IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received December 22, 2019, accepted February 18, 2020, date of publication February 28, 2020, date of current version March 13, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977234

An Exact Method and Ant Colony Optimization for Single Machine Scheduling Problem With Time Window Periodic Maintenance

AMMAR A. QAMHAN, AREF AHMED, (Member, IEEE), IBRAHIM M. AL-HARKAN, AHMED BADWELAN, ALI M. AL-SAMHAN, AND LOTFI HIDRI

Department of Industrial Engineering, King Saud University, Riyadh 11451, Saudi Arabia

Corresponding author: Ibrahim M. Al-Harkan (imalhark@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research at King Saud University through the Research Group under Grant RG-1439-56.

ABSTRACT This paper considers a time window periodic maintenance strategy with different duration windows and job scheduling activities in a single machine environment. The aim is to minimize the number of tardy jobs through the integration of production scheduling and periodic maintenance intervals. A mixed-integer linear programming model (MILP) is proposed to optimize small-sized test instances. Furthermore, an ant colony optimization (ACO) algorithm is developed to solve larger sized test instances. Subsequently, to measure the efficiency of the solutions obtained by ACO, Moore's algorithm is also developed to benchmark with ACO. To test the efficiency and the effectiveness of the ACO algorithm, a set of data for small and large sized problems was generated in which several parameters were adopted and then ten replicates were solved for each combination. The small sized instances were solved by the MILP. Then, the results obtained showed that the proposed ACO was able to obtain the exact solutions within reasonable CPU times, thus, it outperformed the CPLEX solver with respect to CPU. The large sized instances were solved by the Moore's algorithm and compared to ACO. Then, the results obtained showed that the ACO outperforms Moore's algorithm for all the instances tested. It can be concluded that the developed ACOis very efficient and effective in solving the problem considered in this paper.

INDEX TERMS Scheduling, MILP, single machine, periodic Maintenance, ant colony.

I. INTRODUCTION

Production scheduling and production planning have a direct impact on an organization's performance [1]. Efficient organizations integrate production scheduling methodologies and maintenance service strategies to make the best use of machine availability and to ensure that the required quality levels are met, within the required time-frames. Previous studies in single machine environments scheduling assume that the production systems are always available at all time horizons and ignore the maintenance aspect. This, in turn, increases the probability of machine breakdowns [2]. Thus, to ensure the optimal availability of production systems, it is necessary to regulate the production schedule by considering intervention dates of maintenance actions. This paper will investigate a time window periodic maintenance strategy with different duration windows and job scheduling activities in a single machine environment, aiming to minimize the number of tardy jobs through the integration between production scheduling and periodic maintenance intervals. Two different algorithms, ACO and Moore's, will be tested to obtain a high-quality solution to the proposed problem.

The associate editor coordinating the review of this manuscript and approving it for publication was Amjad Gawanmeh.

II. LITERATURE REVIE

Studying the effects of different types of maintenance (periodic, preventive, among others) on different scheduling environments has received considerable attention for many years now. The periodic maintenance policy has been widely used by Chen [3]. Ma et al. and Sanlaville and Schmidt provided comprehensive reviews for cases where intervention dates are fixed and known at the beginning of a scheduling horizon [4], [5]. Cui and Lu recently studied flexible maintenance and release dates on a single machine environment with the objective of minimizing the makespan, and presented a mixed-integer linear programming model for small-sized test instances [6]. For small-to-medium sized problems, a heuristic ERD-LPT and a branch-and-bound algorithm were also proposed. The nature of the flexible maintenance methods in their study depends on the machine running time. To perform maintenance activity on the machine, the processing time on the machine must be less than or equal to T, where T is the time between two periodic maintenances. Ángel-Bello et al. proposed another mixed-integer linear model for minimizing the makespan [7]. Hou et al. considered the problem of a single machine with periodic consecutive maintenance-availability constraints, and sequence-dependent setup costs [8]. They also aimed to find a schedule with an appropriate maintenance strategy to minimize makespan, by proposing a partial maintenance model and incorporating it into a single machine scheduling problem with a deterioration model. Partial maintenance saves time by restoring the machine to be maintained to an available state on order to return to a production system. Similarly, Chung et al. and considered a single machine scheduling problem with batch setups, positional deterioration effects, and multiple optional rate-modifying activities to minimize the total completion time [9]. Laalaoui and 'Hallah used a two-phase heuristic algorithm and tried to maximize the weighted number of a single machine environment subject to scheduled maintenance periods and a common due date. They showed that the problem was strongly NP-hard, and to solve small-sized instances binary multiple knapsacks were proposed [10]. Similarly, Detti et al. addressed a problem arising in a manufacturing environment concerning the joint scheduling of multiple jobs [11].

Furthermore, a variable neighborhood search algorithm was presented for all-sized problems, the computational result showing the effectiveness of the proposed algorithm. Sbihi and Varnier investigated single machine problem scheduling with two scenarios of maintenance activities, with the aim to minimize the maximum tardiness. The first scenario dealt with the problem of when the time between two maintenance activates is fixed, while the second scenario dealt with the problem of when the time between two maintenance activates was not fixed, but depended on the machine working time [12]. Pacheco et al. also used a variable neighborhood search algorithm and studied the problem of sequencing jobs in a single machine with programmed preventive maintenance and sequence-dependent setup times [13]. Wang et al. studied a single machine scheduling problem where deteriorating jobs and flexible periodic maintenance were considered, using a branch and price algorithm [14]. Sun and Geng also aimed to find an optimal schedule in order to minimize the maximum completion time in single machine scheduling [15]. Low et al. presented six heuristic algorithms based on first fit and best fit, dealing with single machine scheduling problems with availability constraints to minimize the makespan. They considered that the machine should stop to maintain after a periodic time interval or change tools after a fixed number of jobs had been processed simultaneously [16]. Mashkani and Moslehi examined the minimization of the makespan on a single machine under a new term called bimodal flexible periodic availability constraints. A binary integer mathematical programming model was presented and they provided an efficient branch-and-bound algorithm and heuristic algorithm to solve the problem using several dominance rules [17].

For such cases where periodic maintenance activities are required, Liao proposed a branch-and-bound algorithm and heuristic to minimize the maximum tardiness once maintenance is performed periodically within a fixed time interval [18], Chen proposed a heuristic for the objective of minimizing the mean flow time of jobs [19]. Chen also proposed a branch-and-bound algorithm to find the optimal schedule [3], discussing the scheduling problem on single machine with the objective of minimizing the number of tardy jobs. Likewise, Hong et al. suggested optimal periodic maintenance policies to determine the interval between the periodic maintenance activities taking into consideration the maintenance costs [20], while Shen and Zhu studied a single machine scheduling problem with periodic maintenance, in which the processing time and repair time were nondeterministic. They used LPT algorithms to solve the problem [21].

The number of tardy jobs has long been considered a performance indicator in a number of research papers on single machine scheduling problems. For example, Lee and Kim presented a mixed-integer programming model and a heuristic consisting of two phases [22], the first phase of which was obtaining an initial solution based on Moore's algorithm, the second phase being more for improvement of the initial solution. Liu et al. further investigated the same problem [23], the authors proposing a branch-and-bound algorithm be applied to the obtained optimal solution. Uzsoy and Martin-Vega developed solutions based on Moore's algorithm for problems with other settings [24]. Their problem was solved under the constraint of periodic maintenance when the time between two consecutive maintenances was fixed. Wang and Xu, and Xu and Xu, studied the problem of maintenance duration when dependent on workload time. In their study, the start time for each maintenance was limited to a certain time window [25], [26]. Under the objective of minimizing a maximum lateness minimization [25] and minimizing the makespan [26], different algorithms were presented to solve the problem, the results showing their effectiveness. Likewise, Qamhan et al. also presented an evolutionary discrete firefly algorithm (EDFA) to solve a real-world manufacturing system problem of job scheduling [27]. Touat et al. focused on flexible maintenance under human resource constraints on single machine scheduling problems to minimize the sum of total weighted tardiness [28]. Two strategies using a fuzzy logic hybrid with a genetic algorithm were proposed to find near-optimal solutions. Zammori et al. proposed a hybrid harmony search algorithm with genetic algorithms [29]. They addressed a problem on single machine scheduling subject to planned

TABLE 1. Model notations.

Indices	
i	$the\ index\ of\ job, \quad i = 1,2, \dots, n;$
j	$the\ index\ of\ job\ position, \quad j = 1,2, \dots, n;$
l	$the\ index\ of\ periodic\ maintenance\ l = 1,2, \dots, L;$
Decision variables	
x_{ijl}	$binary\ decision\ variable\ and\ x_{ijl} = 1\ if\ i\ processed\ in\ position\ j\ in\ the\ period\ l\ )$
U_{jl}	$binary\ decision\ variable\ and\ U_{jl} = 1\ \ if\ the\ job\ in\ position\ j\ in\ the\ period\ l\ is\ tardy.$
f_{jl}	$decision\ variable, the\ compliation\ time\ time\ of\ job\ j\ in\ period\ l$
q_{jl}	$decision\ variable\ for\ the\ virtual\ and\ actual\ due\ date\ of\ job\ j\ in\ period\ l$
tm_l	$decision\ variable, the\ starting\ time\ of\ maintenance\ l$
fm_l	$decision\ variable, the\ compliation\ time\ of\ maintenance\ l$
Parameters	
p_i	$processing\ time\ of\ job\ i$
d_i	$due\ date\ for\ job\ i$
M_l	$time\ duration\ of\ maintenance\ l$
T	$presumptive\ time\ between\ two\ maintenance$
w	$time\ allowance$
A	$a\ large\ enough\ number$

maintenance and sequence-dependent setup times with the objective of minimizing total earliness and tardiness penalties. Later on, Bertolini et al. went on to present and compare new metaheuristics to solve an integrated jobs-maintenance scheduling problem on a single machine subjected to aging and failures [30]. Nie et al. addressed the problem of a fuzzy random time window for maintenance planning on single machine scheduling problems with multi-objective functions of minimizing the total weighted completion time, and maximizing the average timeliness level under a fuzzy environment [31]. Global-local-neighbor particle swarm optimization (PSO) based on the first fit rule as the initial swarm was proposed, and the experimental results showed that the algorithm was practicable and efficient in handling such complex problems. Krim et al. also challenged the single machine scheduling problem using periodic preventive maintenance to minimize the weighted sum of completion times [32]. Results showed that the average percentage error of the best heuristic was less than 10%. Chung et al. addressed the same problem to minimize total completion time with the use of a binary integer programming model [33]. Others such as Chen et al. aimed to minimize the total completion times where the machine had to receive periodical maintenance so that the dirt generated in the process did not exceed the limit [34], while Xu and Xu considered a single machine tool change scheduling problem where tool change durations were workload dependent [35].

In this sense, the addressed problem in this study is to minimize the number of tardy jobs on single machine problem under flexible periodic maintenance within a time window, in addition to the maintenance activates having different durations. To the best of our knowledge, this problem has not yet been studied. The next section will describe the mixed-integer linear programming model. Section 4 will be devoted to the presentation of the proposed algorithm (ACO). Section 5 will present the experimental results. Finally, section 6 will be dedicated to a conclusion and recommendations for future study.

III. PROBLEM FORMULATION AND ANALYSI

A. PROBLEM DEFINITION AND ASSUMPTION

The addressed problem can be summarized as the following; There are n jobs $J = J_1, J_2, \dots, J_n$ to be processed on a single machine, which the machine can only process one job J_i at a time. There is no preemption allowed or precedence relationship between the jobs and all jobs are available at time zero. Each job has due dat $d = d_1, d_2, \dots, d_n$e. The machine is not available in the all-time horizon line and it has a L periodic maintenances $M = M_1, M_2, \dots, M_l$, and the durations of different maintenance activate M_l sare not necessarily equal. The time between two maintenance activates is fixed and equal, T, and each maintenance activity can start in a time window T $\mp$ w where w is the time allowance for the starting maintenance activity. The main task is to optimize the scheduling sequence of the n jobs which minimizes the number of tardy jobs.

B. MIXED-INTEGER LINEAR PROGRAMMING MODEL (MILP)

The following notations in Table 1 are used to formulate the problem:

TABLE 2. Jobs data for the given example.

Job	1	2	3	4	5	6	7	8	9	10
p_i	47	27	94	90	60	13	39	37	16	79
d_i	322	324	278	429	398	352	456	332	364	438

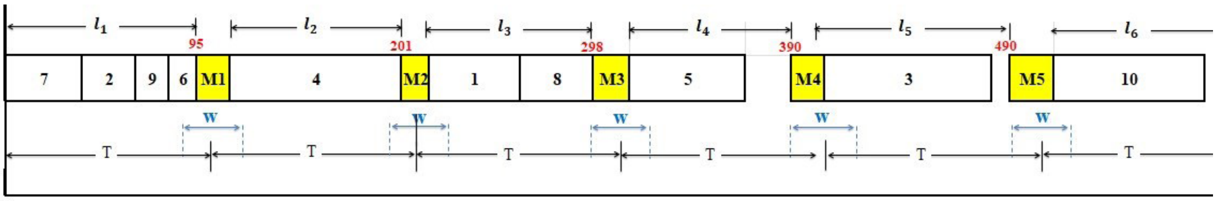

FIGURE 1. Optimal schedule for the example.

TABLE 3. Maintenance data for the given example.

Job	1	2	3	4	5	6	7	8
M_i	16	13	18	12	22	18	24	13
	T=100				*w=10*			

The MILP model can be formulated as follows:

$$\min \sum U \tag{1}$$

Subject to

$$\sum\nolimits_{(l=1)}^{L} \sum\nolimits_{(i=1)}^{n} x_{ijl} = 1, \quad j = 1, 2, \ldots, n \tag{2}$$

$$\sum\nolimits_{(l=1)}^{L} \sum\nolimits_{(j=1)}^{n} x_{ijl} = 1, \quad i = 1, 2, \ldots, n \tag{3}$$

$$\sum\nolimits_{(i=1)}^{n} \sum\nolimits_{(j=1)}^{n} p_i x_{ijl} \leq tm_l - fm_{l-1}, \quad l = 1, 2, \ldots, L \tag{4}$$

$$fm_0 = 0 \tag{5}$$

$$fm_l = tm_l + M_l, \quad l = 1, 2, \ldots, L \tag{6}$$

$$tm_l \geq l * T - w_l, \quad l = 1, 2, \ldots, L \tag{7}$$

$$tm_l \leq l * T + w_l, \quad l = 1, 2, \ldots, L \tag{8}$$

$$fm_{l-1} + \sum\nolimits_{(i=1)}^{n} \sum\nolimits_{(k=1)}^{j} p_i x_{ikl} = f_{jl}, \quad j = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, L \tag{9}$$

$$\sum\nolimits_{(i=1)}^{n} d_i x_{ijl} + A\left(1 - \sum\nolimits_{(i=1)}^{n} x_{ijl}\right) = q_{jl}, \quad j = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, L \tag{10}$$

$$f_{jl} - q_{jl} - AU_{jl} \leq 0, \quad j = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, L \tag{11}$$

$$x_{ijl}, U_{jl} \text{ binary variables} \in \{0, 1\}, \quad i, j = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, \tag{12}$$

$$f_{jl}, q_{jl}, tm_l, fm_l \geq 0, \quad i, j = 1, 2, \ldots, n \quad \& l = 1, 2, \ldots, L \tag{13}$$

Constraint (1) is the objective function to minimize the total number of tardy jobs. Constraints (2 and 3) ensure that each job occupies only one different position and vice versa. Constraint (4) guarantees that the total processing time in each period l is in the allowable range between the starting time of maintenance l and the finishing time of its immediate predecessor l-1. Constraint (5) fixes the dummy maintenance as the first of the schedule. Constraints (6) to (8) are to calculate maintenance starting time and finishing time. Constraints (9) to (11) are to estimate the number of tardy jobs.

C. NUMERICAL EXAMPLE

To validate the proposed mixed-integer linear programming model, an instance with 10 jobs (n = 10) was tested by using a branch and cut method under CPLEX solver. Table 2 shows the input data for the given example. Figure 1 shows the Gantt chart for the optimal schedule.

The number of required maintenance depends on the total processing time and the presumptive time between two maintenances T. In Table 3 the maintenances input data for the given example.

IV. ANT COLONY OPTIMIZATION (ACO

ACO is one of the algorithms used for the discrete optimization problem, having been applied to solve many optimization problems in various domains. For single machine scheduling problems, we can cite the works of Liao and Juan [36].

This algorithm is inspired by the searching behavior of ants for food, which can be described as follows: A group of ants starts searching in several random directions from the nest (this process is initiated only once). During the passage on any path, the ants produce a chemical pheromone to mark their paths. When an ant finds a source of food, it takes a quantity of it and returns to the nest by choosing the path with the largest amount of pheromone. The ant will then start searching from the nest again, choosing the path that contains the largest amount of pheromone. The amount of pheromone is updated every time period. The shortest path will always contain the largest amount of pheromone and therefore all the ants will move along it. To optimize the problem under study in this paper we adopted the ACO algorithm which was used in [36], [37]. The following subsections briefly describe the components of the proposed algorithm.

A. INITIALIZE THE PHEROMON

The initial pheromone is given by the following formula:

$$\tau_0 = K/(n * \sum U)$$

where K is constant, n is the number of jobs and $\sum U$ is the number of tardy jobs by applying the initial heuristic.

Algorithm 1 Heuristic for the Initial Solution

```
Function initial_sol();
Function Heuristic (i,j); /*apply Moore's
Algorithm then calculate heuristic desirability when
job i occupies position j */
A_n ← order the job in a non − decreasing
of their due dates;
i ← 1;
while i ≤ n do
    if A_i have tardy job
        Move the longest processing time into R set
        Update the completion time in A set
    i++;
End
i ← 1; /∗ Estimate the heuristic desirability Heuristic
(i,j)/
while i ≤ n do
    j← 1;
    while j ≤ j max do
        U ← number_of_Tardy for (Heuristic (i,j));
        η (i, u) ← 1/(n∗U);
        j++;
    End
    i ++;
end
```

B. A HEURISTIC FOR THE INITIAL SOLUTIO

We implemented Moore's algorithm (Algorithm 1) as a heuristic for the initial solution. The advantage of using this rule is that it estimates the heuristic desirability better than a random search.

C. MAIN LOOP

Two parameters to control the termination conditions in the main loop (Algorithm 2): the maximum number of

Algorithm 2 Main ACO Loop

```
Function main();
i ← 1, j←0;
while i ≤ Itemax && j≤ Rmax do
    k←1
    while k≤ Number of ants do
        Sequence construction ();
        Local search();
        Pheromone updating();
        U'' ← number_of_Tardy for (ant(k),
        Sequence);
        if U'' is better than U then
            U'' ← U;
            j=0;
        else j++;
        k++;
    i ++;
end
```

non-improvement (Rmax), and the maximum number of iterations (Itemax). In each loop, the algorithm will construct a sequence of n jobs for the m ants, updating the pheromone, initiate local search for the sequence and save the best fit.

D. CONSTRUCT THE JOB SEQUENCE FOR EACH ANT

All ants start each loop with an empty sequence, and step by step each ant independently constructs the unscheduled jobs until a feasible solution is obtained. To choose the job j to be processed in the current position i it should be calculated according to the heuristic desirability $\eta(i, u)$ of this position and the quantity of pheromones $\tau_t(i, u)$ on the arc of that connection between the two. This choice will be made randomly, with a probability of choosing position j given by:

$$j = \left\{ \begin{array}{ll} \arg m \max\limits_{u \in U} \left\{ [\tau_t(i, u)]^\alpha \, [\eta(i, u)]^\beta \right\} & if\ q \leq q_0, \\ S & otherwise, \end{array} \right\}$$

where: U is the set of unscheduled jobs, q is a random number generated between [0, 1], q_0 is a given number between [0, 1] referring to the relative importance of exploitation, α and β are two parameters that control the relative importance of pheromones and desirability. From the above formula if $q \leq q_0$ the maximum value for the unscheduled job j is placed at position; otherwise, the job is selected according to the random variable S which itself is selected from the following probability:

$$p\,(i, j) = \frac{[\tau_t(i, u)]^\alpha \, [\eta(i, u)]^\beta}{\sum_{u \in U} [\tau_t(i, u)]^\alpha \, [\eta(i, u)]^\beta}$$

E. LOCAL SEARC

The procedure consists of swapping two jobs selected randomly, for the possibility of improvement (Algorithm 3).

TABLE 4. The generating conditions of test problems.

Group	Processing times	Due Dates	Tardiness factor T	The relative range of due dates R	Maintenance interval	Maintenance time
instances	U(10, 100)	U(a, b)	{0.2, 0.4,0.6}	{0.2,0.5,0. 8)	{0.3,0.4} for small sized {0.1,0.2} for large sized	U(c, d)

$a = \sum p\left(1 - T - \frac{R}{2}\right)$ & $b = \sum p\,(1 - T + \frac{R}{2})$

$c = 0.02 * \sum p$ & $d = 0.05 * \sum p$

The time allowance for the starting maintenance w = 0.1* Maintenance interval.

TABLE 5. Comparison of ACO with CPLEX.

			n=10		n=12		CPU_Time							
							CPLEX				ACO			
T	R	I	Gap	NO	Gap	NO	Ave.	Max.	Min.	Std.	Ave.	Max.	Min.	Std.
0.2	0.2	0.3	0	10	0	10	59.71	205.89	12.22	73.59	0.12	0.14	0.10	0.01
		0.4	0	10	0	10	71.22	341.84	11.89	103.35	0.10	0.12	0.08	0.01
0.2	0.5	0.3	0	10	0	10	16.14	30.16	11.33	7.22	0.13	0.15	0.11	0.02
		0.4	0	10	0	10	101.34	720.77	10.85	224.27	0.11	0.14	0.09	0.01
0.2	0.8	0.3	0	10	0	10	12.35	20.49	10.64	3.01	0.14	0.16	0.12	0.01
		0.4	0	10	0	10	18.51	86.70	8.98	23.98	0.12	0.15	0.11	0.01
0.4	0.2	0.3	0	10	0	10	68.46	415.22	12.37	129.89	0.12	0.13	0.10	0.01
		0.4	0	10	0	10	159.3	676.98	19.66	207.35	0.11	0.12	0.10	0.01
0.4	0.5	0.3	0	10	0	10	77.2	374.66	13.81	123.47	0.13	0.14	0.12	0.01
		0.4	0	10	0	10	37.59	194.20	12.29	56.70	0.11	0.12	0.10	0.01
0.4	0.8	0.3	0	10	0	10	48.12	160.91	13.01	54.45	0.13	0.15	0.11	0.01
		0.4	0	10	0	10	33.10	79.19	11.48	26.26	0.11	0.13	0.09	0.01
0.6	0.2	0.3	0	10	0	10	183.51	664.80	14.72	229.96	0.12	0.14	0.11	0.01
		0.4	0	10	0	10	61.08	232.32	11.20	84.50	0.10	0.13	0.08	0.02
0.6	0.5	0.3	0	10	0	10	20.55	68.96	11.01	17.99	0.13	0.14	0.11	0.01
		0.4	0	10	0	10	148.39	731.41	12.83	226.48	0.11	0.12	0.10	0.01
0.6	0.8	0.3	0	10	0	10	27.83	114.76	10.72	33.41	0.12	0.15	0.05	0.03
		0.4	0	10	0	10	90.80	614.70	12.36	189.55	0.12	0.13	0.10	0.01

Gap: is the average relative deviation of the solution from optimal solutions, which were obtained by CPLEX.

NO: Number of instances (out of 10 instances) for which the algorithm found optimal solutions.

Ave: the average consumed time to obtain the solution for the 10 instances.

Max: the maximum consumed time to obtain the solution form the 10 instances.

Min: the minimum consumed time to obtain the solution form the 10 instances.

Std: the standard deviation of the consumed time from the average.

F. UPDATING THE PHEROMON

At the end of each cycle (all jobs are scheduled for each ant), the pheromone variables are updated according to the formula:

$$\tau_t\,(i,j) = (1-\rho)\,\tau_t\,(i,j) + \rho\tau_{t-1}\,(i,j)$$

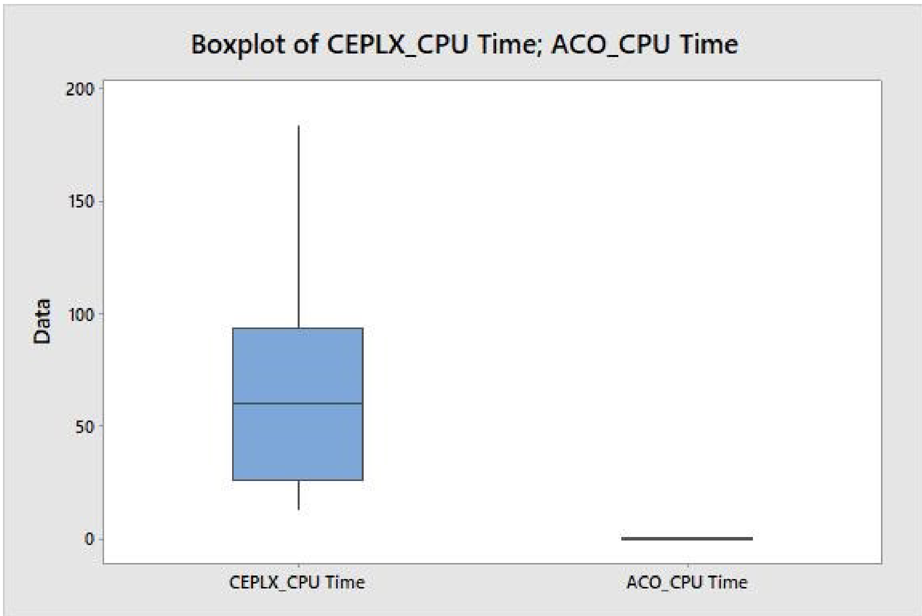

FIGURE 2. Boxplot of consumed CPU time for small-sized instances.

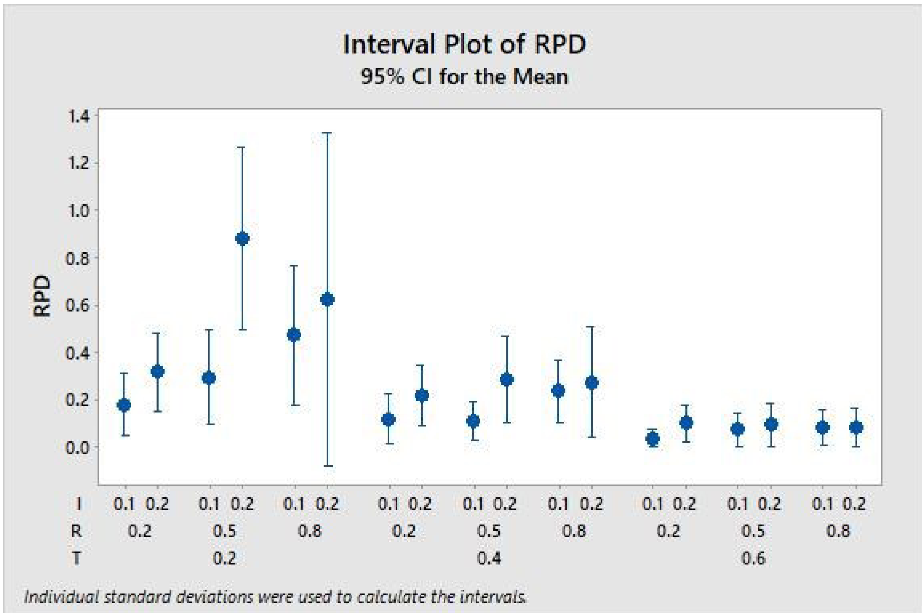

FIGURE 3. Interval Plot of relative percentage devotion for large-size instances.

where $\rho \in [0, 1]$ represents the coefficient of the evaporation rate of pheromones on the arcs between the time t and time (t−1).

V. EXPERIMENTAL RESULT

Two algorithms, ACO and Moore's, were coded in C and run on a PC including an Intel Core i3 CPU running at 2.53 GHz and configured with 3 GB of RAM. To test the efficiency and the effectiveness of the ACO algorithm, a set of data was generated using the parameters that provide [22] ten problem instances for each combination. The parameters are listed in Table 4.

For regulation, the algorithm parameters for a series of pilot runs using the algorithm were conducted, including temporaril stopping the part of the local search because it introduced a noise factor for the algorithm parameters. The recommended values for the problem are as follows: Maximum number of iterations = 100, Maximum number of non-improvement = 50, Number of ants = 10, $\alpha = 0.1$, $\beta = 0.8, \rho = 0.05, q_0 = 0.9$.

The computational results are summarized in Tables 5 and 6.

For small-sized instances, those with 10 to 12 jobs, optimal solutions were obtained from CPLEX solver with a one hour

TABLE 6. Comparison of ACO with moore's algorithm.

T, R	I	n=50 RPD Ave.	Max.	Min.	Std.	HC*	ACO*	n=100 RPD Ave.	Max.	Min.	Std.	HC*	ACO*	CPU TIME HC	ACO
0.2,0.2	0.1	0.19	0.62	0.00	0.22	0	6	0.17	0.88	0.00	0.34	0	3	0.07	0.89
	0.2	0.36	1.00	0.00	0.38	0	7	0.28	0.84	0.00	0.33	0	7	0.06	0.58
0.2,0.5	0.1	0.46	1.36	0.00	0.53	0	8	0.13	0.50	0.00	0.20	0	5	0.06	0.85
	0.2	0.91	2.75	0.00	0.92	0	8	0.86	1.91	0.00	0.75	0	8	0.06	0.64
0.2,0.8	0.1	0.78	1.75	0.00	0.68	0	7	0.17	1.28	0.00	0.41	0	2	0.08	0.90
	0.2	1.25	5.00	0.00	1.96	0	4	0.00	0.00	0.00	0.00	0	0	0.06	0.71
0.4,0.2	0.1	0.20	0.82	0.00	0.29	0	5	0.04	0.29	0.00	0.09	0	3	0.06	0.88
	0.2	0.21	0.71	0.00	0.30	0	4	0.23	0.81	0.00	0.26	0	6	0.06	0.61
0.4,0.5	0.1	0.14	0.50	0.00	0.16	0	7	0.09	0.49	0.00	0.18	0	2	0.06	0.87
	0.2	0.17	0.80	0.00	0.27	0	5	0.40	1.28	0.00	0.46	0	6	0.06	0.64
0.4,0.8	0.1	0.17	0.50	0.00	0.20	0	7	0.30	0.87	0.00	0.34	0	6	0.07	0.87
	0.2	0.22	1.29	0.00	0.40	0	5	0.33	1.80	0.00	0.59	0	5	0.06	0.70
0.6,0.2	0.1	0.04	0.24	0.00	0.08	0	3	0.03	0.26	0.00	0.08	0	3	0.06	0.95
	0.2	0.10	0.43	0.00	0.14	0	5	0.10	0.51	0.00	0.20	0	3	0.08	0.66
0.6,0.5	0.1	0.14	0.52	0.00	0.19	0	4	0.01	0.09	0.00	0.03	0	1	0.06	0.99
	0.2	0.14	0.67	0.00	0.24	0	5	0.06	0.41	0.00	0.13	0	3	0.06	0.68
0.6,0.8	0.1	0.10	0.50	0.00	0.18	0	3	0.07	0.38	0.00	0.13	0	3	0.05	0.93
	0.2	0.11	0.64	0.00	0.21	0	4	0.07	0.41	0.00	0.14	0	2	0.05	0.22

HC*: Number of instances (out of 10 instances) for which Moore's algorithm gave better solutions than ACO.
ACO*: Number of instances (out of 10 instances) for which ACO gave better solutions than Moore's algorithm.
Ave: the average relative deviation of the solution from the best-found solutions for the 10 instances.
Max: the maximum relative deviation of the solution from the best-found solutions for the 10 instances.
Min: the minimum relative deviation of the solution from the best-found solutions for the 10 instances.
Std: the standard deviation of the relative deviation from the average.

Algorithm 3 Local Searchc

```
Function Local search ();
i ← 1;
while i ≤ i_max do
    j← Rand() % n;
    k← Rand() % n;
    if ( j!= k)
        Swap sequence(k,j);
        U'' ← number_of_Tardy for
        (ant, sequence(k, j));
        if U'' is better than U then
            U'' ← U;
            Sequence ← Swap sequence(j, k);
    i ++;
end
```

run time limitation. As Table 5 shows, the proposed ACO was able to obtain the exact solutions within reasonable CPU times and outperformed the CPLEX solver CPU as shown in Figure 2. Results of the test on large-size instances, those with 50 to 100 jobs, show that the ACO outperforms Moore's algorithm for all the instances tested, as shown in Table 6.

In addition, to assess the behavior of the algorithm with different parameters, Figure 3 shows the impact of tardiness factor (T), the relative range of due dates (R) and Maintenance interval factor (I) on the Average Relative Percentage Devotion (ARPD).

VI. CONCLUSION AND RESULTS DISCUSSION

This study proposed two methods to minimize the total number of tardy jobs when scheduling a set of jobs on a single machine influenced by intervention dates of periodic maintenance actions, with different durations and time windows for starting the maintenance actions. The first method gives an exact solution by using CPLEX solver to solve the proposed mixed-integer linear programming model. A near-optimal solution method using an ACO algorithm has also been presented. Different sets of data were generated to validate and test the quality of the proposed algorithms. The results showed that the ACO algorithm, which was able to obtain exact solutions in reasonable CPU times, outperformed the CPLEX solver. In addition, when the ACO compared to Moore's algorithm, the results showed that the ACO outperforms Moor's algorithm for all the instances tested. It can be concluded that the developed AC is very efficient and effective in solving the problem considered in this paper.

ACKNOWLEDGMENT

The authors also thank the Deanship of Scientific Research and RSSU at King Saud University for their technical support.

REFERENCES

[1] K. G. Kempf, P. Keskinocak, and R. Uzsoy, *Planning Production and Inventories in the Extended Enterprise: A State of the Art Handbook*, vol. 1. Springer, 2011.
[2] M. Pinedo, *Scheduling: Theory, Algorithms, and Systems*. Springer, 2012.
[3] W. Chen, "Minimizing number of tardy jobs on a single machine subject to periodic maintenance," *Omega*, vol. 37, no. 3, pp. 591–599, Jun. 2009.
[4] Y. Ma, C. Chu, and C. Zuo, "A survey of scheduling with deterministic machine availability constraints," *Comput. Ind. Eng.*, vol. 58, no. 2, pp. 199–211, Mar. 2010.
[5] E. Sanlaville and G. Schmidt, "Machine scheduling with availability constraints," *Acta Inf.*, vol. 35, no. 9, pp. 795–811, Sep. 1998.
[6] W.-W. Cui and Z. Lu, "Minimizing the makespan on a single machine with flexible maintenances and jobs' release dates," *Comput. Oper. Res.*, vol. 80, pp. 11–22, Apr. 2017.
[7] F. Ángel-Bello, A. Álvarez, J. Pacheco, and I. Martínez, "A single machine scheduling problem with availability constraints and sequence-dependent setup costs," *Appl. Math. Model.*, vol. 35, no. 4, pp. 2041–2050, Apr. 2011.
[8] Y.-T. Hou, D.-L. Yang, W.-H. Kuo, and L.-S. Wu, "A single-machine scheduling problem with a deterioration model and partial maintenance," *J. Statist. Manage. Syst.*, vol. 21, no. 8, pp. 1501–1511, Nov. 2018.
[9] T. Chung, J. N. D. Gupta, and M. Qiu, "Single machine scheduling problem with batch setups involving positional deterioration effects and multiple rate-modifying activities," *Eng. Optim.*, vol. 51, no. 10, pp. 1743–1760, 2019.
[10] Y. Laalaoui and R. M'Hallah, "A binary multiple knapsack model for single machine scheduling with machine unavailability," *Comput. Oper. Res.*, vol. 72, pp. 71–82, Aug. 2016.
[11] P. Detti, G. Nicosia, A. Pacifici, and G. Zabalo Manrique de Lara, "Robust single machine scheduling with a flexible maintenance activity," *Comput. Oper. Res.*, vol. 107, pp. 19–31, Jul. 2019.
[12] M. Sbihi and C. Varnier, "Single-machine scheduling with periodic and flexible periodic maintenance to minimize maximum tardiness," *Comput. Ind. Eng.*, vol. 55, no. 4, pp. 830–840, Nov. 2008.
[13] J. Pacheco, S. Porras, S. Casado, and B. Baruque, "Variable neighborhood search with memory for a single-machine scheduling problem with periodic maintenance and sequence-dependent set-up times," *Knowl.-Based Syst.*, vol. 145, pp. 236–249, Apr. 2018.
[14] T. Wang, R. Baldacci, A. Lim, and Q. Hu, "A branch-and-price algorithm for scheduling of deteriorating jobs and flexible periodic maintenance on a single machine," *Eur. J. Oper. Res.*, vol. 271, no. 3, pp. 826–838, Dec. 2018.
[15] X. Sun and X.-N. Geng, "Single-machine scheduling with deteriorating effects and machine maintenance," *Int. J. Prod. Res.*, vol. 57, no. 10, pp. 3186–3199, Jan. 2019.
[16] C. Low, M. Ji, C.-J. Hsu, and C.-T. Su, "Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance," *Appl. Math. Model.*, vol. 34, no. 2, pp. 334–342, Feb. 2010.
[17] O. Mashkani and G. Moslehi, "Minimising the total completion time in a single machine scheduling problem under bimodal flexible periodic availability constraints," *Int. J. Comput. Integr. Manuf.*, vol. 29, no. 3, pp. 323–341, 2016.
[18] C. J. Liao and W. J. Chen, "Single-machine scheduling with periodic maintenance and nonresumable jobs," *Comput. Oper. Res.*, vol. 30, no. 9, pp. 1335–1347, Aug. 2003.
[19] J. S. Chen, "Single-machine scheduling with flexible and periodic maintenance," *J. Oper. Res. Soc.*, vol. 57, no. 6, pp. 703–710, 2006.
[20] S.-S. Hong, J.-H. Park, and C.-H. Lie, "Optimal preventive maintenance policy with cost-dependent improvement factor," *J. Korean Inst. Ind. Eng.*, vol. 36, pp. 108–116, Jan. 2010.
[21] J. Shen and K. Zhu, "An uncertain single machine scheduling problem with periodic maintenance," *Knowl.-Based Syst.*, vol. 144, pp. 32–41, Mar. 2018.
[22] J.-Y. Lee and Y.-D. Kim, "Minimizing the number of tardy jobs in a single-machine scheduling problem with periodic maintenance," *Comput. Oper. Res.*, vol. 39, no. 9, pp. 2196–2205, Sep. 2012.
[23] M. Liu, S. Wang, C. Chu, and F. Chu, "An improved exact algorithm for single-machine scheduling to minimise the number of tardy jobs with periodic maintenance," *Int. J. Prod. Res.*, vol. 54, no. 12, pp. 3591–3602, 2016.
[24] R. Uzsoy, "Production scheduling algorithms for a semiconductor test facility," *IEEE Trans. Semicond. Manuf.*, vol. 4, no. 4, pp. 270–280, Nov. 1991.
[25] T. Wang and D. Xu, "Single-machine scheduling with workload-dependent maintenance duration to minimize maximum lateness," *Math. Problems Eng.*, vol. 2015, pp. 1–5, Jul. 2015.
[26] Z. Xu and D. Xu, "Single-machine scheduling with preemptive jobs and workload-dependent maintenance durations," *Oper. Res.*, vol. 15, no. 3, pp. 423–436, May 2015.
[27] M. A. Qamhan, A. A. Qamhan, I. M. Al-Harkan, and Y. A. Alotaibi, "Mathematical modeling and discrete firefly algorithm to optimize scheduling problem with release date, sequence-dependent setup time, and periodic maintenance," *Math. Problems Eng.*, vol. 2019, pp. 1–16, Aug. 2019.
[28] M. Touat, S. Bouzidi-Hassini, F. Benbouzid-Sitayeb, and B. Benhamou, "A hybridization of genetic algorithms and fuzzy logic for the single-machine scheduling with flexible maintenance problem under human resource constraints," *Appl. Soft Comput.*, vol. 59, pp. 556–573, Oct. 2017.
[29] F. Zammori, M. Braglia, and D. Castellano, "Harmony search algorithm for single-machine scheduling problem with planned maintenance," *Comput. Ind. Eng.*, vol. 76, pp. 333–346, Oct. 2014.
[30] M. Bertolini, D. Mezzogori, and F. Zammori, "Comparison of new metaheuristics, for the solution of an integrated jobs-maintenance scheduling problem," *Expert Syst. Appl.*, vol. 122, pp. 118–136, May 2019.
[31] L. Nie, J. Xu, and Y. Tu, "Maintenance scheduling problem with fuzzy random time windows on a single machine," *Arabian J. for Sci. Eng.*, vol. 40, no. 3, pp. 959–974, Dec. 2014.
[32] H. Krim, R. Benmansour, D. Duvivier, D. Aït-Kadi, and S. Hanafi, "Heuristics for the single machine weighted sum of completion times scheduling problem with periodic maintenance," *Comput. Optim. Appl.*, vol. 75, no. 1, pp. 291–320, Oct. 2019.
[33] T.-P. Chung, Z. Xue, T. Wu, and S. C. Shih, "Minimising total completion time on single-machine scheduling with new integrated maintenance activities," *Int. J. Prod. Res.*, vol. 57, no. 3, pp. 918–930, 2019.
[34] Y. Chen, L.-H. Su, Y.-C. Tsai, S. Huang, and F.-D. Chou, "Scheduling jobs on a single machine with dirt cleaning consideration to minimize total completion time," *IEEE Access*, vol. 7, pp. 22290–22300, 2019.
[35] Z. Xu and D. Xu, "Single-machine scheduling with workload-dependent tool change durations and equal processing time jobs to minimize total completion time," *J. Scheduling*, vol. 21, no. 4, pp. 461–482, Oct. 2017.
[36] C.-J. Liao and H.-C. Juan, "An ant colony optimization for single-machine tardiness scheduling with sequence-dependent setups," *Comput. Oper. Res.*, vol. 34, no. 7, pp. 1899–1909, Jul. 2007.
[37] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

AMMAR A. QAMHAN received the B.Sc. degree in industrial engineering and manufacturing systems from the Faculty of Engineering and Information Technology, Taiz University, in 2011, and the M.S. degree in industrial engineering from King Saud University, in 2019. He worked as an Engineer with Hayel Saeed Anam (HSA) Head Office, Technical Auditing Yemen Region, from 2012 to 2014. His main research interests include optimization and scheduling.

AREF AHMED (Member, IEEE) received the B.S. degree in industrial engineering from Taiz University, Yemen, in 2013. He is currently pursuing the M.S. degree in industrial engineering with the College of Engineering, King Saud University, Riyadh, Saudi Arabia. Since 2015, he has been a Doctor's and Researcher's Assistant in industrial engineering with the College of Engineering. His research interests include optimization, safety risk assessment, and ergonomic.

IBRAHIM M. AL-HARKAN received the B.S. degree in industrial engineering from King Saud University, and the M.S. and Ph.D. degrees in industrial engineering from The University of Oklahoma, USA. He was the Chairman of the Mechanical Engineering Department for three years and half, also the Chairman of the Industrial Engineering Department for two years, in addition, he was the dean for graduate studies for five years. He was the Vice Rector Assistant for Graduate studies and Scientific Research for Knowledge Exchange and Technology Transfer for one year. He was the General Director of the External Joint Supervision Program (EJSP) for ten years. He is currently the General Director of the Entrepreneurship Institute. He is currently an Associate Professor of industrial engineering with King Saud University, Saudi Arabia. His specialized areas are production planning and control, modeling and simulation of industrial and service systems, and applied operations research. He is one of the founder of the Saudi Industrial Engineering Society (SIES) and currently the Vice President of the society, one of the founders of the Industrial Engineering Council at the Saudi Council of Engineers, one of the founders of the Saudi Chapter of Institute of Industrial engineers, and established the Department of Industrial Engineering, College of Engineering, King Saud University.

AHMED BADWELAN received the B.Sc. degree in mechanical engineering from the Faculty of Engineering, University of Aden, in 2012, and the M.S. degree in industrial engineering from King Saud University, Saudi Arabia. He is currently pursuing the Ph.D. degree with the Industrial Engineering Department, College of Engineering, King Saud University, Saudi Arabia. He is also a Researcher with the Industrial Engineering Department, College of Engineering, King Saud University. His area of expertise is manufacturing systems. His main research interests include manufacturing, production, and quality.

ALI M. AL-SAMHAN received the B.Sc. and M.Sc. degrees in mechanical engineering from King Saud University, Riyadh, Saudi Arabia, and the Ph.D. degree from the School of Manufacturing and Mechanical Engineering, Birmingham University, U.K. He is currently a Professor and the Chairman of the Industrial Engineering Department, King Saud University. He served as the Vice Dean of scientific research at King Saud University. He also serves as a Consultant for leading manufacturers in Saudi Arabia and a member of the SASO Committee for industrial standardization. Also, he assists local manufactures, as a R&D Consultant, in developing more than 30 products introduced to Saudi market 20 years ago. Also, he developed two B.Sc. teaching laboratories in the Industrial Engineering Department, for metal forming and manufacturing automation using PLC courses. His research interests focus on engineering materials and manufacturing processes, monitoring and controlling of manufacturing systems using artificial intelligence, product design and development, automation and mechatronics, and biomechanical engineering. He has many registered patents and published articles in leading journals of *Industrial and Manufacturing Engineering*. He is leading a number of funded projects from different organization in Saudi Arabia.

LOTFI HIDRI received the B.S. degree in mathematics from the Tunisian College of Science, in 1993, the M.S. degree in energetic engineering from the National Engineering School, in 1999, and the Ph.D. degree in operations research from the Tunisian High Institute of Management, in 2007. Since 2012, he has been a Faculty Member with the Industrial Engineering Department, King Saud University. His main research interest is focused in scheduling and transportation.