NEW HEURISTICS FOR MINIMISING TOTAL COMPLETION TIME AND THE NUMBER OF TARDY JOBS CRITERIA ON A SINGLE MACHINE WITH RELEASE TIME

E. O. Oyetunji[1] and **A. E. Oluleye**[2]

[1]Department of Computer Science
University for Development Studies, Ghana
eoyetunji@yahoo.com

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

ABSTRACT

This paper considers the bi-criteria scheduling problem of simultaneously minimising the total completion time and the number of tardy jobs with release dates on a single machine. Since the problem had been classified as NP-Hard, two heuristics (HR9 and HR10) were proposed for solving this problem. Performance evaluations of the proposed heuristics and selected solution methods (HR7 and BB) from the literature were carried out on 1,100 randomly generated problems ranging from 3 to 500 jobs. Experiment results show that HR7 outperformed HR10 when the number of jobs (n) is less than 30, while HR10 outperformed HR7 for $n \geq 30$.

OPSOMMING

In hierdie artikel word die bi-kriteria-skeduleringsprobleem bestudeer waar die totale voltooiingstyd en die aantal take wat laat is op 'n enkele masjien geminimiseer moet word. Verskeie heuristieke word voorgestel en getoets om sodoende die beste benadering te identifiseer.

1. INTRODUCTION

Scheduling is concerned with the allocation of tasks (jobs) to processors (machines) with the aim of optimising (minimising or maximising) a set of objectives. When it is desired to minimise (or maximise) only one objective at a time, the resulting problem is tagged as a single criterion scheduling problem. It is called a bi-criteria scheduling problem when it is desired to optimise two criteria (objectives) at a time, while it is called a multi-criteria (multi-objectives) scheduling problem whenever we seek to optimise three or more criteria at a time. Scheduling objectives provide the basis on which the performances of solution methods to scheduling problems can be evaluated.

A schedule is completely defined by the sequence (order) in which the jobs (tasks) are to be processed on the machines (processors) and the time at which the processing of each job may start and end on the machines. The total cost of a schedule is not a function of one objective, but rather a function of two or more objectives [4]. Since realising this fact, many researchers [6, 13, 15, 21, 22, 23] have worked on bi-criteria and multi-criteria scheduling problems. The effort put into single criterion scheduling problems since the pioneering work of Johnson [7] has paid off. Today a number of solution methods proposed for the bi-criteria and multi-criteria scheduling problems have been either a direct modification or an amalgamation of solution methods to single criterion scheduling problems [19, 23].

Many variants of bi-criteria and multi-criteria scheduling problems have been studied by many researchers. Essentially there are four techniques for combining the criteria [6, 26], which may be categorised as either hierarchical minimisation or simultaneous minimisation. The hierarchical minimisation technique involves a situation in which one criterion is more important than the other. The criterion that is less important is minimised, subject to the constraint that the more important criterion is less or equal to a predetermined constant (for a minimisation problem).

The simultaneous minimisation approach involves minimising all the criteria at the same time. There are three variants of this approach: priori, interactive, and posteriori techniques.

In the *priori* approach, the criteria are combined into a scalar function. Each criterion is assigned a weight (which denotes the relative importance of the criterion) by the Decision Maker (DM). The analyst then constructs solutions that minimise the scalar function.

In the *interactive* (or progressive) approach, a set of solutions that represents trade-offs on the values of the criteria is sought. Partial preference information (trade-offs) is provided by the DM, while the analyst constructs solutions based on the DM's partial preference. The obtained solutions are presented to the DM, who can either accept or reject them. The DM's preferences can be altered if he/she is not satisfied with the solutions presented by the analyst – in which case the analyst begins the search for a new set of solutions based on the DM's new preferences. Again the obtained solutions are presented to the DM. This process continues until the DM is satisfied with the presented solution set.

The *posteriori* approach also involves constructing a set of compromise solutions. In this case, the analyst designs a solution method to seek the set of compromise solutions (Pareto-optimal set), and presents the obtained solutions to the DM who selects the one that best satisfies his/her preferences. The implications of each solution are presented to the DM.

This paper considers the bi-criteria scheduling problem of simultaneously minimising the total completion time (Ctot) and the number of tardy jobs (NT) on a single machine with release time. The priori approach is adopted. The problem consists of a set of n jobs (J_1, J_2, … J_n) with each job having processing time (P_i), release or ready time (r_i), and due dates (d_i). The completion time for processing each job is C_i. A job J_i is said to be tardy ($U_i = 1$) if

it is completed after its due date (i.e. $C_i > d_i$). The sum of individual job completion times is called the total completion time ($\sum_{i=1}^{n} C_i$), while the sum of jobs that are tardy is called the number of tardy jobs ($\sum_{i=1}^{n} U_i$). The total completion time and the number of tardy jobs have relative weights denoted by α and β respectively. Thus the objective function is a linear combination of the total completion time and the number of tardy jobs criteria $(\alpha \sum_{i=1}^{n} C_i + \beta \sum_{i=1}^{n} U_i)$. The case of the equal importance of the two criteria was considered ($\alpha = \beta = 0.5$). Two solution methods (heuristics) were proposed for this problem, and compared with a heuristic called HR7, proposed earlier for the same problem by Oyetunji [17].

2. PREVIOUS WORK

A single performance measure represents only a component of the total cost of a schedule. In practice, decision makers usually have to consider more than one criterion before arriving at a decision [13]. Thus, considering scheduling problems with more than one criterion is more relevant in the context of real life scheduling problems. The problem of scheduling jobs with diverse performance objectives (i.e. jobs having different weights, and in some cases, different performance measures or criteria) was studied by Peha [8]. He called them 'heterogeneous-criteria scheduling problems', and presented an $O(N^2)$ algorithm for the problem. A hybrid-framework (consisting of two modules: an expert-scheduler and a meta-scheduler) for multi-criteria production scheduling problems was proposed by Geyik [5]. The performance criteria considered included makespan, mean flowtime, mean tardiness, and mean machine idle-time.

In 2003 a multi-criteria scheduling system was developed and implemented for a computer assembly shop by Abhyuday et al. [1]. Six performance indices (average flow time, maximum tardiness, customer priority, inventory holding cost, production balancing, and transportation cost between two assembly lines) were considered. The scheduling problem of simultaneously minimising the average flow-time and the maximum tardiness criteria was explored by Leon [11]. Two search procedures (called BMOS and DMOS) were implemented using problem-space neighborhood generation techniques.

Assayad et al. [2] proposed a list scheduling heuristic (called Reliable Bi-Criteria Scheduling Algorithm) for the bi-criteria problem of scheduling data-flow graphs of operations on to parallel heterogeneous architectures. The two criteria considered are, first, minimisation of the schedule length, and second, maximisation of system reliability. Landa and Burke [9] presented an introductory tutorial on the application of multi-objective metaheuristics to the optimisation of some multiple criteria scheduling problems.

Molnar [15] used a discrete event simulation model to propose a genetic algorithm (GA) for multi-criteria scheduling optimisation of order picking activities in an automotive parts warehouse. Mehta et al. [14] presented a Multi-Criteria Scheduling Algorithm (MCSA) based on swapping dispatching rules to yield improved system performance. Rerouting of jobs in queues was conventionally adopted in case of machine breakdowns, while a Selective Rerouting (SR) approach based on the lateral entry of critical jobs in the queue of alternate machine was adopted.

Yves and Emmanuel [25] addressed the problem of dynamically scheduling independent tasks and/or application task graphs in a GridRPC environment. Four heuristics were proposed and compared with the well-known minimum completion time (MCT) algorithm. Their experiment results showed that the proposed heuristics outperformed the MCT

algorithm on several metrics, including the makespan and the response time. Low et al. [12] developed a multi-objective model for solving flexible manufacturing system (FMS) scheduling problems that simultaneously considered three performance measures: minimum mean job flow time, mean job tardiness, and minimum mean machine idle time. Hybrid heuristics were proposed for solving the addressed FMS scheduling problems.

Benoit et al. [3] studied the complexity of the bi-criteria mapping problem for pipeline graphs on communication homogeneous platforms. Several efficient polynomial bi-criteria heuristics were proposed and their relative performance evaluated through extensive simulations. Petrovic et al. [24] presented a GA for multi-objective job shop scheduling problems. The developed tool was used to analyse and solve a real-world problem defined in collaboration with a pottery company. Klusáček et al. [10] proposed a novel schedule-based approach (Earliest Gap-Earliest Deadline First [EG-EDF]) for scheduling a continuous stream of batch jobs on the machines. The proposed solution was compared with some of the most common queue-based scheduling algorithms, such as FCFS, EASY backfilling, and Flexible backfilling. Their experiments showed that the EG-EDF rule was able to compute good assignments, often with a shorter algorithm runtime, compared with the other queue-based algorithms.

The problem of maximising the total profit and minimising the maximum idle time on m unrelated parallel machines where pre-emption is allowed, was explored by T'kindt et al. [27]. The objective function was chosen as a linear combination of the total profit and the maximum idle time. The criteria have relative weights. An algorithm to compute the set of all strict Pareto optima was proposed. Nelson et al. [28] explored the two bi-criteria scheduling problems. These are (1) minimisation of the mean flow time and number of tardy jobs with zero release dates on a single machine, and (2) minimisation of the maximum tardiness and number of tardy jobs with zero release dates on a single machine. Four algorithms were proposed for the problems.

Hoogeveen and Velde [29] studied the bi-criteria single-machine scheduling problem of minimising total completion time and maximum cost, f_{max}. The maximum cost was defined as $\max_{1 \leq j \leq n} f_j(C_j)$, where each f_j denotes an arbitrary regular cost function for job J_j. They proved that the problem is simultaneously solvable in polynomial time. Two algorithms (Algorithm I and Algorithm II) were proposed for this problem. Verma and Dessouky [30] studied the problem of determining a schedule of jobs with unit-time lengths on a single machine that minimises the total weighted earliness and tardiness penalties with respect to arbitrary rational due-dates. The problem was formulated as an integer programming problem, and it was shown that if the penalties meet a certain criterion called the Dominance Condition, then there is an extremal optimal solution to the LP-relaxation that is integral, leading to a polynomial-time solution procedure.

The bi-criteria scheduling problem of minimising the maximum earliness and the number of tardy jobs on a single machine was explored by Azizoglu et al. [31]. First, they examined the problem of minimising maximum earliness while keeping the number of tardy jobs to its minimum value (hierarchical minimisation). An algorithm for generating all the efficient schedules for bi-criteria problems was proposed. Second, a general procedure to find the efficient schedule that minimises a composite function of the two criteria by evaluating only a small fraction of the efficient solutions was developed. Hoogeveen [6] explored multi-criteria scheduling problems by aggregating the criteria into a single function called the composite objective function. He suggested two types of composite objective function: linear and general.

Recently, Oyetunji and Oluleye [22] proposed a method for evaluating the performances of solution methods for bi-criteria scheduling problems. A normalisation scheme (which converts the values of one criterion to the other) was also put forward. Three heuristics (HR4, HR5, and HR6) were proposed by Oyetunji and Oluleye [19] for the bi-criteria problem of simultaneously minimising the total completion time (Ctot) and the number of tardy jobs (NT) on a single machine with release dates. The bi-criteria problem of

minimising the total completion time (Ctot) and number of tardy jobs (NT) on a single machine with release dates was modeled as hierarchical minimisation problems by Oyetunji and Oluleye [20]. Two types of hierarchical minimisation models (one with the total completion time criterion being more important than the number of tardy jobs criterion, and the other with the number of tardy jobs criterion being more important than the total completion time criterion) were explored.

The technique proposed earlier by Oyetunji and Oluleye [22] for assessing the performance of solution methods to bi-criteria problems was extended to multi-objective scheduling problems by Oyetunji [16]. The difference was that the composite objective function was designed to handle situations in which some of the criteria were to be minimised, while at the same time some were to be maximised (mixed multi-objectives scheduling). A new heuristic (HR7) was proposed by Oyetunji [17] for the bi-criteria problem of simultaneously minimising the total completion time and the number of tardy jobs on a single machine with release dates. His experiments show that the HR7 outperformed an earlier heuristic (HR6) for the same bi-criteria problem in terms of both effectiveness (for problems involving 3 to 500 jobs) and efficiency (for problems involving fewer than 30 jobs). For a more detailed review of literature on multi-criteria scheduling, we refer to Hoogeveen [6] and T'kindt and Billaut [32].

3. SOLUTION METHODS

A number of solution methods (proposed and selected) for the bi-criteria scheduling problem of simultaneously minimising the total completion time and the number of tardy jobs on a single machine with release time are discussed below.

3.1 Proposed solution methods

Since this problem is NP-Hard, heuristic approaches are desired to obtain fairly good schedules. Two heuristics (HR9 and HR10) are proposed here for the bi-criteria scheduling problem.

3.1.1 HR9 heuristic

The basic idea in this heuristic consists of choosing a job J_i with the least job allowance (due dates minus processing time) among the set of jobs that have arrived and are available for processing at time t. If the chosen job will be tardy, do not schedule the job: rather choose the next job that has the least job allowance, until all the jobs have been scheduled. The HR9 heuristic steps are outlined below:

HR9 Heuristic steps

Step 0: Initialise
Job_SetA=[J_1, J_2,…,J_n]; The set of given jobs
Job_SetB=[0] The set of scheduled jobs
Job_SetC=[J_1, J_2,…,J_n] The set of unscheduled jobs
Job_SetD=[0] The set of available jobs at time t
t = $\min_{J_i \in Job\_SetA} r_i$ (i.e. the minimum ready time of all jobs)

Step 1: At time t
Update Job_SetD with jobs for which r_i<=t
If r_i <= t then Job_SetD = [J_i]

Step 2: Calculate Job_Allowance[J_i] for all available jobs at time t

Step 3: Choose the job with the least Job_Allowance among the jobs that have arrived at time t from Job_SetD

Step 4: If the job chosen in Step 3 will be tardy, remove it from Job_SetD and go back to Step 3, otherwise go to Step 5.

Step 5: Add the job chosen in Step 3 to Job_SetB and remove the same job from Job_SetC. Compute start time S_i = t and completion time $C_i = S_i + p_i$

Step 6: Compute new time as: t = max (C_i, $\min_{J_i \in Job\_SetC} r_i$); maximum of the completion time or the minimum ready time of the remaining unscheduled jobs.
Step 7: If Job_SetC is not empty, go back to Step 1, otherwise proceed to Step 8
Step 8: Job_SetB is the schedule required.
Step 9: Stop.

3.1.2 HR10 heuristic

The basic idea in this heuristic 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. If the chosen job will be tardy, do not schedule the job, rather choose the next job that has the least processing time until all the jobs have been scheduled. The HR10 heuristic steps are outlined below:

HR10 heuristic steps

Step 0: Initialise
Job_SetA=[J_1, J_2,…,J_n]; The set of given jobs
Job_SetB=[0] The set of scheduled jobs
Job_SetC=[J_1, J_2,…,J_n] The set of unscheduled jobs
Job_SetD=[0] The set of available jobs at time t
Job_SetE=[0] The set of tardy jobs
t = $\min_{J_i \in Job\_SetA} r_i$ (i.e. the minimum ready time of all jobs)
Step 1: At time t
Update Job_SetD with jobs for which r_i<=t
If r_i <= t then Job_SetD = [J_i]
Step 2: Choose the job with the smallest processing time among the jobs that have arrived at time t from Job_SetD
Step 3: If the job chosen in Step 2 will be tardy, remove it from Job_SetD, add the job to Job_SetE, and go to Step 4, otherwise go to Step 5.
Step 4: If at time t Job_SetD is empty, go to Step 6, otherwise go to Step 2
Step 5: Add the job chosen in Step 2 to Job_SetB and remove the same job from Job_SetC. Compute start time S_i = t and completion time C_i = S_i + p_i
Step 6: Compute new time as: t = max (C_i, $\min_{J_i \in Job\_SetC} r_i$);maximum of the completion time or the minimum ready time of the remaining unscheduled jobs.
Step 7: If Job_SetC is not empty, go back to Step 1, otherwise proceed to Step 8
Step 8: Append Job_SetB with Job_SetE in the increasing order of processing times of the jobs in Job_SetE.
Step 9: Job_SetB is the schedule required.
Step 10: Stop.

3.2 Selected solution methods

Many researchers have explored many variants of the bi-criteria scheduling problems. Many of them have adopted dynamic programming, linear programming, or integer linear programming methods, while others have proved the NP-Hard nature of the problems [6]. The literature on the bi-criteria problem considered in this paper (the problem of simultaneously minimising total completion time and the number of tardy jobs with release dates on a single machine - $1 \mid r_i \mid (\alpha \sum_{i=1}^{n} C_i + \beta \sum_{i=1}^{n} U_i)$) appears to be sparse. To the best of our knowledge, only Oyetunji [17] and Oyetunji and Oluleye [18] have proposed solution methods (heuristics) for solving this problem with respect to the job characteristics, shop environments, and criteria under consideration. This is probably due to the fact that the problem is NP-Hard in the strong sense.

Therefore, to compare the performances of the proposed heuristics, a heuristic (HR7) was selected from Oyetunji [17] as well as a branch and bound (BB) method implemented by Oyetunji and Oluleye [19]. The HR7 heuristic systematically combines the AEO and EOO heuristics. (For details of the HR7 heuristic and the BB method, see Oyetunji [17] and Oyetunji and Oluleye [19], page 149, respectively.) The HR7 outperformed the best (HR6) of the heuristics proposed by Oyetunji and Oluleye [18], hence its selection for evaluation.

4. DATA ANALYSIS

For the purpose of proper comparison, the same problems solved by Oyetunji and Oluleye [20] were solved. These consist of 50 problems for each of 22 different problem sizes ranging from 3 to 500 jobs (Fig. 1). In all, 1,100 randomly generated problems were solved. The processing times of the jobs were randomly generated (using the random number generator in Microsoft Visual Basic 6.0) with values ranging between 1 and 100 inclusive. Similarly, the ready times of the jobs were randomly generated with values ranging between 0 and $\sum_{i=1}^{n} P_i$ inclusive, and the due dates of the jobs were randomly generated with values ranging between ($r_i + p_i$) and ($r_i + 2*p_i$) inclusive. The problem data was archived for future reference, and can be made available on request.

A program was written in Microsoft Visual Basic 6.0 to apply the solution methods (HR7, HR9, HR10, and BB) to the problems generated. The program computes the value of the normalised linear composite objective function ($F = 0.5 * \sum_{i=1}^{n} C_i + 0.5 * \sum_{i=1}^{n} U_i$) obtained by each solution method for each problem. The data (values of the normalised linear composite objective function and execution time) was exported to Statistical Analysis System (SAS version 9.1) for detailed analysis. SAS is a very versatile statistical package, and was employed to enable credible conclusions to be drawn from the results. The hardware used for the experiment was a 1.73 GHz T2080 Intel CPU with 1024 MB of RAM.

The general linear model (GLM) procedure in SAS was used to compute the mean value of the normalised linear composite objective function for each problem size (50 problem instances were solved under each problem size) and by solution methods. The test of means was also carried out using the GLM procedure to determine whether or not the differences observed in the mean value of the normalised linear composite objective function obtained by various solution methods are statistically significant. Note that the methodology of Oyetunji [16] was used to obtain the normalised linear composite objective function. Therefore, the normalised linear composite objective function shown in the results is dimensionless.

5. RESULTS AND DISCUSSION

The values of the normalised linear composite objective function obtained by the various solution methods indicate the effectiveness of the solution methods, whereas their efficiency is measured by the execution time (seconds). The mean values of the normalised linear composite objective function obtained by the various solution methods and problem sizes considered are shown in Table 1. As expected, the branch and bound (BB) method gave the minimum mean value of the normalised linear composite objective function (indicating the best performance with respect to effectiveness) for all the problem sizes considered. The HR7 heuristic gave the mean value of the normalised linear composite objective function that was closest to that of the BB method when the number of jobs was less than 30. This was closely followed by the HR10 heuristic. However, when the number of jobs was equal to or greater than 30, the HR10 heuristic gave the mean value of the normalised linear composite objective function that was closest to that of the BB method. This was closely followed by the HR7 and HR9 heuristics in that order (Table 1).

The results presented in Table 1 were subjected to statistical test to determine whether or not the differences observed in the mean values of the normalised linear composite objective function obtained from the various solution methods were significant; the results are summarised in Tables 2-5. The differences in the mean value of the normalised linear composite objective function obtained from BB, HR7, HR10, and HR9 solution methods are not significant (indicating competitive performances among BB, HR7, HR10, and HR9 solution methods) at the 5% level for problems ranging from 3 to 7 jobs (Table 2). However, under the same job configuration, the mean value of the normalised linear composite objective function obtained from the HR6 heuristic is significantly different from that of the BB, HR7, HR10, and HR9 solution methods (Table 2), indicating a poor performance compared with the other four methods.

For problems ranging from 8 to 30 jobs, the differences in the mean value of the normalised linear composite objective function obtained from BB, HR7, and HR10 solution methods are not significant (indicating competitive performances among BB, HR7, and HR10), whereas the mean values of the normalised linear composite objective function obtained from the HR9 heuristic are significantly different from those of the BB, HR7, and HR10 solution methods at the 5% level (Table 3), indicating poor performance compared with the other three methods.

When the number of jobs ranges between 30 and 200 inclusive, the differences in the mean value of the normalised linear composite objective function obtained from BB and HR10 are not significant at the 5% level (Table 4), indicating competitive performances between BB and HR10. Under the same problem loading, the mean value of the normalised linear composite objective function obtained from the HR7 and HR9 heuristics are significantly different from that of the BB and HR10 solution methods at 5% level (Table 4), indicating a poor performance compared with the other two. Table 5 shows that the mean value of the normalised linear composite objective function obtained from the HR10 heuristic is significantly different from that of the HR7 and HR9 heuristics at 5% level for problems ranging from 300 to 500 inclusive, indicating better performance than the HR7 and HR9 heuristics). The BB procedure could not be applied to problems involving more than 200 jobs, as it could not obtain solutions within the maximum allowable time of 60 minutes due to its obvious implicit enumeration characteristic.

The ratio of the values of the normalised linear composite objective function obtained from the HR7, HR9, and HR10 to that of the BB (HR7/BB, HR9/BB, and HR10/BB) was computed for the various problem sizes; the results are shown in Fig. 1. The closeness of the HR7 performance to that of BB when the number of jobs is less than 30 is obvious. And the closeness of the HR10 performance to that of BB when the number of jobs is equal to or greater than 30 is also obvious (Fig. 1). As the number of jobs increases, the performance of the HR10 heuristic gets better and closer to that of the BB method (Fig. 1).

The mean values of the execution time (seconds) taken to obtain results by the various solution methods for the various problem sizes are shown in Table 6. As expected, the BB method took longer (and thus was slower) than the HR7, HR9, and HR10 heuristics (Table 6). The HR7, HR9, and HR10 heuristics were not consistently faster than each other under the same problem loading (Table 6).

The t-tests carried out indicate that HR7, HR9, and HR10 heuristics were significantly faster than the BB for all the considered problem sizes (results not shown). Also, both HR7 and HR10 are significantly faster than HR9 when the number of jobs exceeds 30.

Problem Size	Mean of the normalised composite objective function BB	HR7	HR9	HR10
3x1	0.2868	0.3037	0.3169	0.3161
4x1	0.3862	0.3911	0.3926	0.3920
5x1	0.3616	0.3631	0.4228	0.3989
6x1	0.3905	0.3962	0.4253	0.3976
7x1	0.3971	0.4020	0.4835	0.4302
8x1	0.4003	0.4247	0.5221	0.4357
9x1	0.4215	0.4379	0.5116	0.4577
10x1	0.4225	0.4369	0.5049	0.4500
12x1	0.4180	0.4425	0.5303	0.4700
15x1	0.4555	0.4725	0.5446	0.4830
20x1	0.4755	0.4874	0.5608	0.4884
25x1	0.4690	0.4821	0.5722	0.4845
30x1	**0.4612**	**0.4772**	**0.5670**	**0.4727**
40x1	0.4744	0.4812	0.5704	0.4808
50x1	0.4857	0.4975	0.5854	0.4921
100x1	0.4851	0.4972	0.5921	0.4867
120x1	0.4853	0.5000	0.5986	0.4934
140x1	0.4830	0.4930	0.5965	0.4843
200x1	0.4804	0.4932	0.5971	0.4855
300x1	-	0.4945	0.5999	0.4846
400x1	-	0.4934	0.6013	0.4835
500x1	-	0.4928	0.6022	0.4833

Sample size = 50

Table 1: Mean of the normalised composite objective function by solution methods and problem sizes

	Heuristics			
Heuristics	BB	HR7	HR9	HR10
BB	-	X	X	X
HR7	X	-	X	X
HR9	X	X	-	X
HR10	X	X	X	-

Note * indicates significant result at 5% level; Sample size = 50
X indicates non-significant result at 5% level
\- indicates not necessary

Table 2: Test of means (probability values) of normalised composite objective function for $3 \leq n \leq 7$ problems

	Heuristics			
Heuristics	BB	HR7	HR9	HR10
BB	-	X	<0.001*	X
HR7	X	-	<0.001*	X
HR9	<0.001*	<0.001*	-	<0.001*
HR10	X	X	<0.001*	-

Note * indicates significant result at 5% level; Sample size = 50
X indicates non-significant result at 5% level
\- indicates not necessary

Table 3: Test of means (probability values) of normalised composite objective function for $8 \leq n < 30$ problems

Heuristics	BB	Heuristics HR7	HR9	HR10
BB	-	<0.001*	<0.001*	X
HR7	<0.001*	-	<0.001*	X
HR9	<0.001*	<0.001*	-	<0.001*
HR10	X	X	<0.001*	-

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

Table 4: Test of means (probability values) of normalised composite objective function for $30 \leq n \leq 200$ problems

Heuristics	HR6	Heuristics HR7	HR9	HR10
HR7	<0.001*	-	<0.001*	<0.001*
HR9	<0.001*	<0.001*	-	<0.001*
HR10	<0.001*	<0.001*	<0.001*	-

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

Table 5: Test of means (probability values) of normalised composite objective function for $300 \leq n \leq 500$ problems

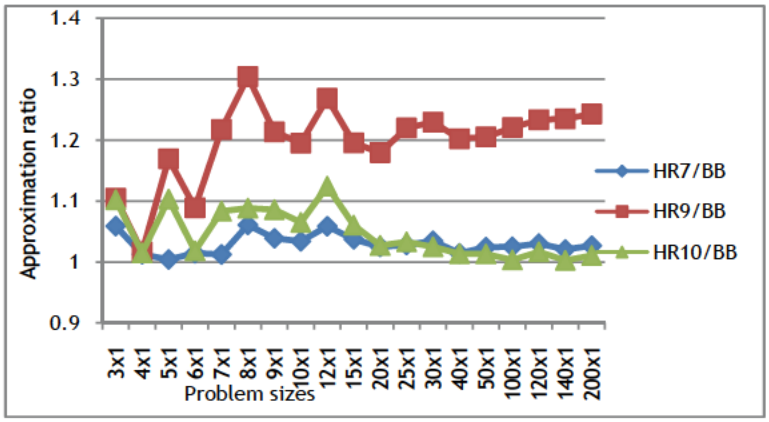

Figure 1: Approximation ratio of the composite objective function by problem sizes

Problem size	Mean of execution time (seconds)			
	BB	HR7	HR9	HR10
3x1	0.8791	0.0021	0.0027	0.0013
4x1	0.6761	0.0030	0.0027	0.0034
5x1	0.8009	0.0041	0.0042	0.0039
6x1	1.0627	0.0051	0.0044	0.0044
7x1	1.5514	0.0067	0.0066	0.0070
8x1	1.8374	0.0083	0.0071	0.0074
9x1	2.2335	0.0102	0.0073	0.0088
10x1	2.7916	0.0122	0.0119	0.0112
12x1	3.0853	0.0175	0.0158	0.0158
15x1	3.4220	0.0247	0.0227	0.0237
20x1	7.4707	0.0432	0.0433	0.0410
25x1	8.9151	0.0650	0.0665	0.0623
30x1	**12.4608**	**0.0912**	**0.0969**	**0.0874**
40x1	18.6614	0.1579	0.1636	0.1519
50x1	34.6055	0.2388	0.5854	0.2577
100x1	102.5728	0.9476	1.1157	1.0239
120x1	140.3210	1.3599	1.6199	1.4872
140x1	270.2090	1.8667	2.3984	2.0297
200x1	480.5320	3.7965	4.8638	4.3243
300x1	-	8.2500	11.5780	10.4256
400x1	-	14.6081	23.1784	19.3045
500x1	-	21.5290	39.0791	27.7815

Sample size = 50

Table 6: Mean of execution time (seconds) by solution methods and problem sizes

6. CONCLUSION

This paper has discussed the bi-criteria scheduling problem of simultaneously minimising the total completion time and the number of tardy jobs with release dates on a single machine. In view of the NP-Hard nature of the problem, two heuristics (HR9 and HR10) were proposed for solving this problem, and compared with both an earlier heuristic (HR7) that was proposed for the same bi-criteria problem, and a branch and bound (BB) method. Results show that the HR7 heuristic outperformed the HR10 heuristic when the number of jobs was fewer than 30. However, for jobs ranges from 30 to 500 jobs inclusive, the HR10 heuristic outperformed the HR7 heuristic. In terms of efficiency, the HR7 and HR10 heuristics were both faster than the BB method.

Therefore, based on performance, the HR7 heuristic is recommended for the bi-criteria problem involving fewer than 30 jobs, while the HR10 heuristic is recommended for the bi-criteria problems involving 30 or more jobs.

7. REFERENCES

[1] **Abhyuday, D., Yung-Nein, Y., & Supphasak, P.** 2003. On using intelligent scheduling for multi-criteria optimisation in a PC assembly model. In *Proceedings of the 31st International Conference on Computers and Industrial Engineering*, San Francisco, California, February, 2003.

[2] **Assayad, I., Girault, A., & Kalla, H.** 2004. A bi-criteria scheduling heuristic for distributed embedded systems under reliability and real-time constraints. *International Conference on Dependable Systems and Networks (DSN 2004), 28 June*

- 1 July 2004, Florence, Italy, Proceedings. IEEE Computer Society 2004, pp. 347-356.

[3] **Benoit, A., Rehn-Sonigo, V., & Yves, R**. 2007. Multi-criteria scheduling of pipeline workflows. In *Proceedings of the 2007 IEEE International Conference on Cluster Computing, 17-20 September 2007*, Austin, Texas, USA. IEEE 2007, pp. 515-524.

[4] **French, S.** 1982. *Sequencing and scheduling*. Ellis Horwood Limited.

[5] **Geyik, F.** 2002. A hybrid-framework for the multi-criteria production scheduling. In *Proceedings of 2rd International Conference on Responsive Manufacturing (ICRM'2002)*, University of Gaziantep, Turkey, 26-28 June, pp. 782-788.

[6] **Hoogeveen, J.A.** 2005. Multicriteria scheduling. *European Journal of Operational Research*, 167(3), pp. 592-623.

[7] **Johnson, S.M.** 1954. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics*, 1, pp. 61-68.

[8] **Peha, J.M**. 1995. Heterogeneous-criteria scheduling: Minimizing weighted number of tardy jobs and weighted completion time. *Journal of Computers and Operations Research*, 22(10), pp. 1089-1100.

[9] **Landa, J.S. & Burke, E.** 2004. A tutorial on multiobjective metaheuristics for scheduling and timetabling. In *Multiple Objective MetaHeuristics (edited by X. Gandibleux, M. Sevaux, K. Sorensen and V. T'Kindt), Springer lecture notes in economics and mathematical systems*, 2004.

[10] **Klusáček, D., Hana, R., Ranieri, B., Marco, P., & Gabriele, C.** 2008. Comparison of multi-criteria scheduling techniques. In *Thierry Priol, Sergei Gorlatch, Paraskevi Fragopoulou (eds), Grid computing achievements and prospects*, pp. 173-184. Springer, 2008.

[11] **Leon, V.J.** 2003. *Multi-objective local search: Scheduling*. Working paper, 2003.

[12] **Low, C., Yukling, Y., & Tai-His, W**. 2006. Modelling and heuristics of FMS scheduling with multiple objectives. *Computers & Operations Research*, 33, pp. 674–694.

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

[14] **Mehta, N.K., Jain, P.K., & Singh, A.** 2006. Multi criteria selective rerouting in job shop by swapping of dispatching rules. *SimTecT 2006 Simulation Conference: Challenges and Opportunities for a Complex and Networked World (SimTecT 2006)*, Melbourne, Australia, 29 May - 1 June, 2006.

[15] **Molnar, B.** 2005. Multi-criteria scheduling of order picking processes with simulation optimisation. *Periodica Polytechnica SER. TRANSP. ENG.*, 33(1–2), pp. 59–68.

[16] **Oyetunji, E.O.** 2009a. Mixed multi-objectives scheduling problems. Paper presented at the 3rd Workshop on Mathematical and Computational Methods in Biology and Medicine (Holistic Modeling and Trends in the Biogeosciences) held at the University of Cape Coast, Cape Coast, Ghana, 21–24 May, 2009.

[17] **Oyetunji, E.O.** 2009b. Truncation and composition of schedules: A good strategy for solving bicriteria scheduling problems. Paper presented at the International Conference on Mathematics and its Applications, University of Ghana, Legon, 16-19 June, 2009.

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

[19] **Oyetunji, E.O. & Oluleye, A.E.** 2008a. Heuristics for minimizing total completion time and number of tardy jobs simultaneously on single machine with release time. *Research Journal of Applied Sciences*, 3(2), pp. 147-152.

[20] **Oyetunji, E.O. & Oluleye, A.E.** 2008b. Hierarchical minimization of total completion time and number of tardy jobs criteria. *Asian Journal of Information Technology*, 7(4), pp. 157-161.

[21] **Oyetunji, E.O. & Oluleye, A.E.** 2008c. Heuristics for minimizing number of tardy jobs on single machine with release time. *South African Journal of Industrial Engineering*, 19(2), pp. 183–196.

[22] **Oyetunji, E.O. & Oluleye, A.E.** 2009. Evaluating solution methods to bicriteria scheduling problems. *Advanced Materials Research*, 62-64, pp. 577-584.

[23] **Stein, C. & Wein, J.** 1997. On the existence of schedules that are near-optimal for both makespan and total weighted completion time. *Operations Research Letters*, 21(3), pp. 115-122.

[24] **Petrovic, D., Alejandra, D., & Sania, P.** 2007. Decision support tool for multi-objective job shop scheduling problems with linguistically quantified decision functions. *Decision Support Systems*, 43(4), pp. 1527-1538.

[25] **Yves, C. & Emmanuel, J.** 2006. Multicriteria scheduling heuristics for GRIDRPC systems. *The International Journal of High Performance Computing Applications*, 20(1), pp. 61–76.

[26] **Van Veldhuizen, D. & Lamont, G.** 2000. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation*, 8(2), pp. 125-147.

[27] **T'Kindt, V., Billaut, J.-C. & Proust, C.** 2001. An interactive algorithm to solve bicriteria scheduling problems on unrelated parallel machines. *European Journal of Operational Research*, 135, pp. 42–49.

[28] **Nelson, R.T., Sarin, R.K. & Daniels, R.L.** 1986. Scheduling with multiple performance measures: The one-machine case. *Management Science*, 32, pp. 464–479.

[29] **Hoogeveen, J.A. & van de Velde, S.L.** 1995. Minimizing total completion time and maximum cost simultaneously is solvable in polynomial time. *Operations Research Letters*, 17, pp. 205–208.

[30] **Verma, S. & Dessouky, M.** 1998. Single-machine scheduling of unit-time jobs with earliness and tardiness penalties. *Mathematics of Operations Research*, 23, pp. 930–943.

[31] **Azizoglu, M., Kondakci, S. & Köksalan, M.** 2003. Single machine scheduling with maximum earliness and number tardy. *Computers and Industrial Engineering*, 45(2), pp. 257-268.

[32] T'**Kindt, V. & Billaut, J.-C.** 2006. Multicriteria scheduling: Theory, models and algorithms. 2nd edition. Springer, Berlin.