FEP WORKING PAPERS
FEP WORKING PAPERS

Research Work
in Progress

n. 474 December 2012

ISSN: 0870-8541

Ant Colony Optimization: a literature survey

Marta S.R. Monteiro [1,2]
Dalila B.M.M. Fontes [1,2]
Fernando A.C.C. Fontes [3,4]

[1] **FEP-UP, School of Economics and Management, University of Porto**
[2] **LIAAD/INESC TEC**
[3] **FEUP-UP, Faculty of Engineering, University of Porto**
[4] **ISR-Porto**

Ant Colony Optimization: a literature survey*

Marta S.R. Monteiro[1†], Dalila B.M.M. Fontes[1], Fernando A.C.C. Fontes[2]

[1] Faculdade de Economia and LIAAD-INESC TEC

Universidade do Porto

Rua Dr. Roberto Frias, 4200-464 Porto, Portugal.

E-mail: martam@fep.up.pt;fontes@fep.up.pt

Tel.: +351-22-0426240

[2] Faculdade de Engenharia da Universidade do Porto and ISR-Porto

Rua Dr. Roberto Frias, 4200-464 Porto, Portugal.

E-mail: faf@fe.up.pt

Abstract

Scientific literature is prolific both on exact and on heuristic solution methods developed to solve optimization problems. Although the former methods have an indisputable theoretical value when it comes to solve large realistic combinatorial optimization problems they are usually associated with large and even prohibitive running times. Heuristic methods, do not guarantee to determine a global optimal solution for a problem but are usually able to find a good solution rapidly, perhaps a local optimum, and require less computational resources. Ant Colony Optimization (ACO) algorithms belong to a class of heuristics based on the behaviour of nature ants. These algorithms have been used to solve many combinatorial optimization problems and have been known to outperform other popular heuristics such as Genetic Algorithms. Therefore, we believe that the number of ACO based algorithms will continue to grow for a long time. The contribution of this work is to provide the reader with a sort of consultation guide for developing ACO algorithms, by presenting a collection of different approaches that can be found in literature, regarding the ACO building blocks.

*This work is funded by the ERDF through the Programme COMPETE and by the Portuguese Government through FCT - Foundation for Science and Technology, projects PTDC/EGE-GES/099741/2008 and PTDC/EEA-CRO/116014/2009.

†Corresponding author.

Keywords: Ant Colony Optimization, Survey, Heuristics, Combinatorial Optimization Problems.
JEL Classifications: C61, C44.

1 Introduction

The idea behind ant algorithms is to adapt and use their communication style which has been proven to be so good in nature, rather than truly mimic the behaviour of real ants. Artificial ants can then be seen and described as communicating agents sharing some characteristics of the real ants, but also incorporating other characteristics for which there is no parallel in nature, (Solimanpur et al, 2004). The overall characteristics are what makes them fit to solve problems, if not optimally, at least by finding very good solutions. A real foraging ant spends all its life travelling between its nest and some food source. It does not then come as a surprise that the first problem solved with an ant algorithm, called Ant System (AS), was the Travelling Salesman Problem (TSP), a well-known combinatorial problem, where the shortest route (path) that visits exactly once each city of a given set of cities, starting and ending at the same city, is to be found.

The very good results that were being achieved with ant algorithms pointed to the broadening of the definition of *path* therefore allowing for the use of this method to solve other problems. Some adaptations of the algorithm had to take place, resulting in the so called Ant Colony Optimization metaheuristic, which is based on the ant system. The definition of the ACO metaheuristic, as a series of generic guidelines that could be very easily adapted to almost all types of combinatorial optimization problems, allowed a boost in the use of this methodology and in the number of researchers and publications in the area. Since then, ACO procedures have been applied to solve a broad set of problems, including: Network Flow Problems (Monteiro et al, 2012), Network Design Problems (Rappos and Hadjiconstantinou, 2004), Assignment Problems (Shyu et al, 2006; Bernardino et al, 2009), Facility Location Problems (Baykasoglu et al, 2006; Chen and Ting, 2008), Transportation Problems (Musa et al, 2010; Santos et al, 2010), Covering Problems (Lessing et al, 2004; Crawford and Castro, 2006; Mehrabi et al, 2009), Location Problems (Pour and Nosraty, 2006), just to mention but a few in the area of combinatorial optimization. Curiously enough, although the TSP was the first problem to be solved by the AS and ACO metaheuristics, it still inspires researchers such as García-Martínez et al (2007), for instance, that have recently used ACO to solve a bi-criteria TSP or Tavares and Pereira (2011) that use the TSP to test an evolving strategy to update pheromone trails.

Although in general ACO algorithms achieve very good results, there are cases where an hybridization with other heuristics or metaheuristics, proves to be necessary. Therefore, in the past few years authors have developed hybrid algorithms between ACO and Local Search (Pour and Nosraty, 2006), Simulated Annealing (Bouhafs et al, 2006), Post Processing Procedures

(Crawford and Castro, 2006), and even with Genetic Algorithms as is the case of (Altiparmak and Karaoglan, 2007). This allowed ant algorithms to achieve even better results in problems too complex to be solved by a single heuristic method.

In the following section we will explore, in detail, the first ant algorithm, that was called the Ant System. Afterwards, we review some of the large number of interesting works that have been developed ever since. We focus our attention mostly in works that have introduced modifications and extensions to the so-called building blocks of ACO algorithms. This is made with the purpose of showing alternative methods, that worked well with specific optimization problems, so that the reader who is developing an ACO algorithm can easily perceive its utility and how to adapt it for the problem at hand.

2 Ant Colony Principles

Ant Colony Optimization principles are based on the natural behaviour of ants. In their daily life, one of the tasks ants have to perform is to search for food, in the vicinity of their nest. While walking in such a quest, the ants deposit a chemical substance called *pheromone* in the ground. This is done with two objectives. On the one hand, it allows ants to find their way back to the nest, such as Hansel and Gretel in the fairytale. And on the other hand, it allows other ants to know the way they have taken, so that the others can follow them. The curiosity is that, because hundreds or even thousands of ants have this behaviour, if one could see the pheromone laid in the ground as a kind of light, the ground would be a large network with some of the arcs brighter than the others. Within the paths created by those arcs would surely be the shortest path between the nest and the food source. This behaviour can be seen as a kind of communication between the ants. If the path has a large concentration of pheromone, this is probably due to its shorter length that allowed ants to travel faster, resulting in a larger number of travels through the path therefore with much more ants depositing pheromone on it. Furthermore, over time the pheromone evaporates and thus its concentration reduces. The more time it takes for the ant to travel from the nest to the food source and back to the nest, the more time the pheromones have to evaporate. This system is thus based both on the positive feedback, i.e. depositing of pheromone attracts other ants to use the same path which will increase the pheromone quantity, and on negative feedback, i.e. dissipating of the pheromone through evaporation leads to lower levels of pheromone thus discouraging other ants. Deneubourg et al (1990) and Goss et al (1989) performed some experiences with real ants and they where able to show that foraging ants can find the shortest path between their nest and some food source, by the use of a chemical substance called pheromone, that they deposit while walking. After these experiments the authors proposed a stochastic model to describe what they had observed. This was the first step leading to an optimization algorithm based on the foraging behaviour of ants. Some years later, Dorigo et al (1996) developed the first foraging ants algorithm which was called *Ant System* and

that was firstly proposed to solve the travelling salesman problem.

2.1 Ant System

The objective of the travelling salesman problem is to find the shortest route between a set of cities, starting and finishing in the same city, going through all cities without visiting each city more than once. This problem is very easily adapted to the idea of the Ant System due to their similarity in concepts: find the shortest path between two points in a graph.

An AS algorithm considers a single ant colony with m artificial ants cooperating with each other. Before the algorithm starts to run each arc linking two different cities is given a certain quantity of pheromone τ_0. This is usually a very small value just enough to ensure that the probability of each arc to be chosen is different from zero. Also, the ants are created.

The algorithm has two main phases, the construction of the tour/solution and the pheromone update. Other important decisions have to be made before the ants can start finding a solution, such as defining the structure (representation) of the solution, or the initial pheromone quantity to be given to each arc. These questions will be discussed further ahead.

At each iteration each ant is randomly placed in a city, from the set of n cities. That city will be the starting point of the tour that is to be constructed by the ant. A solution to the TSP can be represented by a set of n consecutive cities. Therefore, at each step of the construction the ant has to choose, with a given probability, the next city to travel to.

This choice is made by using a *transition rule*, the short expression for *random proportional transition rule*, that uses a combination of attractiveness of the city, which is given by the heuristic information η_{ij} of the problem, and of the fitness of the move, i.e. past usage, which is given by the pheromone quantity τ_{ij}. The transition rule quantifies the probability of ant k, positioned at city i, travelling to city j and it is given by:

$$P_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in J_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta}, \tag{1}$$

where η_{ij}, the *heuristic information* or *visibility* of arc (i, j), is the inverse of the distance between city i and city j, i.e.

$$\eta_{ij} = \frac{1}{d_{ij}}, \tag{2}$$

J_i^k is the set of cities not yet visited by ant k while at city i, and α and β are parameters weighting the relative importance of the pheromone and of the heuristic information, respectively.

Therefore, the closest cities, that is, the ones that the ant can see from where it is standing, will

have a higher visibility value, whereas the others will have a lower one.

The values α and β are two tunable parameters that weight the pheromone information and the heuristic information on the transition rule.

After building the solutions the pheromone values in the arcs are updated. The update is done in two phases. Just before the ants can deposit pheromone in the arcs of their solution, the algorithm applies an *evaporation rate* ρ, with $\rho \in]0, 1]$, to the pheromone present at each arc, see equation (3).

$$\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t). \tag{3}$$

This operation simulates the natural process of evaporation preventing the algorithm from converging too quickly (all ants constructing the same tour) and getting trapped into a local optimum. The value of the evaporation rate indicates the relative importance of the pheromone values from one iteration to the following one. If ρ takes a value near 1, then the pheromone trail will not have a lasting effect, potentiating the exploration of the solutions space, whereas a small value will increase the importance of the pheromone, potentiating the exploitation of the search space near the current solution.

The length S^k of each tour is then calculated and the ants will be allowed to deposit pheromone in every arc of their tour. The pheromone quantity to be deposited in each arc is proportional to the quality of the solution of each ant and to the number of ants to incorporate that arc in its solution, as can be seen in equations (4) and (5).

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t), \tag{4}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{S^k(t)} & \text{if } (i,j) \text{ belongs to the solution of ant } k, \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where Q is a positive proportionality parameter and $S^k(t)$ is the length of the tour constructed by ant k at iteration t. For small problem instances, this update leads to a reduction of the search space thus converging to one where the optimal solution components will have the highest values in the matrix. However, for large instance problems it is known that stagnation is likely to happen, driving the solution to a suboptimal solution rather than to an optimal one. This is why pheromone evaporation is so important.

The previous steps are performed until some stopping criterion is reached, which can be a fixed number of iterations, as was the case, but it can also be the setting of a bound on running time or even the number of solutions evaluated.

The best values for the parameters used in ant algorithms depend both on problem characteristics and on the strategy chosen for searching the solution space. Therefore, before setting values for the parameter, decisions on the search strategy have to be made. Then, the algorithm must be run several times in order to establish the values of the parameters which tend to perform better.

3 Ant Colony Optimization

Meanwhile, some improvements were inserted into the AS such as the introduction of elitist ants into the colony (Dorigo et al, 1996), the ranking of ants (Bullnheimer et al, 1997), and the bounding of the allowed accumulated pheromone in each path (Stützle and Hoos, 1997). Nevertheless, the most important development is the description of the Ant Colony Optimization Metaheuristic by Dorigo and Di Caro (1999) and Dorigo et al (1999). The ACO, which is described in Algorithm 1, is made of general guidelines for the development of algorithms based on foraging ants to solve combinatorial optimization problems.

Algorithm 1 Pseudo-code for Ant Colony Optimization.

```
1: Initialize parameters
2: Initialize pheromone trails
3: Create ants
4: while Stopping criteria is not reached do
5:     Let all ants construct their solution
6:     Update pheromone trails
7:     Allow Daemon Actions
8: end while
```

The main difference from the basic structure of the AS algorithm is the introduction of a *Daemon*. The daemon can perform problem specific operations or centralized operations, which use global knowledge of the solutions, thus having a very active and important role in the algorithm. Note that in contrast to the AS no global knowledge is used since each ant deposits pheromone in its solution despite what the other solutions are like. This is a task that has no equivalence in the nature. The daemon can, for example, control the feasibility of each solution or give an extra pheromone quantity to the best solution found from the beginning of the algorithm or to the best solution in the current iteration. These last operations were already mentioned in previous algorithms but never attributing its responsibility to a main entity in the colony.

Another important feature, frequently used by authors on ant based algorithms is the introduction of Local Search procedures following the construction of the solutions. This is an optional feature that has been proved to be very important in the exploitation of the search space near to good solutions, leading almost always to better performances of the ACO.

3.1 The building blocks of an ACO

ACO algorithms have a set of characterising features that can be considered as their step stones. These features must always be specified, preferably, when describing the algorithm:

- method chosen to construct the solution,
- heuristic information,
- pheromone updating rule,
- transition rule and probability function,
- parameters values, and
- termination condition.

It becomes obvious that the combination of the different techniques, that can be developed for each of them, result in a large diversity of ant colony algorithms, each of which more adequate to a certain class of problems. Within the vast literature on the subject, different proposes can be identified to either improve earlier results or simply to solve a new type of problems. In this section, and for each of these building blocks, we review some of the extremely large number of techniques previously proposed, since it is impossible to track all the work that has been done ever since the early stages of ant algorithms. Nonetheless, the reader is always referred to the works that will be discussed in this section, for further details, as well as to the references therein.

3.1.1 Constructing a solution

The construction of a solution, along with its representation, is one major issue of an ant algorithm, as it is with any other heuristic method, since it will influence the rest of the procedures to be defined. Thus, it plays a crucial role on the success of the algorithm. Besides, it is common knowledge that it has a great effect on the running time of the ACO algorithm (Neumann and Witt, 2010). Therefore, if the construction is badly chosen, the probability of a bad performance is high.

Regarding the solution construction, a critical decision is whether to allow or not that unfeasible solutions are constructed. This decision alone, can have several outcomes, such as:

- allowing the construction of unfeasible solutions and then creating an extra procedure to fix them. This may involve too much of running time effort just to fix the solution;

- allowing the construction of unfeasible solutions and then discarding or penalizing unfeasible solutions. In this case, it may happen that the number of usable solutions is too small (or even nonexistent), and thus the algorithm converges quickly to a suboptimal solution. Even if this is not the case, if there are several constraints being violated then it is too hard to use the penalties since they are of different nature and thus may have opposite behaviours;

- if only feasible solutions are allowed, then the construction procedure may be too complex leading to large running times.

The construction of a solution is influenced by many aspects, such as the problem being solved and the constraints to be considered, the representation chosen for the problem, the investigator preferences, and so on.

Alaya et al (2004) solve a Multidimensional Knapsack problem where a decision on a subset of objects, satisfying a few resource constraints, has to be made in order to maximize the total profit. The solution for this problem only requires the choice of a subset of objects to be introduced in the knapsack, with no order specified, and can then be represented as a string of object identifiers. Each ant starts the construction of their solution by randomly choosing an object to be put in the knapsack. Then, objects are added to the solution, by using a transition rule as defined in equation (1), as long as they do not violate any resource constraint. This way a feasible solution is always obtained.

Rappos and Hadjiconstantinou (2004), in order to design two-edge connected flow networks, use two types of ant colonies sharing information about their pheromone levels. This problem is about configuring a network in order to satisfy demand nodes, provided that an extra arc is considered to keep the network flowing in the case that one of the arcs in the network fails. The solution for this problem is constructed in two phases, each of which solved by a different type of ants. One ant colony is inhabitated by flow ants and the other colony by reliability ants. The number of flow ants is the same as the number of demand nodes and, although they all start constructing their solution from the source node, each ant is assigned to reach just one specific demand node. When all flow ants have constructed their partial solutions, reaching their demand node destination, the network is created. The next step involves the reliability ants whose objective is to decide upon the extra arc, called reliability arc, to be added to the solution. For every flow ant a reliability ant is created and associated with each arc visited by the flow ant. Therefore, for each flow ant there is a set of reliability ants, as many as arcs in the solution of the flow ant. The objective of a reliability ant is to find an alternative path from the root node to the same destination node of the flow ant as long as it does not use a particular arc, from the ones used in the solution of the flow ant. This ACO algorithm provides a single feasible solution at each iteration, which is only entirely defined when all partial solutions of the flow ants have been assembled together, and the extra arc found by the reliability ants is

identified.

Baykasoglu et al (2006) solve a dynamic facility layout problem, where each ant has to decide, for each period t, the location of the n departments considered. The authors use a string with size $t \times n$ to represent the final solution, where the first n consecutive values identify the department locations for the first period, the second n consecutive values give the locations for the second period, and so on. Therefore, to construct a solution, all an ant has to do is to choose $t \times n$ elements of the type *department location*, accordingly to the pheromone levels, and provided that, within a time period, no *department location* is repeated, thus guaranteeing the construction of a feasible solution.

Partitioning and covering problems are solved with ACO algorithms by Crawford and Castro (2006). In this case, given a set of columns and rows, the objective is to choose a subset of columns covering all rows while minimizing cover costs. The solution is represented by a subset of columns. This implies a different approach, from the ones we have been mentioning before, because the solution components are represented by nodes and not by arcs, a fact that simplifies the calculations. The construction is straightforward. Each ant starts with an empty set of columns. Then, the ant adds columns one at the time, based on pheromone values, until all rows are covered. Solutions constructed in this way, can be unfeasible in the partitioning case because a row may be covered by more than one column. That is why post processing procedures, that will try to eliminate redundant columns, are added afterwards in order to turn unfeasible solutions into feasible ones.

In a transportation problem with N supply nodes and M demand nodes, it is known that a solution has, at most, $N + M - 1$ arcs. This observation allows Altiparmak and Karaoglan (2006) to decide on allowing each ant in their algorithm to be able to choose $N + M - 1$ arcs to construct a feasible solution. Each ant starts by randomly choosing an arc from the set of available arcs A, and proceeds the construction by adding, one at the time, the remaining arcs by using pheromone information. The arcs in the set of allowed arcs to be chosen to enter the solution is defined by the demand and supply nodes that have not exceeded already their demand and supply, respectively. In this case the sense of path is not applied since arcs are chosen arbitrarily as long as they satisfy demand and supply constraints.

The Single Source Capacitated Facility Location Problem deals with the location of a set of facilities, with a limited capacity on the supply, and the allocation of a single facility to each customer so as to satisfy customer demands and minimize total costs. Chen and Ting (2008) propose an algorithm to solve it which integrates an Ant Colony System with two types of ants, location ants and assignment ants. Therefore, there are two different solution representations and constructions. Location ants select the facilities to be opened, and their solutions are not uniform, in the sense that each ant can open a different number of facilities, according to:

$$f_a = \left\lfloor \frac{\sum_i d_i}{\sum_j (s_j/m)} \right\rfloor + U[0, r], \tag{6}$$

where f_a is the number of facilities to be opened by ant a, d_i is the demand of customer i, s_j is the supply on node j, m is the number of available locations for the facilities, and r is a pre-specified integer constant with a value between the first term of the sum in equation (6) and m. After selecting the facilities to be opened, assignment ants assign each customer to one and only one facility but they do not acknowledge, at least in this phase, whether the solution is feasible or not, from the supply capacity point of view. The unfeasible solutions are dealt with by using penalties, in the local search phase.

3.1.2 Visibility Information

The heuristic information, also known as visibility, is an extra information available to the ant algorithm which is usually referred to as a kind of local information. Originally used as the inverse of the length of the arc between two cities in the TSP, it has suffered several mutations throughout the hundreds of approaches that have been developed since then.

Lessing et al (2004) studied the influence of the heuristic information, also called the visibility value, in the performance of ant algorithms when solving Set Covering Problems (SCPs). Two types of heuristic information are studied, static heuristic information, where the values are calculated only once, at the beginning of the algorithm, and dynamic heuristic information, in which case the values are calculated at each construction step of each ant, as in the case of the ant-density algorithm. The different heuristic information values used are based on the Column Costs,

$$\eta_j = \frac{1}{c_j}; \tag{7}$$

the (normalized) Lagrangean Costs C_j,

$$\eta_j = \frac{1}{C_j}; \tag{8}$$

the Cover Costs,

$$\eta_j = \frac{card_j(S)}{c_j}, \tag{9}$$

where $card_j(S)$ is the number of rows covered by a column j;

(normalized) Lagrangean Cover Costs,

$$\eta_j = \frac{card_j(S)}{C_j}; \tag{10}$$

Marchiori and Steenbeck Cover Costs,

$$\eta_j = \frac{1}{(c_j/cv(j,S)}, \tag{11}$$

where $cv(j,S)$ is the sum, for all rows covered by column j but not covered by any other column in $S \setminus \{j\}$, of the minimum cover costs;

Marchiori and Steenbeck Lagrangean Cover Costs with Normalized Costs,

$$\eta_j = \frac{1}{(C_j/cv(j,S)}; \tag{12}$$

and finally lower bounds, where the heuristic information for column j is the inverse of the cost of the lower bound obtained by tentatively adding column j.

Each of these heuristic information types was tested with four different ant algorithms, and the results obtained suggest that different types of heuristic information should be used for different types of ant algorithms.

Reimann and Laumanns (2006) use savings values as the heuristic information instead of the usual inverse of the arc cost,

$$\eta_{ij} = S_{ij}, \tag{13}$$

in an ACO algorithm to solve Capacitated Minimum Spanning Tree problems. The savings $S_{ij} = c_{i0} + c_{0j} - c_{ij}$ are related to the cost difference obtained by merging the subtrees of node i and j, previously linked directly to the source node 0. In this case, the larger the savings associated to an arc the higher probability of that arc being selected.

A Capacitated Fixed-Charge Location problem aims at deciding on the supply facilities that must be opened such that they can satisfy all the customers demand at the lowest possible cost. Venables and Moscardini (2006) developed an ACO based algorithm that defines and uses the information of a matrix called the total opportunity matrix T_{ij}. This matrix uses the sum of the differences between the cost of each arc (i,j), that is c_{ij}, and both c_{i^*j} the lower supplying cost from facility i and C_{ij^*} the lower supplying cost to customer j, such that $T_{ij} = (c_{ij} - c_{ij^*}) + (c_{ij} - c_{i^*j})$. At the end, the visibility is set to be the facility visibility, and is defined as the sum on the customers index of the total opportunity cost,

$$\eta_i = \sum_{j=1}^{n} T_{ij}. \tag{14}$$

The lower the T_{ij} the higher the visibility and probability of an arc to be chosen.

In the work of Altiparmak and Karaoglan (2007) the heuristic information is based on the concave nature of the arcs costs in the Transportation Problem to be solved. In this case, the

heuristic information takes into account not only the cost of the arc c_{ij} but also the flow of that arc x_{ij}, that is the unit transportation cost

$$\eta_{ij} = \frac{1}{(c_{ij}/\sqrt{x_{ij}})}. \tag{15}$$

Pour and Nosraty (2006) have solved the NP-hard plant/facility location problem with an ACO algorithm. In this problem, there is a set of existing facilities p_i and a set of locations where new facilities x_j are to be located and the objective is to locate these new facilities, such that the sum of the costs between the facilities is minimized, and each location is assigned a single facility. In this algorithm, the heuristic information used by the authors is defined as the inverse of the product of distances d_i and costs f_i between existing facility i and all new facilities x_j, taking the form of

$$\eta_{ij} = \frac{1}{f_i \cdot d_j}. \tag{16}$$

This way, nearest and lower cost facilities have a better heuristic value.

The Minimum Weight Vertex Cover problem is solved by Shyu et al (2004) with an ACO algorithm where the heuristic information is defined for pairs of the type $(node, ant)$. In it, the heuristic information is defined as the local preference of ant k to choose node j to enter the solution, and it translates into the ratio between the number of arcs linked to node j but not yet covered by ant k and the weight of node j. Being thus defined, this heuristic information is not static because its value changes with each step of the construction of the solution, and also from solution to solution, since ants may construct different solutions.

Crawford and Castro (2006) calculate the value of the heuristic information in a dynamic fashion, to solve Partitioning and Covering problems. At each step of the construction of the solution, the algorithm computes the heuristic information as the per unit cost of covering an additional row, as given bellow

$$\eta_j = \frac{e_j}{c_j}, \tag{17}$$

where e_j is the number of additional rows that are covered by node j when it is added to the partial solution already constructed.

In the Cell Assignment Problem a set of cells must be linked to a set of switches such that each cell is associated to exactly one switch, however switches may be linked to several cells. Shyu et al (2006) define two heuristic information matrices, instead of the usual single one, to be used in an ACO algorithm developed to solve the Cell Assignment Problem. These matrices are associated with the choice of to which switch to move to when located at a certain cell, and vice-versa. On the former, the decision uses a heuristic information function based on the inverse of the partial costs. This heuristic information is dynamic since whenever an arc (c_i, s_j) linking

a cell c_i and a switch s_j is included in the partial solution, the heuristic value for that arc will be updated with the inverse of the partial solution cost constructed to the moment. This update is performed at each step of the construction procedure. Therefore, the higher the partial cost the lower the value of the heuristic information η_{c_i,s_j}. Whenever an ant is located at a certain switch it must choose to which cell to move to. In order to do so, the heuristic information used is the call volume associated with each cell, thus η_{c_i} is defined for cells rather than for arcs. Therefore, the larger the cell volume the higher the value of the heuristic, thus favouring cells with high call volumes to be handled first.

3.1.3 Pheromone Bounds

At some point on the run of an ACO algorithm, the values of the pheromones in the components of the solution, let us say arcs, may be extremely small, almost prohibiting the choice of those arcs, or extremely large which will lead to the construction of the same solution, over and over again. To prevent that from happening one might set upper and a lower bounds on the pheromones. The first work to introduce this mechanism was (Stützle and Hoos, 1997), and the authors define the following pheromone bounds:

$$\tau_{max} = \frac{1}{\rho F^*}, \tag{18}$$

where the pheromone upper bound τ_{max} depends not only on the evaporation rate ρ but also on the total cost of the best solution found so far F^*;

$$\tau_{min} = \frac{\tau_{max}(1 - p_{dec})}{(\frac{n}{2} - 1)p_{dec}}, \tag{19}$$

where the pheromone lower bound τ_{min} depends on the value of τ_{max}, on the the number n of the components of the solution, and on the probability of constructing the best solution p_{dec}, which is a value to be set.

Therefore, whenever a new global best solution is found, τ_{min} must also be updated.

Venables and Moscardini (2006) and Altiparmak and Karaoglan (2007) both define the upper bound τ_{max}, as in equation (18). The minimum pheromone value allowed is given by a fraction of the maximum pheromone value allowed,

$$\tau_{min} = \tau_{max}/a, \tag{20}$$

where a is a parameter value given by the size of the problem both in the work of Venables and Moscardini (2006) and in the work of Altiparmak and Karaoglan (2007). It is easy to see that τ_{min} and τ_{max} are not static values changing whenever a new best solution is found.

Another mechanism also based on pheromone trails is the identification of stagnation. Altiparmak and Karaoglan (2007) use a two phase reinitialization scheme. On the one hand, if more than 50% of the arcs in a transportation network have pheromone values equal to τ_{min}, then τ_{max} and τ_{min} are updated according to the global best solution and all values in the pheromone matrix are set to τ_{max}. On the other hand, if the global best solution has not been updated for 50 iterations, then 10% of the population is randomly generated and will replace the worst solutions.

Blum and Blesa (2005) propose an ACO algorithm to solve edge-weighted k-cardinality tree problems, where the pheromone values are in the interval $[0, 1]$, following the HyperCube Framework defined by Blum et al (2001). In order to define an usable value for each limit, the minimum and maximum pheromone values are as given in equation (21).

$$[\tau_{min}, \tau_{max}] = [0.001, 0.999]\,. \tag{21}$$

The algorithm also incorporates a so-called convergence factor cf, defined in equation (22), in order to estimate the degree of convergence of the system,

$$cf = \frac{\sum_{a \in A(S_k^i)} \tau_a}{k \cdot \tau_{max}}, \tag{22}$$

where k is the cardinality of the problem, A is the set of arcs belonging to the best k-cardinality tree of the iteration, τ_{max} is the already defined maximum pheromone value, and finally τ_a is the pheromone of arc a. By definition cf is a value always between 0 and 1, and the closer cf is to 1, the closer is the system to convergence because the probability to construct again S_k^i is closer to 1. When this happens pheromone values and the best solution are reset.

Bui and Zrncic (2006), which address degree-constrained Minimum Spanning Trees, define the maximum and the minimum value allowed for pheromone levels based on the differences between the cost M of the most expensive arc and the cost m of the cheapest arc, as follows

$$\tau_{max} = 1000 \cdot \frac{(M - m) + (M - m)}{3} \tag{23}$$

and

$$\tau_{min} = \frac{M - 3}{3}. \tag{24}$$

Whenever an arc exceeds τ_{max} it is not reset to τ_{max}, as usual, but rather adjusted to $\tau_{max} - \tau_{ij}^{init}$, where τ_{ij}^{init} is the initial pheromone value for arc (i, j) and is given by $\tau_{ij}^{init} = (M - c_{ij}) + (M - m)/3$. In a similar way, $\tau_{ij} = \tau_{min} + \tau_{ij}^{init}$ whenever the pheromone value goes under τ_{min}. This way, as some of the original information is maintained it is expected that the ant still recognizes good arcs and bad arcs.

Bin et al (2009) use lower and upper bounds for the pheromone values in the arcs of the routes found by the ants, for the vehicle routing problem, which are dependent on the distance d_{0i} between the central supply node 0 and each customer node i. The bounds are given by

$$[\tau_{min}, \tau_{max}] = \left[\frac{Q}{\sum_i d_{0i}}, \frac{Q}{\sum_i 2d_{0i}} \right], \tag{25}$$

where Q is a parameter. These bounds are calculated only once, at the beginning of the algorithm. The algorithm incorporates a mutation operator, in a similar fashion to genetic algorithms, to try to include arcs other than the ones with higher pheromone value, chosen by influence of the probability function. Given two parent tours from a solution, two customers, one from each tour, are randomly selected and exchanged. If this operation turns out to result into unfeasible solutions, then they are fixed by using a repairing mechanism. Thus, two new feasible solutions are always created.

3.1.4 Pheromone Update

In the definition of the ACO metaheuristic the pheromone update has been defined to be performed after all the ants have constructed their solutions. Although it is the recommended/suggested method, it has not been proven to be the best choice for all problems. In fact, different pheromone update schemes have been provided in the literature differing in three key aspects: the moment at which pheromones are updated, the pheromone quantity to be deposited and evaporated, and which ants are allowed to deposit pheromone in their trails.

The work of Talbi et al (2001) is one of those cases where an alternative approach has proven to achieve good results. In order to solve a Quadratic Assignment Problem, the pheromone update instead of reinforcing the components of the best solution found, as is usually done, reinforces every solution $F(S)$ taking into account both the value of the best ($F(S^*)$) and the value of the worst ($F(S^-)$) solutions found, as follows

$$\tau_{ij} = (1 - \rho) \times \tau_{ij} + \frac{\rho}{F(S)} \times \frac{F(S^-) - F(S)}{F(S^*)}. \tag{26}$$

The intention is to weaken the reinforcement, preventing a quick convergence, due to the unusual large number of ants depositing pheromone on their solutions.

A different approach is that of Rappos and Hadjiconstantinou (2004) that was developed to design flow networks that are two-edge connected, that is, that can continue to satisfy the customers demands if any single arc in the network is removed. Having in consideration the nature of the problem, the authors decided to make a distinction between two types of pheromone values associated to each arc. One, $T_e(ij)$, is called the *arc trail* and is related to the fixed cost that

has to be paid for using that arc. The other one, $T_f(ij)$, is called the *flow trail* and is related to the cost of the flow passing through the arc. Flow ants, which can detected and reinforce both pheromone types, are created, as well as reliability ants, that can only detect and reinforce arc pheromone. In each iteration a single solution is produced by the flow ants, and then the solution is made reliable by adding an extra arc by the reliability ants. Both flow trails and reliability trails are updated, at the end of the corresponding construction phase, by initially performing a reduction on the pheromone of all arcs. Then, each reliability ant adds:

$$\Delta T_e(ij) = \frac{1}{b_{ij}} \tag{27}$$

to each arc on its solution, regarding the arc pheromone trail, where b_{ij} is the fixed cost to be incurred by using arc (i, j). Each flow ant adds the following quantities to the arc and flow pheromone trails, respectively, on the arcs of its solution, provided that fixed-costs are only paid once

$$\Delta T_e(ij) = \frac{1}{b_{ij}} \text{ and } \Delta T_f(ij) = \frac{1}{c_{ij} d_j}, \tag{28}$$

where c_{ij} is the cost per unit flow and d_j is the demand of node j. The reason why reliability ants to not deposit pheromone on flow trails is straightforward, the extra arc that they add to the solution does not carry any flow.

In a work by Alaya et al (2004), where multidimensional knapsack problems are solved, the pheromone update is done in such a way that the quantity deposited in each component of the solution includes information about the difference between the objective function value of the best solution of the iteration $F(S^{it})$ and of the global best solution $F(S^*)$,

$$\Delta \tau_{ij} = \frac{1}{1 + F(S^*) - F(S^{it})}. \tag{29}$$

Therefore, the closer the solution is to the global best solution, the higher the quantity of pheromone deposited.

Two pheromone updating rules are proposed in (Shyu et al, 2004), a global and a local one. On the one hand, at the end of each iteration, and after evaporation is applied, the pheromone present on the nodes of the incumbent solution S^* are reinforced with a quantity inversely proportional to the total weight of the nodes in the solution.

$$\Delta \tau_i = \frac{1}{\sum_{j \in S^*} w_j}. \tag{30}$$

On the other hand, the local pheromone updating rule is applied each time the ant adds a node into its solution and is given by

$$\tau_i = (1 - \varphi)\tau_i + \varphi\tau_0 \tag{31}$$

where τ_0 is the initial pheromone laid in every node and φ is the evaporation rate applied locally. This latter rule has the objective of preventing the ants of always choosing the most significant node. Eswaramurthy and Tamilarasi (2009) have also used a similar global and local updating rule but considering arcs instead of nodes. It should be noticed that while Shyu et al (2004) solve the Minimum Weight Vertex Cover problem, Eswaramurthy and Tamilarasi (2009) solve the Job Shop Scheduling problem.

Solimanpur et al (2005), have also considered depositing more pheromone in the components of the solutions closer to the global best solution. In this case, they allow not only the best ant in the iteration to deposit pheromone but also all other ants. The amount of pheromone to be deposited by ant k is given by

$$\Delta\tau_{ij}^k = \lambda \cdot \frac{F(S^*)}{F(S^k)}, \tag{32}$$

where $F(S^k)$ is the solution of ant k, and λ is a scaling factor that must be chosen appropriately such that a quick convergence to a local optima may be avoided. This method clearly encourages search along the vicinities of the global best solution in the hope that a better one can be found nearby.

According to the value defined in equation (22), Blum and Blesa (2005) define a pheromone updating rule rewarding three solutions: the best solution in the current iteration S_k^{ib}, the best global solution to the moment S_k^{gb}, and the restart-best solution S_k^{rb}, that is, the best solution found at the restart of the algorithm. The reinforcement is then not based on the fitness of the solution, that is, the corresponding value of the objective function, but rather on the value of a convergence factor cf, see equation (22), which is computed at every iteration. Each of these three solutions is attributed a different weight k_{ib}, k_{gb}, and k_{rb}, defined in the same manner as above, such that their sum equals 1. The schedule the authors have applied is dependent on cf in such a way as to increase the value of k_{rb} and decrease the value attributed to k_{ib} with the increase of cf, if $cf < 0.99$. The value of the evaporation rate parameter is also dynamic, decreasing with the increase of cf. When a global convergence has been reached, that is, when $cf \geq 0.99$ then the only solution being updated is S_k^{gb} since a reset of the algorithm is to be made, and this is the only solution to be maintained. The pheromone values are updated, as well, initially by evaporating a percentage of the pheromone present in each arc, and then by adding the following pheromone quantity in each arc,

$$\xi_a = k_{ib}\delta(S_k^{ib}, a) + k_{rb}\delta(S_k^{rb}, a) + k_{gb}\delta(S_k^{gb}, a), \tag{33}$$

where $\delta(S_k, a) = 1$ if arc a belong to the solution tree S_k, and 0 otherwise.

Following the work of Bin et al (2009), Yang et al (2007) use an ant-weight pheromone updating strategy based on the ant-density algorithm, in the Improved ACO used to solve the Vehicle Routing Problem. The idea behind it is to incorporate both local and global information about solutions. Therefore, every ant, representing a single route, deposits pheromone in its solution components, following

$$\Delta\tau_{ij}^{k} = \frac{Q}{K \times L} \times \frac{D^k - d_{ij}}{m^k \times D^k}, \tag{34}$$

where Q is the usual proportionality constant parameter, L is the sum of the lengths of all tours in the solution, K is the total number of routes in the solution, D^k is the length of tour k, d_{ij} is the distance between customer i and customer j, and m^k is the number of customers visited in route k. Note that, a solution is only entirely defined when all routes constructed are assembled. The first component $\frac{Q}{K \times L}$, the global pheromone increment, depends on the total length of the solution and on the number of tours, and it represents a compromise between the total cost and the number of vehicles used. The second component $\frac{D^k - d_{ij}}{m^k \times D^k}$, the local pheromone increment, uses the contribution of arc (i, j) to the kth tour, which increases as d_{ij} decreases.

3.1.5 Transition Rule and Probability Function

This may be considered the characteristic that has less suffered from the evolution of ant algorithms. Its initial structure, as given in equation (1), is almost always used in the works of the researchers in the area. Nonetheless, different methods have been introduced mainly associated to the high complexity of the problem to be solved.

The probability distribution used by Bouhafs et al (2006) to calculate the probability of visiting customer j when in customer i, in a Capacitated Location-Routing problem, also incorporates the savings value γ_{ij}, for visiting customer j from customer i:

$$P_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta} \cdot [\gamma_{ij}]^{\lambda}}{\sum_{j \in J_i^k} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}^{\beta}] \cdot [\gamma_{ij}]^{\lambda}} \tag{35}$$

where J_i^k is the set of costumers not yet visited by ant k in its solution and that, by being chosen, do not violate any constraint. The savings value is computed once at the beginning of the algorithm as follows

$$\gamma_{ij} = d_{i0} + d_{i0} - g \cdot d_{ij} + f \cdot |d_{i0} - d_{0j}|, \tag{36}$$

where g and f are parameters, d_{ij} is the distance between nodes i and j, and 0 is the starting node.

Afshar (2005) proposes a new transition rule for ant colony optimization algorithms, that is

given by:

$$P_{ij}^{k}(t) = \frac{\alpha\tau_{ij}(t) + \beta\eta_{ij}}{\sum_{j \in J_i^k}[\alpha\tau_{ij}(t) + \beta\eta_{ij}]}. \tag{37}$$

The strategy is defined to prevent a domination of the pheromone trails in the ants decision, by incorporating an additive form instead of the usual multiplicative form. This way, the author expects both pheromone and heuristic information to have an active role in the decision. This new transition rule comes with a modification of the heuristic value, which is a simple scaling procedure given by:

$$\eta_{ij}^{s} = \frac{\eta_{ij}}{\max(\eta_{ij})}, \tag{38}$$

making every value to be between zero and one regardless of problem size, a difficulty which was already mentioned before.

A probability function based on the one developed by Maniezzo (1999) for the Quadratic Assignment problem is used within an algorithm developed to solve single row layout problems by Solimanpur et al (2005). The function also presents an additive scheme but eliminates the necessity of the parameter β associated to the heuristic value

$$P_{ij}^{k}(t) = \frac{\alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}}{\sum_{j \in J_i^k}[\alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}]}. \tag{39}$$

In this case, it is clear that α must be a number between zero and one, and not any positive number as was the case of the original method. Therefore, if one wishes to prioritize the pheromone information one is implicitly decreasing the importance of the heuristic information, and vice-versa, and there is only one value for which they have the same weight, which is 0.5.

Blum and Blesa (2005) introduced some changes to the transition rule defined for the Ant Colony System (ACS), earlier developed by Dorigo and Gambardella (1997), in order to solve k-minimum spanning tree problems. An ant starts its solution by randomly choosing the first arc to enter the solution tree. Then, at each step of the construction, the next arc a to be added is chosen deterministically if $q \leq 0.8$, and probabilistically if $q > 0.8$, according to equation (40):

$$a = \begin{cases} \arg\min\left\{\dfrac{\tau_a}{w(a)} : a \in A_{NH}(S_{t-1})\right\} & \text{if } q \leq 0.8 \\ l & \text{if } q > 0.8, \end{cases} \tag{40}$$

where τ_a is the pheromone in arc a, $w(a)$ is the weight of arc a, $A_{NH}(S_{t-1})$ is the set of all arcs that do not belong to solution S_{t-1} and have exactly one end-point in S_{t-1}, and where

$$l = \begin{cases} \dfrac{\tau_a / w_a}{\sum\limits_{a'} \in A_{NH}(S_{t-1})} \tau_{a'} / w_{a'} & \text{if } a \in A_{NH}(S_{t-1}) \\ 0 & \text{otherwise.} \end{cases} \tag{41}$$

This rule, assigns equal weight to the pheromone and the heuristic values, hereby represented by $1/w_a$, by eliminating parameters α and β from the exponents of the pheromone and heuristic values respectively. Given that the probabilistic rule is only triggered whenever a random number $q > 0.8$, the search for solutions is in 80% of the cases usually concentrated on relatively good areas.

3.1.6 Parameter Values

The setting of an ant based algorithm can take a long time to achieve in order to produce some useful results. Furthermore, a set of parameter values has also to be defined:

- α - parameter related to the weight of the pheromone concentration in the probability function;
- β - parameter weighting the relative importance of heuristic information in the probability function;
- ρ - pheromone evaporation rate, where $\rho \in]0, 1]$, measures the information that is to be transported to the next iteration;
- Q - parameter weighting the quantity of pheromone to be deposited in each component of the solution;
- τ_0 - initial pheromone value to be deposited in every solution component, to guarantee that every one of them has, at least, a small probability of being chosen;
- number of ants in the colony;
- stopping criterion - the number of iterations to be performed, the number of solutions to be evaluated, maximum allowed running time, and so on.

For each algorithm developed, there can be other parameters to be set, for example, if bounds are imposed on the pheromone values, a p_{best} parameter, as well as, the limit values have to be defined. There are other cases where differences in the definition of the probability function, or the type of ant used, require more parameters. We feel that there is no need to report on these parameters in a section of their own as they tend to be unique for each algorithm. Nonetheless, almost every work that was reviewed in this paper reports to have tried several combinations of parameter values before choosing the ones to be used.

3.2 Books and Surveys

Reviews are very important, specially when someone is starting on a new research area. Therefore, we could not finish this work without referring to some of the detailed and comprehensive reviews. For a good review on early Ant Colony Optimization historical applications we refer to (Cordon et al, 2002). The reader may also find the review of Mullen et al (2009) very interesting, since the authors review the application of ant algorithms in fields such as digital image processing, data mining and other machine learning techniques. In this work we have omitted multi-criteria combinatorial optimization problems. A good work reviewing this type of problems is provided by García-Martínez et al (2007), where the authors, besides providing a survey on previous works also solve a set of instances of the bi-criteria TSP with several ACO algorithms, in order to be able to compare them and discuss their characteristics.

Although a little out-of-date, due to the large number of works that have seen the broad daylight after they have been published, (Bonabeau et al, 1999) and (Dorigo and Stützle, 2004) are still very important references regarding ant based algorithms, providing excellent explanations on ant algorithms and their evolution. The first book gives us an insight on the general social insect behaviour with particular emphasis on ant algorithms. The second book is fully dedicated to ant colony algorithms and surveys several applications of ACO in several fields, such as scheduling, machine learning and bio-informatics. In addition, it also discusses some theoretical findings and is an excellent guide to everyone who wishes to implement ant algorithms.

4 Conclusion

The class of combinatorial optimization problems is prolific in NP-hard problems. But, although some small instances of such problems can be solved with exact methods, heuristics are more adequate to solve large instances as they usually need far less computational resources. Ant Colony Optimization is a metaheuristic initially defined to solve problems within the class of combinatorial optimization, although its frontiers have long been overcome. In this work, we have presented a collection of different approaches that can be found in the literature, regarding the ACO building blocks. The algorithms that were reviewed have been used to solve all sorts of problems, but mainly problems within the combinatorial optimization class. Our objective is to provide a list of alternative methods that can be used for each ACO feature, in order to facilitate the identification of the most adequate technique or simply to inspire an investigator that is thinking on developing his own ACO algorithm.

References

Afshar MH (2005) A new transition rule for ant colony optimization algorithms: application to pipe network optimization problems. Engineering Optimization 37(5):525–540

Alaya I, Solnon C, Ghédira K (2004) Ant algorithm for the multi-dimensional knapsack problem. In: International Conference on Bioinspired Optimization Methods and their Applications (BIOMA 2004, pp 63–72

Altiparmak F, Karaoglan I (2006) An adaptive tabu-simulated annealing for concave cost transportation problems. Journal of the Operational Research Society 59(3):1–11

Altiparmak F, Karaoglan I (2007) A genetic ant colony optimization approach for concave cost transportation problems. In: Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, pp 1685–1692

Baykasoglu A, Dereli T, Sabuncu I (2006) An ant colony algorithm for solving budget constrained and unconstrained dynamic facility layout problems. Omega 34:385–396

Bernardino EM, Bernardino AM, Sánchez-Pérez JM, Gómez-Pulido JA, Vega-Rodríguez MA (2009) A hybrid ant colony optimization algorithm for solving the terminal assignment problem. In: IJCCI 2009 - International Joint Conference on Computational Intelligence

Bin Y, Zhong-Zhen Y, Baozhen Y (2009) An improved ant colony optimization for vehicle routing problem. European Journal of Operational Research 196:171–176

Blum C, Blesa MJ (2005) New metaheuristic approaches for the edge-weighted k-cardinality tree problem. Computers & Operations Research 32:1355–1377

Blum C, Roli A, Dorigo M (2001) Hc-aco: the hyper-cube framework for ant colony optimization. In: Proceedings of Metaheuristics International Conference MIC'2001, Porto, Portugal, July 2001, pp 399–403

Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm Intelligence - from Natural to Artificial Systems. Oxford University Press, New York

Bouhafs L, Hajjam A, Koukam A (2006) A combination of simulated annealing and ant colony system for the capacitated location-routing problem. In: KES (1), pp 409–416

Bui TN, Zrncic CM (2006) An ant-based algorithm for finding degree-constrained minimum spanning tree. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM, New York, USA, GECCO '06, pp 11–18

Bullnheimer B, Hartl RF, Strauß C (1997) A new rank based version of the ant system - a computational study. Central European Journal for Operations Research and Economics 7:25–38

Chen CH, Ting CJ (2008) Combining lagrangian heuristic and ant colony system to solve the single source capacitated facility location problem. Transportation Research Part E: Logistics and Transportation Review 44(6):1099 – 1122, DOI DOI: 10.1016/j.tre.2007.09.001

Cordon O, Herrera F, Stützle T (2002) A review on the ant colony optimization metaheuristic: Basis, models and new trends. Mathware & Soft Computing 9:141–175

Crawford B, Castro C (2006) Integrating lookahead and post processing procedures with aco for solving set partitioning and covering problems. In: ICAISC, pp 1082–1090

Deneubourg JL, Aron S, Goss S, Pasteels JM (1990) The self-organizing exploratory pattern of the argentine ant. Journal of Insect Behaviour 3:159–168

Dorigo M, Di Caro G (1999) The ant colony optimization meta-heuristic, McGraw-Hill Ltd., UK, Maidenhead, UK, England, pp 11–32

Dorigo M, Gambardella LM (1997) Ant colony system: A cooperative learning approach to the travelling salesman problem. IEEE Transactions on Evolutionary Computation 1:53–66

Dorigo M, Stützle T (2004) Ant Colony Optimization. MIT Press, Cambridge, MA

Dorigo M, Maniezzo V, Colorni A (1996) The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics - Part B 26(1):29–41

Dorigo M, Caro GD, Gambardella LM (1999) Ant algorithms for discrete optimization. Artificial Life 5(2):137–172

Eswaramurthy V, Tamilarasi A (2009) Hybridizing tabu search with ant colony optimization for solving job shop scheduling problems. The International Journal of Advanced Manufacturing Technology 40:1004–1015

García-Martínez C, Cordón O, Herrera F (2007) A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. European Journal of Operational Research 180(1):116–148

Goss S, Aron S, Deneubourg J, Pasteels J (1989) Self-organized shortcuts in the argentine ant. Naturwissenschaften 76:579–581

Lessing L, Dumitrescu I, Stützle T (2004) A comparison between aco algorithms for the set covering problem. In: ANTS, pp 1–12

Maniezzo V (1999) Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. INFORMS journal on computing 11:358–369

Mehrabi AD, Mehrabi S, Mehrabi A (2009) A prunning based ant colony algorithm for minimum vertex cover problem. In: International Joint Conference on Computational Intelligence (IJCCI'09), pp 281–286

Monteiro MSR, Fontes DBMM, Fontes FACC (2012) Concave minimum cost network flow problems solved with a colony of ants. Journal of Heuristics (To appear)

Mullen R, Monekosso D, Barman S, Remagnino P (2009) A review of ant algorithms. Expert Systems with Applications 36:9608–9617

Musa R, Arnaout JP, Jung H (2010) Ant colony optimization algorithm to solve for the transportation problem of cross-docking network. Computers & Industrial Engineering 59(1):85 – 92

Neumann F, Witt C (2010) Ant colony optimization and the minimum spanning tree problem. Theoretical Computer Science 411(25):2406 – 2413

Pour HD, Nosraty M (2006) Solving the facility layout and location problem by ant-colony optimization-meta heuristic. International Journal of Production Research 44:5187–5196

Rappos E, Hadjiconstantinou E (2004) An ant colony heuristic for the design of two-edge connected flow networks. In: ANTS Workshop, pp 270–277

Reimann M, Laumanns M (2006) Savings based ant colony optimization for the capacitated minimum spanning tree problem. Computers & Operations Research 33:1794–1822

Santos L, ao Coutinho-Rodrigues J, Current JR (2010) An improved ant colony optimization based algorithm for the capacitated arc routing problem. Transportation Research Part B: Methodological 44(2):246 – 266

Shyu SJ, Yin PY, Lin BMT (2004) An ant colony optimization algorithm for the minimum weight vertex cover problem. Annals of Operations Research 131:283–304

Shyu SJ, Lin BMT, Hsiao TS (2006) Ant colony optimization for the cell assignment problem in pcs networks. Computers & Operations Research 33(6):1713–1740

Solimanpur M, Vrat P, Shankar R (2004) Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing. European Journal of Operational Research 157:592–606

Solimanpur M, Vrat P, Shankar R (2005) An ant algorithm for the single row layout problem in flexible manufacturing systems. Computers & Operations Research 32:583–598

Stützle T, Hoos H (1997) Max-min ant system and local search for the traveling salesman problem. In: IEEE International Conference On Evolutionary Computation (ICEC'97), IEEE Press, Piscataway,NJ, pp 309–314

Talbi EG, Roux O, Fonlupt C, Robillard D (2001) Parallel ant colonies for the quadratic assignment problem. Future Generation Computer Systems 17:441–449

Tavares J, Pereira F (2011) Evolving strategies for updating pheromone trails: A case study with the tsp. In: Schaefer R, Cotta C, Kolodziej J, Rudolph G (eds) Parallel Problem Solving from Nature - PPSN XI, Lecture Notes in Computer Science, vol 6239, Springer Berlin-Heidelberg, pp 523–532

Venables H, Moscardini A (2006) An adaptive search heuristic for the capacitated fixed charge location problem. In: Dorigo M, Gambardella L, Birattari M, Martinoli A, Poli R, Stützle T (eds) Ant Colony Optimization and Swarm Intelligence, Lecture Notes in Computer Science, vol 4150, Springer Berlin / Heidelberg, pp 348–355

Yang Z, Yu B, Cheng C (2007) A parallel ant colony algorithm for bus network optimization. Computer-Aided Civil and Infrastructure Engineering 22:44–55

Recent FEP Working Papers

473	João Correia-da-Silva and Carlos Hervés-Beloso, *Irrelevance of private information in two-period economies with more goods than states of nature*, December 2012
472	Gonçalo Faria and João Correia-da-Silva, *Is Stochastic Volatility relevant for Dynamic Portfolio Choice under Ambiguity?*, October 2012
471	Helena Martins and Teresa Proença, *Minnesota Satisfaction Questionnaire - Psychometric Properties and Validation in a Population of Portuguese Hospital Workers*, October 2012
470	Pedro Mazeda Gil, Oscar Afonso and Paulo Brito, *Skill Structure and Technology Structure: Innovation and Growth Implications*, September 2012
469	Abel L. Costa Fernandes and Paulo R. Mota, *Triffin's Dilemma Again and the Efficient Level of U.S. Government Debt*, September 2012
468	Mariana Cunha and Vera Rocha, *On the Efficiency of Public Higher Education Institutions in Portugal: An Exploratory Study*, September 2012
467	Paulo R. Mota, Abel L. Costa Fernandes and Ana-Cristina Nicolescu, *The Recent Dynamics of Public Debt in the European Union: A Matter of Fundamentals or the Result of a Failed Monetary Experiment?*, September 2012
466	Elena Sochirca, Oscar Afonso and Sandra Silva, *Political rivalry effects on human capital accumulation and inequality: a New Political Economy approach*, September 2012
465	Mariana Cunha and Paula Sarmento, *Does Vertical Integration Promote Downstream Incomplete Collusion? An Evaluation of Static and Dynamic Stability*, August 2012
464	Andreea Stoian and Rui Henrique Alves, *Can EU High Indebted Countries Manage to Fulfill Fiscal Sustainability? Some Evidence from the Solvency Constraint*, August 2012

Editorial Board (wps@fep.up.pt)
Download available at: `http://wps.fep.up.pt/wplist.php`
also in `http://ideas.repec.org/PaperSeries.html`

www.fep.up.pt

School of Economics and Management, University of Porto
Rua Dr. Roberto Frias | 4200-464 Porto | Portugal
Telephone: +351 225 571 100