The Libyan Academy For High Studies

Misurata - Libya

SCHOOL OF ENGINEERING AND APPLIED SCIENCE

DEPARTMENT OF ENGINEERING PROJECT MANAGEMENT

Makespan Minimization for Identical Parallel Machines

By: Yousef Hussein German

A thesis submitted to the Department of Engineering Project Management

In partial fulfillment of the requirements for the degree of master of engineering science

Supervisor:

Prof. Dr. Mohamed A. Elhaj

2015

Abstract

Reducing production time is important factor for companies which their main objectives are to maximize profits and minimize costs. To achieve these objectives one must follow the scientific methods for scheduling production time.

In this research it has been studied the (Makespan Minimization for Identical Parallel Machines). The problem involves an assignment number of jobs (N) to a set of identical parallel machines (m), when the objective is to minimize the makespan (maximum completion time of the last job on the last machine of the system). In the literature the problem is denoted by ($P_m||C_{\max}$).

The objective of this study is to find the optimal schedule (solution) for identical parallel machines scheduling problems, by using hypothetical situation under defined assumptions and constraints.

Mathematical modeling and algorithms methods are used in this research to represent and solve the problem under study.

Integer Linear Programming model (ILP) is used to formulate the problem. Longest Processing Time algorithm (LPT) is used to find (generate) the initial solution, then the (U-1) algorithm is used to improve the initial solution. The solution algorithms are coded in (MATLAB), also LINGO15 software are used to test and evaluate solution algorithms.

The results of study demonstrated that, the mathematical modeling and algorithms methods are powerful tools and more effective for this kind of problems, compared with other methods.

الملخص

يعد تقليص زمن الإنتاج من أهم العوامل بالنسبة للشركات والمؤسسات التي تسعى لتحقيق الأرباح وتقليل التكاليف المصاحبة لعملية الإنتاج إلى أقل قدر ممكن. ولتحقيق هذه الأهداف يجب إتباع الطرق والأساليب العلمية لجدولة زمن الإنتاج وذلك لضمان إستغلال الوقت بشكل أمثل.

تم في هذا البحث دراسة مشكلة تقليص زمن المعالجة الأقصى للآلات المتوازية المتطابقة (Makespan Minimization for Identical Parallel Machines).

الدراسة تتعلق بتخصيص او(معالجة) عدد من الأعمال (N)،على عدد من الآلات المتوازية المتطابقة (m)، وذلك بهدف تقليص زمن المعالجة الاقصى للأعمال التي يتم تنفيذها على هذه الآلات إلى اقل قدر ممكن. في الدراسات السابقة يشار إلى مشكلة تقليص زمن المعالجة الأقصى للآلات المتوازية المتطابقة بـ ($P_m||C_{\max}$).

الدراسة تهدف إلى إيجاد نظام جدولة (حل) أمثل لمشكلة الآلات المتوازية المتطابقة، وذلك باستخدام حالة افتراضية تحت فرضيات وقيود محددة، استخدم فيها اسلوب النمذجة الرياضية والطرق الخوارزمية لتمثيل وحل المشكلة. حيث تم استخدام نموذج البرمجة الخطية الصحيحة (ILP model) لصياغة وتمثيل المشكلة. بينما تم استخدام خوارزمية (LPT algorithm) لإيجاد الحل الإبتدائي ومن تم تطويره بواسطة خوارزمية (U-1 algorithm) للحصول على الحل النهائي.

كما تم إعداد وتنفيد برنامج حاسوبي (U-1) وتشفير خوارزمية الحل بإستخدام لغة البرمجة الهندسية (MATLAB) .وإختبار النتائج ومقارنتها بإستخدام برنامج (LINGO15).

نتائج الدراسة اثبتت مدى فاعلية استخدام اسلوب النمذجة الرياضية والطرق الخوارزمية لحل هذا النوع من مشاكل جدولة الآلات المتوازية المتطابقة، وذلك لما تتميز به من دقة وسرعة الوصول إلى الحل مقارنة بالطرق الأخرى.

Acknowledgements

I would like to express my gratitude to Dr. Mohamed A. Elhaj . For his supervision, advice, and guidance throughout this study.

I would like to extend my sincere thanks to my family for their support and encouragement.

Also I would like to express my thanks to my colleagues .

Table of Contents

Chapter 3 : Mathematical Modeling and Programming

List of Figures

List of Tables

List of abbreviations and notations

The notation	Term
$C_{\max}$	Makespan (maximum completion time)
$C^*_{\max}$	Optimal Makespan (maximum completion time)
P_m	Identical parallel machines
Q_m	Uniform parallel machines
R_m	Unrelated parallel machines
N	Number of jobs
m	Number of machines
n	Number of different processing times
ILP	Integer Linear Programming
LPT	Longest Processing Time
α | β | γ	Graham's notations
P_j	Processing time of job (j)
x_{ij}	The assignment (decision) variable
L_i	The load on machine (i)
C_i	The completion time of the last job on machine (i)
$C_{\max}^{LPT}$	Maximum completion time obtained by LPT algorithm
i	The index of machine
K_j	Number of jobs with processing time P_j in the load of current machine
r_j	Number of jobs with processing time P_j are still unloaded on any machine
P_h	Processing time of job (h) in the next lexicographic load
UB	Upper Bound for the load of all machines
LB	Lower Bound for the load of all machines
FCFS	First Come First Served
argmax	The index of machine with maximum load
argmin	The index of machine with minimum load
LCFS	Last Come First Served
EDD	Earliest Due Date
PMS	Parallel Machines Scheduling

CHAPTER 1

Introduction

1-1 Introduction:

The time is an important factor for companies which their main objectives are to maximize the profits and minimize costs. To achieve these objectives one must follow the scientific approaches and methods.

Scheduling processing times of jobs or tasks is a very common activity for both industrial or non-industrial works. Although the scheduling began to be taken seriously in manufacturing at the beginning of 20^{th} century with the work of (Henry Gantt) and other pioneers, in the mid of 20^{th} century the first scheduling algorithms were formulated [1].

At the end of the World war II, many of scientists who worked in operational research units within British forces, returned to civilian life in universities and industries and applied operations research methodology by using mathematical models and algorithm methods to analyze and find solutions for complex problems in industries. Since then there has been a growing interest in scheduling [2].

In real live there are many problems can be considered as parallel machine scheduling problems. For example parts waiting for processing on production lines, ships - docks in the port, hospital assistance-patients. etc.

Generally scheduling is evaluated by a performance measure or an objective function, a popular performance is the minimization of the makespan or (maximum completion time of the last job).

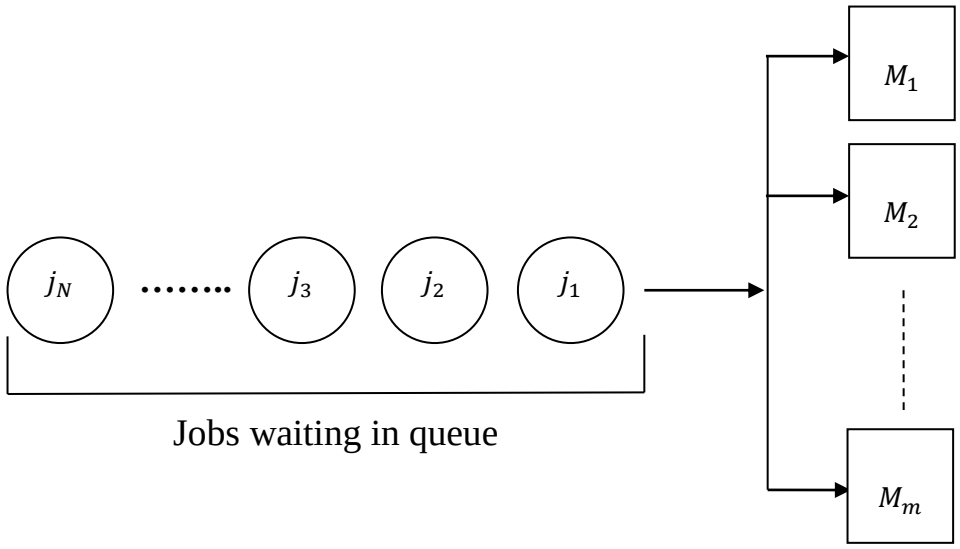
In this research has been studied : Makespan minimization for identical parallel machines by using mathematical modeling and computer programming to formulate and solve the problem.

1-2 Problem Statement:

Parallel machine scheduling can be a very complicated computation process, without using modern technology, especially with a big number of jobs and machines.

In this study the problem deals with number of jobs ($\boldsymbol{N}$ jobs) to be processed on number of identical parallel machines ($\boldsymbol{m}$ machines), when the objective is to minimize makespan (maximum completion time of the last job). The problem denoted by ($P_m||C_{\max}$). Figure (1.1) shows (N) independent jobs on (m) parallel machines. The problem is:

How the jobs can be effectively processed (assigned) on identical parallel machines to minimize the makespan?



Figure(1-1): (m) parallel machines with (N) jobs.

1-3 Assumptions:

To solve the problem statement of this study the following assumptions are used:

1. All machines are identical and able to perform all operations.
2. Each machine can only process one task at any time.
3. Preemption of a job on another machine is not allowed.
4. All jobs are available at time zero.

1-4 Objectives:

1. Conduct an extensive literature review on parallel machines scheduling.
2. Provide mathematical model to represent and formulate the problem.
3. Minimize maximum completion time of the jobs on identical parallel machines.

1-5 Methodology:

1- To complete the study and achieve objectives, integer linear programming (ILP) model has been provided, algorithms and computer program are coded in (MATLAB) language to solve the problem.

2- LINGO15 software was used to evaluate the performance of solution algorithms.

1-6 Limitations:

This study limited by using hypothetical situation under defined assumptions and constraints.

1-7 Study structure:

Chapter 1 :Introduction

This chapter contains the framework of the study that deals with problem statement, assumptions, purposes, methodology and literature review.

Chapter 2 : Machines Scheduling

This chapter provides concepts, definitions and notations of machines scheduling, sequencing rules and parallel machines scheduling.

Chapter 3 : Mathematical Modeling and Programming

This chapter describes the mathematical model which is used to represent and formulate the problem, algorithms solution and computer programming.

Chapter 4 : Application of the model

In this chapter many applications have been investigated to test the performance of solution algorithms.

Chapter 5 : Conclusions and recommendations

Finally, in chapter (5) the summary of solution method and the important points that concluded from the study are displayed, also some suggestions for the future studies.

- **References.**
- **Appendices.**

1-8 Literature Review:

1. Raghavendra .B. V and Murthy A. N. (2011).

In this study the identical parallel machines scheduling problem for minimizing unbalance between the machines or minimizing the makespan is defined as follows: there are number of (n) independent jobs and (m) identical parallel machines. Each job has its fixed processing time. The processing job can be completed by either of machines. A genetic algorithm (GA) is applied for determining the best sequence to minimizing the makespan.

The results of GA approach is compared with the other proposed methods in other papers and found that the proposed GA method gives better results [3].

2. Hashemain Navid, (2010).

The study considered the problem of parallel machine scheduling with multiple planned unavailability periods in the resemble case.

The problem has been formulated as a mathematical programming. An effective algorithm has been developed to solve large-scale practical problems. The algorithm loads machines according to the lexicographical order within a construction and backtracking approach. The results demonstrated that the exact algorithm is able to solve large-scale problems which are not solvable by any other method including integer linear programming (ILP) solver [4].

3. Koulamas Christos, Kyparisis George J, (2009).

In this paper they proposed a modified longest processing time (MLPT) heuristic algorithm makespan minimization problem. The MLPT algorithm schedules the three longest jobs optimally first, followed by the remaining jobs

sequenced according to the LPT rule. The results demonstrated that the MLPT rule has tight worst-case bound of 1.22, an improvement over the LPT bound of 1.28 [5].

4. Chien-Hung Lin and Ching-jong Liao (2008).

In this paper has been studied (minimizing makespan on parallel machines with machine eligibility restrictions) the machines and jobs classified into two levels: high and low levels. A high-level machine can process all jobs while a low-level machine can process only low-level jobs. The objective is to minimize the makespan. A new algorithm has been developed to solve the problem.

The computational experiments showed that, the developed algorithm can find the optimal solution for various sized problems in a short time [6].

5. Sovindik Kaya, (2006).

In this research has been studied the parallel machine scheduling problem subject to availability constraints on each machine. The objectives are to minimize the total completion time and minimize the maximum completion time.

Three heuristic algorithms are developed for the total completion time problem. Also exact and approximation algorithms are proposed for maximum completion time problem. All proposed algorithms are tested through extensive computational results.

For minimizing the total completion time the computational results showed that the improvement algorithm gives very good solutions. And for minimizing maximum completion time the computational experimentation showed that the exact algorithm solve problems and gives excellent solutions [7].

6. Gupta Jatinder N. D. & others, (2004).

In this paper the problem is described as follows: number of (n) jobs available at time zero is to be processed on (m) identical parallel machines. Each job processes without interruption on one of the (m) machines with processing time (p_j). Two simple improvement heuristic algorithms are proposed to minimize the maximum completion time subject to an optimal total flow time. The results showed the proposed heuristics outperformed the existing heuristics making it a better solution methodology for the problem when average performance is the measure of interest [8].

1-9 Comment on previous studies:

There are many studies in the literature dealing with parallel machines scheduling problems. In the above mentioned studies, makespan minimization (maximum completion time)for parallel machines has been calculated and discussed for different instances.

Although the relatively small sized of identical parallel machines problems can be solved by operational methods such as dynamic programming, branch and bond method. But these methods still have limitations with a large number of jobs and when the number of machines are more than two.

The computational experiments in the above studies demonstrated that, the mathematical modeling and algorithms methods are powerful tools and more effective with least effort, compared with other methods, such as CPLEX solver.

CHAPTER 2

Machines Scheduling

2-1 Introduction:

Scheduling is a decision-making process that is used a regular basis in many manufacturing and service industries. It deals with the allocation of resources to tasks over given time periods and its goal is to optimize one or more objectives.

Scheduling playing an important role in most manufacturing, production systems and other types of service industry [9].

During the seventies of the last century, computer scientists discovered scheduling as a tool for improving the performance of computer systems. Furthermore scheduling problems have been investigated and classified with respect to their computational complexity [10].

In practical life there are many scheduling problems types, can be classified depending on machine environment, jobs characteristics, and optimality criteria.

2-2 Scheduling problems classification:

Very common classification of scheduling problems and widely used in the literature is described by three notational form $\alpha \mid \beta \mid \gamma$ and called Graham's notation [1] ,[9].

The (**α**) field describes the machine/scheduling environment.

The(**β**) field provides details of processing characteristics and constraints.

The (**γ**) field describes the objective function to be attained (minimized).

2-2-1 The first field (α): (Machine environment)

Three types of machine environment are defined. However an environment may be divided into the several other environments:

1- Single Machine: the case of single machine is the simplest of all possible machine environments and there is only one machine to process the jobs.
2- Parallel machines: more than one machine is performing the same function. the machines may be identical (P_m) or uniform (Q_m) or unrelated (R_m).
3- Dedicated machines: the machines are specialized for the execution of certain operations. the machines may be Flow Shop (F_m) or Job Shop (J_m) or Open Shop (O_m). Table 2.1 shows (α) field with more details.

Table 2.1: (α) field for common Machine Environment

Environment		Description
Single Machine (1)		There is only one machine to process the jobs.
parallel Machines	Identical (P_m)	All machines have the same speed factor and they can process all the jobs.
	Uniform (Q_m)	Machines with different speeds and each job has a single operation.
	Unrelated (R_m)	There is no relation between machines.
Dedicated Machines	Flow Shop (F_m)	All jobs visit the same machines in the same sequence.
	Job Shop (J_m)	The jobs are passed through machines in different order.
	Open Shop (O_m)	Machines have different speed factors, and jobs should be processed on every single machine.

2-2-2 The second field (β): (processing characteristics / constraints)

Including the constraints such as presence of preemption or not, existence of non-availability periods. Table 2.2 shows (β) field with more details.

Table 2.2: (β) field for common processing characteristics/Constraints.

job characteristics	**Description**
Release Date (r_j)	The job cannot start its processing on a machine before its release date.
Preemptions (*Prmp*)	A job may be interrupted during its processing due to arrival of high priority job.
Precedence (*Prec*)	When one job depends on the completion of another job.
Breakdowns (*Brkdwn*)	machines are not continuously available for processing.
Recirculation (*Recrc*)	When a job visits a machines more than once.
Permutation (*Prmu*)	The processing order of all jobs on one machine is maintained throughout the shop.

2-2-3 The third field (γ): (objective function) describes the performance measure or the optimality criteria. Table 2.3 provides more information.

Table 2.3: (γ) field for common scheduling objective functions.

Objective function	**Description**
Makespan $C_{\max}$	Maximum completion time of the jobs
Total completion time $\sum C_j$	The sum of completion time of all the jobs
Maximum Lateness $L_{\max}$	Worst case of the due date
Total weighted Tardiness $\sum \omega_j T_j$	The sum of weighted tardiness of the jobs
Total weighted completion time $\sum \omega_j c_j$	The sum of weighted completion time of the jobs

2-3 Examples for(α | β | γ)notations:

According to Graham's notation, there are many types of problems can be generated by changing each of three fields (α | β | γ).

1- In the case of study (identical parallel machines scheduling problem) the problem is denoted by ($P_m||C_{\max}$). where:

P_m : in the (α) field denotes to number (m) of identical parallel machines.

$||$: in the (β) field shows that the jobs are not constrained

$C_{\max}$: in the (γ) field shows the optimality criterion is the makespan.

2- Total completion time for single machine ($1||\sum C_i$):

1 : in the (α) field denotes to a single machine.

$||$: in the (β) field shows that the jobs are not constrained

$\sum C_i$: in the (γ) field shows the optimality criterion is total completion time.

3- Total maximum lateness with preemption for uniform parallel machines ($Q_m|prmp|L_{\max}$):

Q_m : in the (α) field denotes to number (m) of uniform parallel machines.

$|prmp|$: in the (β) field shows that the jobs have preemption constraint.

$L_{\max}$: in the (γ) field shows the optimality criterion is maximum lateness.

And so on, with the same above procedure can obtained many types of problems.

2-4 Scheduling rules:

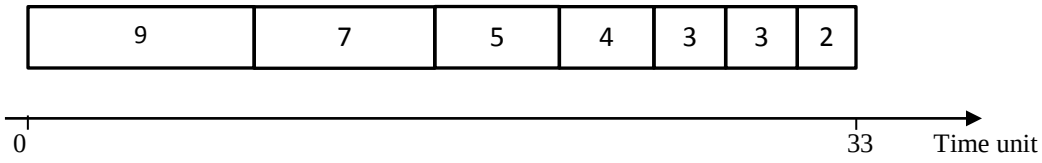
Scheduling rules, also known as (priority or sequencing rules) are used to determine the priority of jobs. Classical parallel machines scheduling problems are often solved by using priority rules (algorithms) and heuristics. These rules are easy to implement and their computational complexity is low. The commonly used priority rules and algorithms are: [11].

1- First Come First Served (FCFS): the job which arrives first at the machine will be served first. this rules is commonly applied in serves centers such as banks.
2- Last Come First Served (LCFS): the job which arrives last at the machine will be served first.
3- Shortest Processing Time rule (SPT): jobs are arranged in ascending order of their processing times. the job with the shortest processing time is processed first.
4- Earliest Due Date rule (EDD): jobs are processed according to the increasing order of their due dates (the job with smallest due date is processed first).
5- Longest Processing Time rule (LPT): jobs are arranged in decreasing order of their processing times. Jobs with large values of processing times are given high priority for scheduling (the job with the longest processing time is processed first).

   For example:

   If there are seven jobs and their processing times are (2 , 5 , 9 , 3 , 3 , 7 , 4). find the sequencing order by using (LPT) rule.

   According to (LPT) rule the jobs will be sorted in decreasing order depending on their processing times as follow: (9 , 7 , 5 , 4 , 3 , 3 , 2). The figure(2-1) shows the sequencing order by using (LPT) rule.



Figure(2-1): The sequencing order by using (LPT) rule.

6- Longest Remaining Processing Time rule (LRPT): this rule is to be applied when job preemptions are allowed. Jobs having longest remaining processing time are scheduled first.

7- Shortest Remaining Processing Time rule (SRPT): this rule is to be applied when job preemptions are allowed. Jobs having shortest remaining processing time are scheduled first.

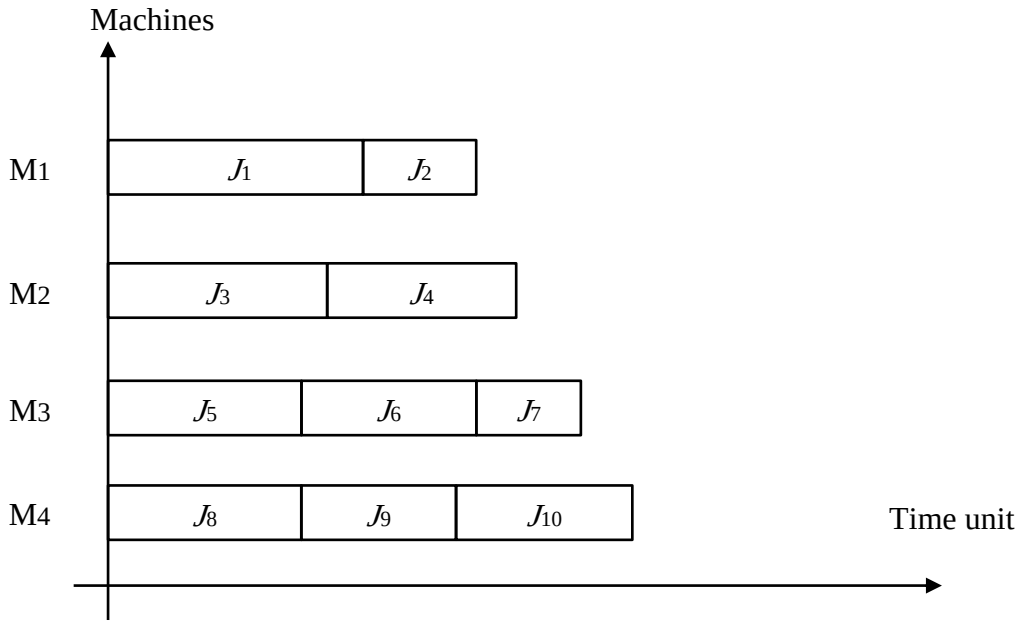
2-5 Parallel Machines Scheduling:

The aim of machine scheduling is to assign jobs to the machines based on related objective function to minimize operating time and increase productivity[3].

Parallel machines scheduling is the task of determining when each operation has to start and finish on each machine and using available resources in efficient manners to execute(assign) jobs or tasks on machines.

Parallel machine scheduling also known as (parallel task scheduling), involves assigning jobs or (tasks) on a set of machines in parallel[12].

The parallel machines can be identical or uniform or unrelated. In this research the case of study is (Identical Parallel Machines). Identical parallel machines are a set of machines have the same speed factor and they can process all the jobs. Figure(2-2) shows Gantt chart for set of machines and jobs.



Figure(2-2): Shows Gantt chart for set of machines and jobs.

In the literature there are different approaches are followed to solve identical parallel machines scheduling problems for minimizing the makespan.

But in the case of processing objects of large scale, heuristic procedure is not yet effective enough, especially the accuracy of the solution need improving.

Genetic (GA) and deterministic algorithms are applied for optimization problems to get optimal solution. [3].

CHAPTER 3

Mathematical Modeling and Programming

3-1 Introduction:

Mathematical modeling and computer programming methods are used to represent and solve the complex problems, such as optimization problems.

There is no single general technique (algorithm) can be followed to solve all mathematical models that can arise in practice. Instead, the type and complexity of the mathematical model dictate the nature of the solution method [13].

3-2 Mathematical model:

A mathematical model is a mathematical representation of an actual situation or problem under study and describe important relationship between variables [14].

The mathematical model has three main components:

1- Objective function: is a mathematical expression that combines the variables to express the goal, which represent profits or costs to maximize or minimize the objective function.
2- Decision variables: are representation to things which can adjust or control to find the values that provide the best value of objective function.
3- Constraints: are a mathematical expression that combines the variables to express limits on the possible solutions [15].

3-3 Mathematical model formulation:

As mentioned in the chapter (2) and according to α | β | γ notations, the identical parallel machines scheduling problem is denoted by $P_m||C_{\max}$.

P_m : in the (α) field denotes to number (m) of identical parallel machines.

| | : in the (β) field shows that the jobs are not constrained.

$C_{\max}$: in the (γ) field shows the optimality criterion (minimizing makespan).

The simplicity of linear functions makes linear models easy to formulate, analyze and find an optimal solution accurately and quickly. The integer linear programming (ILP)model is a very common tool used to represent optimization problems such as the problem at the hand ($P_m||C_{\max}$) can be formulated as (ILP) model.

Where:

$C_{\max}$: the makespan (maximum completion time).

N : number of jobs (integer).

m : number of machines. (integer).

p_j :processing time of job (j) (integer).

x_{ij} : the assignment (decision) variable.

The mathematical model of the problem is:

$$Min \quad C_{\max} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}1)$$

subject to;

$$\sum_{j=1}^{N} p_j x_{ij} \le C_{\max} \qquad i=1,\ldots,m \qquad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}2)$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad j=1,\ldots,N \qquad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}3)$$

$$x_{ij} \in \{0,1\} \qquad i=1,\ldots,m\,, \qquad j=1,\ldots,N \qquad \ldots\ldots\ldots\ldots \quad (3\text{-}4)$$

$$C_{\max} \ge 0 \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3\text{-}5)$$

The first formula (3-1) in the model is the objective function ($C_{\max}$) makespan, which should be minimized.

Constraint (3-2) assures that the load on any machine is equal or less than ($C_{\max}$).

Constraint (3-3)shows that each job must be assigned to exactly one machine.

Constraint (3-4) describes the type of the assignment decision variable (x_{ij})

$$x_{ij} = \begin{cases} 1 & \text{if the job } j \text{ is assigned to machine } i \\ 0 & \text{if the job } j \text{ is not assigned to machine } i \end{cases}$$

Constraint (3-5) shows that, the objective function $C_{\max}$ is integer variable.

Example: There are 10 jobs their processing times given in the table below, the jobs processed by 3 identical parallel machines, each machine can only process one job at any time and pre-emption of the job on another machine is not allowed.

Find mathematical model for the problem to minimize Makespan (C_{max}).

Job No;	1	2	3	4	5	6	7	8	9	10
Processing time	12	10	13	9	8	14	6	3	11	5

The solution: The mathematical model of the problem is:

$Min \ C_{max}$

subject to;

$$\sum_{j=1}^{10} p_j x_{ij} \le C_{max} \qquad i=1,2,3$$

P1*X11 + P2*XI2 + P3*X13 + P4*X14 + P5*X15 + P6*X16 + P7*X17 + P8*X18 + P9*X19 + P10*X110 <= C_{max}
P1*X21 + P2*X22 + P3*X23 + P4*X24 + P5*X25 + P6*X26 + P7*X27 + P8*X28 + P9*X29 + P10*X210 <= C_{max}
P1*X31 + P2*X32 + P3*X33 + P4*X34 + P5*X35 + P6*X36 + P7*X37 + P8*X38 + P9*X39 + P10*X310 <= C_{max}

$$\sum_{i=1}^{3} x_{ij} = 1 \qquad j=1,2,3,4,5,6,7,8,9,10$$

X11 + X21 + X31 = 1
X12 + X22 + X32 = 1
X13 + X23 + X33 = 1
X14 + X24 + X34 = 1
X15 + X25 + X35 = 1
X16 + X26 + X36 = 1
X17 + X27 + X37 = 1
X18 + X28 + X38 = 1
X19 + X29 + X39 = 1
X110 + X210 + X310 = 1

$x_{ij} \in \{0,1\}$, i=1,2,3 , j=1,2,3,4,5,6,7,8,9,10

X11, X12, X13, X14, X15, X16, X17, X18, X19, X110 $\in$ {0,1}

X21, X22, X23, X24, X25, X26, X27, X28, X29, X210 $\in$ {0,1}

X31, X32, X33, X34, X35, X36, X37, X38, X39, X310 $\in$ {0,1}

$C_{\max} \geq 0$.

Variable	Value	Reduced Cost
CMAX	31.00000	0.000000
P1	12.00000	0.000000
X11	1.000000	12.00000
P2	10.00000	0.000000
X12	1.000000	10.00000
P3	13.00000	0.000000
X13	0.000000	13.00000
P4	9.000000	0.000000
X14	1.000000	9.000000
P5	8.000000	0.000000
X15	0.000000	8.000000
P6	14.00000	0.000000
X16	0.000000	14.00000
P7	6.000000	0.000000
X17	0.000000	6.000000
P8	3.000000	0.000000
X18	0.000000	3.000000
P9	11.00000	0.000000
X19	0.000000	11.00000
P10	5.000000	0.000000
X110	0.000000	5.000000
X21	0.000000	0.000000
X22	0.000000	0.000000
X23	1.000000	0.000000
X24	0.000000	0.000000
X25	0.000000	0.000000
X26	0.000000	0.000000
X27	1.000000	0.000000
X28	0.000000	0.000000
X29	1.000000	0.000000
X210	0.000000	0.000000
X31	0.000000	0.000000
X32	0.000000	0.000000
X33	0.000000	0.000000
X34	0.000000	0.000000
X35	1.000000	0.000000
X36	1.000000	0.000000
X37	0.000000	0.000000
X38	1.000000	0.000000
X39	0.000000	0.000000
X310	1.000000	0.000000

3-4 Solution method:

To find the optimal solution (optimal $C_{\max}$)for above mathematical model it should be find the optimal list of scheduling. Therefore, every possible order of jobs should be checked. Assume that if there are (*N*) jobs then (*N*!) permutations should be checked to find the optimal solution. This is very complicate operation even with numbers of job (*N*) relatively are not big. For example if (*N*=10), to find the optimal solution there are (10! = 3628800 order) should be checked.

Although it is not easy to find optimal solution directly for these kinds of problems, using appropriately algorithms can obtain best solutions with the least effort comparing with other methods.

In this situation of study, the solution can be obtained in two steps: by using two algorithms, (U-1) algorithm and the Longest Processing Time (LPT) algorithm.

1. The LPT algorithm using to find (generate) the initial solution.
2. The (U1) algorithm using to improving the initial solution.

3-5 Solution Algorithms:

Simply, the algorithm is a set of logical steps used to solve a specific problem. Since the starting of using the algorithm methods to solve the complicated problems in different fields, there are many types of algorithms have been explored and improved. Although most of the algorithm methods have nearly the same principles, but there is no specific algorithm can be used for all situations, in this study two types of algorithms are used to solve the problem:

3-5-1 LPT algorithm:

The (LPT) algorithm always puts the smaller jobs towards the end of schedule, that makes it easier to balance machines loads.

According to the(LPT) algorithm whenever one of the (*m*) machines is freed, the longest job among number of jobs (*N*) in decreasing order waiting for processing is selected to be next [16].

The next job (j) will be scheduled on machine (i^*) according to the formula:

$$i^* = argmin \left\{ L_i + p_j : i = 1, ..., m \right\} \quad \text{..... (3-6)}$$

Where: L_i is the load on machine (i) , p_j the processing time of job (j).

And the makespan ($C_{\max}$) of any feasible solution is:

$$C_{max} = max \{ C_i : i = 1, ..., m \} \text{ ..... (3-7)}$$

3-5-2 Steps of LPT algorithm:

Step1: sort (*N*) jobs according to the non-increasing order of their processing time.

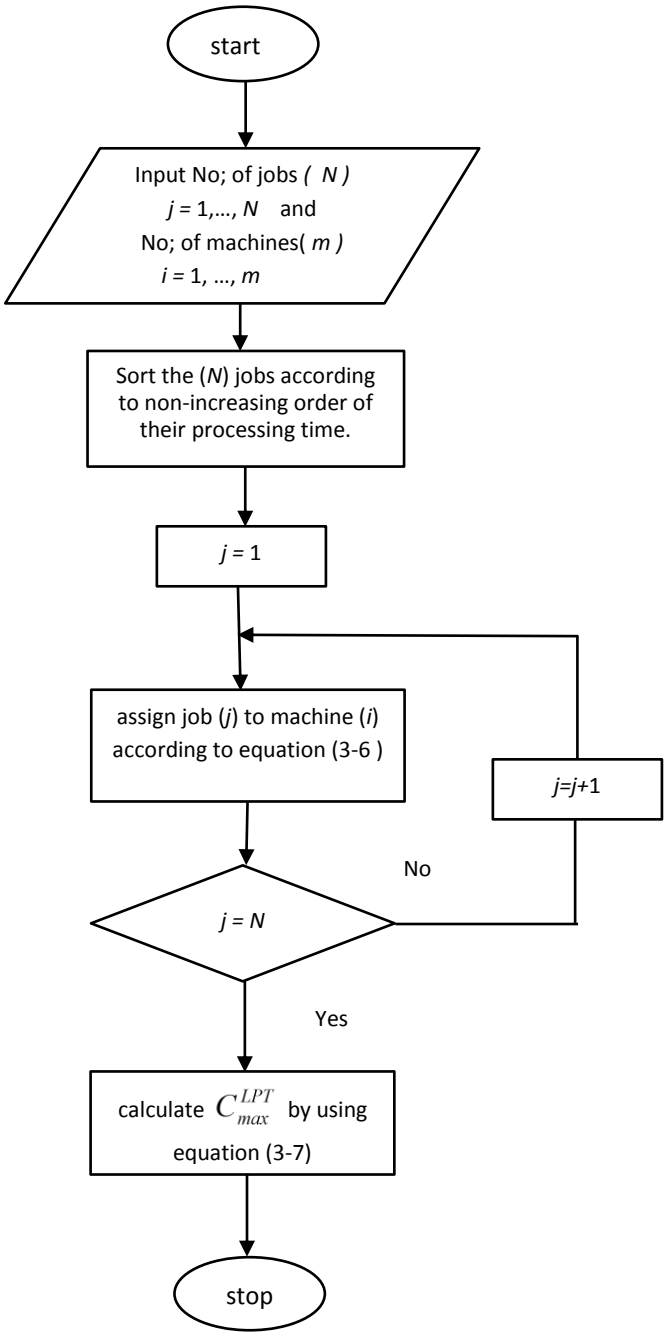
Step 2: set (j=1).

Step 3: assign job (j) to machine (*i*) according to equation (3-6).

Step 4: if j=N (all jobs are allocated) then go to the next step, otherwise set j=j+1 and go to step 3.

Step 5: calculate C_{max}^{LPT} by using equation (3-7).

These steps are shown in the flowchart given in figure (3-1).



Figure(3-1): The flow chart of (LPT) algorithm.

3-5-3 U-1 algorithm:

The U-1 algorithm, used as a main algorithm and based on two types of operations: (construction and backtracking). The load of machines are determined in sequence, one after another. Therefore the potential load of machine (i) with ($1< i \le m$) depends on the loads of the previous machines. And the loads on machines continue until assigning the last job [4].

Assume that, ($C_{\max}$): is the makespan of the current feasible solution. So if the optimal solution has not been explored yet, then its value is not greater than ($C_{\max} - 1$) in the case of integer processing times. Therefore:

$$UB = C_{\max} - 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \text{(3-8)}$$

Where (UB) is the upper bound for the load of all machines in the feasible solution and still to be investigated.

And the lower bound (LB) for all the other loads in the same feasible solution can be found by the next equation:

$$LB = \max\left\{0\,,\ \sum_{j=1}^{N} p_j - (m-1)\,UB\right\} \quad \ldots\ldots\ldots\ldots \text{(3-9)}$$

Equation (3-9) implies that if all machines except one have a total load equal to the upper bound, then the remaining load is the lower bound. After finding a new feasible solution both bounds (upper and lower bound) tighten up.

To ensure that the load on any machine is feasible, the total load on the machine, must be between the lower and upper bounds.

$$LB \le \sum_{j=1}^{n} p_j\, k_j \le UB \quad \ldots\ldots\ldots\ldots\ldots \text{(3-10)}$$

Where: (k_j) is integer, and j =1,…..,n.

3-5-3-1 Construction phase:

There are many feasible solutions can satisfy the formula (3-10). For an implicit enumeration procedure, the feasible solutions must be ordered somehow and enumerated in this order. One easy way to perform this task is to order them in lexicographical order also known as (the dictionary order) or (the alphabetic order), (see appendix 2).

In the construction phase, and to load the machines one by one, the largest solution in the lexicographical order for (n) jobs types is given by formulae (3-11)&(3-12). That is when the machine has no previous load.

$$k_1 = \min \left\{ \left\lfloor \frac{UB}{p_1} \right\rfloor, r_1 \right\} \quad \text{.....} \quad \text{(3-11)}$$

$$k_j = \min \left\{ \left\lfloor \frac{UB - \sum_{h=1}^{j-1} p_h \, k_h}{p_j} \right\rfloor, r_j \right\} \ , \ j = 2 \ , ... , n. \quad \text{.....} \quad \text{(3-12)}$$

The feasibility of the load is checked by formula (3-10).

If the construction is successful (all machines are loaded and all conditions are satisfied), both the upper bound and lower bound are update.

3-5-3-2 Backtracking phase:

The algorithm is always looking for the optimal solution. Therefore the backtracking is applied whenever any of the following two situations are accounted:

1- When the load of a machine is not feasible (cannot satisfy formula (3-10)).
2- When a new feasible solution has been found for all machines (updating makespan).

when the machines has no further feasible load (the load of machines are not satisfy formula (3-10)). Then there is no feasible solution for the current upper bound and optimal makespan ($C^*_{\max}$) is equal to pervious upper bound.

$$C^*_{\max} = UB + 1 \quad \text{..... (3-13)}$$

3-5-3-3 Steps of (U-1) algorithm:

Step 1: set ($i = 1$) and ($t = 1$).

Step 2: use (LPT) algorithm to find the initial solution (upper bound).

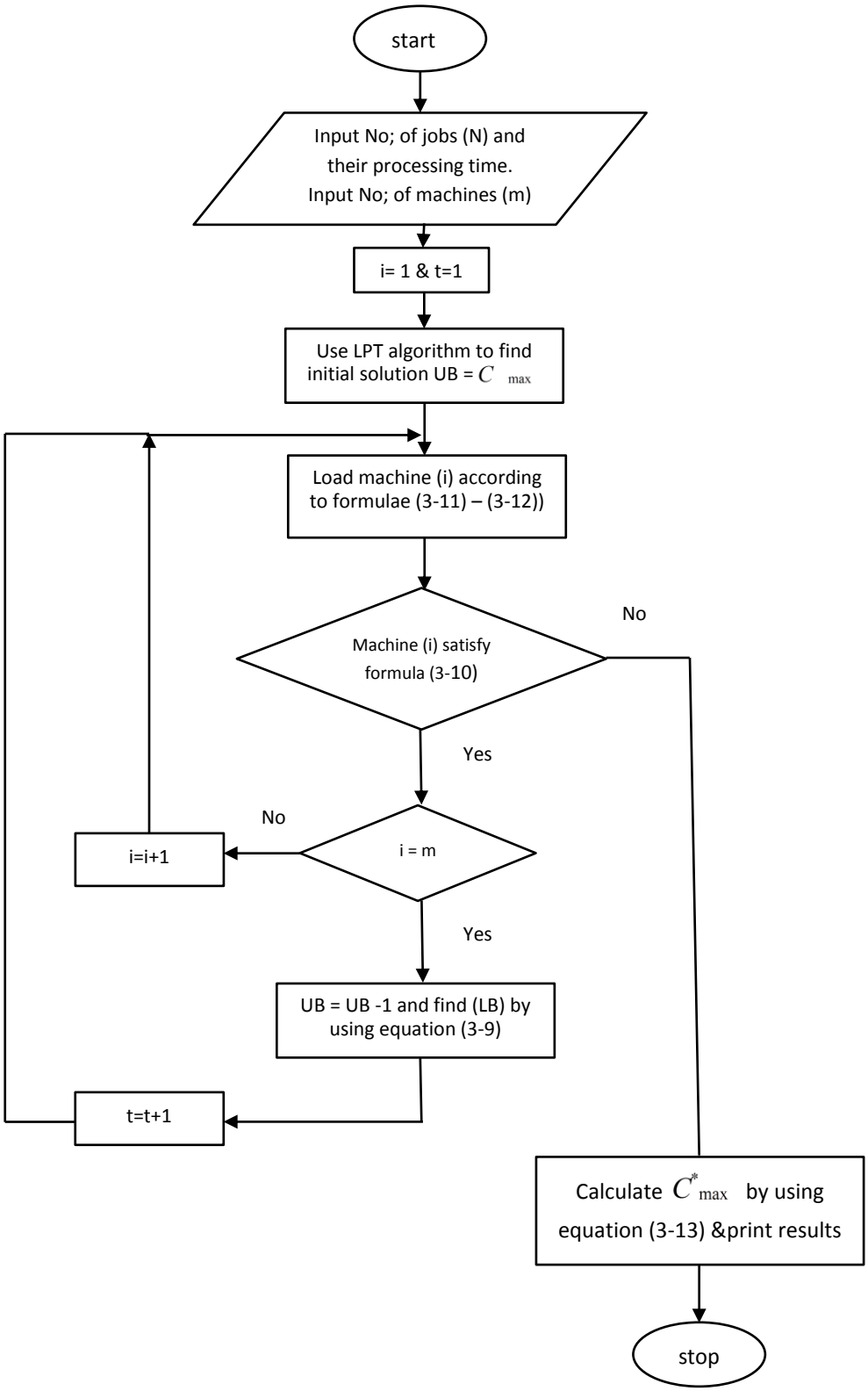
Step 3: load machine (i) according to the formulae (3-11),(3-12).

Step 4: if machine (i) does not satisfy inequality (3-10) go to step (6).

Step 5: if ($i = m$) set ($UB = C_{\max} - 1$) and ($t = t + 1$) and go to step (3). otherwise set ($i = i + 1$) and go to step (3).

Step 6: calculate $C^*_{\max}$ by using equation (3-13).

Step 7: print results and stop.



Figure(3-2): The flow chart of (U-1) algorithm.

3-6 Computer program:

To complete the study and get results, computer program (U-1) has been designed and applied, algorithms solution are coded in (MATLAB) language.[17].

The program is designed to solve variety of problems depending on number of machines (m) and number of jobs (N).

After input number of machines (m) and number of jobs (N) with their processing time, the program can solve the problem by calling some functions and goes through many iterations until finding the best feasible solution. Figure (3- 3) shows the(U-1) program interface.

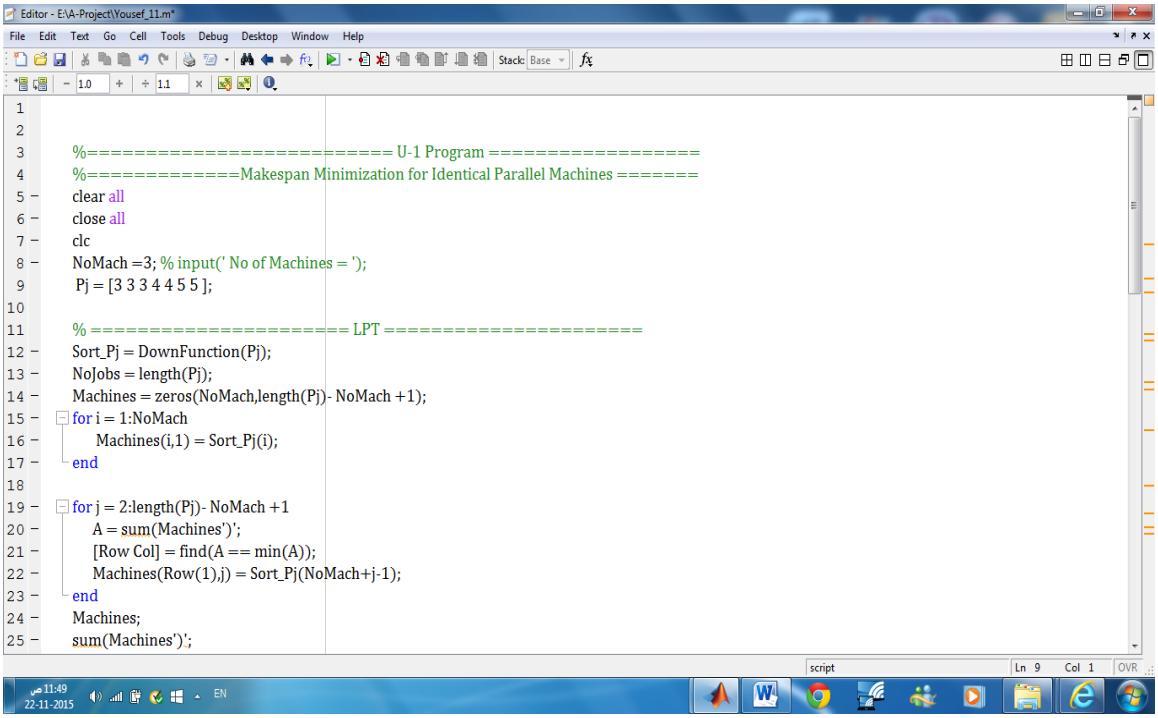


Figure (3- 3): (U-1) program interface.

3-7 Testing (U-1)program:

To test the performance of (U-1) program the ILP model was solved by one of the commercial available solvers to expressing and solving optimization models (LINGO 15) software. [18].

Applications and results in chapter (4) demonstrated that, (U-1) program provide identical results with (LINGO 15) software. Figure (3-4) shows the LINGO15 interface.

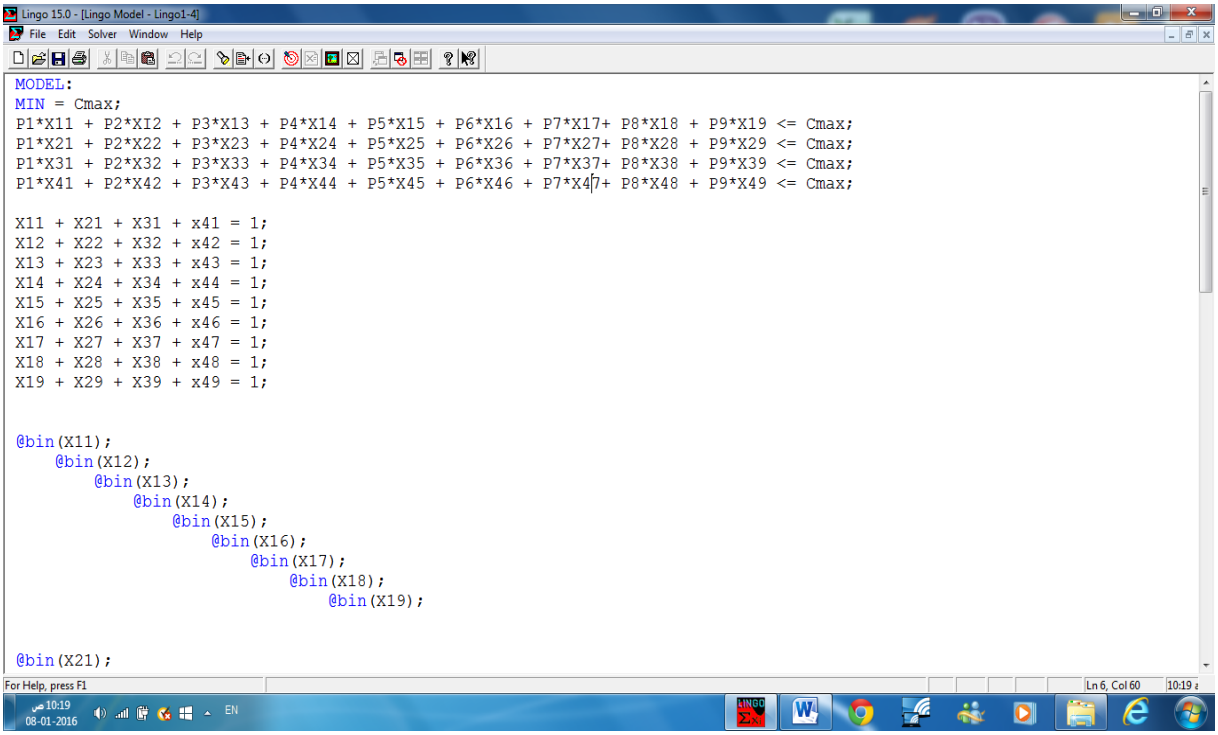


Figure (3- 4): LINGO15 program interface.

CHAPTER 4

validity of the model

4-1 Introduction:

Many computational applications with different levels of difficulties have been carried out to evaluate the performance of ILP model and (U-1 algorithm).

The application are (different number of machines and jobs) to show the

The difficulty of the experiments is described by the number of iteration required by the solution algorithm to find the best feasible solution.

All experiments are carried out on a personal computer (Intel(R) core(TM)i5 CPU 2.30 GHz, RAM 4.00 GHz. System type 64 bit).

4-2 Case (1):

Find the optimal schedule (solution) by using the (LPT) and (U-1) algorithms to minimize maximum completion time (C_{max}) for (7) jobs their processing times (p_j) given bellow:

p_j = [3, 3, 3, 4, 4, 5, 5]

The jobs processed by (3) identical parallel machines, each machine can only process one job at any time and preemption of the job on another machine is not allowed.

The solution:

find the initial schedule (solution):

4-2-1 Using LPT algorithm to find the initial schedule (solution):

Step 1: sort the jobs in non-increasing order (5, 5, 4, 4, 3, 3, 3)

Step 2: set j =1

Step 3: assign job (1) to the machine (i) according to equation (3-6).

$$i = argmin \left\{ L_i + p_j : i = 1, ..., m \right\}$$

The load on machines:

C_1 = P1 + P5 + P7 = 5 + 3 + 3 = 11

C_2 = P2 + P6 = 5+ 3 = 8

C_3 = P3 + P4 = 4+ 4 = 8

Step 4: $j = N = 7$ (all jobs are assigned (allocated)).

Step 5: calculate C_{max} by using equation (3-7).

$$C_{max} = max\{C_i : i = 1, 2, 3\} \rightarrow C_{max} = max\{C_1, C_2, C_3\}$$

$$C_{max} = max\{11, 8, 8\} \quad \rightarrow \quad C_{max} = 11$$

Machines loads and the initial solution obtained by LPT algorithm are illustrated in Figure (4-1) by Gantt chart:

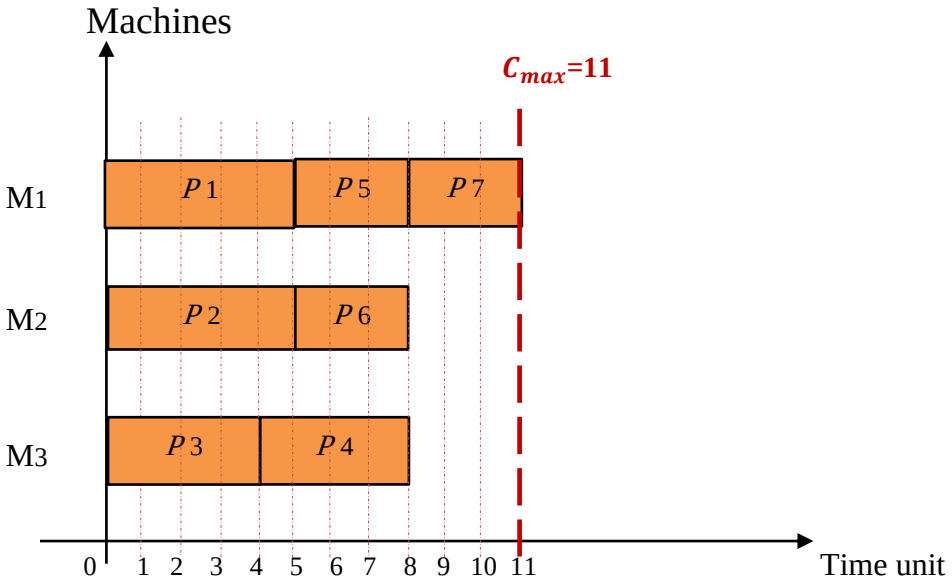


Figure (4-1): The initial solution obtained by LPT algorithm.

4-2-2 Using (U-1) algorithm to improve the initial solution:

Step 1: set ($i=1$) and ($t=1$).

Step 2: use (LPT) algorithm to find the initial solution (upper bound).

At first iteration $UB = C_{max} = 11$

And the lower bound (LB)can be found by equation (3-9):

$$LB = \max\left\{0\,,\ \sum_{j=1}^{N} p_j - (m-1)\,UB\right\}$$

$$LB = \max\left\{0\,,\ \sum_{j=1}^{7} p_j - (3-1)\,11\right\} \rightarrow LB = \max\{0\,,\ 27-(2)\,11\} \rightarrow LB = \max\{0\,,\ 5\} \rightarrow LB = 5$$

Step 3: load machine(1) according to the formulae (3-11)&(3-12).

$$k_1 = \min\left\{\left\lfloor \frac{UB}{p_1} \right\rfloor,\ r_1\right\}$$

$$k_j = \min\left\{\left\lfloor \frac{UB - \sum_{h=1}^{j-1} p_h\ k_h}{p_j} \right\rfloor,\ r_j\right\}\ ,\ j = 2\,,\ldots..,\ n.$$

Therefore:

$$k_1 = \min\left\{\left\lfloor \frac{11}{5} \right\rfloor,\ 2\ \right\} \rightarrow k_1 = \min\{\lfloor 2.2 \rfloor,\ 2\} \rightarrow k_1 = 2$$

$$k_2 = \min\left\{\left\lfloor \frac{UB - \sum_{h=1}^{2-1} p_h k_h}{p_2} \right\rfloor, r_2\right\},\ j=2 \ \rightarrow\ k_2 = \min\left\{\left\lfloor \frac{11-(p_1 k_1)}{4} \right\rfloor, 2\right\},\ j=2$$

$$k_2 = \min\left\{\left\lfloor \frac{11-(5\times 2)}{4} \right\rfloor, 2\right\} \ \rightarrow\ k_2 = \min\{\lfloor 0.25 \rfloor, 2\} \rightarrow\ k_2 = 0$$

$$k_3 = \min\left\{\left\lfloor \frac{UB - \sum_{h=1}^{3-1} p_h k_h}{p_3} \right\rfloor, r_3\right\},\ j=3 \rightarrow k_3 = \min\left\{\left\lfloor \frac{11-(5\times 2+4\times 0)}{3} \right\rfloor, 3\right\},\ j=3$$

$k_3 = \min\{\lfloor 0.33 \rfloor, 3\},\ j=3 \ \rightarrow\ k_3 = 0$ Thus

The load on machine(1) is (C_1):

M1 : $C_1 = p_1 k_1 + p_2 k_2 + p_3 k_3 \ \rightarrow\ C_1 = 5\times 2 + 4\times 0 + 3\times 0 \rightarrow C_1 = 10$

Step 4:

$LB \le \sum_{j=1}^{n} p_j\ k_j \le UB$ ……. (3-10) $\rightarrow\ 5 \le \sum_{j=1}^{n} p_j\ k_j \le 11$

Where the load on machine(1). $C_1 = \sum_{j=1}^{3} p_j\ k_j = 10$

then machine(1) satisfy inequality (3-10)

$i = i+1 \rightarrow i = 1+1 \rightarrow i = 2 \neq m$. then the next step is (3).

Step 3: load machine (2) according to the formulae (3-11)&(3-12).

With the same procedure for loading machine (1). The load on machine (2) is(C_2):

M2 : $C_2 = p_1 k_1 + p_2 k_2 + p_3 k_3 \ \rightarrow\ C_2 = 5\times 0 + 4\times 2 + 3\times 1 \rightarrow C_2 = 11$

Where the load on machine (2). $C_2 = \sum_{j=1}^{3} p_j\ k_j = 11$

then machine 2 satisfy inequality (3-10).

$i = i + 1 \rightarrow i = 2 + 1 \rightarrow i = 3$

The load on machine(3) is (C_3):

M3 : $C_3 = p_1 k_1 + p_2 k_2 + p_3 k_3 \rightarrow C_3 = 5\times 0 + 4\times 0 + 3\times 2 \rightarrow C_3 = 6$

Step 4:

$LB \le \sum_{j=1}^{n} p_j\ k_j \le UB$ ..... (3-10) $\rightarrow 5 \le \sum_{j=1}^{n} p_j\ k_j \le 11$

Where the load on machine (3). $C_3 = \sum_{j=1}^{3} p_j\ k_j = 6$

then machine (3) satisfy inequality (3-10).

Step 5: $i = m = 3$ then Calculate C_{max} by using formula (3-7).

$C_{max} = max\left\{C_i : i = 1, ... , m\right\}$ (3-7)$\rightarrow$ $C_{max} = max\left\{C_i : i = 1, 2, 3\right\}$

$C_{max} = max\left\{C_1, C_2, C_3\right\} \rightarrow C_{max} = max\left\{10, 11, 6\right\} \rightarrow C_{max} = 11$

($UB = C_{\max} - 1$) and ($t = t + 1$).

$UB = C_{\max} - 1 \rightarrow UB = 11 - 1 \rightarrow UB = 10$ and $t = t + 1 \rightarrow t = 2$

That is meaning: the new upper bound is $UB = 10$ and the next iteration is $t = 2$

And the new lower bound (LB)can be found by equation (3-9): $LB = 7$

With the same above procedure the new load on machines are:

$C_1 = 10$

$C_2 = 8$

$C_3 = 9$

Calculate C_{max} by using formula (3-7).

$C_{max} = max\{10, 8, 9\} \rightarrow C_{max} = 10$

$UB = C_{\max} - 1 \rightarrow UB = 10 - 1 \rightarrow UB = 9$ and $t = t + 1 \rightarrow t = 3$

That is meaning: the new upper bound is $UB = 9$ and the next iteration is $t = 3$

And the new lower bound (LB)can be found by equation (3-9): LB=9

the new load on machines are:

$C_1 = 9$

$C_2 = 9$

$C_3 = 9$

i=3 , m=3 $\rightarrow i = m$

Calculate C_{max} by using formula (3-7).

$C_{max} = max\{C_1, C_2, C_3\}$

$C_{max} = max\{9, 9, 9\} \rightarrow C_{max} = 9$

$(UB = C_{\max} - 1)$ and $(t = t + 1)$.

$UB = C_{\max} - 1$

$UB = 9 - 1 \rightarrow UB = 8$ and $t = t + 1 \rightarrow t = 4$

That is meaning: the new upper bound is $UB = 8$ and the next iteration is $t = 4$

And the new lower bound (LB)can be found by equation (3-9): LB=11

the new load on machines are:

$C_1 = 8$; $C_2 = 8$, $C_3 = 8$

Where the load on machines does not satisfy inequality (3-10). Then go to step (6)

Step 6: set $C^*_{\max} = UB + 1$

$C^*_{\max} = 8 + 1 \rightarrow C^*_{\max} = 9$

The optimal solution $C^*_{\max} = 9$.

figure(4-2) shows Gantt chart for the solution and the load on machines .

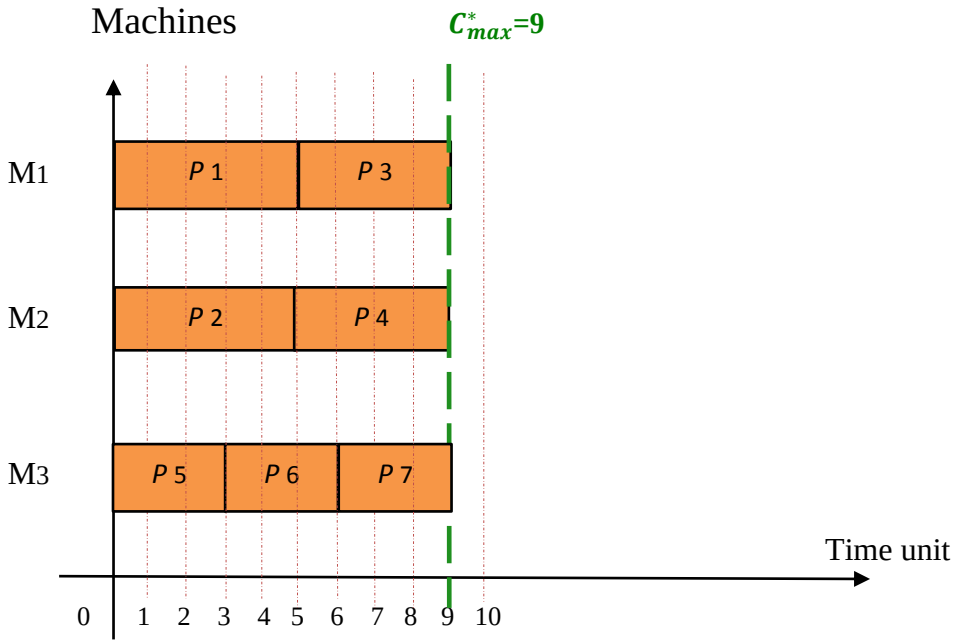


Figure (4-2): The optimal solution (schedule) for Case (1).

Improving the initial solution to obtain the optimal solution (schedule) of the problem shown in figure (4-3).

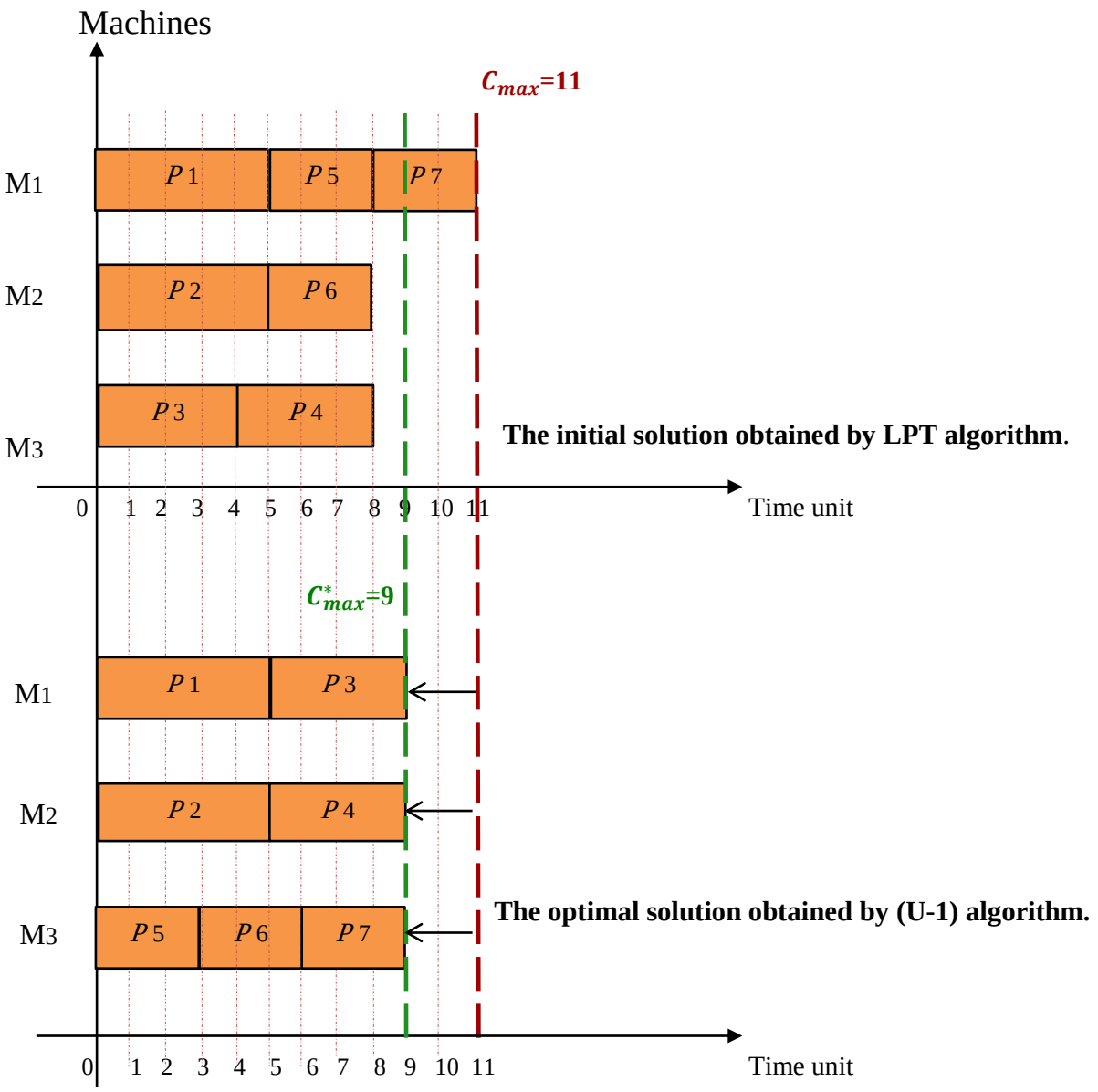


Figure (4-3): Using (U-1) algorithm to Improve the initial solution.

4-2-3 Using (LINGO 15) to solve Case(1):

```
MODEL:
MIN = Cmax;
P1*X11 + P2*X12 + P3*X13 + P4*X14 + P5*X15 + P6*X16 + P7*X17 <= Cmax;
P1*X21 + P2*X22 + P3*X23 + P4*X24 + P5*X25 + P6*X26 + P7*X27 <= Cmax;
P1*X31 + P2*X32 + P3*X33 + P4*X34 + P5*X35 + P6*X36 + P7*X37 <= Cmax;

X11 + X21 + X31 = 1;
X12 + X22 + X32 = 1;
X13 + X23 + X33 = 1;
X14 + X24 + X34 = 1;
X15 + X25 + X35 = 1;
X16 + X26 + X36 = 1;
X17 + X27 + X37 = 1;

@bin (X11);
     @bin (X12);
          @bin (X13);
               @bin (X14);
                    @bin (X15);
                         @bin (X16);
                              @bin (X17);

@bin (X21);
     @bin (X22);
          @bin (X23);
               @bin (X24);
                    @bin (X25);
                         @bin (X26);
                              @bin (X27);

@bin (X31);
     @bin (X32);
          @bin (X13);
               @bin (X34);
                    @bin (X35);
                         @bin (X36);
                              @bin (X37);

P1 = 5;
  P2 = 5;
    P3 = 4;
      P4 = 4;
        P5 = 3;
          P6 = 3;
            P7 = 3;

END
```

The Solution:

```
Global optimal solution found.
  Objective value:                              9.000000
  Objective bound:                              9.000000
  Infeasibilities:                              0.000000
  Extended solver steps:                               0
  Total solver iterations:                            21
```

Variable	Value
CMAX	**9.000000**
P1	5.000000
X11	0.000000
P2	5.000000
X12	0.000000
P3	4.000000
X13	0.000000
P4	4.000000
X14	0.000000
P5	3.000000
X15	1.000000
P6	3.000000
X16	1.000000
P7	3.000000
X17	1.000000
X21	0.000000
X22	1.000000
X23	0.000000
X24	1.000000
X25	0.000000
X26	0.000000
X27	0.000000
X31	1.000000
X32	0.000000
X33	1.000000
X34	0.000000
X35	0.000000
X36	0.000000
X37	0.000000

The solution for Case (1) by LINGO15 software and LPT , U-1 algorithm is shown in table (4-1).

Table (4-1): The solution for case(1) by (LINGO 11 , LPT and U-1).

LINGO 11				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_5 , P_6 , P_7	9	**9**	21
M2	P_2 , P_4	9		
M3	P_1 , P_4	9		
LPT algorithm				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_1 , P_5 , P_7	11	**11**	1
M2	P_2 , P_6	8		
M3	P_3 , P_4	8		
U-1 algorithm				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_5 , P_6 , P_7	9	**9**	4
M2	P_2 , P_4	9		
M3	P_1 , P_4	9		

According to the results in table (4-1) the solution of the problem (C^*_{max}) which obtained by LPT algorithm ($C^*_{max}=11$). And as can be observed the solution by LINGO15 and U-1 program is identical ($C^*_{max}=9$) with different distribution of jobs on machines and number of solution iterations.

4-3 Case(2):

Find the optimal schedule (solution) by using the (LPT) and (U-1) algorithms to minimize maximum completion time (C_{max}) for (12) jobs their processing times (p_j) given bellow:

p_j = [8, 6, 10, 4, 6, 12, 7, 8, 5, 10, 1]

The jobs processed by (3) identical parallel machines, each machine can only process one job at any time and preemption of the job on another machine is not allowed. And comparing the solution with LINGO15 solution.

The solution: the solution for example (2) shown in table (4-2).

Table (4-2): The solution for Case(2) by (LINGO 15 , LPT and U-1).

LINGO 15				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_2 , P_3 , P_4, P_9 , P_{11}	26	**26**	21
M2	P_5 , P_6, P_7	25		
M3	P_1 , P_8, P_{10}	26		
LPT algorithm				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_1 , P_6 , P_9, P_{10}	28	**28**	1
M2	P_2 , P_4 , P_7 , P_{11} ,	25		
M3	P_3 , P_5 , P_8	24		
U-1 algorithm				
Machine	Jobs	C	C^*_{max}	Iteration
M1	P_1 , P_2 , P_{10}	26	**26**	5
M2	P_3 , P_4 , P_5	26		
M3	P_6 , P_7 , P_8, P_9 , P_{11}	25		

According to the results in table (4-2) the solution of the problem (C^*_{max}) which obtained by LPT algorithm ($C^*_{max} = 28$). And as can be observed the solution by LINGO15 and U-1 program is identical ($C^*_{max} = 26$) with different distribution of jobs on machines and number of solution iterations.

4-4 Problems with random number of machines and jobs:

Number of problems with random number of machines and jobs have been generated to make sure and demonstrate effectiveness of the solution algorithm. Table (4-3) represents the results. (for more details and data of problems see appendix 1 and appendix 2 for U-1 program code).

Table (4-3):The solution for number of (m) Machines and (N) Jobs:

No;	Machines	Jobs	C^*_{max}			Iteration	
	(m)	(N)	LINGO 15	LPT	U-1	LINGO 15	U-1
1	2	8	42	42	42	17	2
2	3	11	26	28	26	72	5
3	4	9	12	15	12	88	5
4	5	10	25	25	25	260	2

Table (4-3) represents the results for random number of machines and jobs, the makespan (C^*_{max}) which obtained by LINGO15 and U-1 algorithm for problem 1 and 2 in the table are equal. On the other hand there is a big difference, in the iterations to reach the solution, U-1 algorithm is always faster than LINGO15.

As can be observed in table (4-3) for number of machine more than (5) and number of jobs more than (10), LINGO15 software cannot give results. Because the integer variables for this version are limited by (50 variable). [18].

4-5 Results and Discussion:

In this study the ILP model introduced to represent and formulate the problem($P_m||C_{\max}$), U-1 algorithm designed and coded in MATLAB to solve the ILP model, also LINGO15 software has been used to test and evaluate performance of U-1 algorithm.

Many experiments and problems are used with random number of machines (m) and jobs (N) to demonstrate effectiveness of the algorithm solution.

In this study, the ILP model can optimally solved by ILP solvers for small size of problems. And as can be observed from tables (4-1), (4-2) and (4-3), LINGO15 can solve the problems with small size of number of machines and jobs while U-1 algorithm can solve large size of parallel machines scheduling problem optimally in small number of iteration.

The experiments and results show that the U-1 algorithm dominate the current version of LINGO15.

Many factors impact on the performance of the solution algorithm:

1- The number of jobs (N) to be scheduled is an important factor since it effects directly on the load of the system.
2- The number of machines (m) effect on the distribution of jobs on the machines.
3- The number of different processing times (n) (number of job types).

CHAPTER 5

Conclusions and Recommendations

5-1 conclusions:

In practical life, there are many problems can be modeled as parallel machines scheduling (PMS) problem, (for example):

- Production lines (very common to find more than one machine).
- In ports (ships – docks).
- In hospitals (hospital assistance – patient).

In this study has been used (ILP) model to represent and formulate the problem ($P_m||C_{\max}$), (Makespan minimization for identical parallel machines), LPT and U-1 algorithms coded in MATAB to solve the problem and find the solution.

Also LINGO15 software is used to test and evaluate performance of U-1 algorithm.

The results of study demonstrated that:

1- Mathematical modeling and algorithms methods are powerful tools and more effective for ($P_m||C_{\max}$) problems compared with other methods.
2- Efficiency and performance of the algorithm are evaluated by the number of iterations required to find the optimal solution.
3- The number of feasible solution(s) which generated is equal to number of iteration required to find optimal solution minus one (t -1) and each one of them is better than the previous one.
4- If the number of iteration (t =1) the initial solution obtained by LPT algorithm is optimal.
5- The ILP model solvers such as LINGO15 can solve small and limit number of optimization problems.

5-2 Recommendations:

1- Follow up the studies and modern technology in parallel machines scheduling field.

2- Using the appropriately programming to create computer system for parallel machines scheduling problems.

3- Study the effect of preventive maintenance and machines breakdown on machines scheduling.

4- Applied the study to solve the practical problems.

5- Study other cases of parallel machines problems for example:

- Uniform parallel machines ($Q_m || C_{\max}$) problem.
- Unrelated parallel machines ($R_m || C_{\max}$) problem.
- Dedicated machines (flow shop F_m) or (job shop J_m).

References

References

1. Alharkan Ibrahim M. “Algorithms For Sequencing and Scheduling”, King Saud University, Saudi-Arabia.
2. A . Ravi Ravindran, “Operations Research (Applications)”, Pennsylvania State University, USA, 2009.
3. Reghavendra B. V and Murthy A. N “Workload balancing in identical parallel machines scheduling using genetic algorithm”, paper, ARPN journal of engineering and applied sciences. Vol.6, No.1, 2011.
4. Hashemain Navid, “Makespan Minimization For Parallel Machines Scheduling With Availability constraints”, thesis, Dalhousie University 2010.
5. Koulamas Christos, Kyparisis George.J “A modified LPT algorithm for tow uniform parallel machine makespan minimization problem”, paper, European journal of operational research (61-68), 2009.
6. Lin Chien-Hung and Liao Ching-jong “Minimizing makespan on parallel machines with machine eligibility restrictions”, paper, The Open Operational Research Journal, 2, 18-24, 2008.
7. Sovindik Kaya, “Parallel Machine Scheduling Subject to Machine Availability Constraints”, thesis, Bilkent University, 2006.
8. Gupta Jatinder N. D. & others, “Makespan Minimization On Identical Parallel Machines Subject To Minimum Total Flow-Time”, paper, Journal of the Chinese Institute of Industrial Engineers,Vol.21, No.3,220-229, 2004.
9. Michel L. Pinedo, “Scheduling (Theory, Algorithms and Systems)”, New York University- USA, Fourth Edition, 2011.

10. Peter Bruker, “Scheduling Algorithms”, Germany, Fifth Edition, 2006.
11. Kaabi Jihene and Harrath Youssef “A Survey of parallel machine scheduling under availability constraints”, paper, International Journal of Computer and Information Technology, Vol 3, issue 02, 2014.
12. Chaudhry Imran & others “Minimizing makespan for machine scheduling and worker assignment problem in identical parallel machine models using GA”, paper, Proceeding of the World congress on Engineering. Vol III, London, UK , 2010.
13. Hamdy A. Taha, “Operations Research an introduction”, ninth edition,2011.
14. Wayne L. Winston, “Operations Research (Application and Algorithms)”, Indiana University- USA, Fourth Edition, 2004.
15. John W. chinneck, “introduction to liner programming”, 2001.
16. Michel L. Pinedo, “Planning and scheduling in manufacturing and services”, New York University- USA, 2005.
17. MATLAP . The Math Works (the language of technical computing) R2012a.
18. LINGO15 software , system Chicago – USA, 2015.

Appendices

Appendix 1

Problems data for table (4-3) and solution procedure:

Input data (2 machines and 8 jobs):

Machines (m)	2							
Jobs (N)	8							
(Pj)	P1	P2	P3	P4	P5	P6	P7	P8
Processing time	16	12	7	11	18	10	4	5

```
MODEL:
MIN = Cmax;
P1*X11 + P2*X12 + P3*X13 + P4*X14 + P5*X15 + P6*X16 + P7*X17 +
P8*X18 <= Cmax;
P1*X21 + P2*X22 + P3*X23 + P4*X24 + P5*X25 + P6*X26 + P7*X27 +
P8*X28 <= Cmax;
X11 + X21 = 1;
X12 + X22 = 1;
X13 + X23 = 1;
X14 + X24 = 1;
X15 + X25 = 1;
X16 + X26 = 1;
X17 + X27 = 1;
X18 + X28 = 1;
```

Output:

U-1 program									
C^*_{max}	42								
iteration	2								
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18
	Value	1	1	0	0	0	1	0	0
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28
	Value	0	0	1	1	1	0	1	1

LINGO 15									
C^*_{max}	42								
iteration	17								
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18
	Value	1	0	1	0	1	0	0	0
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28
	Value	0	1	0	1	0	1	1	1

Input data (3 machines and 11 jobs):

Machines (m)	3										
Jobs (N)	11										
(Pj)	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
Processing time	8	6	10	4	6	12	7	8	5	10	1

```
MODEL:
MIN = Cmax;
P1*X11 + P2*X12 + P3*X13 + P4*X14 + P5*X15 + P6*X16 + P7*X17 + P8*X18 +
P9*X19 + P10*X110 + P11*X111 <= Cmax;
P1*X21 + P2*X22 + P3*X23 + P4*X24 + P5*X25 + P6*X26 + P7*X27 + P8*X28 +
P9*X29 + P10*X210 + P11*X211 <= Cmax;
P1*X31 + P2*X32 + P3*X33 + P4*X34 + P5*X35 + P6*X36 + P7*X37 + P8*X38 +
P9*X39 + P10*X310 + P11*X311 <= Cmax;
X11 + X21 + X31 = 1;
X12 + X22 + X32 = 1;
X13 + X23 + X33 = 1;
X14 + X24 + X34 = 1;
X15 + X25 + X35 = 1;
X16 + X26 + X36 = 1;
X17 + X27 + X37 = 1;
X18 + X28 + X38 = 1;
X19 + X29 + X39 = 1;
X110 + X210 + X310 = 1;
X111 + X211 + X311 = 1;
```

Output:

U-1 program												
C^*_{max}	26											
iteration	5											
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18	X19	X110	X111
	Value	1	1	0	0	0	0	0	0	0	1	0
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28	X29	X210	X211
	Value	0	0	1	1	1	0	0	0	0	0	0
M3	Xij	X31	X32	X33	X34	X35	X36	X37	X38	X39	X310	X311
	Value	0	0	0	0	0	1	1	1	1	0	1

LINGO 15												
C^*_{max}	26											
iteration	72											
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18	X19	X110	X111
	Value	0	1	1	1	0	0	0	0	1	0	1
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28	X29	X210	X211
	Value	0	0	0	0	1	1	1	0	0	0	0
M3	Xij	X31	X32	X33	X34	X35	X36	X37	X38	X39	X310	X311
	Value	1	0	0	0	0	0	0	1	0	1	0

Input data (5 machines and 10 jobs):

Machines (m)	5									
Jobs (N)	10									
(Pj)	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Processing time	19	22	3	7	16	25	2	11	5	8

```
MODEL:
MIN = Cmax;
P1*X11 + P2*X12 + P3*X13 + P4*X14 + P5*X15 + P6*X16 + P7*X17 + P8*X18 +
P9*X19 + P10*X110 <= Cmax;
P1*X21 + P2*X22 + P3*X23 + P4*X24 + P5*X25 + P6*X26 + P7*X27 + P8*X28 +
P9*X29 + P10*X210 <= Cmax;
P1*X31 + P2*X32 + P3*X33 + P4*X34 + P5*X35 + P6*X36 + P7*X37 + P8*X38 +
P9*X39 + P10*X310 <= Cmax;
P1*X41 + P2*X42 + P3*X43 + P4*X44 + P5*X45 + P6*X46 + P7*X47 + P8*X48 +
P9*X49 + P10*X410 <= Cmax;
P1*X51 + P2*X52 + P3*X53 + P4*X54 + P5*X55 + P6*X56 + P7*X57 + P8*X58 +
P9*X59 + P10*X510 <= Cmax;

X11 + X21 + X31 + X41 + X51 = 1;
X12 + X22 + X32 + X42 + X52 = 1;
X13 + X23 + X33 + X43 + X53 = 1;
X14 + X24 + X34 + X44 + X54 = 1;
X15 + X25 + X35 + X45 + X55 = 1;
X16 + X26 + X36 + X46 + X56 = 1;
X17 + X27 + X37 + X47 + X57 = 1;
X18 + X28 + X38 + X48 + X58 = 1;
X19 + X29 + X39 + X49 + X59 = 1;
X110 + X210 + X310 + X410 + X510 = 1;
```

Output: U-1

U-1 program											
C^*_{max}	25										
iteration	2										
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18	X19	X110
	Value	1	0	0	0	0	0	0	0	0	0
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28	X29	X210
	Value	0	1	0	0	0	0	0	0	1	0
M3	Xij	X31	X32	X33	X34	X35	X36	X37	X38	X39	X310
	Value	0	0	1	0	0	0	0	1	0	0
M4	Xij	X41	X42	X43	X44	X45	X46	X47	X48	X49	X410
	Value	0	0	0	1	0	1	0	0	0	0
M5	Xij	X51	X52	X53	X54	X55	X56	X57	X58	X59	X510
	Value	0	0	0	0	1	0	1	0	0	1

LINGO 15											
C^*_{max}	25										
iteration	260										
M1	Xij	X11	X12	X13	X14	X15	X16	X17	X18	X19	X110
	Value	0	0	1	1	0	0	1	1	0	0
M2	Xij	X21	X22	X23	X24	X25	X26	X27	X28	X29	X210
	Value	0	0	0	0	0	1	0	0	0	0
M3	Xij	X31	X32	X33	X34	X35	X36	X37	X38	X39	X310
	Value	0	0	0	0	1	0	0	0	0	1
M4	Xij	X41	X42	X43	X44	X45	X46	X47	X48	X49	X410
	Value	0	1	0	0	0	0	0	0	0	0
M5	Xij	X51	X52	X53	X54	X55	X56	X57	X58	X59	X510
	Value	1	0	0	0	0	0	0	0	1	0

Appendix 2

U-1 Program code

```
%============================== U-1 Program ============================
%======== Makespan Minimization for Identical Parallel Machines =======
% Author: Yousef German.
%      Date: 2015.
%=======================================================================
clear all
close all
clc
NoMach = 5;% input(' No of Machines = ');
  Pj = [19 22 3 7 16 25 2 11 5 8  ];
  % ====================== LPT ======================
Sort_Pj = DownFunction(Pj);
NoJobs = length(Pj);
Machines = zeros(NoMach,length(Pj)- NoMach +1);
for i = 1:NoMach
       Machines(i,1) = Sort_Pj(i);
end

for j = 2:length(Pj)- NoMach +1
      A = sum(Machines')';
      [Row Col] = find(A == min(A));
      Machines(Row(1),j) = Sort_Pj(NoMach+j-1);
end
Machines;
sum(Machines')';
% ================ End LPT ======================
%==========================================
UB = max(sum(Machines')');
LB = sum(Pj)-(NoMach-1)*UB;
I = length(Pj);II = I+1;J = 1;a = 0;
while I ~= II
    a = a + 1;
R = find(Sort_Pj == Sort_Pj(J));
P(a)  = Sort_Pj(R(1));
aa(a) = length(R) ;
II    = sum(aa);
J     = II+1;
end
n  = length(aa);
r  = aa;
ii = 0;Cc =[];Mm = [];
[C kk ii] = SetUB(NoMach,UB,LB,P,n,r,aa,ii)

% ---------- LB <= min(C) & UB >= max(C)----------

Cc(1,:) = C; Mm = kk;
Iter = 1;
while LB <= min(C) & UB >= max(C)
    Iter = Iter + 1;
%    if ii == NoMach
       UB = UB - 1;
       LB = sum(Pj)-(NoMach-1)*UB;
```

```
       [C kk ] = NSetUB(NoMach,UB,LB,P,n,r,aa,ii,C);
      Cc(Iter,:)=[]
Mm((Iter-1)*NoMach+1:Iter*NoMach,:)=[];
P
K1_Kn = Mm((Iter-2)*NoMach+1:(Iter-1)*NoMach,:)
% xlswrite('Yousef.xlsx',K1_Kn);
% xlswrite('Yousef.xlsx',P','f7:f9');
end
```