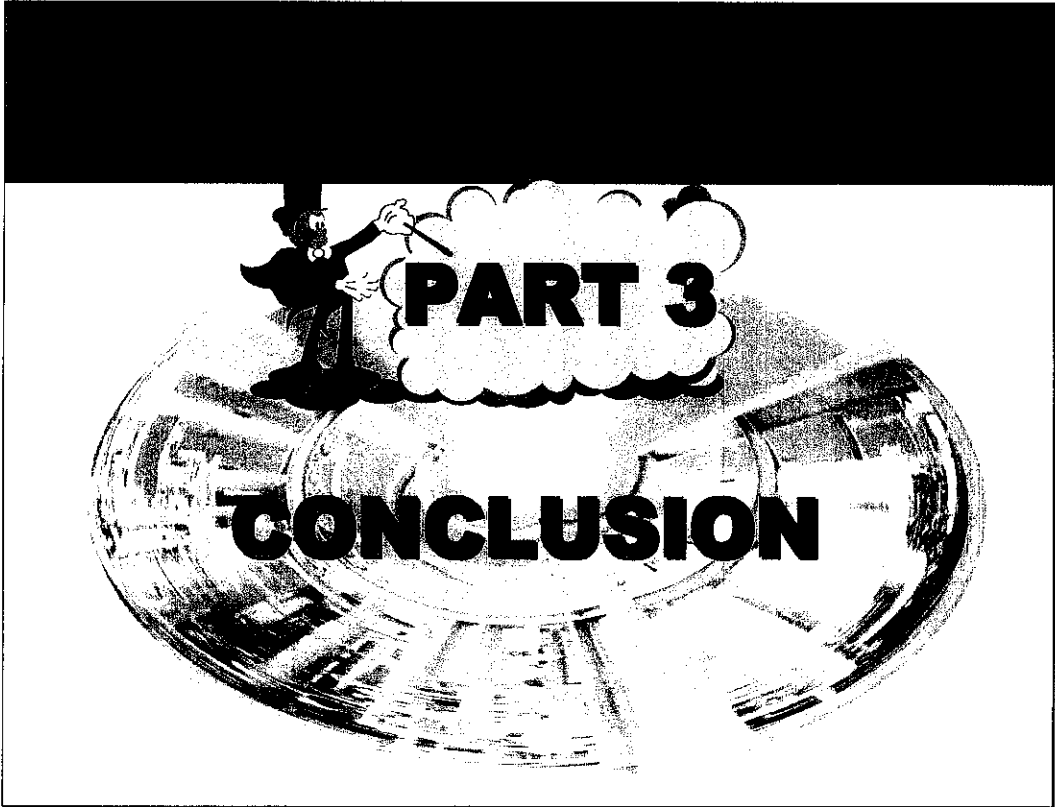
[Redacted]

**Job Precedence: 6-3-10-5-7-1-4-2-9-8**



**CASH in Parallel Processing With Due Dates**

**Job Precedence: 1-10-2-9-3-4-8-5-7-6**



## CONCLUSION

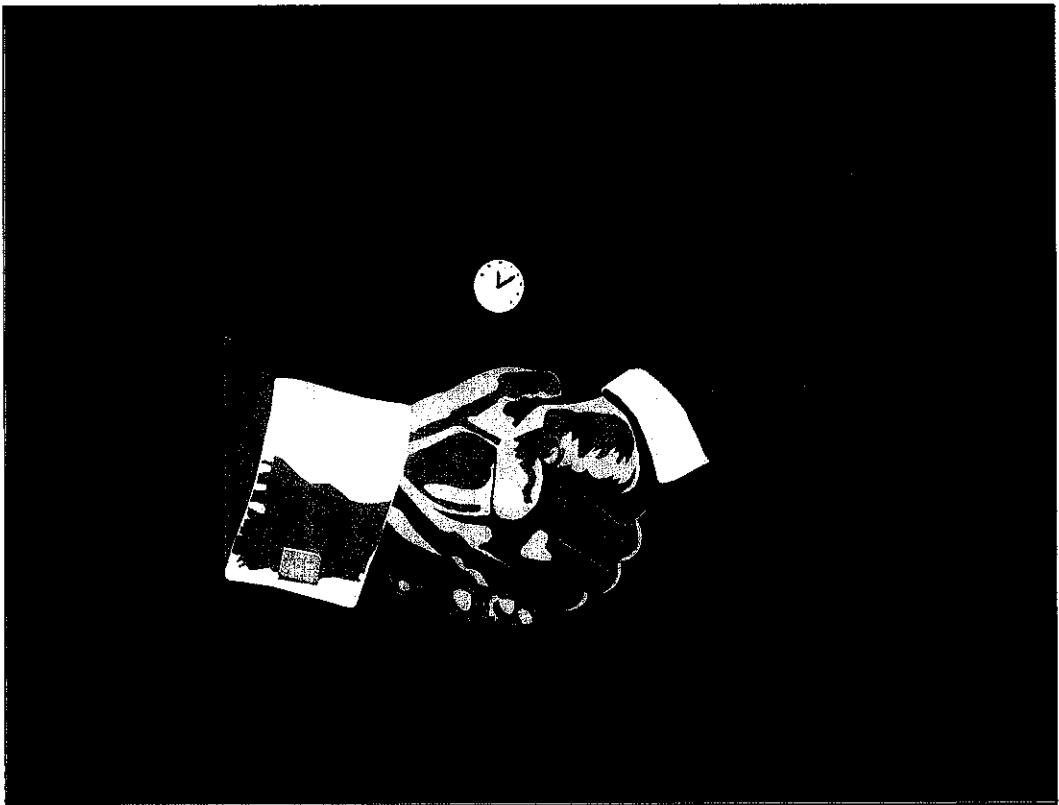
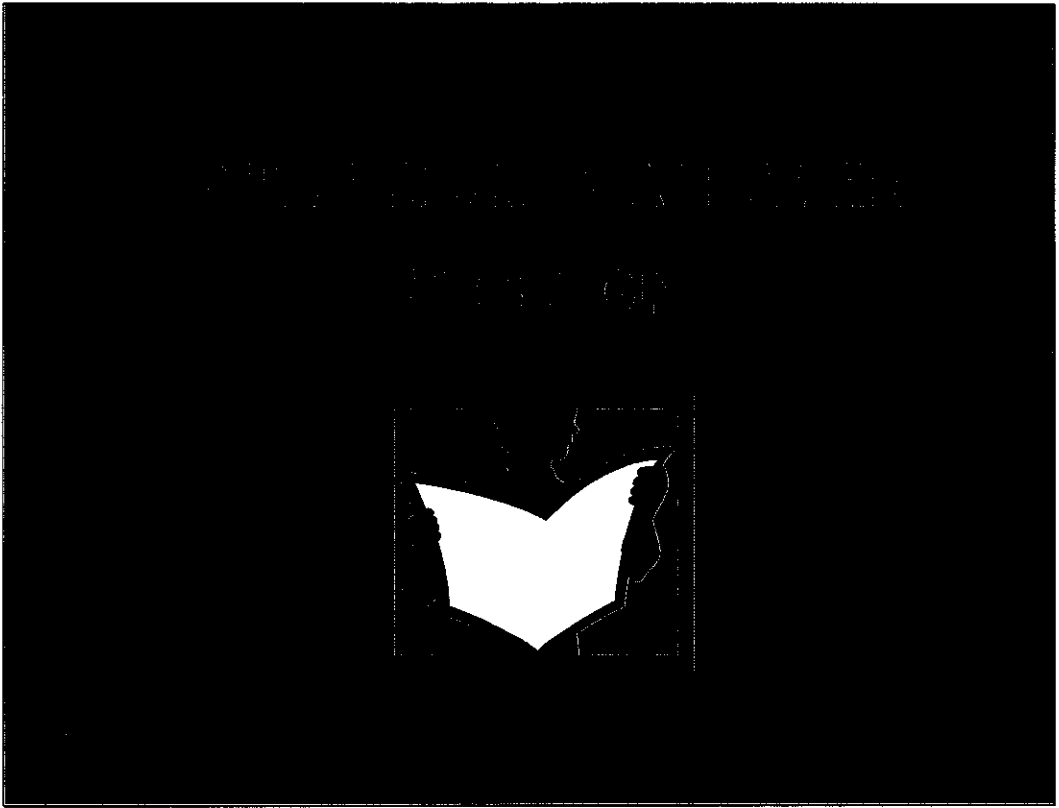
Traditional method in industrial scheduling requires the support of highly involved and complex mathematical tools.

Knowledge system can help to overcome this problem.

- Top three benefits of knowledge system:
1. Faster decision making
  2. Increased productivity
  3. Increased quality of decision making







## **APPENDIX D**

<b>Appendix D.1:</b>	Poster Engineering Design Exhibition (EDX) XIII .	84
----------------------	---	----

In this section, the author would like to take the opportunity to immortalize the poster made during EDX XIII. This poster has been selected among best ten of Final Year Project for Mechanical Engineering Department.

**KNOWLEDGE BASED SYSTEM  
FOR  
PARALLEL MACHINE OPERATION SCHEDULING**



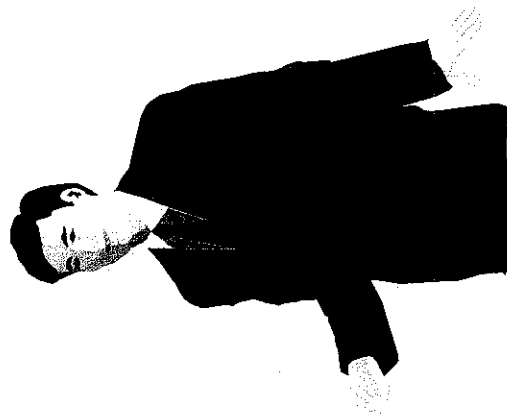
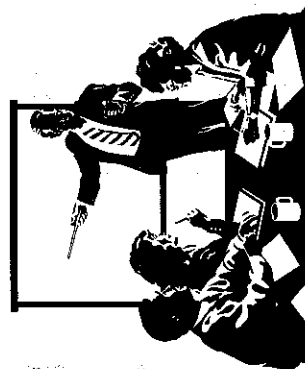
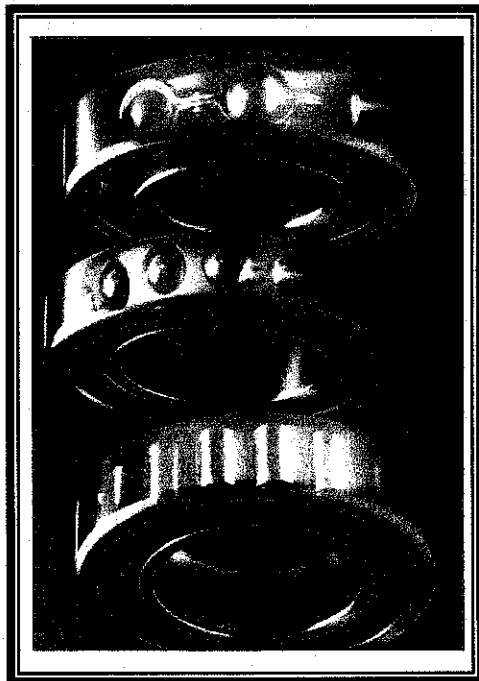
**By:**

**Mohd Razman Bin Ramli**

**0000 1167**

**Supervisor:**

**Mrs Ainul Akmar Bt Mokhtar**



**PROOF**



## PROJECT IDENTIFICATION

- ❖ To engage in the issue of parallel machine operation scheduling from the knowledge based system perspective.

- ❖ Currents operation scheduling - highly involved the complex mathematical tools which time consumption is productivity.

## PROBLEM STATEMENT

- ❖ KBS will provided an integrated system that aided the industrialists to make decision on the parallel operation scheduling issues

- ❖ KBS offers the more reliable, easier, simple handling tools and user-friendly interface

## PROJECT SIGNIFICANT

## VISION

To develop a Knowledge-Based System (KBS) for parallel operation scheduling

## MAIN OBJECTIVES

To develop a knowledge base

- theory of operation scheduling shall be integrated in one source.

To develop an inference engine

- inference engine can be illustrated as the hub of the system where the engine will be a key factor of how the KBS will perform.

To develop a user interface

- a communication interface to establish the interaction between user & the knowledge base.

## TECHNOLOGY

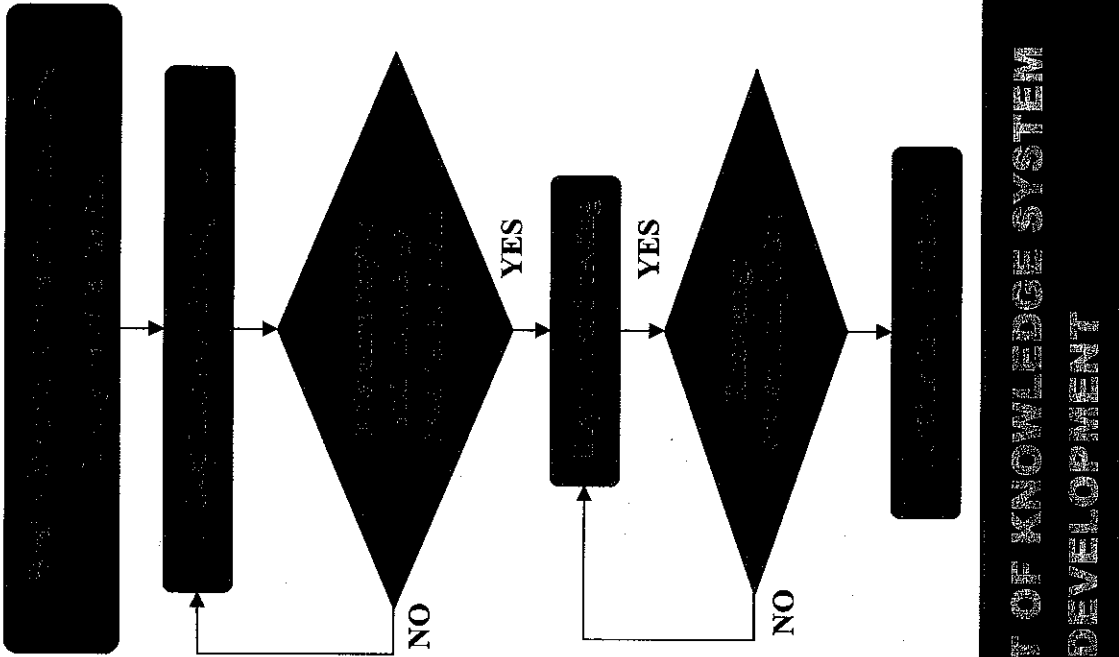
*Operation:* an act performed by machines

*Scheduling:* a programme of work to be done or planned events

*Parallel Machines:* a number of identical machines are available, and jobs can be processed on any one of them.

*Sequencing:* ordering of jobs

*Knowledge Based System:* an interface for user to comprehend the knowledge in a system



# SYSTEM LAYOUT



1

## KNOWLEDGE BASED SYSTEM FOR PARALLEL MACHINE

VERSION 1.1

BY  
MOHD RAZMAN BIN RAMLI  
1167

Engineering 3003  
Innovative Technology PETROBRAS

## OPERATION SCHEDULING

2



WELCOME.WELCOME.WELCOME.WELCOME.WELCOME.

3

## SELECT THE CONDITION

- 1 JOBS OF EQUAL WEIGHTS
- 2 JOBS WITH PRIORITIES BY WEIGHTS
- 3 JOBS WITH DUE DATES

4

## CONDITION 1: JOBS WITH EQUAL WEIGHTS

Before Sorting:

Time	Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7	Job 8	Job 9	Job 10

After Sorting:

Time	Job	Job	Job	Job	Job	Job	Job	Job	Job	Job

Result: Machine 1: Job Job Job Job Job Job Job  
Machine 2: Job Job Job Job Job Job Job  
Machine 3: Job Job Job Job Job Job Job



## CONDITION 2: JOBS WITH PRIORITIES BASED BY WEIGHTS

Before Sorting:

Time	Weight	Ratio	Job	Job	Job	Job	Job	Job	Job	Job

After Sorting:

Time	Job	Job	Job	Job	Job	Job	Job	Job	Job	Job

Result: Machine 1: Job Job Job Job Job Job Job  
Machine 2: Job Job Job Job Job Job Job  
Machine 3: Job Job Job Job Job Job Job

## CONDITION 3: JOBS WITH DUE DATES

Before Sorting:

Time	Due Date	Job	Job	Job	Job	Job	Job	Job	Job	Job

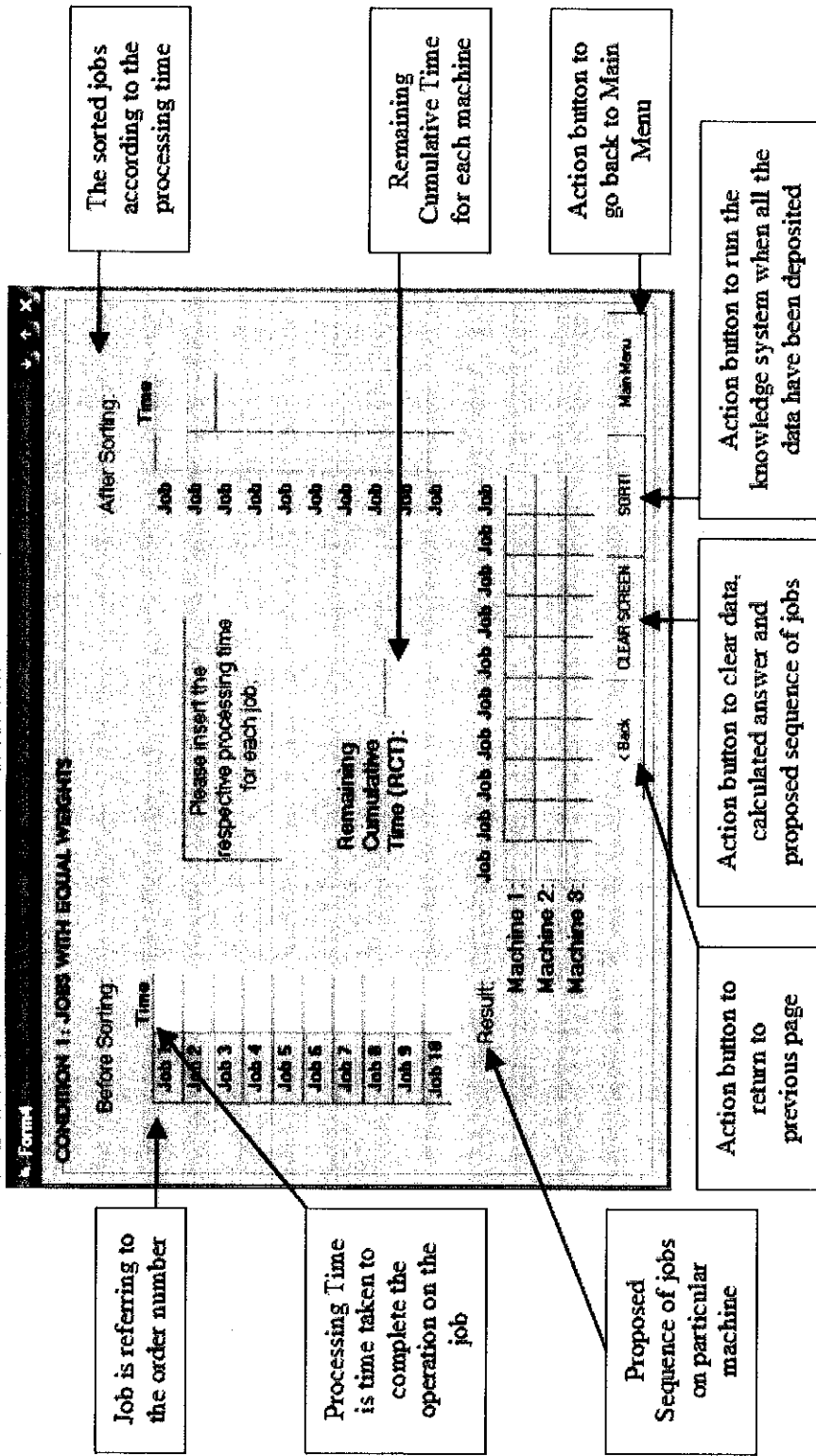
After Sorting:

Time	Due Date	Completion Time	No. of Tasks/Job	Job	Job	Job	Job	Job	Job	Job

Result: Machine 1: Job Job Job Job Job Job Job  
Machine 2: Job Job Job Job Job Job Job  
Machine 3: Job Job Job Job Job Job Job

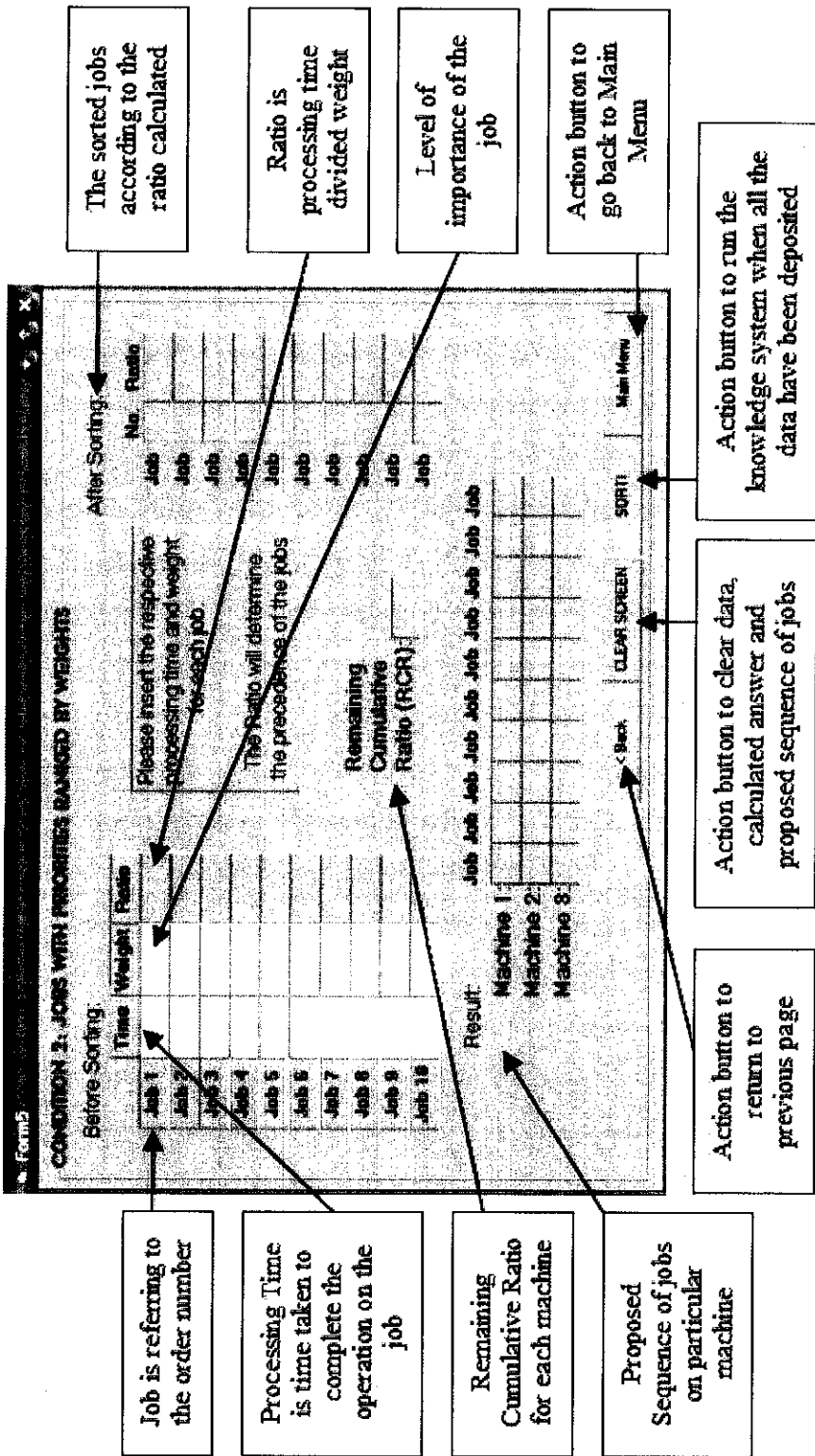
# Jobs with the Equal Weight

4



# Jobs with Priorities Ranked by Weights

5



# Jobs with the Due Dates

6

