Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

A metaheuristic algorithm based on Ant Colony Based approach for the assigning tasks problem to a workforce with different skills

Roosvell Camilo Velandia[a], David Alvarez Martinez[b] and John Willmer Escobar[c*]

[a]*VGG Aplicaciones, Colombia*
[b]*Department of Industrial Engineering, Universidad de los Andes, Bogotá, Bogotá 110110, Cundinamarca, Colombia*
[c]*Department of Accounting and Finance, Universidad del Valle, Cali, Cali 760001, Valle del Cauca, Colombia*

C H R O N I C L E

Article history:
Received: October 23, 2023
Received in revised format: March 20, 2024
Accepted: March 29, 2024
Available online: March 29, 2024

Keywords:
Ant Colony Optimization
Multiskill Workforce Scheduling
Unrelated Parallel Machine Scheduling Problem

A B S T R A C T

This paper studies the problem of assigning tasks to a workforce with different skills. The problem is modeled as an unrelated parallel scheduling problem, incorporating sequence-dependent setup times (UPMSPSDST). Exact methods generally are not able to solve real large problems of UPMSPSDST. Hence, this research introduces an efficient, straightforward metaheuristic solution leveraging the Ant Colony Optimization (ACO) algorithm. The objective is to minimize the total completion time while assigning jobs to unrelated parallel machines with sequence-dependent preparation times. The algorithm establishes a threshold for improving the Ants (solutions) to select only promising ants for the improvement phase, thereby reducing the computational effort performed by local search operators. The proposed ACO algorithm maintains a basic structure and could be extended to solve other scheduling problems. A set of test instances available in the literature has been used to validate the efficiency of the proposed methodology. In addition, the results have been compared with the best previously published works. The ACO algorithm improves 30% of the best-known solutions (BKS) and reaches 30% of the BKS. The results show that the average performance of the ACO algorithm exceeds the average performance of the methods used by the best previously published works for the UPMSPSDST.

1. Introduction

Personnel scheduling or rostering is the process of building work schedules for employees of an organization to satisfy the demand for goods or services (Ernst et al., 2004). Precisely, the problem consists of assigning jobs to qualified personnel while minimizing the required time to satisfy the services and guaranteeing that each requirement is assigned to adequate personnel. This variant of rostering is formally known in the literature as the multi-skill workforce scheduling problem (MSWSP) (Firat & Hurkens, 2012). The MSWSP considers a set of scheduled tasks for a group of technicians for a given time horizon. Generally, scheduling seeks to reduce the maximum completion time for all tasks (called makespan). A multi-skill workforce scheduling problem variant is the work assignment problem (Unrelated Parallel Machine Scheduling Problem with Sequence-Dependent Setup Times - UPMSPSDST), introduced by Cheng & Sin (1990). The UPMSPSDST is considered NP-hard (Pinedo, 2016). The UPMSPSDST has been solved by using exact or approximate techniques. Exact methods find the best solution for small problems, usually with high computing times. In contrast, approximate methods guarantee good solutions within short computing times. Reviews related to workforce scheduling problems (WSP) have been published by Castillo et al. (2009) and De Bruecker et al. (2015). The philosophy of exact early methods of WSP has prioritized independent works (Moodie & Roberts, 1967). The concept of preferential partial order for the work units was added by Muntz & Coffman (1969). Additionally, some research has been conducted on graph theory (Allahverdi et al., 1999). The obtained results for the described works are not promising due to their computational complexity. Therefore,

* Corresponding author.
E-mail address: john.wilmer.escobar@correounivalle.edu.co (J. W. Escobar)

doi: 10.5267/dsl.2024.3.006

the scientific community has proposed approximate optimization models, metaheuristics, and hybrid algorithms for solving WSP. Early works dealing with exact methods for WSP have been proposed by Baker & Magazine (1977), Thompson & Goodale (2006), Seçkiner et al. (2007), and Kuo et al. (2011). Baker & Magazine (1977) consider the problem of scheduling days in continuous operations under various day-off policies when demand for the workforce is given. Thompson & Goodale (2006) introduce a nonlinear representation capturing the staffing problem's nonlinear nature and demonstrates their representation's superiority via an extensive set of labor tour scheduling problems. Seçkiner et al. (2007) present an integer programming model for the hierarchical workforce problem under compressed workweeks. The model is based on the formulation developed by Billionnet (1999) for the hierarchical workforce problem. An Unrelated Parallel Machine Scheduling Problem with Simultaneous Setup Time and Learning Effects has been studied by Kuo et al. (2011). In this scenario, the setup time for each job depends on its past sequence. The goals are to minimize the total absolute deviation of job completion times and the overall workload across all machines.

Othman et al. (2012) consider a workforce scheduling model including human aspects, such as skills training, workers' personalities, workers' breaks, and workers' fatigue and recovery levels. The model aims to reduce hiring, firing, training, and overtime expenses. It considers factors such as the number of high-performing workers terminated, break times, and the average fatigue level among workers. Barrera et al. (2012) consider the multiactivity combined timetabling and crew scheduling problem. The objective is to arrange the minimum number of workers necessary to effectively attend to a group of customers distinguished by specific services that must be matched. Florez et al. (2013) consider the multimode resource-constrained project scheduling problem (MRCPSP) in projects requiring alternative renewable and nonrenewable resources. This work extends the MRCPSP and proposes a new multiobjectiv323456e mixed-integer programming model. Laesanklang et al. (2015) propose an approach based on mixed-integer programming (MIP) with decomposition to solve a workforce scheduling and routing problem, in which a set of workers should be assigned to tasks that are distributed across different geographical locations. Gérard et al. (2016) have studied a multiactivity tour scheduling problem with time-varying demand. The authors introduced four methods to address this issue: a mixed-integer linear programming model, an approach resembling branch-and-price, which utilizes a nested dynamic program to solve subproblems heuristically, a diving heuristic, and a greedy heuristic leveraging the subproblem solver to tackle the given problem. Liu & Liu (2019) investigate a satisfaction-driven bi-objective workforce scheduling problem in which workers process different skills to operate the available parallel machines. The goal is to maximize workforce satisfaction and the number of on-time jobs (i.e., service level). Many approximate methods have been successfully applied. Simulated annealing approaches have been proposed by Anagnostopoulos & Rabadi (2012), Kim et al. (2002), and Lin & Ying (2015). Variable Neighborhood search algorithms have been studied by De Paula et al. (2007), Avalos-Rosales et al. (2015), and Çakırgil et al. (2020). Tabu search approaches have been considered by Yamashita (2000), Chen & Wu (2006), and Lee et al. (2013). Record-to-record algorithms have been introduced by Chen (2015), and local search heuristics have been proposed by Tsang & Voudouris (1995) and Xie et al. (2017). However, population algorithms for WSP problems have been published since 2006 (Cowling et al., 2006; Valls et al., 2009; Arnaout et al., 2010; Vallada & Ruiz, 2011). Recently, there has been a fast-growing population algorithm such as the work proposed by Joo & Kim (2015). This work proposes a genetic algorithm with three dispatch rules (processing time, total machine time, and task sequence).

Fozveh et al. (2016) propose a multiobjective mathematical model for scheduling multiskilled multiobjective workforce problem minimizing the number of night-shift engineers, the workforce's total cost, and maximizing the size of the engaged workforce. A bee colony optimization algorithm has been employed to solve the problem. Algethami et al. (2016) present a genetic algorithm (GA) for workforce scheduling and routing problems. Yurtkuran et al. (2018) solved a workforce scheduling and balancing problem of unpaced subassembly lines with buffers feeding a car manufacturer's paced body assembly line. The objective is to ascertain the minimal workforce for handling split lots at subassembly stations to supply the paced line within a recurring time frame. The authors introduce an enhanced artificial bee colony (ABC) algorithm with a solution acceptance rule and multi-search (SAMSABC). Afzalirad and Rezaeian (2016) proposed an artificial immune system exposed to bacteria or pathogens for solving scheduling problems. Besides, Algethami et al. (2019) consider the workforce scheduling and routing problem to refer to assigned personnel to visits across various geographical locations. The authors introduce an adaptive multiple crossover genetic algorithm to address combined scheduling and routing issues. Using online learning to gauge the operator's effectiveness, they assess a combination of problem-specific and conventional crossovers. Pereira et al. (2020) examined a novel workforce scheduling and routing challenge, the multiperiod workforce scheduling and routing problem with dependent tasks. The aim is to schedule and route teams to minimize makespan, ensuring that all services are completed within the shortest possible timeframe. The authors suggest a mixed-integer programming model based on the ant colony optimization (ACO) metaheuristic. Simeunović et al. (2015) propose a model for complex production systems with unpredictable and volatile demand. Recently, hybrid algorithms combining trajectory techniques with managing populations have been proposed by Abreu & Prata (2018). This solution method combines simulated annealing with a genetic algorithm by using three types of genetic operators (order crossover (OX), partially matched crossover (PMX), and cycle crossover (CX)).

Yaoyuenyong and Nanthavanij (2005) have proposed alternative hybrid approaches for a related problem. Their focus lies on a workforce scheduling issue aimed at minimizing the number of workers required to carry out daily physical tasks while ensuring that energy capacities remain unexceed. Two heuristics and one exact algorithm are then explained—Remde et al.

(2007) study a complex real-world workforce scheduling problem. The authors introduce a reduced variable neighborhood search (rVNS) and hyperheuristic approach to decide which subproblems to tackle. Li et al. (2012) present a hybrid approach of goal programming and metaheuristic search to find compromise solutions for a complex employee scheduling problem, i.e., nurse rostering with many complex and soft constraints. Musliu (2006) introduced novel heuristic methods for automatically generating rotating workforce schedules. The proposed methods include: (1) a tabu search (TS)-based algorithm; (2) a heuristic approach utilizing a min-conflict heuristic (MC); (3) a method combining the min-conflict heuristic (TS-MC) with random walk (TS-RW) within the tabu search algorithm; and (4) a method integrating the tabu mechanism (MC-T), random walk (MC-RW), and both mechanisms (MC-T-RW) within the min-conflicts heuristic. Becker (2020) explores rotational workforce planning, which involves constructing schedules comprising alternating work and rest periods. This study suggests a novel decomposition approach for rotational shift scheduling that applies to the rotating workforce scheduling problem structure and is potentially extendable to other problems.

This work proposes a novel evolutionary ant colony optimization (ACO) algorithm. The logic of the former algorithm is to generate feasible random solutions for the problem under consideration. Each ant leaves an artificial trace (pheromone) of the positive characteristics that make up the solution, thereby creating subsequent generations (Dorigo & Stützle, 2003). The proposed algorithm's efficiency has been compared with the efficiency obtained by the best previous works published in the specialized literature and by an international company's real instance. The proposed approach outperforms the best results found in the literature.

In Section 2, the proposed algorithm is illustrated. Section 3 presents the experiments' design, the developed application's computational results, and its corresponding analysis. Finally, in Section 4, the conclusions and future work resulting from this research are presented.

2. Proposed methodology

The ant colony optimization (ACO) algorithm, initially proposed by Dorigo (1992), has been inspired by the pheromone trail left by certain species of ants and their influence on the subsequent behavior of the community. ACO has been used to address the traveling postman and routing problems (Reinelt, 2003). Due to its good computational performance, the ACO algorithm has been considered to solve NP-Hard problems. ACO algorithms have been implemented for different problems, such as sequential order (Gambardella & Dorigo, 2000), scheduling (Blum, 2005), balancing of the assembly line (Blum, 2008), probabilistic TSP (Balaprakash et al., 2009), 2D-HP protein folding (Shmygelska & Hoos, 2005), DNA sequencing (Blum et al., 2008), protein-ligand coupling (Korb et al., 2007) and the packing-routing (Di Caro & Dorigo, 1998). The ACO's logic generates feasible solutions (ants) randomly based on the population's historical characteristics. Therefore, each ant must leave an artificial trail (pheromone).

Algorithm 1. ACO algorithm for combinatorial optimization problems

```
1  initialization
2  While Termination_condition is not met
3      ConstructAntSolutions
4      ApplyLocalSearch  *Optional
5      UpdatePheromones
6  End While
7  End ACO
```

Algorithm 1. General structure of ACO algorithm (Gendreau & Potvin, 2010)

We seek to maintain the ACO structure (see Algorithm 1) by incorporating affine operators into the mathematical and computational structure of the problem. Therefore, a random construction of solutions is carried out. These solutions could be improved if they pass an established filter. The idea of filtering is to maintain a balance between computational costs, improving inferior quality-built ants is not sought, and the refinement of promising ants is guaranteed.

The proposed ACO pseudocode is presented in Algorithm 2. The methodology seeks to propose an algorithm that can quickly adapt to other versions of the personnel assignment problem. The algorithm establishes the metaheuristic parameters, which consist of the number of ants, the number of generations, the values of α and β, and the threshold value for filtering ants (see Line 1 of Algorithm 1). The pheromone matrix (Pheromones) is initialized with the processing times of the tasks required in the available machines and the configuration times between tasks (see Line 2). Then, the algorithm's iterative scheme is entered. This scheme consists of generating the ants according to the pheromone matrix and the number (see Line 4). Then, an attempt is performed to improve them through a local search defined by the swap and insertion

operators (see Lines 5 to 16). Subsequently, the incumbent's value (the best and reached during the search) and the pheromone matrix (incumbent and pheromone in lines 14 and 17, respectively) are updated.

Algorithm 2. Proposed ACO

```
1   Establish_Parameters(#ants,#iterations,α,β, Incumbent)
2   Pheromones =  Process time
3   For i = 0 to #iterations
4       Population = ConstructionAntSolutions(Pheromones, #ants)
5                                                   For each Ant in Population
6           If FO(Ant) is in Umbral Then
7               While Improvement
8                   For j in Jobs
9                       For l in Jobs
10                          Savings.add(Swap(Ant,j,l)
11                          Savings.add(Insertion (Ant,j,l)
12                  Improvement = Applyoperators(Savings, Ant)
13              End While
14              UpdateIncumbent(Ant)
15          End If
16      End For
17      Updatepheromones(Pheromones)
18  End For
19  Return Incumbent
```

Algorithm 2. General Proposed Approach

The building phase sets the first job randomly (see Line 1 of Algorithm 3) and is assigned to the machine with the shortest total time (see Lines 2 and 3). Subsequently, as long as there are still jobs unassigned, the probabilities of assigning work to the machine are calculated based on (1), only for jobs not assigned based on artificial pheromones (see Lines 3 and 4). Finally, the criterion of assigning the jobs to the machine with the lowest total processing time is used (see Lines 6 and 7).

Algorithm 3. ConstructAntSolutions

```
1   Random = random_between (1, jobs)
2   Best_machine = best_machine (machines, Random)
3   First_job = assign (Best_machine, random)
4   While job is not assigned
5       Probability = equation (1) (job,pheromones) (1)
6       Best_machine = best_machine (machines, jobs)
7       Next_job = assign (Best_machine, Probability)
8   End while
```

Algorithm 3. Ant construction scheme

$$p\left(c_i^j \middle| s_p\right) = \frac{\pi_{ij}^{\alpha}\left[\eta\left(c_i^j\right)\right]^{\beta}}{\sum c_i^l \in N_{(sp)} \pi_{il}^{\alpha}\left[\eta\left(c_i^j\right)\right]^{\beta}} \qquad \forall c_i^j \in N_{(sp)} \tag{1}$$

Eq. (1) allows assigning a probability to the unassigned jobs based on their execution time and the available information on pheromones. In particular, η is the probabilistic value, i.e., the function assigned to each feasible component of a solution;

this value is commonly called heuristic information. Parameters α and β determine the influence of the heuristic information on the algorithm. If $\alpha = 0$, the probabilities are proportional to each work's time unit, and thus, the algorithm's behavior would be similar to a GRASP approach. Additionally, if $\beta = 0$, the probability calculation considers the heuristic information collected from the population.

However, each ant can be modified (*improved*) depending on whether its objective function is within the established *threshold*. Two types of operators are estimated to apply the transition achieving the most significant time savings while searching for a promising solution. Therefore, the swap operator and insert operator are estimated. The swap operator consists of substituting a job i to be performed on machine k for a job j to be performed on machine l. Jobs can be performed on the same machine (intramachine) when $k = l$. Likewise, jobs can be assigned to different machines (intermachine) when $k \neq l$. Details of the operators are illustrated below.

Intra-machine: In Fig. 1, job $i = 1$ is to be performed on machine k, and job $j = 8$ is to be performed on machine l, where $l = k$. Then, the operator changes the sequence of jobs.

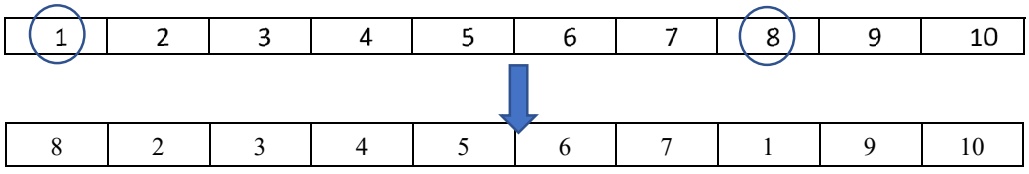

Fig. 1. Swap operator of intra-machine.

Intermachine: In Fig. 2, job $i = 1$ is to be performed on machine k, and job $j = 18$ is to be performed on machine l, where $k \neq l$. Then, the operator changes the sequence of machines k and l.

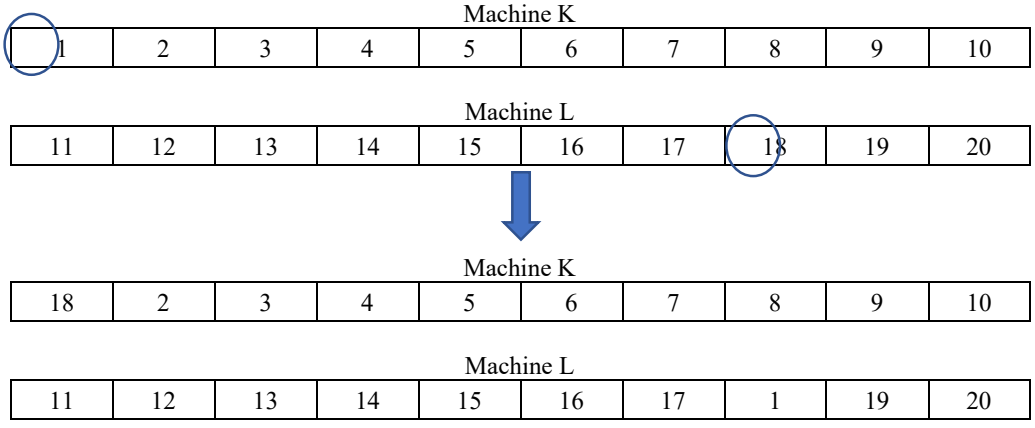

Fig. 2. Swap operator of inter-machine.

Finally, the insertion operator consists of inserting a job i to be performed on machine k in the sequence of jobs performed on machine l. Suppose we have a job $i = 1$ inserted into machine l where $k \neq l$; its insertion is performed in the first position of the sequence formed by the jobs on machine l.

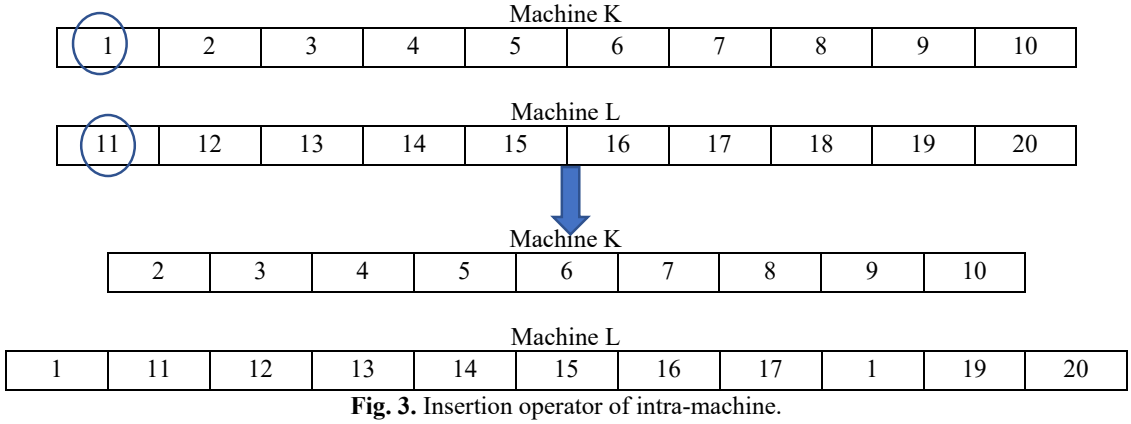

Fig. 3. Insertion operator of intra-machine.

Therefore, the insertion operator transforms machines k and l, as shown in Fig. 3. The proposed algorithm is coded to analyze all possible positions in the job scheduled l, and the proposed algorithm carries out an exhaustive search until the best movements are found. The movements are carried out methodically: the best ones are applied first until all machines have been altered or no improvements are found.

3. Computational results

3.1. Parameters calibration

The set of instances proposed by Abreu & Prata (2018) and a real instance have been used to validate the performance of the proposed methodology. A total of 40 instances of the literature have been used to test the algorithm efficiency. Instances combine 2, 4, 8, and 12 machines and 10, 25, 50, and 100 jobs. These instances are published in Abreu (2019). The results were obtained after executing the algorithm on a computer with an Intel® Core ™ i5-6200U processor and 4.00 GB of memory. The proposed algorithm was implemented in the C# programming language and was executed 30 times for each instance due to immersed randomness. The established parameters were selected experimentally (design experiments with two levels for each variable). The entire battery of test cases previously described and the values of the intervals recommended in the literature were used. The best values of the parameters are illustrated in Table 1.

Table 1
Best values of the parameters

Parameter	Obtained Value
Number of ants	10
Number of generations	10
α	1
β	2
Incumbent	0.3

Source: Owner

3.2. Obtained Results

Table 2 presents the proposed approach's results. These results include the parameter values with which the proposed approach obtained the best performance. Additionally, the genetic algorithm (GA) proposed by Abreu and Prata (2018) has been considered the best solution published in the literature. The information is organized as follows. First, the best solution obtained is compared considering the 30 executions of the proposed ACO versus the AG's ten executions, thus illustrating the solution's value, the computing time, and the GAP (percentage differential between solutions). Next, the average behavior of both methodologies is compared, thus illustrating the average value of the solutions and their GAP. Finally, the confidence intervals of the proposed algorithm illustrate the behavior of the solution's value reached for the sample of 30 executions. The column headings in Table 2 are formalized below.

- **ID:** Name of the Instance
- **Best ACO:** Best results obtained after 30 executions carried out with the proposed algorithm.
- **Best AG:** Best result obtained by Abreu & Prata (2018)
- **Best GAP:** Variation with respect to the results obtained by Abreu & Prata (2018). The value is obtained as follows: $\text{Best GAP} = \frac{(\text{Best AG} - \text{Best ACO})}{\text{Best AG}} \times 100$
- **ACO Time:** Average execution time of the results obtained after 30 iterations of the ACO approach
- **AG Time**: Average execution time reported by Abreu & Prata (2018)
- **Average ACO:** Average of the results obtained after 10 executions performed by the ant colony algorithm
- **Average AG:** Average value reported by Abreu & Prata (2018)
- **Average GAP:** Variation with respect to the results obtained by Abreu & Prata (2018). The value is obtained as follows: $\text{Average GAP} = \frac{(\text{Average AG} - \text{Average ACO})}{\text{Average AG}} \times 100$
- **CI-:** The lower limit of the ACO algorithm runs with a 95% confidence interval.
- **CI+:** Upper limit of the ACO algorithm runs with a 95% confidence interval.

Table 2
Results obtained by ACO for the set of benchmarking instances

ID	Best ACO	Best AG	Best GAP	ACO Time	AG Time	Average ACO	Average AG	Average GAP	CI -	CI +
MP-10x01	**62.00**	**62.00**	0.00	0.10	3.53	62.63	89.00	-29.63	61.14	63.13
MP-10x02	**72.00**	**72.00**	0.00	0.09	3.56	72.40	106.00	-31.70	72.21	72.59
MP-10x03	**61.00**	**61.00**	0.00	0.04	3.54	64.73	105.00	-38.35	63.18	66.29
MP-10x04	**61.00**	**61.00**	0.00	0.08	3.54	64.27	100.00	-35.73	63.46	65.07
MP-10x05	**68.00**	**68.00**	0.00	0.08	3.51	71.80	106.00	-32.26	70.84	72.76
MP-10x06	**80.00**	**80.00**	0.00	0.10	3.56	81.20	120.00	-32.33	80.43	81.97
MP-10x07	**73.00**	**73.00**	0.00	0.64	3.53	78.60	113.00	-30.44	78.03	79.17
MP-10x08	**65.00**	**65.00**	0.00	0.08	3.52	67.20	103.00	-34.76	65.92	68.48
MP-10x09	**59.00**	**59.00**	0.00	0.09	3.49	60.93	96.00	-36.53	59.88	61.98
MP-10x10	**67.00**	**67.00**	0.00	0.07	3.54	71.93	108.00	-33.40	70.75	73.12
MP-25x01	**68.00**	**68.00**	0.00	3.50	8.07	70.87	100.00	-29.13	70.13	71.6
MP-25x02	69.00	**66.00**	4.55	3.89	8.13	76.00	113.00	-32.74	74.89	77.11
MP-25x03	**73.00**	75.00	-2.67	4.00	8.14	80.97	115.00	-29.59	80.22	81.71
MP-25x04	73.00	**69.00**	5.80	3.24	8.10	77.67	111.00	-30.03	76.63	78.71
MP-25x05	74.00	**72.00**	2.78	3.19	8.07	80.13	116.00	-30.92	79.11	81.16
MP-25x06	78.00	**73.00**	6.85	3.86	8.14	81.50	117.00	-30.34	80.66	82.32
MP-25x07	69.00	**68.00**	1.47	3.81	8.12	74.70	113.00	-33.89	73.82	75.58
MP-25x08	74.00	**70.00**	5.71	3.41	8.14	76.37	115.00	-33.59	75.53	77.2
MP-25x09	75.00	**73.00**	2.74	3.45	8.07	81.27	119.00	-31.71	80.28	82.25
MP-25x10	**71.00**	72.00	-1.39	3.73	8.06	77.00	116.00	-33.62	75.78	78.22
MP-50x01	132.00	**125.00**	5.60	3.54	21.17	139.60	176.00	-20.68	137.16	142.04
MP-50x02	**232.00**	237.00	-2.11	19.27	21.40	243.27	288.00	-15.53	241.46	245.07
MP-50x03	238.00	**235.00**	1.28	8.25	21.27	245.33	292.00	-15.98	243.9	246.77
MP-50x04	238.00	**236.00**	0.85	25.16	21.39	248.87	290.00	-14.18	246.92	250.82
MP-50x05	235.00	**231.00**	1.73	27.07	21.27	245.80	287.00	-14.36	243.58	248.03
MP-50x06	230.00	**227.00**	1.32	17.73	21.97	242.87	284.00	-14.48	240.52	245.21
MP-50x07	239.00	**232.00**	3.02	44.98	21.19	244.87	289.00	-15.27	243.23	246.5
MP-50x08	239.00	**235.00**	1.70	14.58	21.31	245.57	287.00	-14.44	243.84	247.3
MP-50x09	237.00	**235.00**	0.85	9.96	21.62	243.30	286.00	-14.93	241.31	245.29
MP-50x10	238.00	**233.00**	2.15	13.76	21.43	245.90	288.00	-14.62	244.18	247.62
MP-100x01	**297.00**	306.00	-2.94	70.44	59.04	316.80	368.00	-13.91	314.21	319.39
MP-100x02	**302.00**	308.00	-1.95	90.24	59.11	318.80	366.00	-12.90	316.22	321.38
MP-100x03	**308.00**	312.00	-1.28	55.40	61.24	320.77	371.00	-13.54	318.65	322.88
MP-100x04	**307.00**	**307.00**	0.00	84.52	58.94	318.03	369.00	-13.81	315.84	320.23
MP-100x05	**309.00**	319.00	-3.13	56.44	59.00	317.90	366.00	-13.14	315.93	319.87
MP-100x06	**299.00**	327.00	-8.56	93.22	59.18	316.53	370.00	-14.45	313.22	319.85
MP-100x07	**308.00**	319.00	-3.45	83.23	58.94	315.03	368.00	-14.39	312.94	317.13
MP-100x08	**310.00**	329.00	-5.78	68.50	59.13	321.37	370.00	-13.14	319.78	322.95
MP-100x09	**311.00**	317.00	-1.89	63.97	58.92	320.80	370.00	-13.30	318.98	322.62
MP-100x10	**305.00**	335.00	-8.96	84.13	59.26	316.13	367.00	-13.86	314.11	318.16
Average			**0.11**	**24.30**	**23.08**	**174.99**	**215.83**	**-23.54**		

Source: Owner

The instances are grouped according to the number of available machines (2, 4, 8, and 12) to synthesize the information in Table 3. For each group, the average percentage of variation was found concerning the results obtained by Abreu & Prata (2018).

Table 3
Summary of the obtained GAPs

Number of machines	Average GAP (%)	Best GAP (%)
2	-33.51	0.00
4	-31.56	2.58
8	-15.45	1.64
12	-13.64	-3.79
Average	**-23.54**	**0.11**

Source: Owner

The analysis of the performance of the proposed algorithm must be approached from three points of view:

- **Best solutions:** The first section found better solutions for 12 instances out of 40. Nine of them were found in the large set. The best result reported in the literature was equal in 12 instances, 10 of which belonged to small instances. In 26 instances, with a maximum deviation of 6.85% for the MP-25x06 instance, the solution built by ACO is inferior to that indicated in previously published research.

- **Computing time:** The proposed algorithm is -34.68% faster than that proposed by Abreu & Prata (2018). However, for the set of larger instances, the average is 26.66% slower due to the high computational cost of the exhaustive search performed in executing the operators implemented in ACO. Indeed, it is necessary to consider the equipment's technical characteristics where the tests have been executed. The used computer is categorized as high to midrange with a score of 4,020. In the work proposed by Abreu & Prata (2018), the computer is categorized as low to midrange with a score of 1,434, according to Passmark (2019).

- **Average value of solutions:** The main strength of the proposed algorithm is its stability. In particular, the average found in all instances is lower than that presented in the literature (23.54%). Finally, a confidence interval with a significance level of 95% is constructed for the average values to show more evidence. The upper limit of the interval is lower than the average reported in the literature.

3.3. Case of study

The company Case of Study is a software development organization specializing in storage logistics. User requirements for application failures or modifications have different levels of technical complexity (Table 4).

Table 4
Type of requirements

Requirement Level	Description
A	Custom installation and developments
B	Logic configuration and operational start-up
C	Report design (Excel, PDF, plain text)
D	Basic functionalities of the program

Source: Owner

The support department has four engineers dedicated to addressing and solving requirements. Now, the categorization of the engineers' capacity to solve each level's requests is given by years of experience in handling the applications, as shown in Table 5.

Table 5
Classification of Engineers

Level	Number of Engineers	Years of experience	Requirement Level
1	1	2 years or more	A, B
2	2	Between 1 and 2 years	B, C, D
3	1	Less than 1 year	C, D

Source: Owner

The level 1 engineer must characterize the requirement as A, B, C, or D. This person must consider the information listed in Table 4 and define the type of request he or she has entered. Subsequently, the engineer must consider the criteria indicated in Table 5 to assign the task to a work team member. Therefore, it is necessary to analyze the following factors: (the type of requirement, availability, and experience of the personnel). Finally, the status of the solution must be verified before the response is given to the customer. The level 1 engineer, as the head of support, performs tasks manually since they do not support technological requirements management software, whose nature makes them challenging to manage. In technological services organizations, directors allocate at least one day of the week for scheduling activities of their work team (Bechtold & Brusco, 1994). Therefore, the level of service suffers when the allocation work is not performed promptly.

Assigning requests to engineers (the support team) can be modeled as assigning tasks to machines to minimize the total time for completing tasks. In particular, engineers are interpreted as machines that work in parallel, where the time required to complete the same task can vary between them; in the literature, these machines are known as unrelated parallel machines. However, requests can be modeled as tasks to be performed sequentially on machines, where the preparation times of tasks depend on the sequence of the performed tasks. This problem is known in the literature as the sequence-dependent setup time. Thus, the company's problem can be modeled as an unrelated parallel machine scheduling problem with sequence-dependent setup times (UPMSPST).

The impact on the company's results is significant since informal means to solve the problem are replaced with the proposed ACO. A tool has been implemented, classifying the received requests according to the type of requirement and appropriately assigning them to the engineers (Tables 4 and 5). The developed application works with storage in the SQL Server database engine. This database has three tables (Machine, Jobs, and Setup) fed by the user directly from the application. The sequence obtained for each machine is detailed on a Gantt chart.

4. Concluding Remarks

This paper proposes a metaheuristic algorithm for the staffing problem modeled as an unrelated parallel machine scheduling problem with a sequence-dependent setup time problem (UPMSPST). The algorithm's structure can be extended to address other scheduling problems by modifying and removing some constraints. Regarding the logic presented, there are two decisive factors in the positive performance of the proposed approach. The first is establishing a threshold for improving solutions since it helps select the promising solutions to which the operators in search of improvements should be applied. In computational terms, this factor significantly affects the execution time. However, establishing probabilities according to information from previously created individuals improves the quality of solutions over time. This pseudorandom behavior allows refinement of the creation, diversification, and introduction of new individuals in the population.

The obtained results by the proposed algorithm outperform the literature results in terms of the best solutions found and the average solutions. For future work, parameters α, β, and Threshold could be calculated dynamically over time in response to various factors, such as other parameters of the algorithm or information obtained during optimization, including the number of generations, the size of the population, and the average number of ants that pass the filter. However, to further diversify the population, it would be convenient to create a genetic mutation operator that drastically modifies some solutions, thereby helping to add a degree of exploration to the algorithm. Additionally, it is proposed to adapt ACO to other variants of the problem while considering, for example, the minimization of delay.

References

Abreu, L. R., & Prata, B. A. (2018). A Hybrid Genetic Algorithm for Solving the Unrelated Parallel Machine Scheduling Problem with Sequence Dependent Setup Times. *IEEE Latin America Transactions, 16*(6), 1715-1722.

Abreu, L. R., (2019). Set of instances for the Unrelated Parallel Machine Scheduling problem with Sequence Dependent Setup Times. Available in: https://www.researchgate.net/publication/315771587_Instances_Tested. Consulted 10 of January of 2019.

Afzalirad, M., & Rezaeian, J. (2016). Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions. *Computers & Industrial Engineering, 98*, 40-52.

Algethami, H., Pinheiro, R. L., & Landa-Silva, D. (2016). A genetic algorithm for a workforce scheduling and routing problem. In 2016 IEEE Congress on Evolutionary Computation (CEC) (pp. 927-934). IEEE.

Algethami, H., Martínez-Gavara, A., & Landa-Silva, D. (2019). Adaptive multiple crossover genetic algorithm to solve workforce scheduling and routing problem. *Journal of Heuristics, 25*(4), 753-792.

Allahverdi, A., Gupta, J. N., & Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *Omega, 27*(2), 219-239.

Anagnostopoulos, G. C., & Rabadi, G. (2002). A simulated annealing algorithm for the unrelated parallel machine scheduling problem. In Proceedings of the 5th Biannual world automation congress (Vol. 14, pp. 115-120). IEEE.

Arnaout, J. P., Rabadi, G., & Musa, R. (2010). 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.

Avalos-Rosales, O., Angel-Bello, F., & Alvarez, A. (2015). Efficient metaheuristic algorithm and re-formulations for the unrelated parallel machine scheduling problem with sequence and machine-dependent setup times. *The International Journal of Advanced Manufacturing Technology, 76*(9-12), 1705-1718.

Baker, K. R., & Magazine, M. J. (1977). Workforce scheduling with cyclic demands and day-off constraints. *Management Science, 24*(2), 161-167.

Balaprakash, P., Birattari, M., Stützle, T., Yuan, Z., & Dorigo, M. (2009). Estimation-based ant colony optimization and local search for the probabilistic traveling salesman problem. *Swarm Intelligence, 3*(3), 223-242.

Barrera, D., Velasco, N., & Amaya, C. A. (2012). A network-based approach to the multi-activity combined timetabling and crew scheduling problem: Workforce scheduling for public health policy implementation. *Computers & Industrial Engineering, 63*(4), 802-812.

Bechtold, S. E., & Brusco, M. J. (1994). Working set generation methods for labor tour scheduling. *European Journal of Operational Research, 74*(3), 540-551.

Becker, T. (2020). A decomposition heuristic for rotational workforce scheduling. *Journal of Scheduling, 23*(5), 539-554.

Billionnet, A. (1999). Integer programming to schedule a hierarchical workforce with variable demands. *European Journal of Operational Research, 114*(1), 105-114.

Blum, C. (2005). Beam-ACO—Hybridizing ant colony optimization with beam search: An application to open shop scheduling. *Computers & Operations Research, 32*(6), 1565-1591.

Blum, C. (2008). Beam-ACO for simple assembly line balancing. *INFORMS Journal on Computing, 20*(4), 618-627.

Blum, C., Vallès, M. Y., & Blesa, M. J. (2008). An ant colony optimization algorithm for DNA sequencing by hybridization. *Computers & Operations Research, 35*(11), 3620-3635.

Çakırgil, S., Yücel, E., & Kuyzu, G. (2020). An integrated solution approach for multi-objective, multi-skill workforce scheduling and routing problems. *Computers & Operations Research, 118*, 104908.

Castillo, I., Joro, T., & Li, Y. Y. (2009). Workforce scheduling with multiple objectives. *European Journal of Operational Research, 196*(1), 162-170.

Chen, J. F., & Wu, T. H. (2006). Total tardiness minimization on unrelated parallel machine scheduling with auxiliary equipment constraints. *Omega, 34*(1), 81-89.

Chen, J. F. (2015). Unrelated parallel-machine scheduling to minimize total weighted completion time. *Journal of Intelligent Manufacturing, 26*(6), 1099-1112.

Cheng, T. C. E., & Sin, C. C. S. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research, 47*(3), 271-292.

Cowling, P., Colledge, N., Dahal, K., & Remde, S. (2006). The trade off between diversity and quality for multi-objective workforce scheduling. *In European Conference on Evolutionary Computation in Combinatorial Optimization* (pp. 13-24). Springer, Berlin, Heidelberg.

De Bruecker, P., Van den Bergh, J., Beliën, J., & Demeulemeester, E. (2015). Workforce planning incorporating skills: State of the art. *European Journal of Operational Research, 243*(1), 1-16.

De Paula, M. R., Ravetti, M. G., Mateus, G. R., & Pardalos, P. M. (2007). Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. *IMA Journal of Management Mathematics, 18*(2), 101-115.

Di Caro, G., & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research, 9*, 317-365.

Dorigo, M. (1992). Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano.

Dorigo, M., & Stützle, T. (2003). *The ant colony optimization metaheuristic: Algorithms, applications, and advances*. In Handbook of metaheuristics (pp. 250-285). Springer, Boston, MA.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European journal of operational research, 153*(1), 3-27.

Fırat, M., & Hurkens, C. A. (2012). An improved MIP-based approach for a multi-skill workforce scheduling problem. *Journal of Scheduling, 15*(3), 363-380.

Florez, L., Castro-Lacouture, D., & Medaglia, A. L. (2013). Sustainable workforce scheduling in construction program management. *Journal of the Operational Research Society, 64*(8), 1169-1181.

Fozveh, K. I., Salehi, H., & Mogharehabed, K. (2016). Presentation of Multi-Skill Workforce Scheduling Model and Solving the Model Using Meta-Heuristic Algorithms. *Modern Applied Science, 2*(10), 194-205.

Gambardella, L. M., & Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing, 12*(3), 237-255.

Gendreau, M., & Potvin, J. Y. (Eds.). (2010). *Handbook of metaheuristics* (Vol. 2, p. 9). New York: Springer.

Gérard, M., Clautiaux, F., & Sadykov, R. (2016). Column generation based approaches for a tour scheduling problem with a multi-skill heterogeneous workforce. *European Journal of Operational Research, 252*(3), 1019-1030.

Joo, C. M., & Kim, B. S. (2015). Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability. *Computers & Industrial Engineering, 85*, 102-109.

Kim, D. W., Kim, K. H., Jang, W., & Chen, F. F. (2002). Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer-Integrated Manufacturing, 18*(3-4), 223-231.

Korb, O., Stützle, T., & Exner, T. E. (2007). An ant colony optimization approach to flexible protein–ligand docking. *Swarm Intelligence, 1*(2), 115-134.

Kuo, W. H., Hsu, C. J., & Yang, D. L. (2011). Some unrelated parallel machine scheduling problems with past-sequence-dependent setup time and learning effects. *Computers & Industrial Engineering, 61*(1), 179-183.

Laesanklang, W., Silva, D. L., & Castillo-Salazar, J. A. (2015). Mixed Integer Programming with Decomposition to Solve a Workforce Scheduling and Routing Problem. In ICORES (pp. 283-293).

Lee, J. H., Yu, J. M., & Lee, D. H. (2013). A tabu search algorithm for unrelated parallel machine scheduling with sequence- and machine-dependent setups: minimizing total tardiness. *The International Journal of Advanced Manufacturing Technology, 69*(9-12), 2081-2089.

Li, J., Burke, E. K., Curtois, T., Petrovic, S., & Qu, R. (2012). The falling tide algorithm: a new multi-objective approach for complex workforce scheduling. *Omega, 40*(3), 283-293.

Lin, S. W., & Ying, K. C. (2015). A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems. *International Journal of Production Research, 53*(4), 1065-1076.

Liu, M., & Liu, X. (2019). Satisfaction-driven bi-objective multi-skill workforce scheduling problem. *IFAC-PapersOnLine, 52*(13), 229-234.

Moodie, C. L., & Roberts, S. D. (1967). Experiments with priority dispatching rules in a parallel processor shop. *International Journal of Production Research, 6*(4), 303-312.

Muntz, R. R., & Coffman, E. G. (1969). Optimal preemptive scheduling on two-processor systems. *IEEE Transactions on Computers, 100*(11), 1014-1020.

Musliu, N. (2006). Heuristic methods for automatic rotating workforce scheduling. *International Journal of Computational Intelligence Research, 2*(4), 309-326.

Othman, M., Gouw, G. J., & Bhuiyan, N. (2012). Workforce scheduling: A new model incorporating human factors. *Journal of Industrial Engineering and Management (JIEM), 5*(2), 259-284.

Passmark, (2019). Sitio web especializado para benchmark de computadores. Disponible en línea: http://www.passmark.com. Consultado 20 February of 2019.

Pereira, D. L., Alves, J. C., & de Oliveira Moreira, M. C. (2020). A multiperiod workforce scheduling and routing problem with dependent tasks. *Computers & Operations Research, 118*, 104930.

Pinedo, M. (2016). Scheduling Theory, Algorithms, and Systems. ed. 5, Editorial Springer, Cap 5.

Reinelt, G. (2003). The traveling salesman: computational solutions for TSP applications (Vol. 840). Springer.

Remde, S., Cowling, P., Dahal, K., & Colledge, N. (2007). Exact/heuristic hybrids using rVNS and hyperheuristics for workforce scheduling. *In European Conference on Evolutionary Computation in Combinatorial Optimization* (pp. 188-197). Springer, Berlin, Heidelberg.

Seçkiner, S. U., Gökçen, H., & Kurt, M. (2007). An integer programming model for hierarchical workforce scheduling problem. *European Journal of Operational Research, 183*(2), 694-699.

Shmygelska, A., & Hoos, H. H. (2005). An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. *BMC bioinformatics, 6*(1), 1-22.

Simeunović, N., Kamenko, I., Bugarski, V., Jovanović, M., & Lalić, B. (2017). Improving workforce scheduling using artificial neural networks model. *Advances in Production Engineering & Management, 12*(4), 337-352.

Thompson, G. M., & Goodale, J. C. (2006). Variable employee productivity in workforce scheduling. *European Journal of Operational Research, 170*(2), 376-390.

Tsang, E., & Voudouris, C. (1997). Fast local search and guided local search and their application to British Telecom's workforce scheduling problem. *Operations Research Letters, 20*(3), 119-127.

Vallada, E., & Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research, 211*(3), 612-622.

Valls, V., Pérez, Á., & Quintanilla, S. (2009). Skilled workforce scheduling in service centres. *European Journal of Operational Research, 193*(3), 791-804.

Xie, F., Potts, C. N., & Bektaş, T. (2017). Iterated local search for workforce scheduling and routing problems. *Journal of Heuristics, 23*(6), 471-500.

Yamashita, D. S. (2000). Tabu search for scheduling on identical parallel machines to minimize mean tardiness. *Journal of Intelligent Manufacturing, 11*(5), 453-460.

Yaoyuenyong, K., & Nanthavanij, S. (2005). Energy-based workforce scheduling problem: mathematical model and solution algorithms. *ScienceAsia, 31*, 383-93.

Yurtkuran, A., Yagmahan, B., & Emel, E. (2018). A novel artificial bee colony algorithm for the workforce scheduling and balancing problem in sub-assembly lines with limited buffers. *Applied Soft Computing, 73,* 767-782.