An International Journal of Optimization and Control: Theories & Applications
ISSN:2146-0957 eISSN:2146-5703
Vol.7, No.2, pp.167-176 (2017)
http://doi.org/10.11121/ijocta.01.2017.00328

RESEARCH ARTICLE

Identical parallel machine scheduling with nonlinear deterioration and multiple rate modifying activities

Ömer Öztürkoğlu

Department of Business Administration, Yasar University, Turkey
omer.ozturkoglu@yasar.edu.tr

ARTICLE INFO

Article History:
Received 05 January 2017
Accepted 23 February 2017
Available 15 July 2017

Keywords:
Parallel machine
Scheduling
Deterioration
Rate-modifying activity

AMS Classification 2010:
90B35, 68M20, 90C10

ABSTRACT

This study focuses on identical parallel machine scheduling of jobs with deteriorating processing times and rate-modifying activities. We consider nonlinearly increasing processing times of jobs based on their position assignment. Rate modifying activities (RMAs) are also considered to recover the increase in processing times of jobs due to deterioration. We also propose heuristics algorithms that rely on ant colony optimization and simulated annealing algorithms to solve the problem with multiple RMAs in a reasonable amount of time. Finally, we show that ant colony optimization algorithm generates close optimal solutions and superior results than simulated annealing algorithm.

1. Introduction

In the last two decades, time-dependent processing times of jobs in scheduling literature have received increasing attention. The awareness of human perspective on scheduling jobs or Total productive maintenance (TPM) on productivity lead researchers think out of the box. Although many studies have to included some type of uncertainties in scheduling or sequencing problems (see [1–3] for details) in literature, some of the issues have been restricted by assumptions so as to simplify the problems. Boudreau et al. [4] discussed some of these issues from the human perspective that labor and task times are assumed to be deterministic and predictable as if they are always available.

In scheduling problems, Gupta and Gupta [5] introduced a variable processing time of a job described by a polynomial function of its starting time to include some dynamic parameters of systems discussed by Gupta et al. [6] and some Russian papers (see [7] for details). Browne and Yechiali [8] also introduced the concept of deteriorating jobs of which their processing time increases as they await to be processed. For example, awaiting steel material in the inventory to be processed might corrode, a drop in the temperature of an ingot needs to be reheated, a delay in medical treatment. Several papers can be given as appropriate examples to deteriorating jobs as a linear function of processing time of a job [5,8,9]. Kunnathur and Gupta [10] proposed a model with piecewise increasing processing times. Mosheiov [11] presented non-linear deterioration according to a job-dependent step function. Ozturkoglu and Bulfin [12] proposed a position-based, nonlinear increasing function of processing time of a job. We also implement a non-linear deterioration in our models. Up to now, all studies have studied on single machine scheduling problem. Additional literatures about time-dependent processing times in a single machine scheduling can be seen in [13] and [14]. Lodree et al. [15] also presented a detailed survey study about sequence-dependent studies from the perspective of human factors.

Mosheiov [11] formulated parallel, multiple machine scheduling problem with a job-dependent step deterioration as an integer program and proposed a heuristic algorithm for the problem. Chen [16] also studied on parallel machine scheduling problem that minimizes total completion time with a consideration of a simple linear deterioration. The author showed that this problem is NP-complete in the ordinary sense, not in the strong sense even with a fixed number of machines. Mosheiov [17] studied parallel, identical machines for makespan minimization of deteriorating jobs with simple linear function of their starting times. Mosheiov [17] showed that the multi-machine scheduling problem is NP-complete by reduction to the single-machine problem and presented an asymptotically optimal heuristic for minimization of makespan. For studies published by 2004, some additional discussions about multi-machine scheduling problems and their NP-completeness can be seen in [14]. Kang and Ng [18] presented a fully-polynomial time approximation scheme (FPTAS) for scheduling linearly deteriorating jobs on m identical parallel machines with the objective of makespan. Ji and Cheng [19] also proposed another FPTAS for parallel-machine total completion time problem with linearly deteriorating jobs. Ji and Cheng [20] developed FPTAS for parallel-machine scheduling of simple linear deteriorating jobs with the objective of minimizing makespan, total completion time and total machine load. Ji and Cheng [20] also showed the makespan problem is strongly NP-hard for the fixed number of machines. All of the above studies considered deteriorating jobs in which the processing time of a job increases due to delay of processing of jobs, depreciation of machines or workers fatigue.

Lee and Leon [21] introduced the concept of rate-modifying activities to the scheduling literature. A rate-modifying activity (RMA) is an activity that affects and changes the speed or rate of the resource. Maintenance activities for machines and rest periods for workers can be given as examples to this concept. Lee and Lin [22] considered single-machine scheduling with the objectives of minimizing makespan, total completion time, total weighted completion time and maximum lateness. In their model, they evaluate the placement of fixed length RMA as well as sequencing of tasks. They proposed polynomial time algorithm for makespan and total completion time problems, and pseudo-polynomial time algorithms for several different objectives. Lee and Lin [22] also studied single machine scheduling problems with rate-modifying activities considering stochastic machine breakdown. In their model, if the RMA is scheduled before a breakdown, then processing times of jobs are reduced. If a breakdown occurs, then repair activity is applied then the resource works with its normal rates. They considered makespan, total completion time and maximum lateness as an objective function. Mosheiov and Sidney [23] developed an efficient polynomial algorithm that minimizes makespan for sequencing tasks with both learning and a RMA. All these studies have considered rate modifying activities on a single machine scheduling without consideration of the deteriorating jobs.

Lodree et al. [15] integrated two distinct concepts, deteriorating jobs and RMAs in scheduling models whose processing times are represented by linear increasing function of their starting times. In several studies, the single machine scheduling problem with deteriorating jobs and multiple RMAs is modeled under different objectives, such as minimizing makespan and total completion time [12,24–26]. In these studies, researchers applied position dependent, non-linear function of processing times for jobs. Additionally, Lee and Wu [27] consider deteriorating jobs with maintenance activities of scheduling jobs on parallel machines. They applied simple, linear deterioration of jobs. In their model, maintenance period is known in advance for each machine. They evaluated makespan for this problem considering both resumable and non-resumable cases. Recently, Dalfard and Mohammadi [28] developed a model for a multi-objective parallel machine scheduling problem with maintenance activity excluding deterioration in processing times. Authors also solved the problem by using simulated annealing and hybrid genetic algorithms. Cheng et al. [29] proposed an improved ant colony optimization algorithm for a parallel machine scheduling problem in which jobs are processed in batches. Wang and Wei [30] showed that an identical parallel machine scheduling problem with linear deterioration and rate-modifying activities can be solvable in polynomial time even the objectives are minimization of total absolute differences in both completion and waiting times. Wang et al. [31] also studied an identical parallel scheduling problem in which machines are deteriorated due to delaying maintenance activities that cause an increment in maintenance time. After a maintenance activity, processing times of jobs decrease. Authors proposed a polynomial time algorithm to solve total completion time for this scheduling problem. Yang and Yang [32] proposed two polynomial time algorithms for unrelated parallel machine scheduling problems with multiple-rate modifying activities.

They considered that processing times of jobs are constant until a rate-modifying activity is performed, afterwards it decreases with a constant rate. Yang et al. [33] developed a polynomial time algorithm to solve unrelated parallel machine scheduling problem with controllable processing times and rate-modifying activities. The cost function in their model comprises total completion time and total job compressions.

To the best of our knowledge, our model is the first to attempt to evaluate the optimal sequence of non-linearly deteriorating jobs and sequences of multiple rate-modifying activities at identical parallel machines. We use fixed length RMA time and position-based, nonlinear deterioration similar to the [12]. The remainder of the paper is organized as follows. In the next section, we present a mathematical model for this problem. In the later sections, we implement ant colony optimization (ACO) and simulated annealing (SA) algorithms to solve this problem. Last, we solve the models and compared their performances.

2. Mathematical model

In our model, we schedule a set of n jobs as $J = \{J_1, J_2, ..., J_n\}$ and at most b number of RMAs on identical R set of parallel machines, $R = \{R_1, R_2, ..., R_m\}$. Jobs are non-preemptive and each job is assigned to only one machine. We assume that jobs and machines are available at the beginning of the scheduling, and jobs are available when a machine is available for processing. A RMA can be given only after a job is completely processed at a machine (jobs are non-resumable). The rest of the model parameters are given in Table 1.

Initial processing time of jobs (p_j) are the same at any machines. Deteriorating processing times of jobs are nonlinear, increasing functions of p_j based on the positions of assigned jobs. If a job is assigned to the i^{th} position after the beginning of the schedule or a given RMA, p_{ji} is formulated in the equation (1) defined by [12]. Additionally, we assume that jobs revert to their base processing time as soon as a RMA is performed.

$$p_{ji} = (1+\alpha)^{i-1} \cdot p_j \tag{1}$$

Table 1. The mathematical model parameters..

i	the position number of scheduled jobs at machines
j	the index number of jobs
k	the position number where an RMA is given before processing a job at the k^{th} position
m	the index number of machines
α	constant deterioration rate of processing time of jobs, $0 < \alpha \leq 1$
q	fixed period of time to perform an RMA
p_j	the initial (base) processing time of job j at identical machines
p_{ji}	is the processing time of deteriorated job j at position i
x_{ijkm}	1, if job j is assigned to the i^{th} position after an RMA at the k^{th} position on machine m, otherwise 0
y_{km}	1 if an RMA is assigned at position k on machine m
C_{im}	completion time of the job in position i on machine m

Hence, the developed integer programming (IP) model can ben described as followings.

$$\min \quad Z = C_{max} \tag{2}$$

subject to

$$C_{max} \geq C_{nm} \quad \forall m \in R \tag{3}$$

$$C_{1,m} = \sum_{j=1}^{n} p_{j1} \cdot x_{1j0m} \tag{4}$$

$$C_{im} = C_{(i-1)m} + \sum_{k=1}^{i}\sum_{j=1}^{n} p_{jk} \cdot x_{ij(i-k)m} + q_i \cdot y_{im}, \quad \forall i = 2, ..., n \quad \text{and} \quad \forall m \in R \tag{5}$$

$$\sum_{j=1}^{n}\sum_{k=0}^{i-1} x_{ijkm} = 1, \quad \forall i = 1, ..., n, \forall m \in R \tag{6}$$

$$\sum_{i=1}^{n}\sum_{k=0}^{i-1}\sum_{m \in R} x_{ijkm} = 1, \quad \forall j = 1, ..., n \tag{7}$$

$$x_{kjim} \leq y_{(i+1)m}, \quad \forall m \in R, \forall i = 1, ..., k-1, \forall j \in J, \forall k = 2, ..., n \tag{8}$$

$$\sum_{i+1}^{n} y_{im} \leq b, \quad \forall m \in R \tag{9}$$

$$x_{ijkm} \in \{0,1\}, y_{im} \in \{0,1\} \tag{10}$$

The equation (2) is the objective of minimizing makespan where $C_{max} = \max\{C_{z1}, C_{z2}, ..., C_{zm}\}$, maximum of the completion time of the last job z

in each machine m. This is formulated by Equation (3). Equations (4) shows the completion time of the jobs at the first position based on base processing times of jobs. Equation (5) calculates the completion time of jobs at the later positions considering nonlinear deterioration of processing times and RMA time if given. Equation (6) restricts that each positions on each machine can only take one job. Equation (7) shows that each job should be assigned only to one position on each machine. Equation (8) arranges the order of RMAs based on the scheduled jobs. To make this constraint clear, if a job is assigned to position 2 after given a RMA at the end of the first position at machine 1 ($x_{2j11} = 1$), then y_{21}, which represents that a RMA is given at the beginning of position 2 at machine 1, should be one. Equation (9) controls the maximum number of allowable RMAs in the sequence. Additionally, equations (10) are the binary constraints.

Mosheiov [11] showed that multi-machine scheduling with linear deterioration is NP-hard even for two machines, our problem is also NP-hard because this is an extended model with the decision of optimal sequence of RMAs in a optimal sequence of jobs of which their processing time is nonlinearly deteriorated. Lee and Wu [27] also claimed that the scheduling problems with linearly deteriorating jobs and maintenance period are also NP-hard.

3. Ant colony optimization algorithm

Because our problem is also in NP-hard class, in this section we propose a unique ant colonoy optimization (ACO) algorithm which is originally developed by [34]. Sankar et al. [35] studied decentralized distributed scheduling problem in a parallel machine shop environment applying an ACO algorithm to the problem. Tkindt et al. [36] proposed an ACO algorithm and a heuristic based on simulated annealing (SA) algorithm for two serial machine scheduling problem for minimizing both makespan and total completion time together. Alaykiran et al. [37] proposed an ACO algorithm to solve hybrid flow shop problems considering makespan as an objective. Arnaout et al. [38] and Arnaout [39] proposed an ACO algorithm to non-preemptive, unrelated parallel machine scheduling problem with machine- and job sequence-dependent setup times. They compared the algorithm with tabu search algorithm and one of the existing heuristics in literature. They showed that ACO algorithm outperformed the other algorithms. Rossi and Boschi [40] developed a heuristic basis on a genetic algorithm (GA) and ACO for the flexible manufacturing systems. In their heuristic, GA and ACO co-evolve in parallel so as to improve the performance of the algorithm. Behnamian et al. [41] integrated three heuristics, ACO, SA and variable neighborhood search (VNS) to solve the parallel machine scheduling problem with sequence-dependent setup times for minimizing the makespan.

In our algorithms, we use a permutation based encoding which represents the sequence of job positions, split parameters and RMA parameters. There are $(m - 1)$ number of split parameters (s_i) in the encoding shows the position where jobs are distributed to machines. r_{im} represents the position of given maximum of b number of RMAs at machine m. Hence, the encoding can be shown as $\{J_1, J_2, ...J_n| s_1, s_2, ..., s_{m-1}|r_{11}, ..., r_{x1}|...|r_{1m}, ..., r_{xm}|\}$. Additionally, $0 \leq r_{11} \leq ... \leq r_{k1} \leq s_1 < ... < s_{m-1} \leq r_{1m} \leq ... \leq r_{xm}$. For example, let encoding scheme $\{3\,5\,8\,1\,4\,2\,9\,6\,7\,10\,|\,4\,|\,2\,|\,7\}$ represent a solution for a scheduling 10 jobs at parallel two machines with at most one RMA. Hence, jobs 3, 5, 8 and 1 are scheduled at machine 1 because s_1 is 1, other jobs are at machine 2. An RMA is scheduled after job 5 at machine 1 ($r_{11} = 2$ meaning that RMA is given after the second position at machine 1) and one RMA is given after job 9 at machine 2. If any one of the RMA factors is equal to 0 (start position) or the same as split factor, it means that RMA is not actually needed at that machine. The rest of the parameters for our ACO algorithm is given in Table 2.

We solve our problem with ACO in two stages: sequencing and assigning. In the sequencing stage, we allocate n jobs to the positions like assigning them to a single machine. In the assigning stage, we firstly split jobs into machines and allocate RMAs in each machine. The sequencing and assignment are based on the pheromone amounts on trails or positions and computed by probabilistically as in equation (11). After calculating probability of assigning next job, we select the job based on simple tournament selection applying under q_0 strategy. q_0 strategy is used in classical ACO to balance the exploration and exploitation. If a random number is greater than q_0, we use tournament selection, otherwise we select the job which has the maximum value of $(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}$. After constructing a full schedule, we select the split factors and then positions of RMAs.

Table 2. The model parameters for the ACO algorithm.

a	number of ants in population
T	number of iterations
$\tau_{ij}(t)$	the intensity of the pheromone trail on the path between jobs i and j at time t
$\eta_{ij}(t)$	the heuristic value (visibility) $1/p_j$ where p_j is the processing time of job j at time t
$\lambda_k^{s_i}(t)$	the intensity of the pheromone on positions k for split factors at time t
$\mu_k^{rim}(t)$	the intensity of the pheromone on positions k for RMA factors at time t
α	the relative importance of the pheromone trail
β	the relative importance of the visibility
Δ	addition of pheromone on trail ij or between positions kn at time t
ρ	evaporation factor, $0 < \rho < 1$
ξ	factor of the online pheromone update, usually $= 0.1$
τ_0	initial pheromone amount on all paths and positions
τ_{min}	minimum allowance of the pheromone amount on paths or positions
τ_{max}	maximum allowance of the pheromone amount on paths or positions

$$P_{ij}^{y}(t) = \frac{(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{l \in N_i^y} (\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}, \quad \forall y = 1, .., a \tag{11}$$

$$P_{sk}^{y}(t) = \frac{(\lambda_k)^{\alpha}}{\sum_{i \in N_s} (\lambda_k)^{\alpha}}, \quad \forall y = 1, .., a \tag{12}$$

$$P_{xk}^{y}(t) = \frac{(\mu_k)^{\alpha}}{\sum_{l \in N_x} (\mu_k)^{\alpha}}, \quad \forall y = 1, .., a \tag{13}$$

where, $P_{ij}^{y}(t)$ is the probability of selecting job j if job i is scheduled for ant y at time t. N_i^y is the neighborhood of ant y that comprises unscheduled jobs up to that time. N_s is the whole available positions for splitting, and N_x is the appropriate positions for placing an RMA. P_{sk}^{y} and P_{xk}^{y} are the probability of selecting positions for splitting and RMAs, respectively, and these only account for the pheromone amounts at the positions.

We also use two pheromone updating processes, local and global updates, in our ACO algorithm. In the local update, as soon as an ant constructs a full schedule including splits and RMAs they change the pheromone amount based on the equations (14), (15) and (16).

$$\tau_{ij}(t+1) = (1-\xi) \cdot \tau_{ij}(t) + \xi \cdot \tau \tag{14}$$

$$\lambda_k(t+1) = (1-\xi) \cdot \lambda_k(t) + \xi \cdot \lambda \tag{15}$$

$$\mu_k(t+1) = (1-\xi) \cdot \mu_k(t) + \xi \cdot \mu \tag{16}$$

In the global update, after all ants construct a solution, firstly some pheromone is evaporated, then ants contribute to the appropriate paths and positions based on the equations (17), (18) and (19).

The pheromone amount is added to ij path if jobs i and j are scheduled consecutively in the sequence of a machine. After global updates, if the pheromone amount on trail or at positions exceeds maximum level or goes below the minimum level, we set the pheromone amounts to the closest limit.

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \sum_{y=1}^{a} \Delta_y \tag{17}$$

$$\lambda_k(t+1) = \rho \cdot \lambda_{ij}(t) + \sum_{y=1}^{a} \Delta_y \tag{18}$$

$$\mu_k(t+1) = \rho \cdot \mu_{ij}(t) + \sum_{y=1}^{a} \Delta_y, \tag{19}$$

where, $\Delta_y = \max\{C_{yz1}, C_{yz2}, ..., C_{yzm}\}$, the maximum of completion time of the last job in each machine based on the given solutions by ant y.

The pseudo-code for our ACO algorithm is explained as in the followings. In this algorithm, we implement a local search procedure (LS1) to improve the solution of an ant at every g number of iterations.

In this local search, if base processing time of a randomly chosen job is larger than the the scheduled job at its previous position, we simply remove that job in the sequence of the machine and insert this job either to the first position or any position after a given RMA. Hence, we aim to decrease the amount of deterioration by performing the local search.

```
Initialize parameters, pheromone amounts, and ants
Do{ until termination criterion is met.
    Do { for all ants
        Construct a full schedule, including splits and RMAs using tournament selection
        Evaluate the makespan of an ant
        If an ant is better than the best ant, update best ant
        Otherwise, at each $g$ iterations do a local search (LS1) on the ant, then compare it with the best ant again
        Do online pheromone update after each ant
        Do global pheromone update after all ants controlled by ($\tau_{min}$, $\tau_{max}$)
    }
}
```

4. Simulated annealing algorithm

Kirkpatrick et al. [42] proposed an iterative, stochastic, neighbor-based search method based on the analogy of heating and cooling of materials. Koulamas [43] developed a polynomial decomposition heuristic to minimize total tardiness on parallel machines. They embedded SA algorithm into their heuristic to do local search by swapping jobs. Park and Kim [44] suggested SA and TS algorithms that minimize order holding costs for scheduling these orders on identical parallel machines with ready times and due date. TS, SA and GA techniques are implemented to the scheduling problems with non-preemptable jobs on identical parallel machines [45]. Hindi and Mhlanga [46] developed SA and steepest descent algorithms to solve the makespan minimization problem of jobs having simple, linear and general linear deterioration on identical parallel machines. Kim et al. [47] proposed a SA algorithm to minimize total tardiness for scheduling jobs on unrelated parallel machines with sequence-dependent setup times. As discussed before, SA is also implemented to the parallel machine scheduling problem in studies [36, 41].

We use the same encoding that we discussed in the previous section in our SA algorithm. In this study, we modified the canonical SA algorithm by embedding two local searches so as to improve the solutions. The first one is the LS1, which is explained in the previous section. The second local heuristic (LS2) considers whole neighborhood of the scheduled jobs in the machine. It utilizes simple swap operator. Hence, LS2 swaps all jobs in a machine and if it finds a better solution than the current one, it updates the current solution. Additionally, the split factors and RMA factors are randomly chosen at the beginning of each iteration. The pseudo-code for this algorithm is as in the following.

```
Initialize parameters and randomly generate an initial solution
Do{ until termination criterion is met.
    Do { for all stages
        Move the split and RMA factors randomly for the schedule
        Implement a local search (LS2) for the given split and RMA factors
        If the temporary solution is better than the current, update the current solution
        Otherwise accept the non-improving solution probabilistically
        If the current solution is better than the best, update best
        If there is no improvement on the best solution for $g$ number of iterations, implement a local search (LS1) on the best solution.
    }
}
```

The probability of accepting non-improving solutions is calculated using Equation (20) defined by [42]. Based on this equation, probability of accepting non-improving solutions is higher at the higher temperatures, and it decreases as the temperature decreases.

$$P = \exp\left(-\frac{\Delta E}{c \cdot T'}\right), \tag{20}$$

where, ΔE is the difference of the makespan between the temporary solution and the current solution. c is a Boltzmann constant, and T is the current temperature level. The other model parameters used in the SA algorithm are: T_0 is initial temperature, π is the cooling parameter ($0 < \pi < 1$) and ν is the number of moves at each stage where the local search is applied.

5. Simulation results

In this study, we only work on identical two parallel machines ($m = 2$). However, our models are capable of considering multiple machines. We solve our parallel machine scheduling problem for 10, 20, 30 and 50 jobs to investigate the effects of heuristics. We randomly generate processing times of jobs based on a uniform distribution that resides between 1 and 160. We use 10 different data set for each job set separately. We assume that RMA time is 5 unit time, and deterioration rate is selected 0.08. After a simple tuning operation by trial-and-error, we decide to have 15 ants in the population in ACO algorithm. The evaporation rate is 0.70 so as to enhance more diversity. The other ACO parameters are $\alpha = 1$, $\beta = 2$, $\xi = 0.1$, $q_0 = 0.1$, $\tau_{min} = 0.1$ and $\tau_{max} = 0.49$. Last, the algorithm is terminated after 1000 iterations. The SA algorithm is simulated for 500 iterations and 5 stages at each iteration. Initial temperature is selected 100, the cooling parameter is 0.99, and the Boltzmann constant is 0.4. Finally, the mathematical model (MM) is coded and solved in IBM ILOG OPL CPLEX Optimization Studio 12.6. Because of the NP-Hardness of the problem, solution time is restricted to 3 hours. If a global optimum integer solution is not obtained by 3 hours, we used the best integer objective value found for the comparisons. The ACO and SA algorithms are coded with JAVA. They all are run on Intel(R) Core(TM) i5 3.5 GHz PC with 2GB ram. Additionally, we also conduct 10 replications on the ACO and SA algorithms for each data set. We use the simple average of the ten replications for the comparisons.

Tables 3, 4 and 5 show the detailed solutions of the algorithms. In Table 3, “Gap(%)” represents absolute tolerance on the gap between the best integer objective and the objective of the best node remaining when the model is terminated. As seen in this table, the gap increases as the number of jobs increases, especially for 50 jobs due to NP-hardness of the problem. “Best found solution” column in Tables 4 and 5 show the best solution obtained by ACO and SA, respectively at the end of iterations. “CPU time” in these tables is the computational time that algorithms spend to obtain the solution over iterations. The average computational times of ACO algorithm for 10-, 20-, 30- and 50 job-problems are 0.6, 2, 4.8 and 15 seconds. For SA, these are 0.2, 3, 7.8 and 95.8 seconds. As seen, SA algorithm takes more time than ACO as the number of jobs increases due to intensive local search.

In order to compare solution quality of ACO and SA algorithms, we calculate their percentage error that represents the difference of their solutions with MM as a basis on mathematical model solutions. These errors are given in Table 6. As seen in the table, for small number of jobs ACO provides close solutions to the mathematical model. The average error of ACO is about 4% for 30 jobs and it takes only 4.8 seconds. However, the SA algorithm does not provide as good solutions as the ACO algorithm. The average error of SA is about 7.7% for 30 jobs. In terms of computational time, even though SA is faster than ACO for 10 jobs, SA requires more time than ACO as the number of jobs increases. The reason of this observation is due to the intensive local search algorithm in SA.

6. Conclusion

In this study, we developed an integer programming model for parallel machine scheduling with deteriorating jobs and rate modifying activities. We consider non-linearly increasing function of processing times based on the sequence of the job. We also consider rate-modifying activities to recover the loss in processing times of the jobs. Because this problem is NP-hard, we proposed two meta-heuristic algorithms that rely on Ant colony optimization and Simulated annealing algorithms. In the ACO algorithm, we proposed different pheromone update schemes for the problem. We run our mathematical model and heuristics with two identical parallel machines for four different sets of jobs; 10, 20, 30 and 50. Results show that the ACO algorithm performs better than SA and generates close optimal solutions for with an average error of 0.7%, 1.6%, 4% and 8.8% for 10, 20, 30 and 50 jobs, respectively. In terms of computational time, ACO is also superior than SA as numbers of job increases. For future studies, the proposed ACO algorithm might be powered by an efficient local search algorithm to obtain closer solutions to the best found solutions.

References

[1] Mohring, R. and Rademacher, F., *An Introduction to Stochastic Scheduling Problems* . In: Neumann, K. and Pallaschke, D. (Eds.), Contributions to Operations Research, Springer, Berlin, (1985).

[2] Righter, R., *Stochastic Scheduling* . In: Skaked, M. and Shanthikumar, G. (Eds.), Academic Press, San Diego, CA, (1994).

[3] Pinedo, M., *Scheduling, Theory, Algorithms and Systems* . Prentice-Hall, Englewood Cliffs, NJ, (1995).

[4] Boudreau, J., Hopp, W., McClain, J., and Thomas, L., On the Interface Between Operations and Human Resources Management . *Manufacturing and Service Operations Management*, 5(3), 179–202, (2003).

[5] Gupta, J. and Gupta, S., Single Facility Scheduling with Nonlinear Processing Times . *Computers and Industrial Engineering*, 14, 387–393, (1988).

[6] Gupta, S., Kunnathur, A., and Dandapani, K., Optimal Repayment Policies for Multiple Loans . *OMEGA*, 15(4), 323–330, (1987).

[7] Tanaev, V., Gordon, V., and Shafransky, Y., *Scheduling Theory, Single-stage Systems* . Kluwer, Dordrecht, (1994).

[8] Browne, S. and Yechiali, U., Scheduling Deteriorating Jobs on a Single Processor . *Operations Research*, 38, 495–498, (1990).

[9] Gawiejnowicz, S. and Pankowska, L., Scheduling Jobs with Varying Processing Times . *Information Processing Letters*, 54(3), 175–178, (1995).

Table 3. Mathematical model solutions.

	Best Found Integer Solution				Gap (%)			
n	10	20	30	50	10	20	30	50
Data 1	536.85	459.62	1250.6	2526.53	0.00	0.01	0.01	1.13
Data 2	340.9	473.94	1545.4	2646.81	0.02	0.00	0.08	2.01
Data 3	306.92	466.66	1539.9	2648.31	0.00	0.10	0.01	5.08
Data 4	394.83	943.87	1153.8	2848.57	0.00	0.01	0.00	1.70
Data 5	437.03	248.17	1434.8	3010.16	0.01	0.00	0.00	4.43
Data 6	475.68	232.98	1991.3	2530.86	0.01	1.6	0.01	3.11
Data 7	495.38	1156.5	1646.00	2758.91	0.0	0.00	0.02	1.16
Data 8	360.28	857.80	1158.70	2271.40	0.03	0.01	0.01	6.85
Data 9	315.61	665.27	1768.8	2320.72	0.01	0.06	0.05	3.60
Data 10	453.61	885.32	1364.8	2581.11	0.00	0.00	0.01	1.20

Table 4. Ant colony optimization algorithm solutions.

	Best Found Solution				CPU Time (sec.)			
n	10	20	30	50	10	20	30	50
Data 1	541.32	469.06	1314.68	2758.19	0.6	2.0	4.8	15.0
Data 2	344.10	475.57	1605.71	2899.70	0.6	2.0	5.0	14.8
Data 3	310.13	469.94	1593.11	2682.29	0.6	1.8	4.8	14.4
Data 4	399.25	964.71	1208.12	3042.29	0.6	2.0	5.0	15.2
Data 5	443.52	253.84	1492.18	3212.44	0.6	2.0	4.8	14.8
Data 6	478.37	233.61	2041.39	2741.60	0.6	2.0	4.8	15.4
Data 7	497.35	1178.58	1718.31	2963.06	0.6	2.0	4.8	15.0
Data 8	365.52	878.28	1210.17	2503.02	0.6	2.0	4.8	14.8
Data 9	319.61	677.15	1841.23	2535.55	0.6	2.0	4.8	14.8
Data 10	455.71	906.78	1422.92	2808.14	0.6	2.0	4.8	15.4

Table 5. Simulated annealing algorithm solutions.

	Best Found Solution				CPU Time (sec.)			
n	10	20	30	50	10	20	30	50
Data 1	546.70	470.72	1309.87	2885.99	0.2	3.0	13.6	95.0
Data 2	345.06	480.00	1669.84	3186.85	0.2	3.0	14.0	96.0
Data 3	313.08	470.48	1673.91	2867.50	0.2	3.2	14.0	94.8
Data 4	408.04	991.29	1263.99	3216.51	0.2	3.0	13.8	95.8
Data 5	447.87	265.17	1534.22	3464.48	0.2	3.0	14.0	95.0
Data 6	496.52	236.13	2086.18	2940.05	0.2	3.0	13.8	95.6
Data 7	510.22	1202.72	1758.61	3189.91	0.2	3.0	13.8	95.6
Data 8	369.02	918.02	1282.63	2583.84	0.2	3.0	14.0	95.6
Data 9	336.81	705.75	1930.59	2723.22	0.2	3.0	13.8	95.6
Data 10	466.09	945.45	1461.22	2893.22	0.2	3.0	13.8	95.6

[10] Kunnathur, A. and Gupta, S., Minimizing the Makespan with Late Start Penalties Added to Processing Times in a Single Facility Scheduling Problem . *European Journal of Operational Research*, 47(1), 56–64, (1990).

[11] Mosheiov, G., Scheduling Jobs With Step-Deterioration ; Minimizing Makespan on a Single and Multi-Machine . *Computers and Industrial Engineering*, 28(4), 869–879, (1995)

[12] Ozturkoglu, Y. and Bulfin, R. L., A Unique Integer Mathematical Model for Scheduling Deteriorating Jobs with Rate-Modifying Activities on a Single Machine. *The International Journal of Advanced Manufacturing Technology*, 57(5-8), 753–762, (2011).

[13] Alidaee, B. and Womer, N., Scheduling with Time Dependent Processing Times: Review and Extensions. *Journal of the Operational Research Society*, 50(7), 711–721, (1999).

[14] Cheng, T., Ding, Q., and Lin, B., A Concise Survey of Scheduling with Time-Dependent Processing Times . *European Journal of Operational Research*, 152, 1–13, (2004).

[15] Lodree, E., Geiger, C., and Jiang, X., Taxonomy for Integrating Scheduling Theory and Human Factors:

Table 6. The percentage gap among MM, ACO and SA algorithms.

	ACO vs. MM				SA vs. MM			
n	10	20	30	50	10	20	30	50
Data 1	0.83	2.05	5.21	9.17	1.83	2.42	4.74	14.23
Data 2	0.94	0.34	3.90	9.55	1.22	1.28	8.05	20.4
Data 3	0.91	0.69	3.46	8.67	2.01	0.82	8.70	16.17
Data 4	0.83	2.21	4.71	6.80	3.35	5.02	9.55	12.92
Data 5	0.75	2.28	4.00	6.72	2.48	6.85	6.93	15.09
Data 6	0.57	0.27	2.52	8.33	4.38	1.35	4.76	16.17
Data 7	0.40	1.91	4.39	7.40	3.00	4.00	6.84	15.62
Data 8	0.81	2.39	4.44	10.20	2.43	7.03	10.70	13.76
Data 9	0.80	1.79	4.09	9.26	6.72	6.08	9.15	17.34
Data 10	0.46	2.42	4.26	8.80	2.75	6.79	7.06	12.09
Average	0.73	1.64	4.09	8.49	3.02	4.16	7.65	15.38

Review and Research Opportunities . *International Journal of Industrial Ergonomics*, 39, 39–51, (2009).

[16] Chen, Z., Parallel Machine Scheduling with Time Dependent Processing Times . *Discrete Applied Mathematics*, 70, 81–93, (1996).

[17] Mosheiov, G., Multi-machine Scheduling with Linear Deterioration. *Infor*, 36, 205–214, (1998).

[18] Kang, L. and Ng, C., A Note on a Fully Polynomial-Time Approximation Scheme for Parallel-Machine Scheduling with Deteriorating Jobs . *International Journal of Production Economics*, 109, 180–184, (2007).

[19] Ji, M. and Cheng, T., Parallel-Machine Scheduling with Simple Linear Deterioration to Minimize Total Completion Time . *European Journal of Operational Research*, 188, 341–347, (2008).

[20] Ji, M. and Cheng, T., Parallel-Machine Scheduling of Simple Linear Deteriorating Jobs . *Theoretical Computer Science*, 410, 3761–3768, (2009).

[21] Lee, C.-Y. and Leon, V., Machine Scheduling with Rate-Modifying Activity . *European Journal of Operational Research*, 128, 493–513, (2001).

[22] Lee, C.-Y. and Lin, C.-S., Single Machine Scheduling with Maintenance and Repair Rate-Modifying Activities . *European Journal of Operational Research*, 135, 495–513, (2001).

[23] Mosheiov, G. and Sidney, J., New Results on Sequencing with Rate Modification . *Information Systems and Operational Research*, 41(2), 155–163, (2003).

[24] Ozturkoglu, Y., A Bi-Criteria Single Machine Scheduling with Rate-Modifying-Activity. *Gazi University Journal of Science*, 26(1), 97–106, (2013).

[25] Kim, B. S. and Ozturkoglu, Y., Scheduling a Single Machine With Multiple Preventive Maintenance Activities And Position-Based Deteriorations Using Genetic Algorithms. *The International Journal of Advanced Manufacturing Technology*, 67(5-8), 1127–1137, (2013).

[26] Ozturkoglu, Y., An Efficient Time Algorithm for Makespan Objectives. *An International Journal of Optimization and Control: Theories & Applications (IJOCTA)*, 5(2), 75-80, (2015).

[27] Lee, W.-C. and Wu, C.-C., Multi-Machine Scheduling with Deteriorating Jobs and Scheduled Maintenance . *Applied Mathematical Modeling*, 32, 362–373, (2008).

[28] Dalfard, V. M. and Mohammadi, G., Two Meta-Heuristic Algorithms for Solving Multi-Objective Flexible Job-Shop Scheduling with Parallel Machine and Maintenance Constraints. *Computers & Mathematics with Applications*, 64(6), 2111–2117, (2012).

[29] Cheng, B., Wang, Q., Yang, S., and Hu, X., An Improved Ant Colony Optimization for Scheduling Identical Parallel Batching Machines With Arbitrary Job Sizes. *Applied Soft Computing*, 13(2):765–772, (2013).

[30] Wang, J.-B. and Wei, C.-M., Parallel Machine Scheduling With a Deteriorating Maintenance Activity And Total Absolute Differences Penalties. *Applied Mathematics and Computation*, 217(20), 8093–8099, (2011).

[31] Wang, J.-J., Wang, J.-B., and Liu, F., Parallel Machines Scheduling With a Deteriorating Maintenance Activity. *Journal of the Operational Research Society*, 62(10), 1898–1902, (2011).

[32] Yang, D.-L. and Yang, S.-J., Unrelated Parallel-Machine Scheduling Problems with Multiple Rate-Modifying Activities. *Information Sciences*, 235, 280–286, (2013).

[33] Yang, D.-L., Cheng, T., and Yang, S.-J., Parallel-Machine Scheduling With Controllable Processing Times and Rate-Modifying Activities to Minimise Total Cost Involving Total Completion Time and Job Compressions. *International Journal of Production Research*, 52(4), 1133–1141, (2014).

[34] Dorigo, M., Maniezzo, V., and Colorni, A., Positive Feedback as a Search Strategy . *Technical Report 91-016, Dip. Elettronica,Politecnico di Milano, Italy*, (1991).

[35] Sankar, S., Ponnambalam, S., Rathinavel, V., and Visveshvaren, M., Scheduling in Parallel Machine Shop: An Ant Colony Optimization Approach . *Industrial Technology, ICIT, IEEE Industrial Conference*, pages 276–280, (2005).

[36] Tkindt, V., Monmarche, N. Tercinet, F., and Laugt, D., An Ant Colony Optimization Algorithm to Solve a 2-Machine Bicriteria Flowshop Scheduling Problem . *European Journal of Operational Research*, 142, 250–257, (2002).

[37] Alaykiran, K., Engin, O., and Doyen, A., Using Ant Colony Optimization to Solve Hybrid Flowshop Scheduling Problems . *International Journal of Advanced Manufacturing Technology*, 35, 541–550, (2007).

[38] Arnaout, J.-P., Musa, R., and Rabadi, G., Ant Colony Optimization Algorithm to Parallel Machine Scheduling Problem with Setups . *4th IEEE Conference on Automation Science Engineering, Washington DC, USA*, pages 578–582, (2008).

[39] Arnaout, J.P. and Rabadi, G. and Musa, R. A Two-Stage Ant Colony Optimization Algorithm to Minimize the Makespan on Unrelated Parallel Machines with Sequence-Dependent Setup Times . *Journal of Intelligent Manufacturing*, 21(6), 693-701, (2010).

[40] Rossi, A. and Boschi, E., A Hybrid Heuristic to Solve the Parallel Machines Job-shop Scheuling Problem . *Advances in Engineering Software*, 40, 118–127, (2009).

[41] Behnamian, J., Zandieh, M., and Ghomi, S., Parallel-Machine Scheduling Problems with Sequence-Dependent Setup Times using an ACO, SA and VNS Hybrid Algorithm . *Experts Systems with Applications*, 36, 9637–9644, (2009).

[42] Kirkpatrick, S., Gelatt, C., and Vecchi, M., Optimization by Simulated Annealing . *Science*, 220, 671–680, (1983).

[43] Koulamas, C., Decomposition and Hybrid Simulated Annealing Heuristics for the Parallel-Machine Total Tardiness Problem . *Naval Research Logistics*, 44, 105–125, (1997).

[44] Park, M.-W. and Kim, Y.-D., Search Heuristics for a Parallel Machine Scheduling Problem with Ready Times and Due Dates . *Computers and Industrial Engineering*, 33(3-4), 793–796, (1997).

[45] Józefowska, J., Mika, M., Różycki, R., and Waligóra, G., Local Search Metaheuristics for Discrete-Continuous Scheduling Problems . *European Journal of Operational Research*, 107, 354–370, (1998).

[46] Hindi, K. and Mhlanga, S., Scheduling Linearly Deteriorating Jobs on Parallel Machines: A Simulated Annealing Approach . *Production Planning and Control*, 12(1), 76–80, (2001).

[47] Kim, D.-W., Kim, K.-H., Jang, W., and Chen, F., Unrelated Parallel Machine Scheduling with Setup Times Using Simulated Annealing . *Robotics and Computer Integrated Manufacturing*, 18(3-4), 223–231, (2002).

Ömer Öztürkoğlu *is an Assistant Professor in Business Administration at Yasar University, Izmir, Turkey. He teaches related courses to Logistics Facilities, Warehousing, and Production and Operations Analysis. In general, his research interests are production and operations systems analysis and design.*