IJMSOR Vol. 1, Issue. 1
January - June, 2016

Multiobjective Scheduling using an Ant Colony System in a Mineral Analysis Laboratory

Programacion Multiobjetivo de las Operaciones en un Laboratorio de Análisis de Minerales usando Colonia de Hormigas

dx.doi.org/10.17981/ijmsor.01.01.02

Research Article - Reception Date: February 4, 2015 - Acceptance Date: April 7, 2016

Alexander Bustacara Prasca
Universidad del Atlántico
alexanderbustacaraa@gmail.com, Barranquilla, Colombia

Eduardo Becerra Rueda
Universidad del Atlántico
edbecerra4@yahoo.com, Barranquilla, Colombia

Fabricio Niebles-Atencio
Universidad de la Costa
fabricioniebles@gmail.com, Barranquilla, Colombia

To reference this paper:
A. Bustacara Prasca, E. Becerra Rueda and F. Niebles-Atencio "Multiobjective Scheduling using an Ant Colony System in a Mineral Analysis Laboratory", *IJMSOR*, vol. 1, no. 1, pp. 8-19, 2016. DOI: dx.doi.org/10.17981/ijmsor.01.01.02

Abstract-- This paper considers the problem of scheduling a given set of samples in a mineral laboratory, located in Barranquilla Colombia. Taking into account the natural complexity of the process and the large amount of variables involved, this problem is considered as NP-hard in strong sense. Therefore, it is possible to find an optimal solution in a reasonable computational time only for small instances, which in general, does not reflect the industrial reality. For that reason, it is proposed the use of metaheuristics as an alternative approach in this problem with the aim to determine, with a low computational effort, the best assignation of the analysis in order to minimize the makespan and weighted total tardiness simultaneously. These optimization objectives will allow this laboratory to improve their productivity and the customer service, respectively. A Multi-objective Ant Colony Optimization algorithm (MOACO) is proposed. Computational experiments are carried out comparing the proposed approach versus exact methods. Results show the efficiency of our MOACO algorithm.

Keywords-- Scheduling; Ant Colony Optimization; Multiobjective Optimization

Resumen-- Este trabajo considera el problema de programar un conjunto de muestras en un laboratorio de análisis de minerales ubicado en Barranquilla Colombia. Teniendo en cuenta la complejidad intrínseca del proceso y la gran cantidad de variables involucradas, este problema es considerado como NP-duro en sentido estricto. Por lo tanto, es posible encontrar una solución óptima en un tiempo razonablemente corto solo para instancias pequeñas, las cuales en general no reflejan la realidad en la industria. Por esta razón, se propone el uso de metaheurísticas como enfoque alternativo en este problema con el fin de determinar, con un costo computacional bajo, la mejor secuencia para el análisis de las muestras que optimice el makespan y la tardanza total ponderada simultáneamente. Estos objetivos de optimización permitirán al laboratorio mejorar su productividad y el servicio al cliente respectivamente. Un Algoritmo Multiobjetivo de Colonia de Hormigas (MOACO) es presentado aquí. Experimentos computacionales son realizados para comparar el algoritmo propuesto con respecto a métodos exactos. Los resultados obtenidos muestran la eficiencia de nuestro algoritmo MOACO.

Palabras claves-- Programación de Operaciones; Optimización por Colonia de Hormigas; Optimización Multiobjetivo

Address correspondence to Fabricio Niebles-Atencio, Universidad de la Costa, Faculty of Engineering, Department of Industrial Engineering, Calle 50 N° 55-66. Barranquilla, Colombia, E-mail:fabricioniebles@gmail.com

IJMSOR vol. 1 no. 1, pp. 8-19. January - December, 2016
Barranquilla. ISSN 2539-5416 Online

I. Introduction

Scheduling is one of the hard optimization problems found in real industrial contexts. Generally speaking, scheduling is a form of decision-making that plays a crucial role in manufacturing and service industries. According to Pinedo [1], scheduling problems deal with "the allocation of resources to tasks over given time periods and its goal is to optimize one or more objectives". This work focuses on the Scheduling process for a specific configuration of a mineral laboratory located in Barranquilla (Colombia), which is in charge of reception, identification, preparing and analysis on samples of coal and coke according to the customer requirements either for certifying the quality of the material and evaluating the feasibility to open a coal mine, or for selling the coal after mining, or even for knowing the physics and chemical properties of it for a customer utilization, for example in thermoelectric plants and steel companies. Such samples might come from Explorations, where samples are obtained by perforation processes. Exploitations and Development, in which the samples are obtained from exploitation fronts, stationary or during samples transportation in piles, ships, wagons, trucks, conveyors, etc; also, some samples come from customers directly. These different ways in which the laboratory receives the samples and the variability of the coal market together with external problems related to the mining sector (union strikes, closing of mines for breaking the law, and the bad conditions of colombian roads, etc.), generates that the demand of the laboratory can't be known in advanced and therefore, is quite difficult to plan or schedule the samples for 46 types of analysis that the laboratory carries out.

On the other hand, a specific sample might require a previous preparation or not and when it arrives to the laboratory, it is possible one or more analysis to be required depending of customer needs. Thus, taking into account the high volume of samples and the quantity of analysis for each one, the capacity of laboratory is -in some cases- exceeded, which is estimated in 70 samples for preparation during a day trip (8 hours) and for processing is among 35 and 45 samples for a day trip, depending on the analysis performed. This situation generates an accumulation of samples in a queue and, therefore, a delay in the deliver to customers. Additionally, the number and types of analysis to be performed to a sample is known only when this sample is received and identified; this, added to the samples that are being processed and the priority system –which is defined depending on the customer importance-, makes impossible to estimate with precision the due date.

At this moment, the scheduling of samples in the laboratory is done in a manual way at the beginning of a day trip taking into account the analysis in process and those that couldn't be completed on the last day trip and the availability of machines and workers. The analysis is done in this way because the laboratory does not have a formal methodology that allows performing this tasks in an effective way and therefore taking better decisions in the manner of assigning samples to the resources so that the laboratory can supply the demand, and at the same time increase the utilization rate of machines and decrease the idle time of this machines. Therefore, taking into account the natural complexity of the process and the large amount of variables involved, the scheduling process is considered as a NP-hard problem in strong sense.

As in a large number of real-life optimization problems in economics and business, the NP-hardness of a scheduling problem means that it cannot be solved in an exact (optimal) manner within a reasonable amount of time. Hence, the use of approximate algorithms is the main alternative to solve this type of problems. According to Talbi [2], approximate algorithms can be classified in two classes: dedicated heuristics and meta-heuristics. The former are problem-dependent and are designed and applicable to a particular problem. The latter are called meta-heuristics procedures and represent more general approximate algorithms applicable to a large variety of optimization problems. Meta-heuristics solve instances of problems that are believed to be hard in general, by exploring the usually large solution search space of these instances. These algorithms achieve this by reducing the effective size of the space and by exploring that space efficiently.

With the improvement of computing performance, the past 25 years have witnessed the development of numerous meta-heuristic algorithms in various communities that sit at the intersection of several fields, including Artificial Intelligence, Computational Intelligence, Soft Computing, Mathematical Programming, and Operational Research. Most of the meta-heuristics imitate natural metaphors to solve complex optimization problems (e.g., evolution of species, annealing process, ant colony, particle swarm, immune system, bee colony, and wasp swarm). Meta-heuristics are more and more popular in different research areas and industries. Scheduling is one of the hard optimization problems found in real industrial contexts for which several meta-heuristic procedures have been successfully applied [3]. Among the set up to date available meta-heuristic algorithms, Ant Colony Optimization (ACO) emulates the behaviour of "real" ants to find the optimal path between their nest and a food source.

The objective of this paper is to solve a multi-objective scheduling problem in a real context, which is a mineral laboratory. Objective functions are defined as the minimization of the total completion time of all jobs (or makespan) and total weighted tardiness. Our solution approach is based on a Multi-objective Ant Colony Optimization Algorithm. As presented next in this paper, ACO algorithms have been used to solve various combinatorial optimization problems, including different scheduling problems [4]. For the particular case of a mineral laboratory, to the best of our knowledge there is no evidence of using metaheuristics for scheduling in this kind of configuration. However, for scheduling problems the use of metaheuristics is very effective and widely used as in academic context like in real problems.

The rest of this paper is organised as follows: Section 2 is devoted to present the review of literature related to the solution of some particular scheduling problems. Section 3 shows the formulation and mathematical model of the problem under study. Section 4 presents in detail the proposed ACO algorithm, while Section 5 and 6 is devoted to computational experiments and the analysis of results. This paper ends in Section 7 by presenting some concluding remarks and suggestions for further research.

II. Literature Review

The scientific literature has extensively reported academic works and real-life applications regarding the utilization of metaheuristics for solving scheduling problems. Although most of these works have focused on single objective applications, we have some applications that involve the optimization of two or more objectives simultaneously, especially in shop scheduling problems [5]-[6]. Lei [7] presented a survey paper of Multiobjective scheduling problems for production planning. In the same way, various intelligent heuristics and meta-heuristics have become popular such as Simulated Annealing (SA), Tabu Search (TS), Multi-Agent System (MAS), Genetic Algorithm (GA) and ACO [8]. For example, authors like: [10], [11], [12], and [13], have used those metaheuristics for solving different kind of shop scheduling problems where one of more objectives are optimized and – in general - one of them is the makespan. At the same way, authors like: [11] [12], and [14] have used some heuristics and metaheuristics for different types of scheduling problems like production planning, aircraft schedule or route planning.

On the other hand, ACO approaches imitate the behavior of real ants when searching for food. Some observations have shown that although an ant has limited capacities, it can with the collaboration of the other ants; find the shortest path from a food source to their nest without visual cue. To perform complex tasks, a colony of ants uses a chemical substance called "pheromone", which they secrete as they move along. The pheromone provides ants with the ability to communicate with each other. Being very sensitive to this substance, an ant chooses in a randomly way the path comprising a strong concentration of this substance. Thus, when several ants cross the same space, an emergence of the shortest path is obtained. The ACO algorithms use systems formed by several artificial ants. These latter not only simulate the behavior of real ants described above, but also (i) apply additional problem- specific heuristic information, (ii) can manage the deposited quantity of pheromone according to the quality of the solution; moreover it is possible to have various types of pheromone, and (iii) has a memory which is used to store the search history. Each ant uses the collective experience to find a solution to the problem [8][9]. The first ACO algorithm is called ant system (AS) [15]. It has been used to solve the traveling salesman problem (TSP). Then, AS was improved and extended. The improved versions include the ant colony system (ACS) [16], the Max–Min ant system (MMAS) [17], etc. The ACO algorithms have been also successfully used for solving a range of combinatorial optimization problems: the vehicle routing problem, the quadratic assignment problem, etc. Likewise, they have been applied to miscellaneous scheduling problems such as: single machine, parallel machines, flow shop; job shop [18], open shop [19]; the general shop scheduling or group shop scheduling (including the job shop and the open shop) [20]; the hybrid flow shop [21]-[23] and the hybrid job shop.

Finally, like we previously mentioned, to the best of our knowledge we observe that no work has been previously proposed in the academic literature and real applications to study scheduling problem with simultaneous optimisation of the makespan and total weighted tardiness. These criteria are very important in real context because in many cases the companies have the aim of increasing its productivity (which can be obtained with the minimization of makespan) and while increasing the level of service customer (which it can gets minimizing total weighted tardiness). Therefore, in this paper we study this gap in the literature by proposing an Ant Colony Optimisation (ACO) algorithm to solve this complex scheduling problem.

III. Problem Formulation

A. System Description

Each sample that comes to the laboratory, is identified, registered and assigned to the respective analysis as we see in figure 1. Some samples that comes to a particular customer do not pass for the activities or sub process of the stage 1, which consists in preparing samples for the corresponding analysis. Then, this same sample can take only one or all the analysis performed in stages 2 to 6 following the order or precedence showed in figure 1.

As we show later in model formulation, it is necessary to define a decision variable for establish what kind of analysis has to be performed to each sample and also, to know which stages and sub process are necessary to process this sample. The types of analysis are described in Table 1

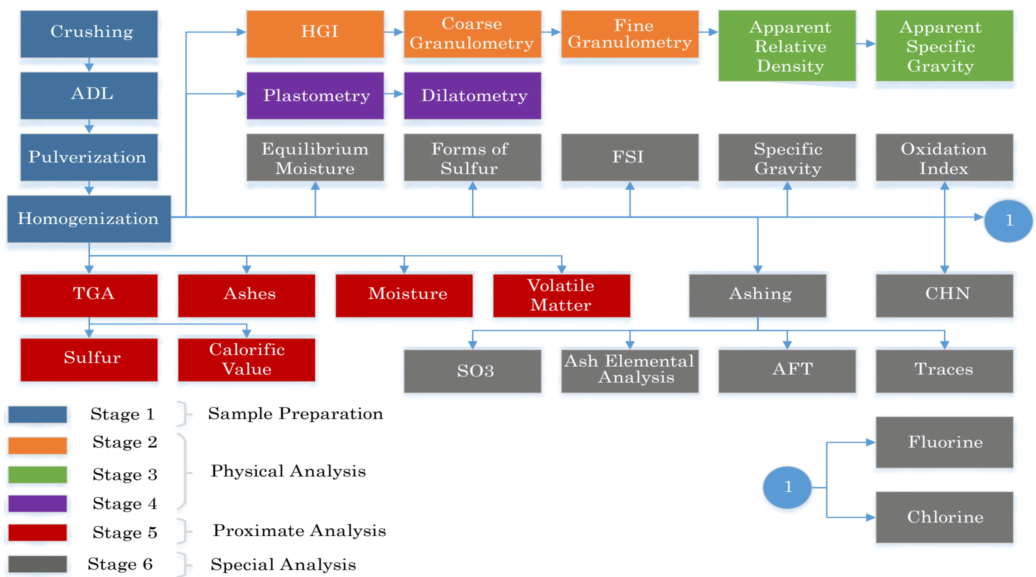

Fig. 1. Illustration of laboratory configuration and the different types of analysis performed.

TABLE 1. DESCRIPTION OF THE DIFFERENT TYPES OF ANALYSIS PERFORMED, PROCESS AND STAGES.

Type of analysis	Process (u)
Crushing	1
ADL	2
Pulverization	3
Homogenization	4
HGI	5
Coarse Series Granulometry Coal and Coke	6
Fine Series Granulometry	7
Apparent Relative Density Coal	8
Apparent Relative Density Coke	9
Plastometry	10
Dilatometry	11
TGA	12
Ashes	13
Moisture	14
Volatile Matter	15
Sulfur	16
Calorific Value	17
Ashing	18
SO3	19
Ash Elemental Analysis	20
AFT	21
Trace Elements	22
Equilibrium Moisture	23
Forms of Sulfur	24
FSI	25
Australian Specific Gravity	26
Oxidation Index	27
CHN	28
Fluorine	29
Chlorine	30

Therefore, taking into account the information previously mentioned, the formulation of mathematical model that describes the problem under study is presented below.

B. Mathematical model

1) Model parameters:

n: *Number of jobs (samples) To be scheduled.*

m: *Number of process*

p_{ik}: $i = 1, \ldots, n; k = 1, \ldots, m;$ *processing time of job i at process u* d_i: *due date of job i*

h_i: *Priority of job i; 3 if is high,2 if is middle and 1 if is low*

Q_k: $k = 1, \ldots, m;$ *Capacity of process u*

S_{ik}: *Binary parameter,iqual to 1 for assigned analysis to job i,if this analysis has to use the process k; 0 Otherwise*

2) Variables:

X_{ijk}: $i, j = 1, \ldots, n; k = 1, \ldots, m;$ *Binary variable,iqual to 1 if the job i is processed in position j in the process k; 0 Otherwise*

r_{ik}: *Inicialization time of job i in the process k*

C_{ik}: *Completion time of job i in the process u of stage l*

T_i: *Tardiness of job i*

W_i: *weight of job i*

C_{max}: *Makespan*

3) Objective Function:

$Min \sum T_i \times W_i$

$Min\ C_{max}$

4) Subject to:

(A)

$$\sum_{j=1}^{n} X_{ijk} = S_{ijk}$$

$\forall i = 1, \ldots, n; \forall k = 1, \ldots, m$

(B)

$$\sum_{i=1}^{n} X_{ijk} = S_{jk}$$

$\forall j = 1, \ldots, n; \forall k = 1, \ldots, m$

(C)

$$C_{ik} = C_{i(k-1)} + \frac{[\sum_{i=1}^{n} S_{ik}]}{Q_k} \times p_{ik} + \sum_{k=1}^{m} (X_{ijk} \times p_{ik})$$

Where $\frac{[\sum_{i=1}^{n} S_{ik}]}{Q_k}$ *is the representation of the integer part of this operation.*

(D)

$r_i11 = 0$
$r_i2 = C_i1$
$r_ii3 = C_i2$
$r_i4 = C_i3$
$r_i5 = C_i14$
$r_i6 = C_i5$
$r_i7 = C_i6$
$r_i8 = C_i7$
$r_i9 = C_i8$
$r_i10 = C_i4$
$r_i11 = C_i10$
$r_i12 = C_i4$
$r_i13 = C_i4$
$r_i14 = C_i4$
$r_i15 = C_i4$
$r_i16 = C_i4$
$r_i17 = C_i4$
$r_i18 = C_i4$
$r_i19 = C_i18$
$r_i20 = C_i18$
$r_i21 = C_i18$
$r_i22 = C_i18$
$r_i23 = C_i4$
$r_i24 = C_i4$
$r_i25 = C_i4$
$r_i26 = C_i4$
$r_i27 = C_i4$
$r_i28 = C_i4$
$r_i29 = C_i4$
$r_i30 = C_i4$
$\forall i = 1, \ldots, n$

(E)

$$W_i = \frac{hi}{\sum_{i=1}^{n} h_i}$$

$T_i = Max(0, C_{max} - d_i) \quad \forall i = 1, .., n$

(F)

$C_{max} = Max(C_{ik}) \quad \forall i = 1, .., n; \forall k = 1, .., m$

(G)

$C_{ik} \geq 0, \quad \forall i = 1, .., n; \forall k = 1, .., m$

$T_i \geq 0, \quad \forall i = 1, .., n$

The restriction (A) implies that each job has to be processed only once at machines assigned; if the corresponding analysis does not need using some machine, then the machine is not used. The restriction (B) indicates that each job has to be assigned to one machine if the job analysis implies the use of this machine. (C) Refers to termination times of jobs in each machine. In restriction (D), we appreciate the order flow in the samples at the machines. These last two restrictions make sure also, that the jobs is not overlapping in every machines. The sets of constraints (E) and (F) define the criteria C_{max} and $\sum T_i \times W_i$ that are minimized. Finally, the restriction (H) ensures that the value of these criteria are not negatives.

V. Proposed ACO algorithm

Ant Colony Optimization (ACO) is a meta-heuristic approach proposed by [24] and improved in later research (e.g. see [17] and [18]). The common behaviour of all variants of ant-based algorithms consists on emulate "real" ants when they find the optimal path between their nest and a food source. Several studies have applied ACO to solve different discrete and continuous optimisation problems [24]. One of these applications involves scheduling problems, as pointed out by [4] in a recently published extended literature review paper.

In this paper, we use the Ant Colony System (ACS) approach to solve the multi-objective scheduling problem under study. The following elements have to be defined [20]:

- An appropriate model to represent pheromones
- The mechanism to update the pheromone trail
- A heuristic function employed to provide information about the problem under study

These elements are employed to guide the selection of a job to be executed at a given time in the system analysed. This impacts the system behaviour. In order to obtain feasible solutions, job routing sequence has to be respected at each step when building a solution. This is ensured by using a restricted candidate list of all jobs that may be carried out at a given time of the schedule.

A. Constructive procedure

While a feasible solution is constructed, each ant k independently performs a sequence of processing jobs at his step. Hence, each ant k has to make two decisions: On the one hand, it has to select a job from the restricted candidate list L_k, but on the other hand, the ant has to select the position in which this processing job will be carried out. In order to respect processing precedence constraints, our solution approach consist of solving the problem stage by stage (a stage is a specific place of laboratory where particular types of analysis are performed). Solutions are hence constructed by repeating it at each processing stage in the system. Selected jobs are then registered successively in a list that also shows their position in the processing sequence. This avoids the procedure to select a job more than once in the same sequence. The structure of the proposed algorithm is presented in Figure 2.

```
Procedure ACO-Metaheuristic
Initialize parameters ( )
  While Stopping condition is not met do
    For each ant k
      For each stage
        Select the next processing job applying transition
        state rule
        Assign a position to selected processing job
        Update pheromone trails locally in order to reduce
        this quantity for the selected job and position
      End-For
    End-For
  Evaluate solution ( )
  Update pheromone trails globally in order to increase this
  quantity in current best solution
  End-While
End-Procedure
```

Fig. 2. Description of the proposed ACO algorithm for the multiobjective scheduling problem.

B. Solving job sequence problem

The procedure uses a state transition rule to select job from the set L_k (referred to the nodes that have not yet visited by ant k). This rule is called random proportional rule. It gives the probability for ant k to select job, based on the pheromone trail τ and the heuristic information η. This rule is formally described by the following expression:

$$s = \begin{cases} \underset{u \in L_k(r)}{\operatorname{argmax}} \{|\tau(r,u)|^{\alpha} \cdot |\eta(r,u)|^{\beta}\} & \text{if } q \le q_0 \\ S & \text{otherwise} \end{cases} \tag{1}$$

Where S is a random variable chosen from the following probability distribution (Dorigo and Gambardella 1997):

$$P_k(O_{ru}, S) = \frac{|\tau(r,u)|^{\alpha} \cdot |\eta(r,u)|^{\beta}}{\sum_{u \in Lk(r)} |\tau(r,u)|^{\alpha} \cdot |\eta(r,u)|^{\beta}} \tag{2}$$

Where α and β respectively corresponds to relative weights of values τ(r, u) and η(r, u) in the rule; q, with $0 \le q \le 1$, is a value randomly chosen from a uniform distribution, and q_0, with $0 \le q_0 \le 1$, is a selection parameter that determines the relative importance between intensification and diversification strategies. That is, if $q > q_0$, the system will trend to diversification; but if $q < q_0$, the system will trend to intensification.

1) Model to represent pheromones

As explained before, it is necessary to define an accurate model to represent the level of pheromones. This level of pheromones establishes the "desirability" of having two given jobs next to each other in the sequence. That is, $\tau(i, j)$ represents the convenience of having job j immediately after job i. Hence, the level of pheromones determines the sequencing order of jobs at each machine. It is also used to represent past experiences of ants with regard to the selection of a job from the list of candidates.

2) Heuristic information

The heuristic information gives specific information about the problem under study. It is used to estimate the convenience of that a job has to be processed at position. For the case of this research, the heuristic Information is computed taking into account the relative weight of each job, the time that is received, and the due date. According to this, the heuristic information is obtained by the following equation:

$$\eta(i,j) = \frac{W_i}{dt_{due\ date(i)} - dt_{arrival(i)}} \tag{3}$$

Where:
W_i = Relative weight of job
$dt_{due\_date(i)}$ = Due date of job i
$dt_{arrival(i)}$ = Time of job i is received

3) Updating pheromone trails

Both local and global strategies for updating pheromone trails are defined in our algorithm. The details are explained next.

Local updating of pheromone trail

The local updating of pheromone trail is executed for each ant once it has built a solution. This rule aims at not to influence the behaviour of other ants.

A mechanism is defined to evaporate the pheromone level of the job and position selected by the ant *k*. This selection becomes less attractive to other ants. This also aims at diversifying the paths that ants are taken and hence avoid convergence to a local optimum. Modification on the pheromone trail is performed using the following expression:

$$\tau(i,j) = (1-\rho_l)\cdot\tau(i,j) + \rho_l\cdot\tau_0 \tag{4}$$

Where τ_0 is the initial level of pheromone and ρ_l, with $0 \leq \rho_l \leq 1$) is the local evaporation parameter of pheromones.

Global updating of pheromone trail

Once ants have finished their paths, the global rule for updating pheromone trail is applied. This rule intensifies the level of pheromone on paths that allows ants to obtain a better solution. In the next iteration, this path will have a high probability of being chosen. The following expression is employed:

$$\tau(i,j) = (1-\rho_g).\tau(i,j) + \rho_g.\Delta\tau(i,j) \tag{5}$$

With:

$$\Delta\tau(i,j) = \begin{cases} (ET_b)^{-1} & \text{if } (i,j) \text{ belongs to the best solution} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Where ET_b is the best solution that is obtained by multiplying the objective functions (makespan and number of tardy jobs) by their respective weights. In other words, ET_b is the lowest value from over possible schedules obtained by the ants. In addition, ρ_g (with $0 \leq \rho_g \leq 1$) is the pheromone evaporation parameter.

V. Experimental Environment and Parameter Setting

This section first describes the datasets employed for the extended experimental study. Afterwards, the process employed to setup the parameters of the MOACO algorithm is described. Finally, the analysis of results is presented, as well as a comparison with an exact method and the laboratory's method.

The algorithm was coded using Visual Basic for Applications (VBA) ®. Experiments were carried out on a PC with processor Intel Pentium ® Dual Core with 2.40 GHz and 4.0 GB of RAM.

A. Benchmark instances

Datasets employed in our experiments were taken from the historical files of laboratory. We considered data from three months with high demand in July, August and September of 2014. These dataset employed samples (jobs) with different types of priority according to the scale defined in Section 3. A total of 30 replications of MOACO algorithm were carried out for each dataset in order to compare with the company schedule and with an exact method. For each replication, we compute the value of makespan and total weighted tardiness with its respective weight in the objective function, which was established in 0.5 for each one.

B. Parameters setting and convergence

Several parameters have to be defined to run the MOACO algorithm. Preliminary runs were carried out in order to setup the values of such parameters with a representative instance for each problem size. As performance metrics, the makespan and the total weighted tardiness were considered since these are the objective functions of the problem under study.

For the number of ants, we tested with 15, 20 and 30 ants in the system (these values correspond to the number of jobs to be scheduled according to some datasets employed in the computational experiments). Our performance analysis did not find any significant difference between such values. Hence, the number of ants was set to be 15, just for searching computational efficiency (the higher the number of ants, the higher the computational time).

Regarding the parameters, the values presented in Table 2 were considered in the preliminary experimental design. According to Dorigo and Gambardella (1997), these factors have a great impact on the algorithm's behaviour and, as a consequence, on the quality of solutions. The values of α, β, ρ, pheromones and q_o were defined according to the works of Kalouli et al. (2009, 2010), who studied a multi-objective scheduling problem with a configuration relatively similar to the system under consideration in this study with both makespan and total earliness/tardiness as objective functions.

Table 2. Values of parameters for algorithm setup

Parameter	Values
α	1 and 2
β	2 and 3
ρ	0.01 and 0.1
q_o.	0.8 and Log(*iter*)/Log(*num_iter*), where *iter* and *num_iter* are, respectively, current iteration and total number of iterations
Quantity of pheromones	0.01 and 0.1

A mixed factorial design with four replications was performed (Niebles-Atencio et al., 2012). The analysis of results was done using STATGRAPHICS ® software. Results showed that the best combination of values for the different parameters is α=2, β=2, ρ=0.01, pheromones=0.01 and q_0=0.8. These values were employed further during the full computational experiments. After running several instances, we observed that the algorithm converged after 2000 iterations (see Figure 3). Hence, this was the number of iterations we have set for running all experiments onwards.

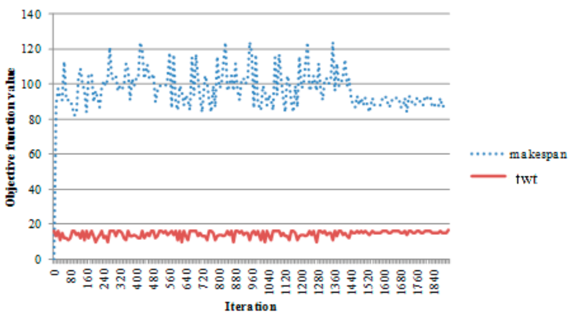

Fig. 3. Convergence of MOACO algorithm.

III. Results

This section presents the analysis of results of our computational experiment. All the experiments were run for system configurations among 15 and 70 jobs (that is the maximal capacity of the laboratory for a day trip). As we previously said, the performance of the proposed MOACO will be here compared against both real or laboratory schedule, and the exact solution using a Mixed-Integer Linear Programming model. The MILP model was coded and run using AMPL® version 8.0 for MS Windows®. Because of the problem of computational complexity, the model was run for instances with number of jobs less and equal to 45. Hence, optimal solutions were obtained for these small sub problems. For instances with higher number of jobs, to get a solution was not possible in a reasonable time, even for the professional version of AMPL.

The general structure of the set of comparisons is presented in Table 3. We first present the global performance of the proposed MOACO algorithm. Table 4 summarizes the obtained values of minimum, maximum and average objective functions values over the sets of instances. As ACO algorithms are probabilistic in nature, thirty replications were carried out. Hence, a minimum and a maximum value for each objective function correspond to the worst and the best value obtained at a given replication, while the average is computed over the set of thirty replications.

Table 3. General structure of comparisons

Jobs	MOACO	MILP	LABORATORY
15	X	X	X
20	X	X	X
25	X	X	X
30	X	X	X
35	X	X	X
40	X	X	X
45	X	X	X
50	X		X
55	X		X
60	X		X
65	X		X
70	X		X

Table 4. Average values obtained by MOACO for the objectives functions

Jobs	C_{max}			TWT		
	Min	*Max*	*Avg.*	*Min*	*Max*	*Avg.*
15	1089	1198	1516	0,47	0,55	0,51
20	1799	1979	1858	1,58	1,82	1,7
25	1740	1914	2351	2,89	3,32	3,11
30	2738	3012	2043	0,29	0,34	0,32
35	1154	1269	2316	0,05	0,06	0,05
40	3258	3584	2628	3,26	3,75	3,51
45	1748	1923	2035	3,98	4,57	4,27
50	2129	2342	1406	4,04	4,65	4,34
55	550	605	1083	0	0	0
60	1512	1663	2103	3,2	3,68	3,44
65	2493	2742	2372	1,48	1,7	1,59
70	2026	2229	2127	2,85	3,28	3,07

Additionally, in order to maintain certain coherence in the experimental analysis, a relative deviation index, in percentage, was employed, as shown in Equation (7) and (8), where corresponds to the averages values of the objective function x (i.e., makespan or total weighted tardiness) obtained using proposed algorithm ACO for representative instances (number of jobs). Also, and corresponds to the values of the objective functions using the MILP model, and laboratory approach. These values are shown in Table 5. Note that a negative value of %dev means that the proposed ACO algorithm outperforms the method against it is compared with.

$$\%dev = \frac{F_x^{ACO} - F_x^{MILP}}{F_x^{MILP}} \times 100\% \tag{7}$$

$$\%dev = \frac{F_x^{ACO} - F_x^{LAB}}{F_x^{LAB}} \times 100\% \tag{8}$$

Table 5. Comparison between MOACO algorithm, MILP model and laboratory's approach.

	MILP		MOACI		LABORATORY		Desviation between ACO and MILP model		Desviation between ACO and Laboratory Approach	
Numbers of jobs	C_{max}	TWT	C_{max}	TWT	C_{max}	TWT				
18	716,82	2,74773	752,6	2,745502361	1254,216667	3,254945602	4,99%	0,08%	-39,99%	-15,65%
21	585,7	0	550,3166667	0	1086,666667	0	6,04%	0,00%	-49,36%	0,00%
24	1750,64	7,40878	1921,25	5,838428204	2273,333333	6,34067644	9,75%	21,20%	-15,49%	-7,92%
27	1490,08	3,97539	1914,2	4,010057253	2586,6	5,150300772	28,46%	0,87%	-26,00%	-22,14%
28	1304,2	3,16926	1512,166667	3,196206019	5187,616667	3,813792735	15,95%	0,85%	-70,85%	-16,19%
34	1044,42	0	1044,416667	0	2067,833333	0	0,00%	0,00%	-49,49%	0,00%
35	1959,35	3,99913	1747,583333	3,975150148	1496,95	4,215229234	10,81%	0,60%	16,74%	-5,70%
37	1164,26	0,473643	1088,583333	0,474393009	2359,366667	0,506240656	6,50%	0,16%	-53,86%	-6,29%
41	1512,97	0,0635623	1154,2	0,049624885	2783,833333	0,267827422	23,71%	21,93%	-58,54%	-81,47%
42	2874,85	0,840928	3052,033333	0,838352739	12274,4	7,27130652	6,16%	0,31%	-75,13%	-88,47%
43	1261,68	1,19757	1126,85	1,19486152	1800,05	1,247865605	10,69%	0,23%	-37,40%	-4,25%
43	1151,28	0,815308	1873,433333	0,829390456	1800,05	0,829800233	62,73%	1,73%	4,08%	-0,05%
44	1816,88	1,02595	1749,933333	1,080516053	8072,283333	4,369192826	3,68%	5,32%	-78,32%	-75,27%
45	2132,09	1,561114	1798,783333	1,580250444	2023,616667	1,64192463	15,63%	1,22%	-11,11%	-3,76%

When comparing the proposed MOACO algorithm with laboratory's schedule, we observe that MOACO is outperformed in only two cases (number of jobs 35 and 43) for makespan criteria. For the rest of instances, MOACO always performed better with a significantly difference. On the other hand, for total weighted tardiness criteria, MOACO is only for two cases (number of jobs 21 and 34) equal to the laboratory approach. For the rest, MOACO outperforms with a great difference (more than 80% in some cases) to laboratory's schedule. Thus, the MOACO approach is widely the best alternative (in comparison for the laboratory approach) for scheduling samples at company's laboratory.

On the other hand, when comparing the proposed MOACO algorithm with MILP model, we observe that, although MILP model outperforms to MOACO in almost all instances, there is no a statistical difference among these two approach for both objective functions, as we see in Table 5 that shows the z test for mean comparisons was performed using MS-Excel ®. Therefore, we can say, with significantly statistical evidence, that MOACO algorithm is as effective as the MILP model, with the plus that MOACO schedules the total amount of jobs but MILP model does not. Hence, MOACO algorithm can be used as an effective decision support tool for scheduling process in the mineral laboratory, in order to minimize both: makespan and total weighted tardiness.

In Tables 8 and 9, we present the results for 20, 50 and 100 job instances. Again, we notice that MOACO was better than Laboratory's approach

Table 6. Statistical Test for mean comparisons of MOACO algorithm and MILP model.

Criteria	**MILP**		**MOACO**	
	Cmax	**TWT**	**Cmax**	**TWT**
Mean	1854,2717	1793,1937	2,14163	2,07913
Var	693398,32	482826,811	3,79412	3,18026
N	30	30	30	30
hypothetical mean difference	0		0	
Z	0,27589		0,11594	
P(Z<=z) (one-tailed)	0,39131		0,45384	
Critical Value of Z (one-tailed)	1,64485		1,64485	
Critical Value of Z (two-tailed)	0,78262		0,90769	
Statistical value of Z (two-tailed)	1,95996		1,95996	

Table 7. Average values obtained by MOACO for the objectives functions

Jobs	C_{max}			**Total Weighted Tardiness**		
	Min	***Max***	***Avg.***	***Min***	***Max***	***Avg.***
20	627	666	649	4099	4245	4146
50	1674	1744	1709	20114	21808	21423
100	3843	3915	3879	161449	163033	162245

for 50 and 100 job instances in both objective functions. However, MILP produced a better makespan than MOACO for instances with 50 and 100 jobs. At the same time, in figures 4 and 5 we present in a graphic way, the percentage of deviation shown in table 5:

Table 8. Average values of makespan and total weighted tardiness for 20 and 50 jobs instances

Approach	20 jobs				50 jobs			
	C_{max}	*%dev*	∑∝Tj	*%dev*	C_{max}	*%dev*	∑∝Tj	*%dev*
MOACO	649	-	4146	-	1709	-	21423	
MILP	663	-2%	2704	53%	1525	12%	21235	1%
LAB.	652	0%	3825	8%	2528	-32%	30314	-29%

Table 9. Average values of makespan and total weighted tardiness for 100 jobs instances.

Approach	100 jobs			
	C_{max}	*%dev*	∑∝Tj	*%dev*
MOACO	3879	-	162245	
MILP	3565	9%	121335	34%
LAB.	4567	29%	160514	1%

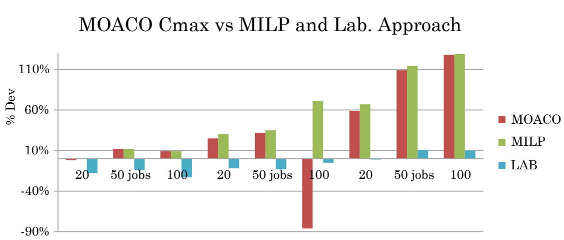

Fig. 4. % of Deviation of Cmax from MILP and Lab. Approach vs. MOACO.

Finally, in Figures 5, 6, 7 and 8 we show the Pareto sets comparisons between MOACO with MILP and MOACO with Laboratory's approach:

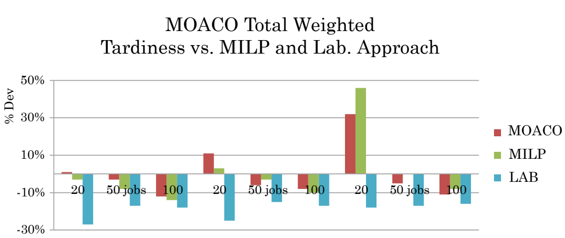

Fig. 5. % of Deviation of Total Weighted Tardiness from MILP and Lab. Approach vs. MOACO.

Finally, in Figures 6, 7, 8 and 9 we show the Pareto sets comparisons between MOACO with MILP and MOACO with Laboratory's Approach:

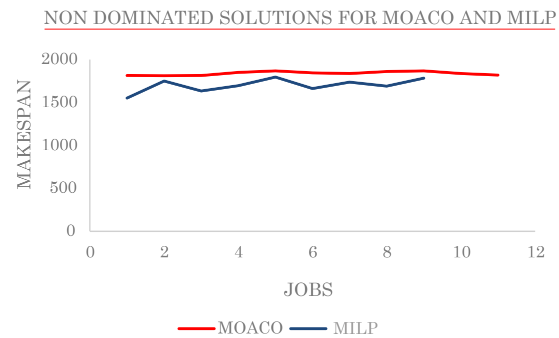

Fig. 6. Pareto Sets for Jobs instances from MOACO and MILP approach for makespan.

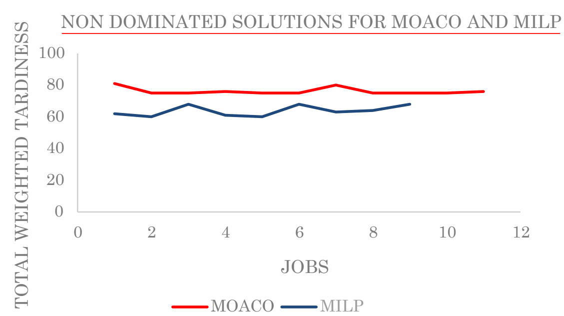

Fig. 7. Pareto Sets for Jobs instances from MOACO and MILP approach for Total Weighted Tardiness.

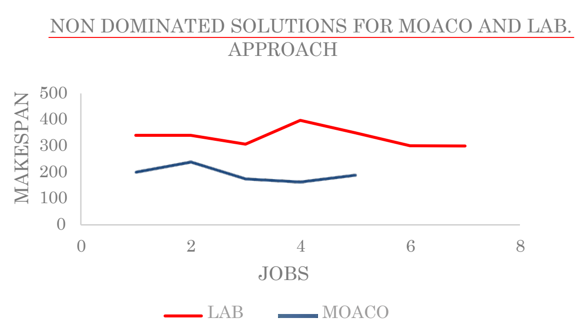

Fig. 8. Pareto Sets for Jobs instances from MOACO and Laboratory's approach for Makespan.

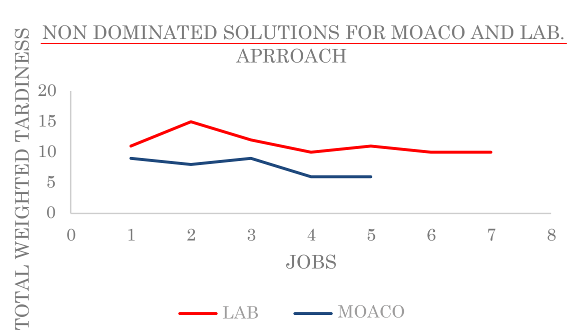

Fig. 9. Pareto Sets for Jobs instances from MOACO and Laboratory's approach for Total Weighted Tardiness.

As we can see in both sets of figures, the gap in the Pareto sets between MOACO and MILP approach is shorter than the gap in the Pareto sets between MOACO and Laboratory's approach. This means that the performance for MOACO and MILP is very close in comparison for performance between MOACO and Laboratory's approach. In this last case, the performance of MOACO is considerably better than company's approach, especially for makespan objective.

VII. Concluding remarks and further research

This paper studied the job scheduling problem in a real and difficult context of a mineral laboratory. Since in the scientific literature ACO algorithms have shown to be good solution procedures for solving complex scheduling problems, an ACO algorithm was proposed in this paper to solve the multi-objective case of minimising makespan and total weighted tardiness. Computational experiments were carried out using historical datasets. Results showed that very good solutions can be found using our ACO algorithm in comparison with the exact method and laboratory's schedule with a considerably less time and computational effort. It is worthwhile to note that the quality of the solution is not affected by an increase in the number of jobs to be scheduled. Therefore, this computational tool is an effective decision support model that allows scheduling the samples in the laboratory in order to increase the customer level and its productivity simultaneously, even for a large number of jobs (samples). Then, this tool can be replicated in the different branches of laboratory for its flexibility and compatibility with laboratory's software. For further research, the ACO algorithm could be hybridised in order to improve much more its performance when solving instances with large number of jobs. Heuristics or meta-heuristics such as Greedy Randomised Adaptive Search Procedure (GRASP), Genetic Algorithms or Simulated Annealing seems to be good options to be used for this hybridisation. Other opportunities for further research consist on adapting our ACO algorithm to solve for this situation, other types of multi-objective scheduling problems with objectives such as number of tardy jobs, total completion time, earliness, work in process etc. In addition, other versions of the problem can be considered. For example, problems with specific time windows for sample arrivals or stochastic processing times.

References

[1] M. L. Pinedo, Scheduling - Theory, Algorithms, and Systems. Springer, 2012.

[2] E.G. Talbi, Metaheuristics: From Design to Implementation. Wiley Publishing, 2009.

[3] L. Jourdan, M. Basseur, and E.-G. Talbi, "Hybridizing exact methods and metaheuristics: A taxonomy," Eur. J. Oper. Res., vol. 199, no. 3, pp. 620–629, Dec. 2009.

[4] R. F. Tavares Neto and M. Godinho Filho, "Literature review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research," Eng. Appl. Artif. Intell., vol. 26, no. 1, pp. 150–161, Jan. 2013.

[5] H. Hoogeveen, "Multicriteria scheduling," Eur. J. Oper. Res., vol. 167, no. 3, pp. 592–623, Dec. 2005

[6] T'kindt, V., Billaut, J.-C. Multicriteria scheduling: Theory, models and algorithms. Berlin: Springer, 2006

[7] D. Lei, "Multi-objective production scheduling: a survey," Int. J. Adv. Manuf. Technol., vol. 43, no. 9–10, pp. 926–938, Oct. 2008.

[8] S. Khalouli, F. Ghedjati, and A. Hamzaoui, "A metaheuristic approach to solve a JIT scheduling problem in hybrid flow shop," Eng. Appl. Artif. Intell., vol. 23, no. 5, pp. 765–771, Aug. 2010.

[9] S. Khalouli, F. Ghedjati, and A. Hamzaoui, "An Ant Colony System Algorithm for the Hybrid Flow-Shop Scheduling Problem," Int. J. Appl. Metaheuristic Comput., vol. 2, no. 1, pp. 29–43, 2011.

[10] S. G. Ponnambalam, V. Ramkumar, and N. Jawahar, "A multiobjective genetic algorithm for job shop scheduling," Prod. Plan. Control, vol. 12, no. 8, pp. 764–774, Jan. 2001.

[11] V. A. Armentano and J. E. Claudio, "An Application of a Multi-Objective Tabu Search Algorithm to a Bicriteria Flowshop Problem," J. Heuristics, vol. 10, no. 5, pp. 463–481, Sep. 2004.

[12] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, "A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria," Comput. Oper. Res., vol. 36, no. 2, pp. 358–378, Feb. 2009.

[13] J. Chang, G. Ma, and X. Ma, "A New Heuristic for Minimal Makespan in No-Wait Hybrid Flowshops," in 2006 Chinese Control Conference, 2006, pp. 1352–1356.

[14] H. Allaoui and A. Artiba, "Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints," Comput. Ind. Eng., vol. 47, no. 4, pp. 431–450, Dec. 2004.

[15] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in European Conference of Artificial Life, 1991, pp. 134–142.

[16] 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.

[17] T. Stützle and H. H. Hoos, "MAX-MIN Ant system," Futur. Gener. Comput. Syst., vol. 16, no. 9, pp. 889–914, Jun. 2000

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," IEEE Trans. Syst. Man. Cybern. B. Cybern., vol. 26, no. 1, pp. 29–41, Jan. 1996.

[19] C. Blum, "Beam-ACO—hybridizing ant colony optimization with beam search: an application to open shop scheduling," Comput. Oper. Res., vol. 32, no. 6, pp. 1565–1591, Jun. 2005.

[20] C. Blum and M. Sampels, "Ant colony optimization for FOP shop scheduling: a case study on different pheromone representations," in Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), 2002, vol. 2, pp. 1558–1563.

[21] K. Alaykýran, O. Engin, and A. Döyen, "Using ant colony optimization to solve hybrid flow shop scheduling problems," Int. J. Adv. Manuf. Technol., vol. 35, no. 5–6, pp. 541–550, May 2007.

[22] S. Khalouli, F. Ghedjati, and A. Hamzaoui, “Hybrid approach using ant colony optimization and fuzzy logic to solve multi-criteria hybrid flow shop scheduling problem,” in Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology - CSTST ’08, 2008, p. 44.

[23] S. Khalouli, F. Ghedjati, and A. Hamzaoui, “An integrated ant colony optimization algorithm for the hybrid flow shop scheduling problem,” in 2009 International Conference on Computers & Industrial Engineering, 2009, pp. 554–559.

[24] A. Colorni, M. Dorigo, and V. Maniezzo, “Distributed optimization by ant colonies,” in European Conference of Artificial Life, 1991, pp. 134–142.

[25] M. Dorigo and T. Stützle, Ant Colony Optimization. Cambridge MA: The MIT Press, 2004.