Minimizing Makespan on Single Machine with Release Dates

E. O. Oyetunji

Department of Computer Science
University for Development Studies, Ghana
Email: eoyetunji@yahoo.com

A. E. Oluleye
[2]Department of Industrial and Production Engineering
University of Ibadan, Nigeria
Email: ayodeji.oluleye@mail.ui.edu.ng

Abstract

This paper considers the single criterion scheduling problem of minimizing makespan on a single machine with release dates. The problem is, by nature, NP-Hard, hence approximation algorithms are desired in order to solve the problem. An algorithm (called NAL) is proposed for the problem. The NAL algorithm is compared with the branch and bound (BB) method and a test algorithm (AEO) selected from the literature. The three solution methods were tested on a set of randomly generated problems (ranging from 10 to 500 jobs). Experimental results show that the proposed algorithm performed competitively with the BB method and outperformed the AEO algorithm.

Keywords: Scheduling, Algorithm, Single machine, Single criterion, Makespan, Release dates

1. Introduction

In scheduling, there are many objectives or criteria by which the performances of solution methods can be measured. These criteria may be studied singly (single criterion scheduling problems) or combined (multi-criteria scheduling problems). Being a major cardinal point in scheduling problems, many researchers have carried out extensive study of scheduling criteria [1, 2, 3, 4]. Not less than about 29 different scheduling criteria have been identified [4]. One scheduling criterion that has received the attention of researchers over the years is makespan (also called schedule length or maximum completion time) and is often denoted by C_{max}. Makespan is the completion time of the last scheduled job and is directly proportional to the production costs. Thus, makespan is a very important scheduling criterion [5, 6], hence, its choice in this study.

The problem of minimizing makespan on a single machine with zero release dates is trivial and can be solved optimally using the shortest processing time (SPT) rule [7]. The introduction of non-zero release date constraints makes the problem NP-Hard [8, 9]. Many variants of this problem have been explored by several researchers [10, 11, 12, 13].

Jeng and Lin [10] considered the single-machine scheduling problem of minimizing the maximum completion time (makespan) for a set of independent jobs. They explored the variant in which the processing times of the jobs are non-linear step function of their starting times and due dates. A pseudo-polynomial time dynamic programming algorithm was proposed for the problem. The problem of minimizing the makespan in a single machine with convex decreasing resource dependent processing times was explored by Kaspin and Shabtay [11]. They considered the two cases (where the job release dates are identical and the general case of non-identical job release dates). An $O(n)$ algorithm was proposed for the case of identical job release dates while an $O(n^2)$ algorithm was proposed for the case of non-identical job release dates.

Shuguang Li et al. [12] considered the problem of scheduling jobs with release times and non-identical job sizes on a single batching machine with the aim of minimizing makespan. An approximation algorithm with a worst-case ratio of $2 + \varepsilon$ was proposed for the problem. Kubzin et al. [14] explored the classical problem of minimizing makespan on a two machine flowshop with zero release dates and the processing of jobs being interrupted by an unavailability period of a machine. They considered both the resumable scenario in which the processing can be resumed when the machine next becomes available, and the semi-resumable scenarios in which some proportion of the processing is repeated but the job is otherwise resumable. For the resumable scenario, a fully polynomial-time approximation schemes based on an extended dynamic programming algorithm was proposed while a 3/2-approximation algorithm was proposed for the semi-resumable scenario.

Hurink [15] considered the general scheduling problem of minimizing makespan on a single machine with release dates. He stated that the problem may be reducible to the problem of minimizing maximum lateness on a single machine with zero release dates by replacing $d_j = K - r_j$ where K (constant) > max r_j.

The literature on the general scheduling problem of minimizing makespan on a single machine in which the release dates and processing time are known and fixed positive intergers (i.e. $1|r_i|C_{\max}$ problem) appears sparse. We are unaware of any study in which a polynomial-time approximation algorithm has been proposed for this problem. In this study, we proposed a polynomial-time approximation algorithm (called NAL) for solving this problem.

The paper is organized as follows: Section 1 deals with general introduction and a brief review of the literature. The description of the scheduling problem being explored is given in section 2 while the concept of machine idle time is discussed in section 3. The proposed and selected solution methods are discussed in section 4 while section 5 covers data analysis. Section 6 deals with results and discussions while the paper is concluded in section 7.

2. The Problem

The scheduling problem being explored in this paper is described as follows: Given the general one-machine scheduling problem where a set J of n jobs has to be sequenced on a machine in order to minimize the makespan (also called schedule length or maximum completion time). It is assumed that only one job can be processed at a time and the arrival time of every job J_i at the machine is known and denoted by r_i (release date). Also, each job J_i needs p_i time units on the machine (processing time).

The time the processing of job J_i starts on the machine (start time) is designated as s_i with the property:

$$s_i \geq r_i \quad \text{………….(1)}$$

while, its completion time (C_i) is defined as:

$$C_i = s_i + p_i \,. \quad \text{…………(2)}$$

Also, the makespan (C_{max}) is defined as:

$$C_{max} = max\,(C_1, C_2, \ldots, C_n) \quad \text{…………(3)}$$

Using the notations of Graham et al. [16], the problem being considered is represented as

$$1 \mid r_i \mid C_{\max}$$

It is assumed that pre-emption is not allowed and that the problem is static and deterministic i.e. number of jobs, their processing times, and ready times are all known and fixed. The assumptions reflect many real-life problems which are often being encountered.

3. Machine idle time

Machine idle time is the period of time in which the machine is not processing any job during the course of production. This may occur at any time (beginning, middle or end) of production. For the particular machine environment (single machine) and job characteristics (existence of release dates and pre-emption is not allowed) being considered in this paper, machine idle time is not desirable because they prolong the makespan which is being minimized. Therefore, they should be reduced as much as possible. It should be noted that, because of release dates constraints, it may be practically impossible to totally eliminate the idle time on the machine. Suppose we are given a hypothetical 4-job single machine scheduling problem whose data are shown in Table 1. The gantt chart of a hypothetical schedule (1 4 2 3) is as shown in Fig. 1. The idle time on the machine may be classified into two (Natural and Forced idle times).

Table 1. Data for a hypothetical 4-job problem

Jobs	**Processing time (p_i)**	**Release dates (r_i)**
1	4	1
2	3	7
3	5	2
4	1	6

3.1 Natural Idle Time

This type of machine idle time occurs as a result of release dates constraints. For example, if the machine has completed the current job and the other jobs are yet to arrive, there will be natural idle time (see Fig. 1). The scheduler cannot do anything about the natural idle time.

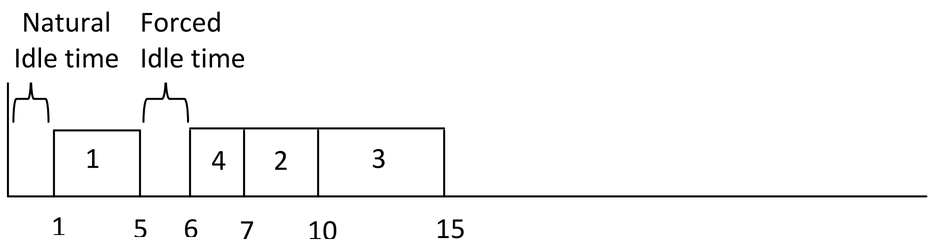

Fig. 1: Gantt chart of a hypothetical schedule showing forced and natural machine idle time

3.2 Forced Idle Time

This type of machine idle time occurs as a result of the schedule constructed and not necessarily the release dates constraints. For example, in Fig. 1 above, the second idle time is forced because at time t = 5 job 3 has arrived but was not scheduled. The scheduler can do something (reduce or eliminate it) about the forced idle time. For instance, if the scheduler decide to alter his/her schedule so that the schedule becomes 1 3 4 2 as shown in Fig. 2, the forced idle time (which existed in Fig. 1) has now been completely eliminated.

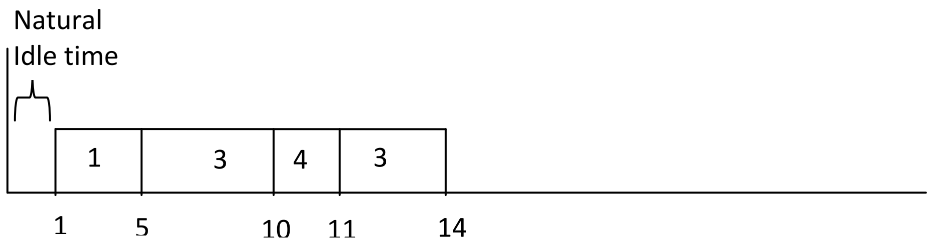

Fig. 2: Gantt chart of a hypothetical schedule showing only natural machine idle time

4. Solution Methods

The makespan or schedule length is an important scheduling criterion as it has direct correlation with the production costs. Because of the release dates constraints, the problem is NP-Hard, hence, approximation algorithms are desired to solve this important scheduling problem. Based on the analysis carried above and our observation of the effects of idle time on the machine with respect to the scheduling problem being considered, an algorithm is proposed for solving this problem. To compare the performance of the proposed algorithm, an algorithm proposed by Oyetunji and Oluleye [17] for minimizing total completion time on a single machine with release dates was selected from the literature based on performance. Also, a branch and bound (BB) method was implemented. The three solution methods are described below.

4.1 The New Algorithm (NAL)

The new algorithm (called NAL) is modification of the AL1 algorithm proposed by Oyetunji [18] for the single criterion problem of minimizing total completion time on single machine with release dates. The NAL algorithm steps are described below.

NAL Algorithm Steps

Step 1: Obtain a schedule using the AL1 algorithm of Oyetunji [18] $i = 1$

Step 2: Consideration for job in position i in the AL1 schedule. Test for idle time on the machine at position i. If $r_i > C_{i-1}$ then there is idle time on the machine, hence proceed to Step 3 otherwise go to Step 4.

Step 3: Test for the type of idle time (Forced or Natural) on the machine at position i. If at time $t = C_i$ there is at least one unscheduled job with $r_i \leq C_i$ then the idle time is Forced, hence select the job with the smallest processing time among the jobs whose $r_i \leq C_i$ and schedule this job to position i in the NAL schedule and go to Step 5; otherwise the idle time Natural, hence go to Step 4.

Step 4: Schedule the job in position i in the AL1 schedule to position i in the NAL schedule, then proceed to Step 5.

Step 5: $i = i+1$, if $i \leq n$ (number of jobs) then go back to Step 2; otherwise proceed to Step 6.

Step 6: Stop.

4.2 Branch and Bound (BB) Method

The branch and bound solution method for the problem was implemented as follows: The value of makespan at each node was computed and the node that gave the minimum value of makespan was used to determine branching. At the terminal node, when all the jobs have been fully assigned, the node was noted and became the solution to the considered problem.

4.3 AEO Algorithm

The AEO algorithm was proposed for the single criterion scheduling problem of minimizing the total completion time of jobs on a single machine with release time and was selected for test based on its outstanding performance. The basic idea in this algorithm consists of choosing a job J_i with the least processing time among the set of jobs that have arrived and are available for processing at time *t* until all the jobs have been scheduled. The algorithm cleverly selects the job to process each time the machine becomes idle or a new job arrives.

5. Data Analysis

The proposed algorithm (NAL) was compared with the AEO algorithm and the BB method. The three solution methods were tested on a set of 900 (18 problem sizes ranging from 10 to 500 jobs by 50 problem instances) randomly generated problems. The processing times of the jobs were randomly generated (using random number generator in Microsoft visual basic 6.0) with values ranging between 1 and 100 inclusive. The ready times were also randomly generated with values ranging between 0 and 24 inclusive.

A program was written in Microsoft visual basic 6.0 to apply the solution methods (NAL, AEO and BB) to the problems generated. The program computes the value of makespan obtained by each solution method for each problem. The data was exported to Statistical Analysis System (SAS version 9.2) for detailed analysis. The hardware used for the experiment is a 1.73 GHz T2080 Intel CPU with 1024 MB of main memory.

The test of means was carried out using the GLM procedure so as to determine whether or not the differences observed in the mean value of makespan obtained by various solution methods are statistically significant. The results obtained are presented and discussed in section 5.

6. Results and Discussions

The three solution methods were evaluated on 50 problem instances under 18 different problem sizes (10 to 500 jobs). Based on the minimum mean value of makespan (effectiveness), a ranking order of BB, NAL and AEO was obtained for all the problem sizes ($10 \leq n \leq 500$) considered (Table 2). To compare the performance of NAL and AEO algorithms with that of the BB method, approximation ratios (NAL/BB and AEO/BB) was computed and plotted (Fig. 3). It is clear that the NAL algorithm performed competitively with the BB method. Also, the performance of NAL algorithm is better than that of the AEO algorithm when compared with the BB method (Fig. 3). When the number of jobs (n) is greater or equal to 100, the performance of NAL and AEO algorithms almost equals that of the BB method (Fig. 3). The test of means carried out show that the differences in the mean value of makespan obtain by all the three solution methods for all the considered problem sizes is not significant at 5% level (Table 3). This confirms the competitive performance of the NAL and AEO algorithms compared to BB method.

In order to measure the efficiency of the solution methods, the execution time (secs) taken by each solution method to obtain solution to an instance of a problem was computed. The mean values of execution time over the fifty problem instances solved under each problem size are as shown in Table 4. To show the exponential nature of the time complexity function of the BB method, the mean values of the execution time has been plotted (Fig. 4). On the average, the BB method spent about 2 hours (7360.04 secs) in solving an instance of problem involving 200 jobs while the AEO and NAL algorithms spent 4.4 secs and 0.25 secs respectively on the same number of jobs (Table 4). Therefore, both NAL and AEO algorithms are faster than BB (Fig. 4). The prohibitive time required by the BB method was the reason why it could not be applied to problems involving more than 200 jobs. Fig. 5 shows that the NAL algorithm is faster (more efficient) than the AEO algorithm.

The mean value of execution time taken by the NAL and AEO algorithms is significantly different from that of the BB method at 5% level (Table 5). Also, at the same level, the mean value of execution time taken by the NAL algorithm is significantly different from that of the AEO algorithm (Table 6). This, thus, confirms the superiority (with respect to efficiency) of the NAL algorithm over both the AEO algorithm and BB method.

Table 2 : Mean value of Makespan by Problem Sizes and Solution Methods

Problem	Mean of Makespan		
Size	BB	NAL	AEO
10x1	500.54	504.64	508.56
15x1	774.36	777.16	779.72
20x1	963.70	964.12	966.36
25x1	1291.12	1292.96	1296.80
30x1	1536.12	1537.54	1540.72
35x1	1742.60	1743.38	1747.50
40x1	2040.84	2041.60	2044.40
45x1	2247.42	2248.14	2249.40
50x1	2587.14	2587.16	2592.28
100x1	5065.94	5066.14	5067.98
150x1	7484.00	7484.32	7486.84
200x1	10087.96	10088.20	10090.36
250x1		12540.86	12542.16
300x1		15092.16	15093.30
350x1		17596.78	17598.18
400x1		20080.60	20081.50
450x1		22735.86	22736.70
500x1		25220.70	25221.54

Sample size=50

Table 3 : Test of means (probabily values) of makespan for $10 \leq n \leq 200$ problems

	Solution Methods		
Solution Methods	**BB**	**NAL**	**AEO**
BB	-	>0.5x	>0.5x
NAL	>0.5x	-	>0.5x
AEO	>0.5	>0.5x	-

Note x indicate non significant result at 5% level; Sample size = 50
- indicate not necessary

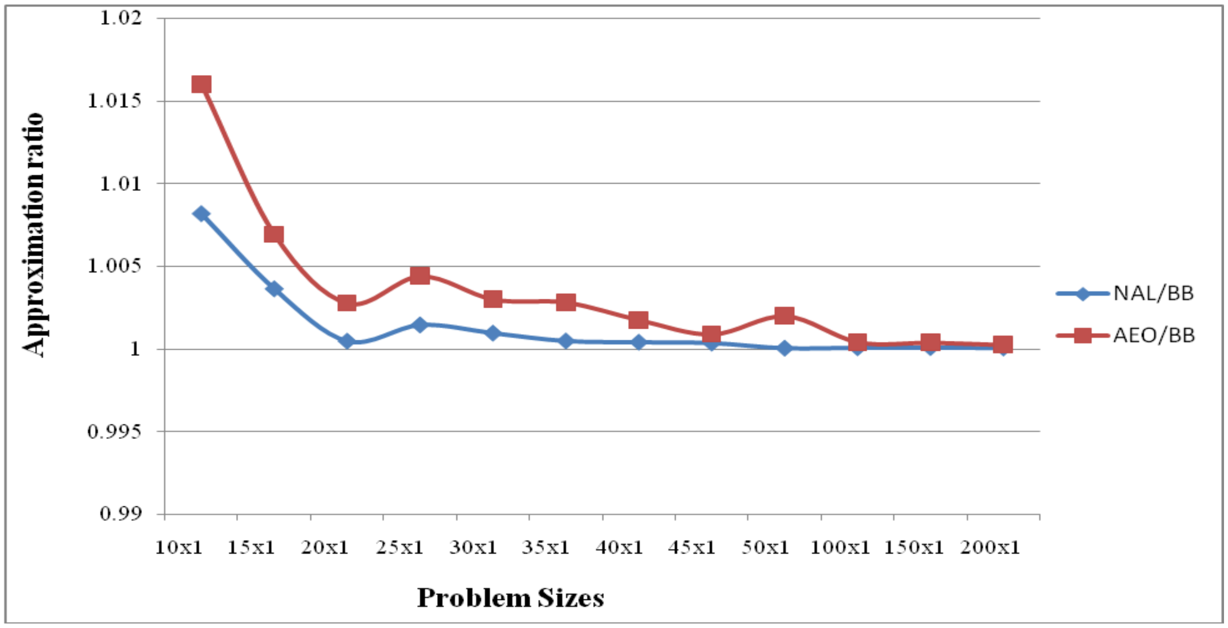

Fig. 3 : Approximation ratio NAL and AEO compared with BB for 10 ≤ n ≤ 200 problems

Table 4 : Mean value of execution time (secs) taken by solution methods for various problem sizes

Problem	Mean of execution time (secs)		
Size	BB	NAL	AEO
10x1	0.0700	0.0517	0.0120
15x1	0.3200	0.0542	0.0236
20x1	0.8500	0.0564	0.0443
25x1	2.0200	0.0581	0.0869
30x1	4.2000	0.0606	0.0918
35x1	7.4100	0.0680	0.1276
40x1	12.5000	0.0680	0.1617
45x1	20.3000	0.0686	0.2122
50x1	28.8900	0.0711	0.2642
100x1	477.6800	0.1111	0.9856
150x1	2571.4600	0.1669	2.2363
200x1	7360.0400	0.2464	4.3989
250x1		0.3464	6.2343
300x1		0.4367	9.0162
350x1		0.5583	12.2758
400x1		0.6992	15.9392
450x1		0.8820	21.0477
500x1		1.1091	23.9270

Sample size=50

Table 5 : Test of means (probabily values) of execution time for $10 \leq n \leq 200$ problems

	Solution Methods		
Solution Methods	**BB**	**NAL**	**AEO**
BB	-	<0.005*	<0.005*
NAL	<0.005*	-	>0.5x
AEO	<0.005*	>0.5x	-

Note * indicate significant result at 5% level; Sample size = 50

x indicate non significant result at 5% level

- indicate not necessary

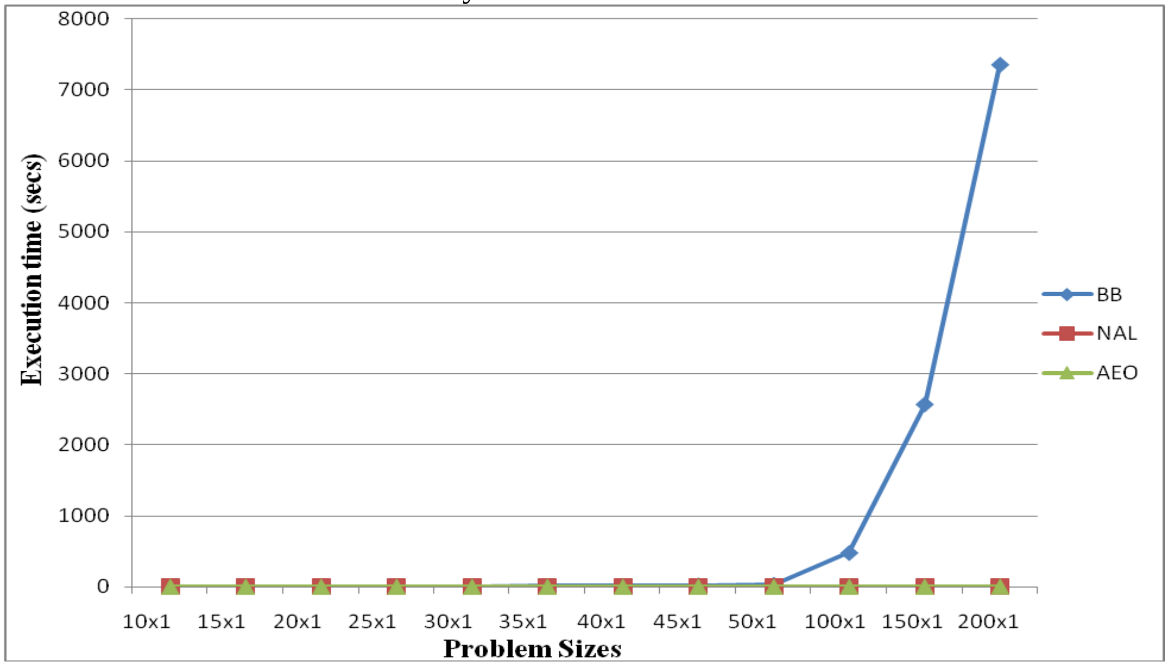

Fig. 4: Comparison of execution time (secs) taken by BB, NAL and AEO for $10 \leq n \leq 200$ problems

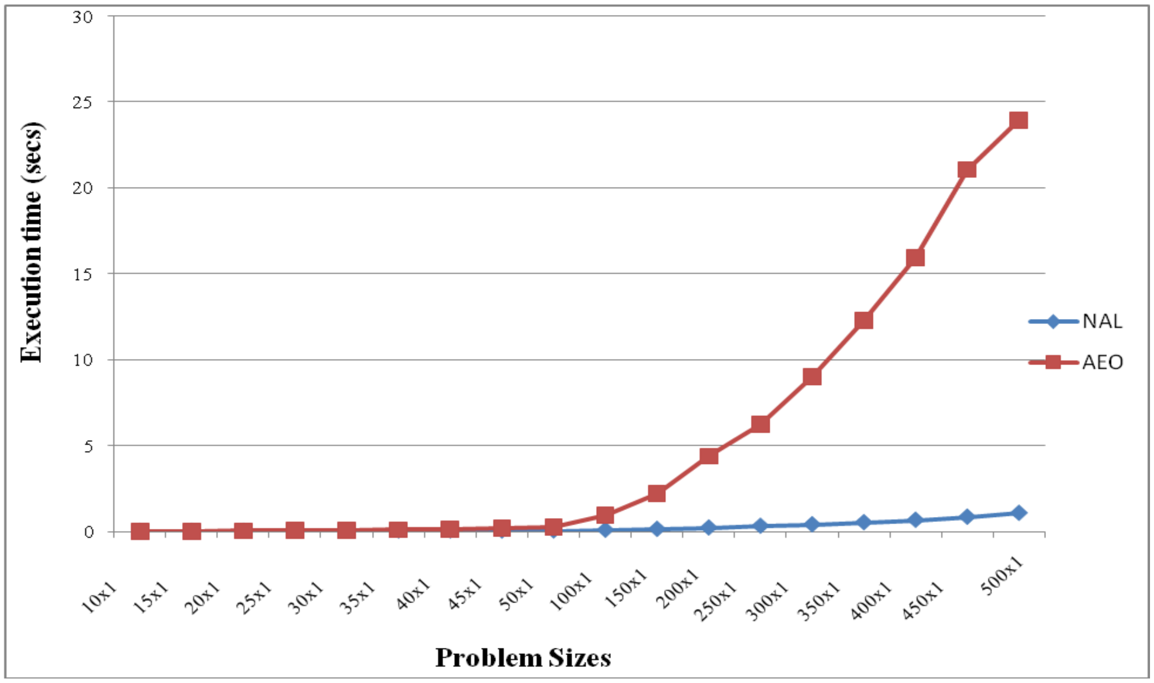

Fig. 5: Comparison of execution time (secs) taken by NAL and AEO for $10 \le n \le 500$ problems

Table 6 : Test of means (probabily values) of execution time for $250 \le n \le 500$ problems

	Solution Methods	
Solution Methods	**NAL**	**AEO**
NAL	-	<0.001*
AEO	<0.001*	-

Note * indicate significant result at 5% level; Sample size = 50
- indicate not necessary

7. Conclusion

The problem of minimizing the makespan (schedule length) on a single machine with release dates has been explored in this paper. It has been observed that the idleness on the machine is one of the factors that prolong the makespan. Two types of idle time (Forced and Natural idle times) have been identified. This observation led to the development of an algorithm (NAL) for minimizing the makespan on a single machine with release dates. The proposed algorithm was compared with the branch and bounds method and the AEO algorithm selected from the literature.

The performances of the three solution methods were evaluated with respect to both effectivess (closeness of the value of the makespan to the optimal) and efficiency (how fast solution can be obtained i.e. a measure of execution speed). Therefore, based on both effectiveness and efficiency, the NAL algorithm is recommended for the scheduling problem of minimizing the makespan on a single machine with release dates.

References

[1] Conway, R. W., Johnson, B. M., Maxwell, W. L. (1960). An experimental investigation of priority dispatching, J. Ind. Eng., 11, 221-230.

[2] Gere, W. S. (1962). A heuristic approach to job shop scheduling. PhD. Thesis, Carnegie Institute of Technology.

[3] Beenhakker, H. L. (1963). Development of alternate criteria for optimality in the machine sequencing problem, PhD. Thesis, Purdue University.

[4] Oyetunji, E.O. (2009). Some common performance measures in scheduling problems. Research Journal of Applied Sciences, Engineering and Technology, 1(2), 6-9.

[5] Oluleye, A.E., Oyetunji, E.O. (1999). Performance of some static flowshop scheduling heuristics. Directions in Mathematics, 315-327.

[6] Shahzad, A. (2010). A Single Machine Scheduling Problem with Individual Job Tardiness based Objectives, available at: http://oro.univ-nantes.fr/sujets-09-10/shahzad.pdf; (accessed on 13th August, 2010).

[7] Karger, D., Stein, C., Wein, J. (1997). Scheduling Algorithms, A chapter written for the CRC Handbook on Algorithms (Boca Raton 1997).

[8] Nagar, A., Haddock J., Heragu,S. (1995). Multiple and bicriteria scheduling: A literature survey, European Journal of Operational Research, 81(1), 88-104.

[9] Nieberg, N. (2010). Single Machine Scheduling. available at: http://www.or.uni-bonn.de/lectures/ss10/scheduling_data/sched10_2.pdf; (accessed on 13th August, 2010).

[10] Jeng, A., Lin, B. (2004). Makespan minimization in single-machine scheduling with step-deterioration of processing times. Journal of the Operational Research Society, 55, 247–256.

[11] Kaspin, M., Shabtay, D. (2004). Convex resource allocation for minimizing the makespan in a single machine with job release dates. Computers & Operations Research, 31, 1481–1489.

[12] Shuguang, L., Goujun, L. Xiaoli, W., Qiming, L. (2005). Minimizing makespan on a single batching machine with release times and non-identical jobs sizes. Operations Research Letters, 33, 157-164.

[13] Bao-Qiang, F., Guo-Chun, T., Shu-Xia, Z. (2008). Scheduling jobs on a single machine with inventory operations, *In proceeding of 7th International Symposium on Operations Research and Its Applications (ISORA'08), Lijiang, China, October 31–Novemver 3, 2008*, pp. 344–350.

[14] Kubzin, M. A., Potts, C. N., Strusevich, V. A. (2009). Approximation results for flow shop scheduling problems with machine availability constraints. Computers & Operations Research, 36(2), 379-390.

[15] Hurink, J.L. (2010). Scheduling (LNMB Course), Lecture 3, available at; http://wwwhome.math.utwente.nl/~hurinkjl/sched/sl3-handout.pdf; (accessed on 13th August, 2010).

[16] Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discrete Math. 5, 287-326.

[17] Oyetunji, E.O., Oluleye, A.E. (2007). Heuristics for minimizing total completion time on single machine with release time. Advanced Materials Research, Vols 18-19, pp. 347-352.

[18] Oyetunji, E.O. (2006). The Development of Scheduling Heuristics for the Bicriteria problem on a single machine. PhD thesis, University of Ibadan, Nigeria.