Hindawi Publishing Corporation
Advances in Artificial Intelligence
Volume 2012, Article ID 312132, 12 pages
doi:10.1155/2012/312132

Research Article

Ant Colony Optimisation for Backward Production Scheduling

Leandro Pereira dos Santos,[1] Guilherme Ernani Vieira,[2,3,4] Higor Vinicius dos R. Leite,[4,5] and Maria Teresinha Arns Steiner[4]

[1] *Instituto Federal do Parana, Assis Chateaubriand, 80230-150 Curitiba, PR, Brazil*
[2] *Petrobras S.A., 41770-395 Salvador, BA, Brazil*
[3] *Doutor Jose Peroba 225, Apartment no. 1103, 41.770-235 Salvador, BA, Brazil*
[4] *Department of Industrial Engineering, Pontifical Catholic University of Parana, 80215-901 Curitiba, PR, Brazil*
[5] *Department of Management, Universidade Tecnológica Federal do Paraná, 80230-901 Curitiba, PR, Brazil*

Correspondence should be addressed to Guilherme Ernani Vieira, g.e.vieira@hotmail.com

Received 7 May 2012; Accepted 31 July 2012

Academic Editor: Deacha Puangdownreong

The main objective of a production scheduling system is to assign tasks (orders or jobs) to resources and sequence them as efficiently and economically (optimised) as possible. Achieving this goal is a difficult task in complex environment where capacity is usually limited. In these scenarios, finding an optimal solution—if possible—demands a large amount of computer time. For this reason, in many cases, a good solution that is quickly found is preferred. In such situations, the use of metaheuristics is an appropriate strategy. In these last two decades, some out-of-the-shelf systems have been developed using such techniques. This paper presents and analyses the development of a shop-floor scheduling system that uses ant colony optimisation (ACO) in a backward scheduling problem in a manufacturing scenario with single-stage processing, parallel resources, and flexible routings. This scenario was found in a large food industry where the corresponding author worked as consultant for more than a year. This work demonstrates the applicability of this artificial intelligence technique. In fact, ACO proved to be as efficient as branch-and-bound, however, executing much faster.

1. Production Scheduling Still a Differential for Competitiveness

The globalised world economic scenario makes entrepreneurial competitiveness unavoidable and being competitive has become an indispensable prerequisite to organisations that strive for success. Within this context, manufacturing activities become especially important for they decisively influence performance, directly affecting (and being affected by) forecast, planning, and scheduling decisions.

Shop-floor production scheduling, which within the hierarchical production planning covers disaggregate and detailed decisions in short time frame, consists in allocating activities (production orders or jobs) to resources, by obeying sequencing and setup restrictions, with focus on getting the best possible results from limited available resources, and, at the same time, aiming at reducing production costs and meeting service levels as fast and efficiently as possible. To make all this happen in cases where production and financial resources are limited and restrictions are many, adequate algorithms techniques and intelligence are necessary. Almost four decades ago, Garey et al. [1] classified production scheduling problems as being NP-hard, which in practical ways means that it is very difficult for one to obtain an optimal solution through exact algorithms and also demand unacceptable execution (computer or effort) time. The difficulty in using exact techniques for the solution of these problems leads to the use of approximate methods, known as heuristics or metaheuristics, which try to find good acceptable solutions (not necessarily optimal ones) within reasonable computer time.

In this study, a metaheuristic known as ant colony optimisation (ACO) was applied to a specific production scheduling problem found in productive systems having

(i) one processing stage,

(ii) parallel resources with different production capacities,

(iii) backward scheduling (based on due dates), and,

(iv) products with many possible (flexible) production routings.

This particular type of production scenario was found in a large food industry that makes, among many other different products, chocolate bars and eggs—mostly for Easter festivities—common in many countries worldwide, especially in Latin America.

For this particular company, production needs to make all forecasted orders by a given due data, and since demand is highly seasonal, most of the labor is hired under temporary contracts. One can imagine that better production schedules imply less money in hiring temporary workers. Because Easter is an important date for these chocolate products (retailers must receive products one to two months prior to the Easter festivities), backward scheduling approach is also used in this type of scenario.

From the literature, one can see that the ACO metaheuristic has been applied to solve complex production scheduling. C. J. Liao and C. C. Liao [2], for example, presented an ACO algorithm applied to agile manufacturing. Shyu et al. [3] proposed an application of ACO to a scheduling shop-floor problem with two machines. Rajendran and Ziegler [4] analyzed two ACO scheduling algorithms for flow-shops. Bauer et al. [5] implemented ACO for solving a production-scheduling problem with one machine. Lin et al. [6] conducted a study using ACO for production scheduling and also proposed the inclusion of two new features inspired by real ants. Ying and Lin [7] also used ant colony systems to solve production scheduling problems.

For the evaluation of the implemented ACO algorithm, production schedules are analyzed according to two performance measures:

(a) maximum completion time or makespan (i.e., total time needed to manufacture all production orders), and

(b) computer processing time (effort) for the creation of a production schedule.

To measure the proposed ACO's efficiency, comparisons are made with a similar system implemented using branch-and-bound optimisation. For these comparisons, different scenarios (configurations) of the proposed problem are tested. This paper explains the proposed ACO implemented and all tests and analysis performed.

The paper is organised as follows. Section 2 presents a bird's eye view on the ACO metaheuristic. Section 3 details the manufacturing scenario. Section 4 shows how the ACO software was implemented. Section 5 describes the design of experiments (DOEs) performed, tests, and analysis conducted. Main conclusions and suggestions for future studies are presented in Section 6.

2. ACO Metaheuristic

As a background support to understand how ACO was used, this section briefly shows the use of ACO metaheuristic to the travelling salesman problem (TSP) and also describe the ACO metaheuristic applied to production scheduling.

2.1. ACO Metaheuristic Applied to TSP. In ant colony optimisation, a given number of ants leave their nest to search for food and there are many possible paths an ant can take to get there. During their walk, ants leave pheromone, which is a substance that tells other ants about paths they can take for food.

Each ant will do a certain number of trips from the nest to the source of food and back to the nest. In each of these trips, the ants deposit in the performed path a certain quantity of pheromone. There will be a standard quantity in the case that the path travelled by the ant does not present improvements compared to the best previous track, otherwise, there will be a larger quantity of pheromone, in case the path travelled by the ant is shorter than the previous best path.

Meanwhile, there is a continuously decrease in the existing quantities of pheromone in all paths, due to the pheromone evaporation. Finally, the choice of the paths is based in a probability which depends of the quantity of the pheromone on a given arc and its distance. It is important to emphasize that the smaller the path, the greater will be the concentration of the pheromone, and consequently, the greater will be the probability of being chosen.

TSP consists in a set of localities to be visited, each one only once, by any agent which, after completing a loop (cycle), has to go back to the origin position. The goal of this problem is to find the path tour that forms a tour passing through all cities. An instance of TSP can be represented by the valued graph $G(V, E)$, where V represents the nodes (localities to be visited) and E represents the arcs in the graph, where each arc has a cost given by the distance. This distance is denoted by d_{ij} and indicates the distance between the city i and j, as

$$d_{ij} = \sqrt{\left(X_i - X_j\right)^2 + \left(Y_i - Y_j\right)^2}. \quad (1)$$

We assume that exist m ants in the system and that each ant has the following characteristics.

(i) Chooses the next city to be visited with a probability which depends on the distance and the quantity of pheromone in the arcs which link every two cities.

(ii) In order to force the ants to perform a feasible tour, transfers to cities which have already been visited are discarded until a tour is completed.

(iii) When a loop is completed, each ant deposits a certain quantity of pheromone on the arc (i, j) visited.

Be $\tau_{ij}(t)$ the intensity of pheromone in arc (i, j) in time t. Each ant in time t chooses the next city to which it will go in time $(t + 1)$. Defining one ACO iteration as the n movements realised by m ants in the interval $(t, t+1)$, then the n iterations of each ant form a loop, that is, each ant realises a tour passing by all the cities. On every one, the intensity of the pheromone is updated by

$$\tau_{ij}(t + n) = \rho\tau_{ij}(t) + \Delta\tau_{ij}, \quad (2)$$

where ρ is a coefficient (constant) with (1-ρ) representing the pheromone evaporation between the times t and $(t + n)$ of the arc (i, j) and $\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$, where $\Delta\tau_{ij}^{k}$ is the quantity of the pheromone deposited in the arc (i, j) by the k-th ant between the times t and $(t + n)$. The coefficient ρ has to be adjusted in a value smaller than "1" in order to avoid unlimited accumulation of pheromone. Normally, the intensity of pheromone in time 0, $\tau_{ij}(0)$, is adjusted as an integer positive constant c.

The rule in order to satisfy the constraint that each ant visits n different cities is to associate to each ant a list, called tabu list, which stores the cities which have already been visited and forbids the ant visit them again before the tour has already been completed. When a tour is completed, the tabu list is used to calculate the present solution of the ant (i.e., the distance travelled in the path). It is defined tabu_k as for the vector which grows dynamically and contains the tabu list of k-th and and tabu_s the s-th city visited by ant k at the present tour.

Defining attractiveness η_{ij} as the quantity $1/d_{ij}$, we determine the probability of transaction of city i to city j by the k-th ant as

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha}.\left[\eta_{ij}\right]^{\beta}}{\sum_{k\in \text{Allowed}_k}\left[\tau_{ik}(t)\right]^{\alpha}.\left[\eta_{ik}\right]^{\beta}}, & \text{if } j \in \text{Allowed}_k, \\ 0, & \text{otherwise;} \end{cases} \tag{3}$$

where $\text{Allowed}_k = \{N - \text{tabu}_k\}$, being N the number of cities of the problem and α and β are parameters that control the relative importance of the intensity of the pheromone versus the attractiveness. In this way, the transaction probability is a combination between the attractiveness and the intensity of the pheromone in time t.

According to Dorigo et al. [8], there are many different forms to compute the value of $\Delta\tau_{ij}^{k}(t, t + 1)$. One of them is denominated Ant-cycle which is calculated as

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k}, & \text{if the } k\text{-th ant goes through arc}(i, j) \\ & \text{in its tour between the times } (t, t + 1), \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where Q is a constant and L_k is the length of the way travelled by the k-th ant.

2.2. ACO Metaheuristic Applied to the Production Scheduling. Based on Dorigo et al. [9], Ventresca and Ombuki [10] and Mazzucco Jr. [11], the representation of the production scheduling problem in the form of ant systems may be built through a disjunctive graph. This graph can be defined as $Q = (V, A, E)$, where V is the set of vertexes of the graph, which corresponds to the set of operations to be scheduled, represented here by O. Two fictional operations, described as "0" and "$N + 1$," are also added to the set V, that is, $V = \{0, O, N + 1\}$, representing the origin (nest) and the destination (food source) nodes.

Group A is a set of arcs connecting consecutive operations from the same job (task, activity, or production order), the arcs that connect the operation 0 to the first real operation of each job and the last operation of each one of them to operation $(N + 1)$.

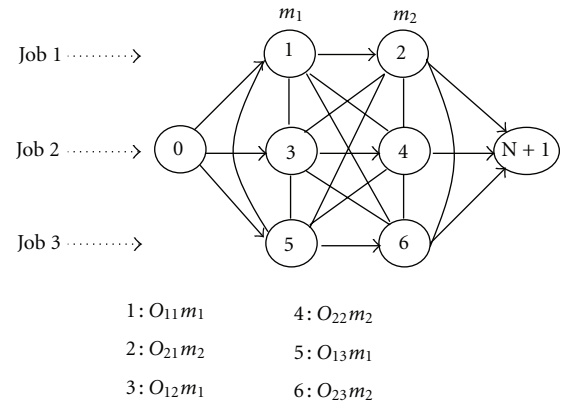


Figure 1: Ant system graph for 3 jobs and 2 machines, Dorigo et al. [9].

Group E is a set of the edges connecting two operations to be performed by the same resource (in this production scenario, a machine), and it may be expressed as $E = \{\{v, w\}/M_v = M_w\}$.

Each arc has a pair of numbers $\{\tau_{ij}, \eta_{ij}\}$, which are a concentration of pheromone and visibility, respectively. The last one can be considered as the operation processing time at node J.

Figure 1 is a representative ant system graph of a production scheduling problem. In this graph, nodes represent the operations to be performed and each operation corresponds to the processing of a certain quantity of product in a single machine. Each of these operations belongs to a determined job J_1, J_2, and J_3 respectively. In this example, a job can be defined as a production order with a certain quantity of product that must pass through two machines (m_1 and m_2). This way, a job is defined as a set of operations. In the graph, the operations numbered 1, 3, and 5 are executed by the same machine m_1. Likewise, operations 2, 4, and 6 are executed by the same machine m_2 and belong to jobs J_1, J_2, and J_3, respectively. The initial and final operations, labeled "0" and "$N + 1$," are fictional, that is, they are not performed; however, they are required by the metaheuristics. They only exist so that the oriented graph, by being oriented, may have an initial and final operation (nest and food nodes in ACO representation). Since they do not have a processing time, these operations do not affect the production scheduling process.

The nodes numbered from 1 to 6 represent the operations to be scheduled and each operation can be symbolically presented as O_{ijm}, where i: $1, \ldots, N$ represents the operation; j: $1, \ldots, K$ represents the job; m: $1, \ldots, M$ represents the machine. Each operation is indexed according to its position number and the job it belongs to. Node 1, for instance, corresponds to operation O_{11m1}, which means that if an ant passes through this node, the system will schedule the first operation of job 1 to machine 1 (O_{11m1}). Node 2 corresponds to operation O_{21m2}, meaning that operation 2

(second operation) of job 1 can be assigned to machine 2. That happens to all the nodes corresponding to a determined operation, as shown on the right side of the figure.

An orientation set of all the edges transforms the graph in Figure 1 into an oriented graph and represents one of the possible solutions to the modeled problem. In the same way, an orientation set defines a sequence or permutation of the operations processed by each machine in M.

3. The Manufacturing Scenario Considered in this Project

The problem covered in this research consists in optimising production scheduling systems having only one processing stage with parallel resources and flexible routings. In other words, any product has one single operation (or processing stage), which may require one or more resources. Hence, each job J has only one operation $J_j = \{O_{1j}\}$ and the objective is then to schedule a set of S jobs: $S = \{J_1, J_2, \ldots, J_S\}$ within a minimum time-frame (makespan). This scenario was found in a particular food industry where part of this study was accomplished.

The considered productive system is characterised by having parallel resources, that is, an operation can be formed by more than one machine or productive resources (as explained below). Thus, there is a set of M machines available, where $M = \{m_1, m_2, \ldots, m_M\}$. It is worth mentioning such machines can be different in capacity and efficiency and, therefore, the system present different processing capacities for the same product.

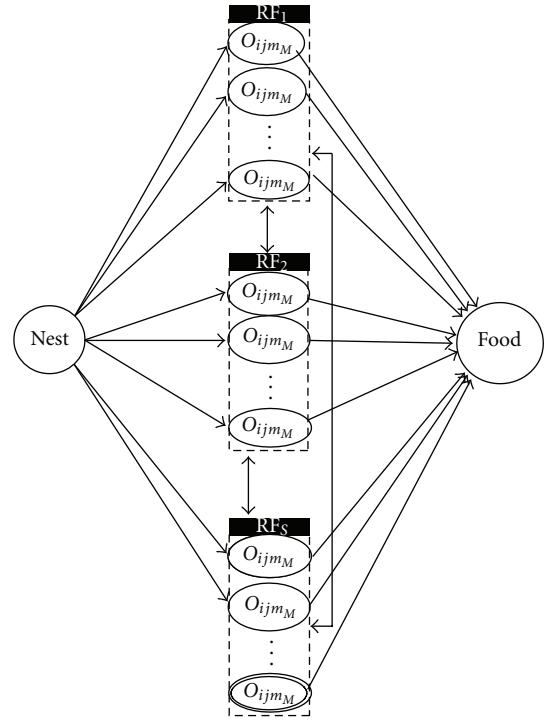
Another important characteristic of the considered productive system is that products can have flexible routings, which means that a particular product can have more than one possible process plan. It is assumed in this paper that each job J can be processed by any of the M machines, that is, each job J has a flexible routing. Schematically, $J_1 = \{\mathrm{RF}_1\}$, $J_2 = \{\mathrm{RF}_2\}$, and $J_S = \{\mathrm{RF}_S\}$, where RF_1 is the flexible routing of job 1, RF_2 is the flexible routing of job 2, and $\{\mathrm{RF}_S\}$ is the flexible routing of job S. Each flexible routing is formed by a set of similar operations. Schematically, $\mathrm{RF}_1 = \{O_{11m1}, O_{11m2}, \ldots, O_{12mM}\}$, $\mathrm{RF}_2 = \{O_{12m1}, O_{12m2}, \ldots, O_{12mM}\}$, and $\mathrm{RF}_S = \{O_{1Sm1}, O_{1Sm2}, \ldots, O_{1SmM}\}$, where O_{11m1} represents operation 1 of job 1 processed by machine 1, O_{12m1} represents operation 1 of job 2 processed by machine 1, and O_{1Sm1} represents operation 1 of job S processed by machine 1.

The system implemented also uses backward scheduling. Each job J has a due date that must be met. The problem thus consists in scheduling all operations in order to minimise the total time needed for their execution (makespan), bearing in mind the delivery due date of all products. There are also other restrictions in this scenario.

(a) Suppliers lead times: the system should not schedule an order if needed material(s) is (are) not available.

(b) Setup times: this is in fact a setup dependent scheduling algorithm.

The food company studied considers minimizing total production time (makespan) as the main optimisation objective and, therefore, this is the objective function used by the ACO system implemented. This performance measure represents the length (total time) of the production schedule. In other words, it is the ending time of the last job scheduled minus the starting time of the first job scheduled. One can also consider makespan as the ending time of the last operation to be processed minus the starting time of the first machine that begins its operation.



S: Number of jobs to be scheduled

M: Number of machines available

Figure 2: ACO graph of a production scheduling problem with one processing stage, parallel resources, and flexible routings.

Although minimizing the maximum makespan is the objective of the ACO system developed, this research also considered the total computer time (effort) as a performance measure to evaluate ACO against another optimisation technique: branch-and-bound (BB), which was used in another study and is used in this paper as a benchmark to help us evaluate the efficiency of the proposed ACO.

4. The ACO Production Scheduling Implemented

As previously mentioned, each product process plan (routing) is made of a single operation (monostage). Therefore, one can say that each flexible routing has a set of similar operations and when one of these operations is chosen, the others are discarded.

Under the proposed ACO graph model, each job's flexible routing (RFs) corresponds to a set of nodes (see dashed square lines at Figure 2). In the beginning these are "virtual" models, available for an ant to pass through. Once an ant

chooses and passes through a node, it becomes a chosen node, meaning that the operation of this job has been scheduled (all remaining similar possible operations in the RF are discarded).

As previously mentioned, the initial node (nest) and the final node (food) are parts of the ACO graph. These nodes are fictitious, that is, are not actual production scheduling activities, while all other nodes correspond to operations that can be scheduled. The edges correspond to the time or duration of the operation to be scheduled and there are no edges linking operations from the same group (RF), that is, from the same job, considering that they will be eliminated if they are scheduled. There are, however, nonoriented edges that link the operation groups (RF) and indicate that all the jobs need to be scheduled, regardless of the sequencing (see connecting arrows among dashed squares).

Ants leave the "Nest" aiming to find a source of "Food," however, all jobs must be scheduled before the ant gets to the food node (the complete path between nest and food nodes comprises a feasible schedule). In the beginning, the ants find different sources of food, and as the algorithm evolves, the number of food sources converges to the "best food sources."

4.1. Structure of the Implemented Software. The ACO technique uses the natural behavior of an ant colony, which tries to find the shortest paths between nest and the food through the communication among the agents (ants) using pheromone, a path of "smell for food" for the other ants.

The ACO system has a set of configuration parameters that impact the quality and performance of the algorithm. In this project, six ACO parameters were analysed.

(i) Number of ants (NA): the quantity of ants simultaneously searching for food (each ant will create a possible schedule).

(ii) Number of travels of each ant (NT): it is the number of times that each ant will travel between the nest and the source of food.

(iii) Quantity of initial pheromone (QIP): it is the quantity of pheromone that already exists in all the paths before the ants start their travels.

(iv) Quantity of added pheromone (QAP): it is a quantity of pheromone left by an ant in the path taken between the nest to the food nodes. In this implementation, pheromone is added right after the ant reaches the food node.

(v) Evaporation percentage (EP): it is the quantity of pheromone lost (that evaporates) in each path as time passes.

(vi) Best response valorisation (BRP). The better the solution found by an ant, the more pheromone is added to its path.

The basics of the ACO system implemented are described in Figure 3. It briefly consists of (a) reading the input data; (b) initializing the system, that is, the ants and their respective tabu and feasible nodes' lists; (c) creating the graphs for each ant, with its respective nodes and edges. The stopping criterion is based on the total number of ants and the numbers of travels (from nest to food) each ant must achieve. During the schedule process, each ant goes from node to node, in the ACO graph each ant keeps track of its onw path being created.

The logic of this process lies on the fact that as an ant leaves the nest it starts scheduling production orders until it reaches the food (which means that needed jobs have been scheduled). The scheduling of an order means that a given node was selected and has been included in the ant's path to food.

An ant randomly chooses the next operation (or the next node to go to) based on the operation processing time and on the quantity of pheromone present at edges connecting the node where the ant currently is and the other possible nodes. The more pheromone exists in an arc (edge), the greater the probability for an ant to select it.

When an ant schedules all of its possible operations, it means that it has reached a source of food and one travel is complete. The solution found by each ant is analysed, and depending on the response quality, each ant's path created will get a specific quantity of pheromone, according to previously established quality criteria (the better the schedule the more pheromone are deposited in every arc in the path). Since the probability of selecting a node also depends on quantity of pheromone in the arc, better solutions tend to influence other ants to choose the same path in the future.

When all ants complete the number of needed travels, the scheduling process ends and the best response is presented.

4.2. The ACO System Class Structure. The ant colony optimisation algorithm was implemented following an object-oriented structure. A total of eight object classes were implemented are briefly explained next and the dependency among these classes are depicted at Figure 4.

(i) ACO_SFS implements setup and configuration procedures, run (execute) commands, evaporation execution, savings, and so forth ("SFS" stands for shop-floor scheduling.)

(ii) Ant_SFS implements: an ant, tabu list procedures, keeps track of edges taken, current node position, calculates probabilities, verifies next possible (feasible) nodes, moves to next node, updates pheromone, and so forth.

(iii) Config_ACO_SFS procedures for the creation and configuration of the ACO.

(iv) Graph_SFS implements the graph where ants will walk through. A graph is basically a list of nodes, a list of edges. In the very beginning, a graph has only two nodes: nest and food.

(v) Edge_SFS an edge is a set of two nodes (begin and end node) and an information about pheromone. Remember that different ants can walk through the same edge and are affected by the same pheromone in the edge.

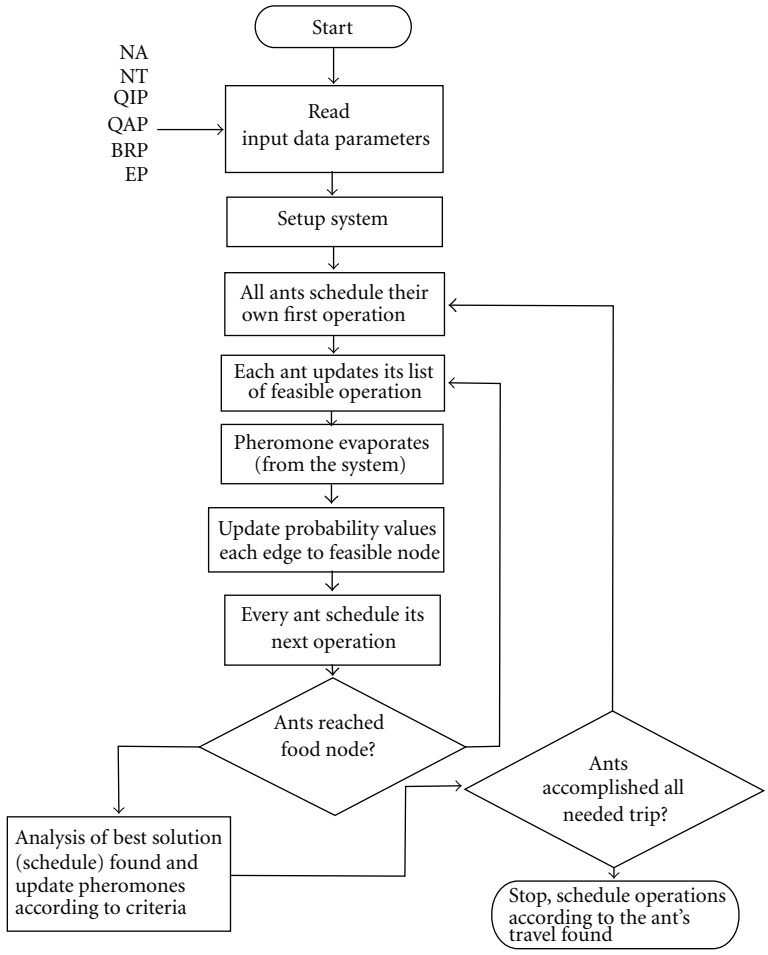


Figure 3: ACO software general flowchart.

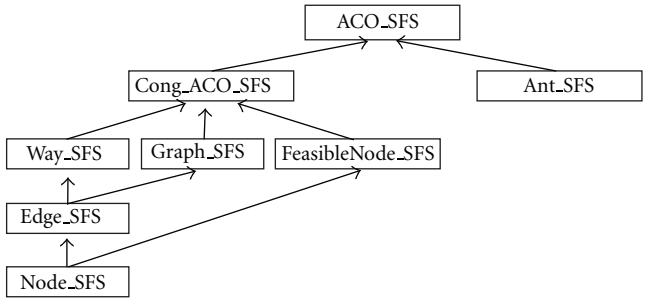


Figure 4: Object-oriented ACO classes.

(vi) Node_SFS a node corresponds to an operation that can be schedule. A production order is basically made of a set of possible operations.

(vii) FeasibleNode_SFS a feasible node is node in the graph were the ant can go to, which means that the ant can "schedule" that operation.

(viii) Way_SFS this class implements the path an ant takes. It is basically a list of edges.

Several other classes were also developed. These classes implemented the production scheduling logic and attributes, such as object classes to model resources, production orders, products, production routings (process plans), calendars, setup matrices, among others (for conciseness reasons, these classes will not be explained.) A screenshot of the system is shown at Figure 5.

5. Experiments Planning and Analysis

Two types of experiments have been made in this project: (a) factorial experiments (2^k) have been used to verify the influence of each ACO configuration parameter in the objective function (makespan + computer time); (b) experiments to verify the ACO efficiency in relation to branch-and-bound have been carried out through an analysis of variance.

5.1. Influence of the ACO Configuration Parameters on Performance. The objective of this first type of analysis is twofold: (a) to understand how the values of the input (or configuration) parameters affect the solution quantity and computer executing time needed; (b) these 2^k factorial experiments will

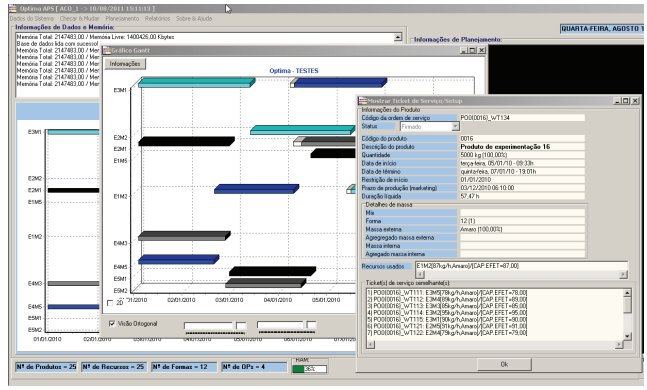


Figure 5: A screenshot of the ACO shop-floor scheduling system developed.

Table 1: Data for the analysis of the considered 26 factorial experiments.

	Scenarios	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
BRP	Low	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
	High	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
NA	Low	20	20	20	20	10	10	10	10	10	10	10	10	10	10	10	10
	High	50	50	50	50	20	20	20	20	20	20	20	20	20	20	20	20
QIP	Low	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	High	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
QAP	Low	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
	High	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50
NT	Low	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
	High	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
EP	Low	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	High	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Number of machines		3	3	3	3	6	6	6	6	10	10	10	10	10	10	10	10
Number of orders		15	20	30	45	15	20	30	45	15	20	30	45	15	20	30	45

also be used to help us identify the best (optimum) values for the ACO system configuration parameters.

In the 2^k factorial analysis performed, two levels for the six parameters (BRP, NA, QIP, QAP, NT, and EP) are considered leading to 64 (2^6) different ACO configuration to be tested. The low and high levels for this analysis are shown on Table 1.

After defining the scope of the 2^k experiments, an analysis of variance (ANOVA) was performed. For this, each one of the 64 configuration scenarios was run sixteen times (sample size was chosen arbitrarily). Table 2 shows the ANOVA experiment results obtained.

By the analysis of Table 2 and based on the theory about ANOVA, one can conclude that the parameters BRP, EP, NT QIP, and QUAP significantly affected the problem results. The only parameter that did not seem to affect performance was the number of ants (NA) always according to a 95% of confidence level. It is possible that the difference between the low and high values for the NA parameter was not enough to cause any significant impact on the system performance and also, possibly the number of travels has hiden some of the effect of this parameter. A detailed analysis is given next.

If the best response valorisation parameter is set too low, the quantity of pheromone deposited in the path will not be sufficient to "make" other ants to follow such path. On the other hand, if BRP is set too high, it forces all ants to follow a single path, prematurely converging to a solution that might be far from a good one.

Parameter EP (evaporation rate) determines the quantity of pheromone lost (that evaporates) in the paths as time passes by. The fact that a higher quantity of substance evaporates within a specific period of time can be important, so that a local minimum solution is not chosen.

Parameter NT (number of travels that each ant must execute) clearly affected the solution quality. As said before, this parameter probably hid the NA effect on performance.

The importance of parameter QIP (quantity of initial pheromone in the system) is noticed when compared to the quantity of pheromone added after a travel is complete. Setting QIP too high and using low QAP may not influence ants to take the best paths. But both parameters significantly affect the response quality.

Finally, as explained previously, parameter NA (number of ants in the system) had no significant influence in the

Table 2: ANOVA results for the experiments 2^6.

	ANOVA				
Parameter	SS	DOF	SA	F_o	Ferritic (F_c)
BRP	315.30	1	315.30	96.51	>4
EP	19.74	1	19.74	6.04	>4
NT	72.01	1	72.01	22.04	>4
QIP	35.27	1	35.27	10.79	>4
QAP	17.39	1	17.39	5.32	>4
NA	9.03	1	9.03	2.76	<4
ERROR	186.22	57	3.27		
T	533.96	63			

SS: sum of squares, DOF: degrees of freedom, SA: square average, F_o: F observed.

Table 3: Summary of the 2^k experiment results.

Parameter	F_o	F critic	Evaluation
BRP	96.51	4	Significant
EP	6.04	4	Significant
NT	22.04	4	Significant
QIP	10.79	4	Significant
QAP	5.32	4	Significant
NA	2.76	4	Not significant

quality of the problem responses. This result may also be caused by the fact that in the implemented ACO, the pheromone is only added to the paths after an ant reaches the food node, not during the search. This, however, is closer to the natural behavior of ants, which by carrying food back to their nest, scracth their bellies on the ground, "leaving" the pheromone. One would assume, however, that the number of ants do affect the answer, however, for the two levels considered in this ACO system (20 and 50 ants) and the strong influence of the NT (number of travels), this did not come true. Table 3 brings a summary of this analysis, showing which variables are and are not significant in the proposed experiment 2^k executed.

The main conclusion in the first phase of this study is that most of the configuration parameters considered, which affect directly or indirectly the quantity of pheromone in a path, significantly impact the response quality (except the number of ants, as explained before).

In the next phase, the configuration parameters used for comparative tests are set, according to the methodology proposed by Montgomery and Runger [12], which basically consists in choosing the variable level in which the sum of the response averages is higher. This way, the parameters chosen were

$$\begin{aligned} &\text{BRP} = 5;\ \text{NA} = 50;\ \text{QIP} = 10;\\ &\text{QAP} = 50;\ \text{NT} = 100;\ \text{EP} = 5. \end{aligned} \tag{5}$$

As previously mentioned, the parameter referring to the number of ants in the system did not seem to be significant for schedule quality (makespan). According to Montgomery and Runger [12], in such cases, a specific parameter level must be chosen so as to optimise economy, operation or any other strong technical factor when executing the algorithm. This was the only parameter to be changed, by attributing 10 as its value. In so doing, there was a decrease in the computer time required to achieve the response and the maintenance of its quality.

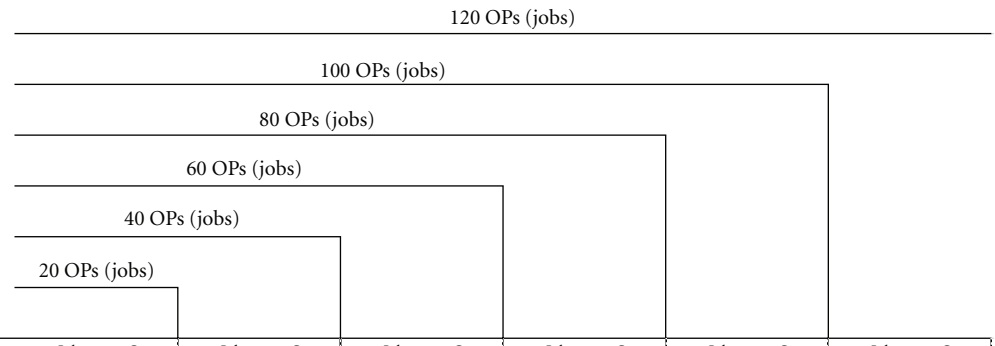
Table 4: Scenarios for the execution of the ACO efficiency tests.

Scenario	Number of possible machines to execute the operation	Number of jobs OPs
1	5	20
2	5	40
3	5	60
4	10	20
5	10	40
6	10	60
7	10	80
8	10	100
9	10	120

5.2. Analysing the ACO Metaheuristics Efficiency in Comparison to Branch-and-Bound. In order to test the ACO metaheuristics efficiency in production scheduling optimisation (in scenarios similar to the one adopted in this paper), comparative tests of the solution generated by the ACO metaheuristics were made with solutions obtained through the use of a branch-and-bound optimisation method (already implemented by the authors of this article).

For the execution of this second phase of these experiments, a productive system that makes 20 different items was considered. The experiments also considered products with several possible process plans (i.e, routings). These scenarios considered five possible machines for the operations and a total of 20, 40, or 60 jobs. Other six scenarios considered ten possible machines for the operation, and 20 to 120 jobs to be scheduled. This is summarised at Table 4.

Considering a scenario with 5 machines and a particular product A, this product's operation could be done at machine I, II, III, IV, or V. Production capacities in each machine are 5, 6, 7, 8, and 9 units/hour, respectively (so machines



	Job	Quant.	Job	Quant.	Job	Quant.	Job	Quant.	Job	Quant.	Job	Quant.
Product A	1	220	21	730	41	110	61	110	81	700	101	700
Product B	2	150	22	560	42	300	62	75	82	530	102	530
Product C	3	450	23	580	43	225	63	225	83	550	103	550
Product D	4	370	24	370	44	740	64	185	84	340	104	340
Product E	5	890	25	960	45	445	65	445	85	930	105	930
Product F	6	450	26	790	46	900	66	225	86	760	106	760
Product G	7	670	27	450	47	335	67	335	87	420	107	420
Product H	8	1.350	28	645	48	675	68	675	88	615	108	615
Product I	9	850	29	375	49	425	69	425	89	345	109	345
Product J	10	760	30	490	50	1.520	70	380	90	460	110	460
Product K	11	490	31	760	51	245	71	245	91	730	111	730
Product L	12	375	32	850	52	750	72	188	92	820	112	820
Product M	13	645	33	1.350	53	323	73	323	93	1.320	113	1.320
Product N	14	450	34	670	54	900	74	225	94	640	114	640
Product O	15	790	35	450	55	395	75	395	95	420	115	420
Product P	16	960	36	890	56	480	76	480	96	860	116	860
Product Q	17	370	37	370	57	740	77	185	97	340	117	340
Product R	18	580	38	450	58	290	78	290	98	420	118	420
Product S	19	560	39	150	59	280	79	280	99	120	119	120
Product T	20	730	40	220	60	365	80	365	100	190	120	190

Figure 6: Product quantity in production orders.

are not identical). Considering a scenario with 10 machines, product A could go to machine I, II, III, IV, V, VI, VII, VIII, IX, or X. Production capacities are 5, 6, 7, 8, 9, 10, 11, 12, 13, and 14 units/hour, respectively.

Each of the 9 scenarios has a number of POs (production orders) to be scheduled. Here, each PO corresponds to a Job that comprises a given quantity of a particular product that need to be made. In this part of the paper, it was considered that each scenario would be characterised by having 20, 40, 60, 80, 100, or 120 POs. Figure 6 shows the quantity of each product to be produced in each PO per scenario.

The methodology used for the comparative analysis of the (ACO and BB) techniques was the hypothesis test with two samples, in order to determine the difference between two averages and the difference between two variances.

Table 5 shows the results obtained with the experiments of the previously described scenarios, in which the column named BB brings the results achieved with branch-and-bound and the column named ACO, the results with ACO.

Firstly, an F test was applied to the results of the two response variables studied: objective function (makespan) and computer time (effort). Such test aimed to verify whether the variances between the two techniques were significantly different. After that, a T-test could be applied to verify how significant the difference between the result was, and then, to infer about the ACO technique efficiency production scheduling optimisation.

Table 5: Makespan results and computer time for the studied scenarios.

Scenario	Makespan [hours]		Computer time [s]	
	BB	ACO	BB	ACO
1	14.01	13.29	1	14
2	25.51	26.33	3	56
3	38.24	38.26	14	93
4	10.84	11.30	547	28
5	18.26	19.05	1,298	112
6	24.37	29.20	3,254	186
7	36.41	38.36	8,620	357
8	40.72	43.81	19,865	800
9	56.89	62.39	23,838	2,867

5.2.1. The F-Test for the Response Variables (Prestep). The F-test was executed to verify the difference in variance between the two samples, regarding both makespan and computer time needed to achieve the best results. This is a prestep

Table 6: The F test for the makespan variables.

	BB	ACO
Average	29.47	31.33
Variance	219.61	265.60
Observations	9	9
DOF (degrees of freedom)	8	8
F_o	0.83	
$P\ (F \leq f)$ uni-caudal	0.40	
F critic uni-caudal (F_c)	0.29	

Table 7: F test for computer time variances (seconds).

	BB	ACO
Average	6,382	501
Variance	85,393,845	848,108
Observations	9	9
DOF (degrees of freedom)	8	8
F_o	100.69	
$P\ (F \leq f)$ uni-caudal	0.00	
F critic uni-caudal (F_c)	3.44	

to define the appropriate test to compare averages (Section 5.2.2). For this test, a package called Data Analysis from Microsoft Excel was employed, more specifically, the F-Test function "two samples for variances," with a significance level $\alpha = 5\%$. Table 6 shows the results obtained.

Since $F_o > F_c$ (0.83 > 0.29), the null hypothesis must not be accepted, once it assumes that the variances of the two samples are equal. Hence, there is statistical evidence that the difference between the variances is significant, which means that, within a 95% confidence level, the variances in makespan obtained with ACO is different from the one obtained using BB. Table 7 presents the F-test result regarding the variance analysis of the computer needed to achieve the results for each technique.

Since $F_o > F_c$ (100.69 > 3.44), the null hypothesis shall not be accepted, once it assumes that the variances of the two samples are equal, that is, there is evidence, again, that the difference between the variances is significant concerning the computer time (effort).

Summing up, F-tests showed that the variance regarding the minimisation of maximum makespan was highly better using BB compared to ACO. However, computer (processing) time variance using ACO was with less than using BB.

5.2.2. T-Test for the Response (Objective) Variables Considered. F-test results for the two response variables showed that for both cases, the variances between the samples are different. That leads to the analysis of the difference between the results averages of the two studied objectives (makespan and computer time) through the use a *T-test employing two samples and presuming different variances.* To do so, the Data Analysis tool from Microsoft Excel was again used, more specifically the *T-test function: two samples presuming different variances, with a significance level* $\alpha = 5\%$. Table 8 presents these results.

Table 8: T test for the makespan analysis.

	BB	ACO
Average	29.47	31.33
Variance	219.61	265.60
Observations	9	9
Hypothesis of mean difference	0	
DOF (degrees of freedom)	16	
Stat t	−0.25	
$P\ (T \leq t)$ uni-caudal	0.40	
t critic uni-caudal	1.75	
$P\ (T \leq t)$ bi-caudal	0.80	
t critic bi-caudal	2.12	

Table 9: Computer time T test.

	BB	ACO
Average	6,382	501
Variance	85,393,845	848,108
Observations	9	9
Hypothesis of mean difference	0	
DOF (degrees of freedom)	8	
Stat t	1.9	
$P\ (T \leq t)$ unicaudal	0.047	
t critic unicaudal	1.8	
$P\ (T \leq t)$ bicaudal	0.094	
t critic bicaudal	2.3	

Referring to the makespan between the ACO and BB techniques, the null hypothesis must be accepted, once the P-value = 40% is higher than the significance level adopted (5%), that is, the difference between the averages is not significant. This conclusion is interesting because it says that ACO is behaving as well as BB, regarding minimisation of makespan (in this productive scenario and for the ACO and BB softwares implemented).

T-test results regarding the computer time of the two compared techniques are presented in Table 9. So, regarding computer time (effort), the null hypothesis must be rejected, since the P-value (4.7%) is lower than the significance level adopted (5.0%). In other words, there exists significant difference between the averages. By looking at the averages shown on Table 9, one can confirm that ACO runs much faster that BB.

Despite the fact that the makespan was similar using ACO and BB, Ant Colony executed much faster than Branch-and-Bound. Table 10 summarises these results.

The objectives that guided this research were supported by two basic ideas. First, the intention to verify whether the ACO metaheuristics was a feasible technique for solving backward production scheduling optimisation in monostage productive systems, with parallel manufacturing resources, different production capacities, and flexible routings. Second, the study also intended to evaluate the efficiency of ACO compared to the branch-and-bound technique.

Table 10: Result summary of the ACO efficiency analysis.

	Makespan [hours]		Computer time [seconds]	
Scenario	BB	ACO	BB	ACO
1	14.01	13.29	1	14
2	25.51	26.33	3	56
3	38.24	38.26	14	93
4	10.84	11.30	547	28
5	18.26	19.05	1298	112
6	24.37	29.20	3254	186
7	36.41	38.36	8620	357
8	40.72	43.81	19865	800
9	56.89	62.39	23838	2867
	F_o	F_c	F_o	F_c
F test	0.83	0.29	100.69	3.44
For the variance:	Statistically different		Statistically different	
	t	α	t	α
T test	40.0%	5.0%	4.7%	5.0%
For the average:	Statistically not different		Statistically different	

By using statistical T-tests and F-tests, the ACO metaheuristics efficiency was, in fact proved, taking into account the quality of the generated production plan (in terms of makespan) and computer time required for the creation of production schedules.

Although there was no enough statistical difference between BB and ACO regarding makespan, if one assumes that BB gives a good answer, then ACO will perform similarly. In terms of computer time, however, ant colony performed much faster than branch-and-bound. It is important to emphasise that these results were obtained for the type of production scenarios considered.

6. Final Considerations

This paper focused on verifying whether ant colony optimisation could be effectively applied to a production scheduled problem found in some types of food industries operating with backward scheduling and considering monostage productive systems, parallel resources, and flexible routings. This analysis studied the metaheuristics configuration variables regarding their influence on the variations and averages of response variables.

It showed that the ACO configuration parameters (best response valorisation, evaporation, quantity of initial pheromone, quantity of added pheromone, and number of travels) proved to be significant in relation to their influence in the response quality (95% reliability). The only variable among the studied ones that did not prove to be significant was the number of ants in the system.

Besides verifying which system configuration variables affect the algorithm response quality, these experiments also helped to set the ACO configuration variables later used to test the efficiency of the ACO method. Hence, regarding the ACO technique efficiency analysis, quality was measured in terms of makespan and computer time spent to achieve good responses, while efficiency analysis was done by comparison of ACO results with branch-and-bound optimisation. Regarding makespan, it was not possible to point out any significant difference between the two methods. This led us to conclude that ACO is as efficient as BB in backward production scheduling in monostage problems with routing flexibility.

Regarding computer time, T-tests revealed that the ACO technique runs much faster than branch-and-bound in solving the type of production scenario considered, with large number of resources and jobs (production orders or tasks). One could verify that the time needed to achieve the response increases in higher proportion using branch-and-bound than using the ACO technique. It is important to point out that the conclusions from this study refer strictly to the type of production problems herein covered.

For future studies, some suggestions are

(i) productive systems with more than one processing stage must be considered,

(ii) implementation (and tests) in a forward scheduling environment,

(iii) different ACO implementation characteristics can also be tested, like, for instance, enabling evaporation to occur in each move of the ants and not only when they reach a food source (the same thing for pheromone deposit),

(iv) different metaheuristics configuration (setup) variables could be tested, and

(v) other possible implementations could consider a multiobjective function, with other objectives, such as minimizing lateness, average flow time, setups, and resources' idleness.

References

[1] Garey, MR, and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.

[2] C. J. Liao and C. C. Liao, "An ant colony optimisation algorithm for scheduling in agile manufacturing," *International Journal of Production Research*, vol. 46, no. 7, pp. 1813–1824, 2008.

[3] S. J. Shyu, B. M. T. Lin, and P. Y. Yin, "Application of ant colony optimization for no-wait flowshop scheduling problem to minimize the total completion time," *Computers and Industrial Engineering*, vol. 47, no. 2-3, pp. 181–193, 2004.

[4] C. Rajendran and H. Ziegler, "Two ant-colony algorithms for minimizing total flowtime in permutation flowshops," *Computers and Industrial Engineering*, vol. 48, no. 4, pp. 789–797, 2005.

[5] A. Bauer, B. Bullnheimer, R. F. Hartl, and C. Strauss, *An Ant Colony Optimizations Approach for the Single Machine Total Tardiness Problem*, Department of Management Science. Universidade de Vienna, Vienna, Austria, 1999.

[6] B. M. T. Lin, C. Y. Lu, S. J. Shyu, and C. Y. Tsai, "Development of new features of ant colony optimization for flowshop scheduling," *International Journal of Production Economics*, vol. 112, no. 2, pp. 742–755, 2008.

[7] K. C. Ying and S. W. Lin, "Multiprocessor task scheduling in multistage hybrid flow-shops: an ant colony system approach," *International Journal of Production Research*, vol. 44, no. 16, pp. 3161–3177, 2006.

[8] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive feedback as a search strategy," Tech. Rep. 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milano, Italy, 1991.

[9] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 26, no. 1, pp. 29–41, 1996.

[10] M. Ventresca and B. Ombuki, "Ant colony optimization for job-shop scheduling problem," Tech. Rep., Department of Computer Science, St. Catharines, Canadá, 2004.

[11] J. Mazzucco Jr., *Uma abordagem híbrida do problema da Programação da produção através dos algoritmos simulated annealing e genético [tese de doutorado (Doutorado Engenharia de Produção, UFSC)]*, Florianópolis, 1999.

[12] D. C. Montgomery and G. C. Runger, *Estatística Aplicada à Engenharia*, LTC, Rio de Janeiro, Brazil, 2004.