جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعة ديالى

كلية العلوم

قسم علوم الرياضيات

خوارزميات حل لمسألة امثلية متعددة المقاييس

رسالة مقدمة إلى مجلس كلية العلوم– جامعة ديالى

وهي جزء من متطلبات نيل شهادة الماجستير في علوم الرياضيات

من قبل الطالب

انمار صبري حسن

بكلوريوس علوم الرياضيات / كلية العلوم/ الجامعة المستنصيرية 2004

إشراف

ا.م.د عدوية علي محمود النعيمي

1442هـ **2021 م**

Chapter One

Scheduling Problem

1.1 Scheduling Problem (Definition and Classification)

1.1.1 Machine Scheduling

There has been a large number of researches on production scheduling problems since the original of mathematical formulations typically, this entails assigning machines to process tasks (jobs) over time in order to refine certain output parameters, either precisely or roughly.

The literature can be divided into two major categories:

a- Deterministic scheduling research: where all problem parameters are considered to be well-known.
b- Stochastic scheduling research: where at least some parameters are random variables.

In deterministic scheduling research a large view is taken and multiple machines are often modeled. The deterministic approach is to plan the work through the machines over a period of time in the best possible way, given a specific objective to optimize. The implicit assumption here is often that a schedule can be executed directly as developed.

However, several scholars have recently recognized that this unlikely scenario exists in many manufacturing settings, and have attempted to apply the deterministic approach to circumstances involving some complexity [14].

1.1.2 Basic Scheduling Concepts

We begin by adding some key notations, focusing on performance parameters without going into details about system environments, etc. We assume that there are n jobs, denoted by 1,...,n, that must be scheduled on a set of machines that are always available from time zero onwards and can only handle one job at a time.

We only state the notation used in this study for the single machine, jobs j (j=1,...,n):

p_j: This means the processing time.

r_j: A release date of job j, i.e. the earlier time of which the p_j of job begin.

d_j: This means the due date.

W_j: This means the weighted.

We can now compute for job j for a given sequence of jobs:

1- The completion time C_j.
2- The flow time $F_j=C_j-r_j$.
3- The lateness $L_j= C_j-d_j$.
4- The tardiness $T_j=\max\{ C_j -d_j,0\}$.
5- The earliness $E_j=\max\{ d_j-C_j,0\}$.
6- The unit penalty $U_j=1$ if $C_j> d_j$ and $U_j = 0$ if $C_j\leq d_j$.

The following performance criteria appear frequently in the literature [15].

For a given scheduling δ we compute:

1- $C_{max}(\delta) = \max_j (C_j)$ (maximum completion time).
2- $E_{max}(\delta) = \max_j (E_j)$ (maximum earliness).
3- $L_{max}(\delta) = \max_j (L_j)$ (maximum lateness).
4- $T_{max}(\delta) =\max_j (T_j)$ (maximum tardiness).
5- $\sum (W_j) C_j (\delta)$ = total (weighted) completion time.
6- $\sum (W_j) E_j (\delta)$ = total (weighted) earliness.
7- $\sum (W_j) T_j (\delta)$ = total (weighted) tardiness.
8- $\sum (W_j) U_j (\delta)$ = total (weighted) number of tardy jobs.

All these criteria except for E_{max} and $(\sum W_j) E_j$ are regular, i.e., the value of the objective function can not be decreased by inserting idle time into the schedule.

1.1.3 The problem classification

A notation which is commonly used to formulate scheduling problem is based on three fields: **α/β/γ** [7]. In this notation, α describes the machine environment, i.e. the structure of the,

-Single machine or multiple machines.

- Machines that are the same or that are different.

The field β explains the problem's constraints as well as other processing conditions. Among the constraints that can exist [14], [16]:

-Preemption allowed or not, i.e. whether the processing of jobs can be

Interrupted and resumed.

-Whether special processing conditions (release date, due date, setup times,

etc,) specified or not, and if these are not and if these are deterministic or

stochastic.

-Fixed or dynamic arrival of jobs, etc.

The criteria (**γ**) used to evaluate the equality of the scheduling include:

-Minimum completion time or make span C_{max}.

-Maximum earliness $E_{max} = \max (E_j)$ for j=1,…,n.

-Maximum tardiness $T_{max} = \max (T_j)$ for j = 1,…,n, etc.

1.1.4 A few examples of scheduling problems

We give a few examples on three fields classification of scheduling problems. The $1/r_i/\sum W_i\ C_i$ is the problem of minimizing total weighted completion time on one machine subject to non- trivial release dates. The $1//\sum W_i\ C_i+T_{max}$ is the problem of determining the best order for jobs to be processed on a single machine in order to minimize the amount of the overall weighted completion period and the maximum tardiness.

1.1.5 Some Types of Machine Scheduling Problems

Due to the large variety of machine scheduling problems, several classification schemes have been proposed based on different dimensions [17]. The number of available machines and how they are arranged (see figure (1-1)) is example of such dimension. The simplest problem is the one-machine sequencing problem: all jobs must be processed on a single machine, and no two jobs can be processed at the same time. There are four types of scheduling issues in a work store [5]:

a-Scheduling a single machine [5]

The single machine scheduling problem entails allocating a single resource to a collection of jobs. This is achieved by creating a series that involves each job and assigning the jobs to their respective sources. Each job should have a priority, a ready time, a processing time, and a due date assigned to it. On the basis of this data and the job series, the value of the performance measure can be computed. This problem grows in complexity at an exponential rate as the number of jobs to be scheduling increases.

b-Flow Shop Scheduling [5]

In each of the m machines, there are n jobs to process, i.e., each job consists of m steps or operations. Each job is processed in the same order through the processing stages, i.e., from the first to the last machine. The problem is to find the sequence in which the jobs should be processed so that the given objectives are achieved.

c-Job Shop Scheduling [5]

This is a general case of the flow shop scheduling problem, in which each job is not necessarily sequenced through the machines in the same way. As in a flow shop, there are n jobs with m operations each of which are predefined and fixed; for example, $J_m/d_j/C_{max}$ denotes a work shop configuration in which all jobs have a due date and the aim is to minimize the maximum completion time.

d-Scheduling of Open Shops [5]

The open shop is a more general case of the job shop scheduling problem as before, there are n jobs consisting of m steps to be processed in m machines. Each work is sequenced differently across the machine, and determining the best sequences for each of the n jobs is part of the problem. And, in addition to the work processing schedule, the sequence of steps for each job must be decided.

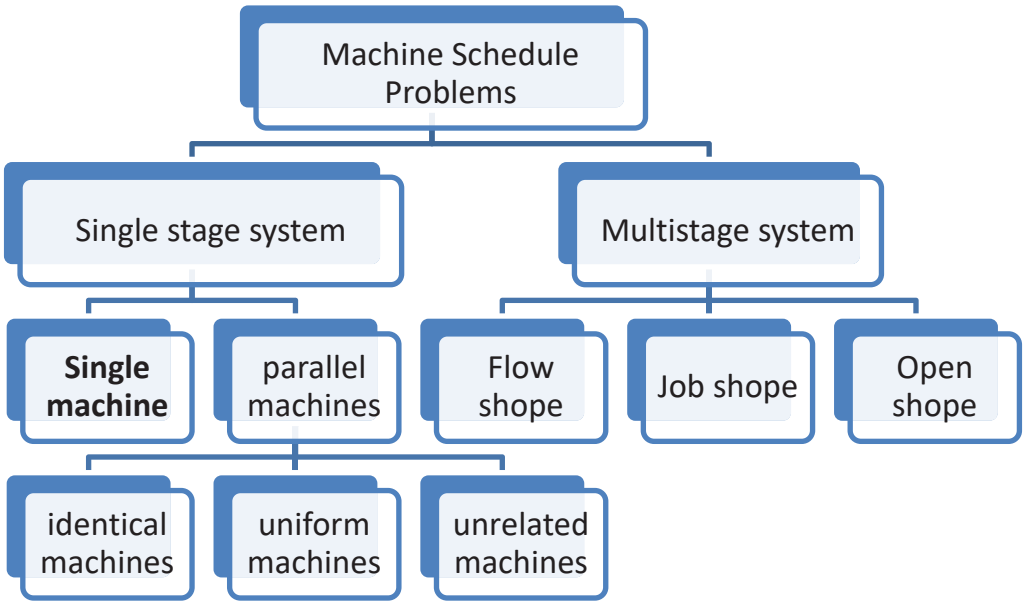

Figure 1-1: Classification of machine Schedule Problems [17]

Definition: P- type problem (1.1.6)

The problems for which the polynomial bounded algorithm for solving this problem, were found.

Definition: NP- type problem (1.1.7)

The problem that haven't a polynomial bounded algorithm to solve this problem.

1.2 Solution Approaches

1.2.1 Basic Rules and Main Results to Find Optimal Solution for p-Type Machine Scheduling Problem

The common basic rules which help us in finding a solution for scheduling problem:

1- Smith rule or **SPT** (shortest processing time) rule, that is, sequencing the jobs are listed in non-descending order of processing time. This rule solves the problem $1//\sum_j C_j$ [18] more general is the **SWPT** rule, that is sequencing the

jobs in non-decreasing order of their processing time to the weight ratio which solves the problem $1//W_jC_j$.

2- The earliest due date or the **EDD** rule, which solves the problem $1//L_{max}$ by sequencing the jobs in non-decreasing order of their due dates. For the $1//T_{max}$ query, this rule also minimizes T_{max} [19].

3- The longest processing time or the **LPT** rule, i.e., sequencing the jobs in a non-increasing order of processing time [20], which minimize $\sum_j Ej$ for the problem $1/C_j \leq d_j/ \sum_j E_j$.

4- The minimum slack time or **MST** rule, that is, sequencing the jobs in non-decreasing order of their slack times ($s_j=d_j-p_j$). In a single machine environment with ready time set at zero, which solves $1//E_{max}$ problem [6].

1.2.2 Mathematical Programming to Solve NP-hard Machine Scheduling Problems

There are some mathematical programming techniques used to solve the combinatorial optimization problem, these techniques are also used for scheduling problems [21].

Many scheduling problems can be formulated as a (mixed) programming problem; in that case, standard integer programming solution procedure can be used. Both Conway et al. [22], and Baker and Schrage [23] discussed integer-programming formulation of scheduling problems.

Complete enumeration method generates schedules one by one, searching for an optimal solution. This method lists all possible schedules and then eliminates the non-optimal schedules from the list, leaving those, which are optimal. Clearly searching for an optimal schedule among all possible schedules using complete enumeration is not suitable even for problems of small size.

The dynamic programming (DP) method is an implicit enumeration technique that can be used to solve any optimization problem that can be solved by solving the derived recurrence relation for this problem [21].There are some difficulties for this method, one of them is the difficulty of finding a good way for brake down problem into stages so that a convenient computation is rather large, which means that the computation grows to exponential rate with increasing in the size of problem.

Branch And Bound (BAB) method is a general method for solving many types of combinatorial optimization problem. BAB method is the most wildly solution technique that is used in scheduling [24]. This method is the typical example of the implicit enumeration approach, which can find an optimal solution by systematically examining subset of feasible solution. The procedure is usually described by means of search tree with nodes that correspond to these subset. From each node for a partially complete solution there grows a number of new branches which replaces the original one by set of new smaller problems that are mutually exclusive. There are two forms of branching that are commonly used:

1- The forward branching, that is the jobs are sequenced one by one from the beginning.
2- The backward branching, that is, the jobs are sequenced one by one from the end.

To minimize an objective function Z, for a particular scheduling problem, the BAB method successively partitions the problem into subsets by using a branching procedure and computes bound by using a lower bounding procedure. These methods are used to exclude subsets that are found to be devoid of any optimal solution. This eventually leads to at least one optimal solution. The lower bound (LB) on the solution to each created sub problem is calculated using the bounding method. For each node we calculate a (LB) which is the cost of the

scheduling jobs (depending on the objective function and the cost of the unscheduled jobs (depending on the derived lower bound)). If this node has a value (LB) greater than or equal to the upper bound (UB) then this node is can called the upper bound is usually defined as the minimum of the values of all feasible solutions currently found. If branching reaches a full sequence of jobs, that sequence is evaluated, and if its value is less than the current upper bound (UB), this (UB) is reset to that value. We repeat the procedure until all nodes have been considered, that is, LB≥UB for all nodes in the search tree. An optimal solution for this problem is a feasible solution with this (LB).

In BAB procedure one can introduce dominance rules (if possible) to specify whether a node can be eliminated before computing its (LB) which reduce the computation time by ignoring the calculations of the dominated nodes and their successors.

1.2.3 The Heuristic Method

It's evident (from the previous section) that utilizing mathematical programming algorithms to tackle a particular problem could take a lot longer than it usually does for large problems. Indeed, even for a minor issue, there is no guarantee that a solution will be found immediately. Sometimes we use a heuristic scheduling instead of the optimal schedule, that is, we can find near optimal solution. Reeves [25] the heuristic technique is defined as follows: A heuristic which seek good (i.e., near optimal) solution at a reasonable computational cost without being able to guarantee either feasibility or optimality, or even, in many case to state how close to optimality a particular feasible solution.

1.3 Multicriteria Scheduling

Many researchers have been working on multiple criteria scheduling with the majority of work being on bi-criteria scheduling. When two criteria are used instead of one, the problem becomes more realistic. One criterion can be chosen to represent the manufacturer's concern while the other could represent consumer's concern. Several studies evaluate the literature on multiple criteria scheduling. Nager et al.(1995) [9], and Tkindt and Billaut (1999) [26] review a special version of the problem, Lee and Vairaktarakis (1993) [27] review a particular variant of the problem in which one criterion is fixed to its greatest feasible value and the other criterion is attempted to be optimized under this restriction.

Hoogeveen (1992) [7] studied a number of bi-criteria scheduling problems. Most real-world optimization problems have several, often conflicting objectives. Therefore, the optimum for a multiobjective problem is typically not a single solution. It is a set of solutions that trade-off between objectives.

There are three different sorts of problems with many criteria that can be detected:

The first of these problems entails determining all sequences that minimize a first objective. The optimal sequence for that task is picked from among those that minimize a second objective. Assume we've decided on two performance criteria to consider, say f and g. If $\boldsymbol{f}$ is more important than $\boldsymbol{g}$, then the approach is to find the optimum value with respect to criterion $\boldsymbol{f}$, say $\boldsymbol{f}^*$, and choose from among the set of optimum schedules for $\boldsymbol{f}$ the one that performs best on $\boldsymbol{g}$, such an **approach** is called **hierarchical** optimization or **lexicographical** optimization, which is denoted by Lex ($\boldsymbol{f}$,$\boldsymbol{g}$) in the third field of the $\boldsymbol{\alpha/\beta/\gamma}$ notation scheme. This method is known as a hierarchical approach.

When the criteria are weighted differently, ***the second*** of these multiple criteria problems is an objective function that can be expressed as the sum of weighted functions and turns the difficulties into a single criterion scheduling problem. Simultaneous optimization is the name given to this method, as well as the ***third type*** of multiple criteria issues. Also in these multiple criteria problems, both criteria are going to be considered as equally important. The issue is determining a sequence that achieves both goals. To solve this problem, the main concept is that we select a subset of solutions from a larger set that contains efficient solutions.

Theorem (1.3.1) [8]

There is an extreme schedule that minimizes F (f,g) if the composite objective function F (f,g) is linear.

المستخلص

في هذه الرسالة ، يتم النظر في مسألة الجدولة متعددة المقاييس على ماكنة واحده. المقاييس الثلاثة المستخدمة هي وقت الاتمام الكلي $\sum C_j$ والتأخير الكلي $\sum T_j$ والحد الاقصى للوقت المبكر E_{max}.

نهدف في هذه الدراسة الى ايجاد الجدول الزمني الامثل والتقريبي للوظائف $j=1,2,\ldots,n$ ، j لتقليل دالة الهدف متعدد المقاييس للدالة $\sum C_j + \sum T_j + E_{max}$.

لحل هذه المسالة نستخدم خوارزمية التقيد والتفرع (BAB) لايجاد الحل الامثل للمسالة. وقمنا باشتقاق القيد الادنى (LB) لاستخدامه في خوارزمية التفرع والتقيد (BAB) . يتم اجراء التجارب الحسابية لخوارزمية BAB على مجموعة كبيرة من مشكلات الاختبار. هنا تظهر صعوبة NP لهذه المسألة اي انه ليس من الممكن دائما العثور على الحل الامثل بسرعة والمسألة حلت الى $n=13$.

لذلك بدلا من البحث عن الحل الامثل بجهد حسابي هائل نستخدم خوارزميات البحث المحلية لايجاد حلول قريبة من الحل الامثل مع وقت حسابي اقل. تم تطبيق ثلاث خوارزميات بحث محلية ، طريقة النسب (DM) والتلدين المحاكي (SA) والخوارزمية الوراثية (GA) لهذه المسالة.

تتم مقارنة الخوارزميات DM و SA و GA مع خوارزمية BAB من اجل تقيم فعالية طرق الحل. تمت صياغة الاستنتاجات على كفاءة الخوارزميات، بناءً على نتائج التجارب الحسابية.

.