One Machine Scheduling Problem With Release dates and Tow Criteria

Al-Zuwaini Mohammad Kadhim
Math. Dep. College of Computer Science and Mathematics, Thi-qar University

Mohanned M. Kadum
Comp. Dep. College of Computer Science and Mathematics, Thi-qar University

مسألة جدولة الماكنه الواحده بوجود أوقات تحضير

محمد كاظم الزويني
قسم الرياصيات
كلية علوم الحاسبات والرياضيات
جامعة ذي قار

مهند محمد كاظم
قسم الحاسبات
كلية علوم الحاسبات والرياضيات
جامعة ذي قار

المستخلص:

يقدم هذا البحث خوارزمية التفرع والتقيد لترتيب مجموعه من النتاجات على الماكنة الواحده، الهدف تصغير الكلفة الكلية لزمن انسياب النتاجات وعدد النتاجات المتأخره عندما يكون للنتاجات ازمنه تحضير غير متساوية.

تضمن البحث قيد ادنى وحلول كفوءه لبعض الحالات الخاصة وقواعد هيمنه للحد من تفرعات شجرة البحث في طريقة التفرع والتقيد. وقد وجد الحل الأمثل لغاية ٤٠ نتاج.

Abstract.

This paper presents a branch and bound algorithm for sequencing a set of jobs on a single machine scheduling with the objective of minimizing total cost of flow time and number of tardy jobs, when the jobs may have unequal ready times. Lower bound; efficient solutions, dominance rules for this problem and a computational experience will also be included.

Computational experience with instances having up to 40 jobs shows that the lower bound is effective in restricting the search.

Key words. Flow time, tardy jobs, scheduling, release date

1. Introduction:

The problem of sequencing n jobs on one machine under different assumptions and multiple criteria are considered extensively. The objective function to be minimized consists of two criteria with unequal ready times: sum of flow time denoted by $\sum F_i$ plus total number of late jobs denote by $\sum u_i$. We assume that the two criteria have the same importance. Denote this problem by $1/r_i/\sum(F_i+u_i)$.

Problem $1/r_i=0/\sum F_i$ is well known and can polynomially solved (Smith 1956). For $1/r_i/\sum F_i$ problem was shown NP-complete by Lenstra et al. (1977)[8]. Mason and Anderson (1991)[10] examine the static sequencing problem of ordering the processing of jobs on a single machine so as to minimized the average weighted flow time. It was assumed that all job had zero ready times, and that the jobs are grouped into classes. Kellerer et al. [6] provide an approximation algorithm for the corresponding $1/r_i/\sum F_i$ problem, with a ratio guarantee of $O(\sqrt{n})$. Zghair (2000) [15] used un efficient branch and bound technique with a suitable lower bound and proved some dominance rules for $1/r_i/\sum F_i$ problem

For $1/r_i/\sum u_i$ problem has been shown NP-hard [2]. Several special cases yield polynomial algorithm. The $1//\sum u_i$ problem can be solved in O(n log n) steps by using Moor's algorithm [11]. Peter and Mikhail (1996) [12] studied the single machine scheduling problem to minimize the weighted number of late jobs. Shao and Milan (2003) [13] considered a single

machine scheduling to minimize the number of late jobs under uncertainty. They proposed a rather general model based on an algebraic approach. Greogorio et al. (2004) [5] used a scheduling techniques to minimize the number of late jobs in workflow systems by organizations to control and improve business processes. Their work present some of the problems of using scheduling results in ordering cases in a workflow and tackles two of them: the uncertainties on the cases, processing times and routing. Marjan and Han (2008) [9], present a new model to deal with the stochastic completion times, which is based on using a chance constraint to define whether a job is on time or late. They have studied minimizing the number of late jobs problem for four classes of stochastic processing times. The $1/r_i/\sum u_i$ **problem can be solved if there are agreeable due dates (i.e. there is a renumbering of the job so that** $r_i \le r_{i+1}$ **and** $d_i \le d_{i+1}$ $(i = 1, 2, \ldots, n-1)$ **in O(n^2) steps by Kise et al. [7].**

For the composite criteria, Emmons (1975) [4] considered a multiple criteria, in which the primary criterion is to minimize the number of tardy jobs while the secondary criterion is to minimize the sum of completion times. The computational experiments he carried out indicated that the additional computational effort to continue to optimality to be remarkably little. Van Wassenhove and Luda (1980) [14] consider the problem of single machine schedule to minimized the flow jobs and maximum tardiness. Abdul-Razaq and zghair [1] considered the total cost of completion time and the number of tardy jobs. They assumed that all jobs available for processing at time zero (i.e. r_i=0 for each job), and they solved the problem with up to 20 jobs.

In this paper we extend the work in [1] and [4]. The aim in this study is to minimize the total cost of flow time and number of tardy jobs with unequal release dates, that is we assume that the jobs are available for processing with difference times(i. e. $r_i \neq 0$**). Then our problem is strongly NP-hard, since the** $1/r_i/\sum(C_i + u_i)$ **problem NP-hard [1, 4].**

2. Problem Formulation

To state our scheduling problem more precisely, we are given a set N of jobs, N= {1, 2, . . ., n} is to be processed one job at the time, on a single machine. For each job i, the processing time p_i , the due date d_i and the ready time r_i are given. For a given processing order of jobs the

completion time C_i , for job i, ($i \in N$) be computed. The objective is to find a processing order of the jobs that minimizes the sum of the total cost of flow time $\sum F_i$ and the number of tardy jobs $\sum u_i$, with release dates. This is denoted by $1/r_i/\sum(F_i + u_i)$.

Let σ : the set of permutation schedules ($|\sigma| = n!$).

δ : a permutation schedule, ($\delta \in \sigma$)

F_i : the flow time of job i, ($i \in \delta$).

$C(\delta)$: the total completion times of schedule δ , ($C(\delta) = \sum_{i\in\delta} c_i$)

$F(\delta)$): the total flow times of schedule δ , ($F(\delta) = \sum_{i\in\delta} F_i$)

$U(\delta)$: No. of tardy jobs of schedule δ , ($U(\delta) = \sum_{i\in\delta} u_i$).

Where $C_1 = r_1 + p_1$, $C_i = \max\{C_{i-1}, r_i\} + p_i$, $i = 2,..., n$

$$F_i = C_i - r_i$$

$$u_i = \begin{cases} 1 & \text{if } C_i > d \\ 0 & o.w. \end{cases}$$

Then the objective is to find a schedule δ of the jobs ($\delta \in \sigma$) that minimize total cost $Z(\delta)$, where:

$$Z(\delta) = F(\delta) + U(\delta)$$

The mathematical form of our problem can be formally stated as:

$$Min\{Z(\sigma)\}$$

Subject to :

$$\left.\begin{array}{l} C_i \geq p_i \\ p_i, r_i \text{ and } d_i \geq 0 \end{array}\right\} \quad \ldots(p)$$

For define such a sequence more precisely.

A sequence $\delta^* \in \sigma$ is optimal in problem (p) if there no sequence $\delta \in \sigma$ such that:

$$Z(\delta) < Z(\delta^*)$$

Similarly, we say that a sequence δ_1 dominates a sequence δ_2 when, $Z(\delta 1) < Z(\delta 2)$.

3. Dominance Theorems.

If it can be shown that an optimal solution can always be generated without branching from a particular node of the search tree, then that node is dominated and can be eliminated. Dominance rules usually specify whether a node can be eliminated before its lower bound is calculated. Clearly, dominance rules are particularly useful when a node can be eliminated which has a lower bound that is less than the optimum solution [15].

Theorem (3-1)

For problem (p). If $r_i \le r_j$, $p_i \le \mathrm{p_j}$ and $\mathrm{d_i} = d_j$ **then** $i \prec j$ **in optimal solution.**

Proof:

Let π **be a sequence of jobs in which the job i preceds job j,** $r_i \le r_j$, $p_i \le \mathrm{p_j}$ **and** $d_i = \mathrm{d_j}$**. Let T be a completion time of job (**$i-1$**) in** π**. Let** π' **be a sequence has the same jobs order of** π **except the job j precedind job i. There are three cases:**

Case 1: $\mathbf{r_i \le r_j \le T}$

Let a=$\mathbf{r_j - r_i}$ **and b=T-**$\mathbf{r_j}$

$\mathbf{(F_i + F_j)}_{i,\, j\in\pi} = \mathbf{a+2b+2\,p_i + p_j}$, $\mathbf{(F_i + F_j)}_{i,\, j\in\pi'} = \mathbf{a+2b+2\,p_j + p_i}$

$\mathbf{(F_i + F_j)}_{i,\, j\in\pi} - \mathbf{(F_i + F_j)}_{i,\, j\in\pi'} = \mathbf{p_i - p_j < 0}$ **….(1)**

if job i$\in\pi$ **is late, then job j** $\in\pi$ **is late and i, j** $\in\pi'$ **are late, then (1) is hold. If job i**$\in\pi$ **is early and j**$\in\pi$ **is late, that is,** $u_j = 1$**. If jobs** $i, \mathrm{j} \in \pi$ **are early, then** $j, i \in \pi'$ **are early also. So (1) is satisfied.**

Case 2: $r_i \le \mathrm{T} \le \mathrm{r_j}$

(i) if $T + p_i \ge \mathrm{r_j}$**, then**

$(F_i + \mathrm{F_j})_{\mathrm{i,j}\in\pi} = a + p_i + \mathrm{p_j} + \mathrm{c}$, $(F_i + \mathrm{F_j})_{i,\, j\in\pi'} = \mathrm{a + b + p_i} + 2p_j$

Where $a = T - \mathrm{r_i}$, $\mathrm{b = r_j - T}$ and $\mathrm{c = T + p_i} - r_j$

$(F_i + \mathrm{F_j})_{\mathrm{i,j}\in\pi} - (F_i + \mathrm{F_j})_{i,\, j\in\pi'} = c - (p_i + b) < 0$ **… (2)**

If job $\mathrm{i} \in \pi$ **is late then, job** $j \in \pi$ **is late and** $i, \mathrm{j} \in \pi'$ **are late also. In** π **if i is early and j is late, then in** π' **job j is either early or late and job i is late and (2) is hold. If** $i, \mathrm{j} \in \pi$ **are early, then** $j \in \pi'$ **is early,** $i \in \pi'$ **is either early or late and (2) is hold.**

(ii) **if** $r_j \geq T + p_i$ **, then**

$(F_i + \mathrm{F_j})_{i,j\in\pi}$ - $(F_i + \mathrm{F_j})_{i,j\in\pi'} = -(p_j + b)$ **, where** $b = r_j - (T + p_i)$ **.**

In π **if job i is late then job j is late and** $i, \mathrm{j} \in \pi'$ **are late. If job** $i \in \pi$ **is early and job** $j \in \pi$ **is late, then jobs** $i, \mathrm{j} \in \pi'$ **are late. If** $i, \mathrm{j} \in \pi$ **are early, then** $\mathrm{j} \in \pi'$ **is early and** $\mathrm{i} \in \pi'$ **either early or late.**

Case 3

$T \leq r_i \leq r_j$ **,**

(i) **if** $b \geq p_i$, $\mathrm{b} = \mathrm{r_j} - r_i$ **, then**

$(F_i + \mathrm{F_j})_{i,j\in\pi}$ - $(F_i + \mathrm{F_j})_{i,j\in\pi'} = -(p_j + b) < 0$

(i) **if b <** p_i **, then**

$(F_i + \mathrm{F_j})_{i,j\in\pi}$ - $(F_i + \mathrm{F_j})_{i,j\in\pi'} = p_i - (p_j + 2b) < 0$

Since $p_i, \mathrm{p_j}, \mathrm{r_i}$ and $\mathrm{r_j}$ **integers and** $u_i \in \{0,1\}$ **, then the theorem is hold.** □□

Theorem (3.2)

Let δ_k **be a partial schedule,** $k \subset N$ **, for** $i, \mathrm{j} \in \bar{k} = N - k$ **and** $C = C(\delta_k) = \sum_{i=1}^{k} C_i$, **if** $C_i \leq r_j$ **, then** (δ_k, i) **dominates** (δ_k, j) **.**

Proof:

Dessouky and Deogun [3] showed that if $C_i \leq r_j$ **, then** (δ_k, i) **dominates** (δ_k, j) **for** $1/r_i/\frac{1}{n}\sum F_i$ **problem.**

Since $C_i \leq r_j$ **, then there exist the idle time (i. e. the machine will be waited until receive the job j, we denoted by** I_j **) in the case** (δ_k, j) **,** $I_j \geq p_i$ **(because** $C_i \leq r_j$ **. So** (δ_k, i) **dominates** (δ_k, j) **, for** $1/r_i/\sum(F_i + u_i)$ **problem.** □□

Theorem (3.3)

Let δ_k **be a partial sequence,** $k \subset N$ **, for** $i, \mathrm{j} \in \bar{k}$ **and** $C = C(\delta_k) = \sum_{i=1}^{k} C_i$, **if** $C \geq r_j$, $C \geq r_i$ **and** $p_i < p_j$ **,then job i precedes job j** (*i.*e. $\mathrm{i} \prec \mathrm{j}$).

Proof:

Zghair [15] showes that $i \prec j$ **for the** $1/r_i/\sum F_i$ **problem under the same conditions, since** $p_i < p_j$ **and** $u_i \in \{0,1\}$**. So** $i \prec j$ **for** $1/r_i/\sum(F_i + u_i)$**.□**

Theorem (3.4)

If for two adjacent jobs i and j, $(i, j \in N)$ **we have** $p_i < p_j$ **and** $r_i \le r_j$**, then we only conceder schedule in which** $i \prec j$**.**

Proof:

A proof by the method of adjacent pair wise interchange is analogous to the proof of theorem (3.3).□□

Let I_i, I_j denote to idle times of jobs i and j respectively then we can state the following result.

Theorem (3.5)

If δ_k **be a partial sequence,** $k \subset N$**, for** $i, j \in \bar{k}$**, such that $I_i < I_j$ and $r_i + p_i < r_j + p_j$. Then** $i \prec j$ **in optimal schedule.**

Proof:

With the same conditions [15], shows that $i \prec j$ **for problem** $1/r_i/\sum F_i$ **. Existence the idle times I_i and I_j means that** $r_i > C(\delta_k)$ **and** $r_j > C(\delta_k)$**. Since** $I_i < I_j$**, and** $r_i \le r_j$**, Hence in** (δ_k, j) **the job i is waited. Let a_i be denote to the wait i, since** $a_i \ge 1$ **and the number of late jobs is the same in** (δ_k, i) **and** (δ_k, j)**, then** $i \prec j$ **in** $1/r_i/\sum(F_i + u_i)$ **problem.□□**

4. Optimal Solution:

In this section we shall gives optimal solutions for our problem (p) when the data of problem satisfy some conditions.

Theorem (4.1)

For $1/r_i/\sum(F_i + u_i)$ **problem:**

If $r_i = r$ **and** $d_i = d$**, for every** $i \in N$**. Then** SPT **rule give an optimal solution.**

Proof:

Let $\pi = (\sigma, \mathrm{i}, \mathrm{j}, \sigma')$ **be a schedule where** σ, σ' **are tow partial schedules and i, j are tow jobs with** $p_i \le p_j$ **. Let** $\pi' = (\sigma, \mathrm{j}, \mathrm{i}, \sigma')$ **be another schedule has the same job's order as in** π **except the jobs i and j, and let T be a completion time of jobs in subschedule** σ **.**

Let $G_i = F_i + U_i$

First: if $r \le T$ **, then let** $a = T - r$

(I) if $d < T + P_i$ **we have**

$\sum_{i,\,\mathrm{j}\in\pi} G_i = 2a + 2p_i + p_j + 2$ **(i and j are late)**

$\sum_{i,\,\mathrm{j}\in\pi'} G_i = 2a + p_i + 2p_j + 2$ **(i and j are late)**

$\sum_{i,\,\mathrm{j}\in\pi} \mathrm{G_i} - \sum_{i,\mathrm{j}\in\pi'} \mathrm{G\,i} = p_i - p_j < 0$

(II) if $T + p_i < d < T + p_j$ **, then**

$\sum_{i,\,\mathrm{j}\in\pi} \mathrm{G_i} - \sum_{i,\mathrm{j}\in\pi'} \mathrm{G_i} = p_i - p_j - 1 < 0$

(III) if $T + p_j < d < T + p_i + p_j$ **, then**

$\sum_{i,\,\mathrm{j}\in\pi} \mathrm{G_i} - \sum_{i,\mathrm{j}\in\pi'} \mathrm{G_i} = p_i - p_j < 0$

(IV) if $T + p_j + p_j \le d$ **, then the jobs i and j are early in**

π **and** π' **, the theorem is hold.**

Second: if $r > T$ **, then in the same way above we show that the theorem is true (integral) by putting** $a = 0$ **and** $t = r$ **. So** *SPT* **rule gives an optimal solution for problem** $1/r_i = r,\ \mathrm{d_i} = d / \sum(F_i + u_i)$ **.** □

Theorem (4.2)

Moor's algorithm (MA) gives an optimal solution for problem $1/r_i = r,\ \mathrm{p_i} = p / \sum(F_i + u_i)$ **.**

Proof:

Since all jobs have constant release date r and constant processing time p. Then any sequence give minimum sum of flow time, and

$\sum_{i=1}^{n} F_i = \sum_{i=1}^{n} C_i - nr$ **, where** $C_i = \sum_{j=1}^{i} p_j = ip,\ \mathrm{i} = 1, 2, \ldots, \mathrm{n}$ **.**

Then the minimum of objective function depend only on $\sum u_i$, but MA gives minimum for $\sum u_i$ [11]. So MA gives an optimal solution for problem $1/r_i = r,\ \mathrm{p_i} = p/\sum(F_i + u_i)$.□

Let ERD be denoted to the earless release dates rule, according to this rule, jobs are sequenced from beginning to end on the basis of an ascending order of their ready time.

Theorem (4.3)

If for all job $i \in N$, $p_i = p^*$ and $\mathrm{d_i} = \mathrm{d}^*$, Then ERD schedule gives an optimal solution for problem (p).

Proof:

A proof by the method of adjacent pair wise interchange is analogous to the proof of theorem (3.1). □

5. Branch and Bound Algorithm:

We now give the main feature of our branch and bound algorithm and its implementation. Prior to their application, the ERD schedule as a heuristic is used to generate an upper bound on the total cost of an optimal schedule. Also, at the root node of the search tree an initial lower bound on the total cost of an optimal schedule is obtained by modify the lower bound in [1].

Lower Bound:

Let $S1 = \sum_{i=1}^{n} C_i$ which is obtained by SPT rule and $S2 = \sum_{i=1}^{n} u_i$ which is obtained by Moor's Algorithm. Abdul-Razaq and Zghair [1] prove that $Z1 = S1 + S2$ is a lower bound for $1/r_i = 0/\sum(C_i + u_i)$ problem.

Since problem $1/r_i = 0/\sum(C_i + u_i)$ is a spatial case of our problem P, then $Z = Z1 - \sum r_i$ is a lower bound for problem (P) because $F_i = C_i - r_i$, but this lower bound is a weak. To modify it, we shall use the relaxation method as follow:

Relaxed the release date by assumed that all jobs have the same release date r^*, where $r^* = \min\{r_i\}$, and problem P reduce to $1/r_i = r^*/\sum(F_i + u_i)$, and

$LB = nr^* + Z$ is a lower bound of the original problem (P).

Our algorithm uses a forward sequencing branching rule, for which nodes at level L of the search tree correspond to initial partial sequences in which jobs are sequenced in the first L positions. Before the branching we see that: if the data of problem satisfied the conditions of any theorem in section (4), then this problem has an optimal solution and not need to branch in search tree of the branch and bound method. If the lower bound LB equals the upper bound UB, then UB is optimal solution and not need to branch. If not equals the following attempt is made to eliminate nodes which are based on (i) the dominance theorems which are stated in section (3) are applied from level one. (ii) from the second level the dominance theorem of dynamic programming is applied, an adjacent jobs interchange to compare the total cost for the two jobs most recently added to the final partial sequence with corresponding total cost when these two jobs are interchanged in position: if the former total cost is larger than the later, then the current node is eliminated, while if both total cost are same, some convention is used to decide whether the current node should be discarded.

For all nodes that remain after the dominance tests are applied, a lower bound is computed. If the lower bound for any node grater than or equal to smallest of the previously generated upper bounds, then the node is discarded.

6. Computational experience:

In this section, we report results of computational tests to assess the effectiveness of the branch and bound algorithm. Algorithm was coded in Fortran Power Stations(FORTRAN 90) and runs on Pentium IV HP. Compaq Computer with a 2.8 GHz processed and 256 Mb of RAM memories. Whenever a problem remained unsolved with in time limit of 100 seconds, computation was abandoned for that problem.

Test problems with 5, 10, 15, 20, 25, 30, 35 and 40 jobs were generated as fallows. For each job $i \in N$, an integer processing time p_i was generated from the uniform distribution [1, 10]. An integer den date d_i was generated for each job i in the same way as those of [1]. And an integer ready time r_i was generated for each i from the uniform distribution [0, PR], where P $= \sum_{i=1}^{n} P_i$ **and** $R \in \{0.2, 0.5, 0.8, 1.1\}$.

Twenty problems were generated for each value of n, results comparing the initial lower bound(LB), and upper bound (UB) with the optimal solution on the total cost for n=35 are given in table 1. Average computation times in second are given in table 2.

Table 1 gives the result of computations at the root node of the search tree for the 35 jobs test problem. One lower and upper bounds are given for each for each problem, together with the optimal solution value. We observe from table 1 that the lower and upper bounds were deviations from the optimum are very small for the other problems.

We observe from table 2 that problems with (5, 10, 15 and 20) jobs are solved satisfactory, average computation times and numbers of nodes become large for n=25, 30, 35 and 40 with one problem not solve for 30 and 35 jobs and two problems not solve for n= 40.

Also table 2 shows average computation times, number of unsolved problems and the numbers of solved problems that require not more than 200 nodes, that require over 200 nodes and not more than 500 nodes, that require over 500 and not more 1000 nodes and that require over 1000 nodes.

Table (1)

Compare the lower bound and upper bound with optimal solution for n= 35.

No.	LB	UB	Opt
1	1944	2632	1977
2	2558	3392	2589
3	1921	2725	1951
4	1919	2933	2020
5	2036	3254	2111
6	2213	3348	2334
7	1718	2880	1841
8	1887	3210	1946
9	1766	2423	1808
10	2065	2655	2111
11	2025	2905	2109
12	1809	2732	1849
13	1886	2537	2013

14	**2267**	**2890**	**2347**
15	**1629**	**2466**	**1665**
16	**1892**	**2769**	**1965**
17	**1954**	**2862**	**2034**
18	**2037**	**2423**	**2133**
19	**1790**	**3121**	**1842**
20	**1615**	**2694**	**1615**

Table 2

No. of problems solved with out branching; unsolved in limit times and No. of nodes in search tree.

					No. of nods			
N	OLB	OUB	NB	UN	-200	-500	-1000	1000-
5	14	3	11	_	9	_	_	_
10	_	_	_	_	20	_	_	_
15	_	_	1	_	18	1	_	_
20	_	_	_	_	10	6	1	3
25	_	_	_	_	_	14	1	5
30	_	_	_	1	_	7	3	9
35	_	_	_	1	_	1	13	5
40	_	_	_	2	_	1	9	8

OLB: lower bound gives optimal solution

OUB: upper bound gives optimal solution

NB: problems solved with out branching

UN: problems unsolved in the limit time

-200 problems required to solved not more 200 nodes

-500 problems required to solved over 200 not more 500 nodes

-1000 problems required to solved over 500 not more 1000 nodes

1000- Problems required to solved over 1000 nodes.

7- Conclusion

In this paper, we have presented a branch and bound algorithm for single machine scheduling with a composite objective.

The sum of flow time and the number of tardy jobs with release dates. $1/r_i/\sum(F_i+u_i)$ **. Our branch and bound is able to solve problems, with up to 40 jobs.**

Although we decompose the problem in to subproblems with simpler structure and modify the lower bound in [1] by using relaxation method. Our results indicate that problems with a composite objective and unequal release date are still much harder to solve. This bound is valid lower bound for $1/r_i/\sum(F_i+u_i)$ **problem which is a general case of our problem. Prospect for the future are fairly optimistic that more effective and new lower bound techniques and dominance rules are being developed for such NP- hard problems.**

References

[1] Abdul-Razaq T. S. and Zghair M. K.," One optimal machine scheduling to minimize total coast of complation time and number of tardy jobs", J. Basrrh Researches. Vol. 25 Par. 3. 83-93 (2000).

[2] Chen B., Potts C. N. and Woeginger G. J., " A review of machine scheduling: complexity, Algorithms and Approximbility", Handbook of compinatorial optimization D. Z. Du and P. M. Pardalos (Eds) pp. 21-169, 1998 Kluwer Academic Publishers.

[3] Dessouky, M. L. and Deogun, J. S. "Sequencing jobs with unequal ready times to minimize mean flow time", SIAM, J. Comput. Vol. 10, No. 1 Feb. (1981), pp.192-202.

[4] Emmons, H. "One machine sequencing to minimize mean flow time with minimum number tardy", Naval Res. Logist. Quart 22 (1975), pp. 582-592.

[5] Gregorio Baggio, Jacques Wainer and Clarence Ellis, "Applying scheduling techniques to minimize the number of late jobs in workflow systems." ACM symposium on applied computing 2004.

[6] Kellerer H., Tautenhohn T. and Woeginger G. J. "Approximability and non-approximability results for minimizing total flow time on a single machine." Proceedings of the 28th Annual ACM symposium on theory of Computing (1996), 418-426 SIAM Journal on Computing.

[7] Kise H., Ibarki T. and Mine H., "A Solvable Case of the one machine scheduling problem with ready and due times." Oper. Res. 26 (1978) 121-126.

[8] Lenstra J. K., "Sequencing by enumerative methods" Mathematisch Centrum, Amsterdam (1977).

[9] Marjan Yan den Akker and Han Hoogeveen, "minimizing the number of late jobs in a stochastic setting using a chance constraint." Journal of Scheduling V. 11 No. 1 Feb.2008. pp 59-69

[10] Mason A. J. and Anderson E. J., "minimizing flow time on a single machine with job classes and setup times" Naval Research Logistics, vol. 38, pp. 333-350 (1991).

[11] Moore J. M. "An n job, one machine sequencing algorithm for minimizing the number of late jobs.", Management Sci. 19(1973) 1063-1066.

[12] Peter Brucker and Mikhail Y. kovalyov, "Single machine batch scheduling to minimize the weight of late jobs", Mathematical methods of O. R. V.43, N.1 Feb.(1996) pp. 1-8.

[13] Shao Chin Sung, Milan vlach, "Single Machine scheduling to minimize the number of late jobs under uncertainty", Fuzzy set and systems 139 (2003) 421-430.

[14] Van Wassenhove L. N. and Gelders L. F., "Solving a bicriterion scheduling problem," European Journal of Oper. Res. 4(1980)42-48.

[15] Zghair M. K. , "Single machine scheduling to minimize total cost of flow time with unequal ready times", Journal of Basrah Researches V.25, par. 3 (2000) pp. 94-108.