Quantitative Analysis and Performance Study of Ant Colony Optimization Models Applied to Multi-Mode Resource Constraint Project Scheduling Problem

Antonio Gonzalez-Pardo[1], Javier Del Ser[2,3,4], and David Camacho[1]

[1] Escuela Politécnica Superior, Universidad Autónoma de Madrid, 28049, Madrid, Spain
{antonio.gonzalez,david.camacho}@uam.es.
[2] OPTIMA Area, TECNALIA, 48160 Bizkaia, Spain
javier.delser@tecnalia.es
[3] University of Basque Country EHU/UPV, 48009 Bizkaia Spain
javier.delser@ehu.es
[4] Basque Center for Applied Mathematics, 48009 Bizkaia Spain

Abstract. Constraint Satisfaction Problems (CSP) belongs to this kind of traditional NP-hard problems with a high impact in both, research and industrial domains. However, due to the complexity that CSP problems exhibit, researchers are forced to use heuristic algorithms for solving the problems in a reasonable time. One of the most famous heuristic algorithms is Ant Colony Optimization (ACO) algorithm. The possible utilization of ACO algorithms to solve CSP problems requires the design of a decision graph where the ACO is executed. Nevertheless, the classical approaches build a graph where the nodes represent the variable/value pairs and the edges connect those nodes whose variables are different. In order to solve this problem, a novel ACO model have been recently designed. The goal of this paper is to analyze the performance of this novelty algorithm when solving Multi-Mode Resource-Constraint Satisfaction Problems. Experimental results reveals that the new ACO model provides competitive results whereas the number of pheromones created in the system is drastically reduced.

Keywords: Ant Colony Optimization, Oblivion Rate, Resource-Constraint Project Scheduling Problems, Pheromone Control

1 Introduction

One of the compounding paradigms within the set of NP-hard problems is related to as Resource-Constraint Project Scheduling Problem (RCPSP) [14]. This family of problems is defined by a set of variables that need to be assigned with a particular value taking into account a set of restrictions that establish constraints among the different values assigned to the variables. Therefore, any Constraint Satisfaction Problem (CSP) is represented with a triple (X, D, C)

where $X = \{x_1, x_2, \dots x_n\}$ represents the set of objects that composes the problem, $D = \{d_1, d_2, \dots d_n\}$ is used to describe the domains that contain the different values for the objects described in X, and C represents the set of constraints that relates the objects with their values [3, 14].

There is a wide number of complex research and industrial problems that can be modelled as a CSP, the main techniques, algorithms and methods obtained from this area have been applied in the last decades to real domains with an increasing level of complexity (i.e. scheduling and planning problems, energy optimization, man-power scheduling, travel and car routing optimization, etc...) [1, 8, 11, 12, 14].

Due to the inherent complexity of CSP problems, it is common to use Computational Intelligence algorithms (such as Ant Colony Optimization (ACO)) to solve these problems. In order to use ACO for solving CSP problems, the solution space must be represented as a graph (called *decision graph*) over which the ACO algorithm is executed. The standard approaches for building this *decision graph* presents several drawbacks that make it difficult to apply this approach to CSP instances of moderate to high dimensionality. To solve this problem, a new CSP-graph based model was proposed in [6], where the reduction in the size of the *decision graph* results in a fast growth in the number of pheromones. In order to control this increase rate of the number of pheromones created in the model, a new heuristic called *Oblivion Rate* was included in the model. This model has been applied to the *N-Queens problem* [6], the *Resource-Constraint Project Scheduling Problem* [7] and the *Lemmings Game* [9]. The contribution of this work is the analysis of the performance of the *Oblivion Rate* heuristic in a new family of RCPSP problems called, *Multi-Mode Resource Constraint Project Scheduling Problems.*

This paper is structured as follows: Section 2 contains the description for the RCPSPs. Section 3 details the implementation of the ACO model used to RCPSP, including the definition of the new decision graph, the behaviour of the ants, and the *Oblivion Rate* heuristic used in this work. The performance of the selected model is analyzed in Section 4 and the conclusions extracted from this work are outlined in Section 5.

2 Resource-Constraint Project Scheduling Problem

This work gravitates on the use of ACO algorithms to the Resource-Constraint Project Scheduling Problem (RCPSP) [2, 4]). The goal of this class of problems is to find an optimal schedule of the activities that compose a project subject to the availability and demand of different resources required to undertake these tasks. In mathematical terms, a project is composed by a set of activities $\mathcal{J} = \{0, \dots, n+1\}$, a set of resource types $\mathcal{Q} = \{1, \dots, q\}$ and a specific number of resources for each resource type $r_q \forall q \in \mathcal{Q}$. A project composed by n activities has always $n + 2$ activities in the set $\mathcal{J}$ because activity 0 and $n+1$ are dummy activities explicitly included to represent the start and end of the project and do not imply any duration nor need for resources.

Each activity can be executed in one or more different modes. If activities can be executed only in one mode, the problem is labeled as *Single-Mode*. Likewise, if activities can be executed in more than one mode, the problem is called *Multi-Mode*. The modes of a given activity represent different ways to execute this activity. For the same activity, modes differ in both the duration needed to complete the activity and the set of resources required for its accomplishment. Formally, the set of different modes of the activity j is denoted as $Mode_j$, the duration of activity j executed with mode m is denoted as d_{jm} and it requires r_{jmq} units of the resource $q \in \mathcal{Q}$. Moreover, s_j denotes the time when activity j started the execution, and f_j denotes the time when such an activity has finished. Note that $f_j = s_j + d_{jm}$ because the execution of any activity cannot be interrupted.

Each project may also contain *precedence constraints* that establish relations of time interdependence between the different activities that compose the project. If a given activity j has a precedence constraint with activity i, activity i cannot be executed until activity j has finished (i.e. $s_i \geq f_j$). By considering these constraints each activity can be assigned two lists, namely, $\mathcal{P}_j$ and $\mathcal{S}_j$, which contain its direct predecessors and successors. It is relevant to note that activity 0 is the only start activity and hence has no predecessors. Likewise, activity $n + 1$ is the only end activity and consequently, has no successors.

A solution for a RCPSP is schedule for the different activities that compose the project. This schedule is composed by the start time for all the activities that compose the project, $\mathbb{S} = \{s_x \mid \forall x \in \mathcal{J}\}$ and the different execution modes for the activities. For a given schedule, the start time is the initial time for activity 0 (s_0) and the finish time is the time for activity $n+1$ (f_{n+1}). The best solution is those with a minimum makespan [13], i.e. the difference between its finishing and starting times ($f_{n+1} - s_0$). A schedule will be declared *feasible* if it satisfies the following constraints:

- All the activities are scheduled, and each of them is executed once.
- Any activity must not be started before all its predecessors have finished. $s_i \geq f_j \mid \forall j \in \mathcal{P}_i, i \in \mathcal{J}$.
- At any time t, the sum of resources required for the activities in execution must not exceed the resource capacities of the project.

3 The Selected ACO Model for Constraint Satisfaction Problems

This section describes the model (first proposed in [6]) that allows a significant reduction in the size of the decision graph. The goal of this model is to create smaller graphs than the ones created in the literature. This section provides a detailed description of the different components that defines the model which are:

- A new decision graph, which is smaller than the ones created in the classical approaches.
- The new ants' behaviour: the reduction in the size of the graph yields a slightly more complex behaviour of the ants.
- The new heuristic called *Oblivion Rate* needed to control the number of pheromones created in the system.

The classical procedure to model any CSP as a graph is by creating as many nodes as pairs $< \textit{variable}, \textit{value} >$ available in the problem, and connecting those nodes whose variables are different. More formally, the resulting graph is defined as $G = (V, E)$ where:

$$\begin{aligned} V &= \{\langle X_i, v\rangle \,| X_i \in X \text{ and } v \in D(X_i)\} \\ E &= \left\{(\langle X_i, v\rangle , \langle X_j, w\rangle) \in V^2 | X_i \neq X_j\right\} \end{aligned} \tag{1}$$

where nodes V represent the pairs $< \textit{variable}, \textit{value} >$, and E represents edges connecting those nodes whose associated variables X are different from each other.

There are several pitfalls regarding this representation but the most important are related to the size of the resulting graph and the type of CSPs that can be represented. In this sense, if the problem has N variables and each of them can take M different values, the resulting graph will contain $N \cdot M$ nodes. As the graph is almost fully connected, the number of edges is $(N \cdot M) \cdot (M \cdot (N-1)) = N^2 \cdot M^2 - N \cdot M^2 \cong N^2 \cdot M^2$. This observation implies that problems composed by many variables or by variables that could take on a high number of different values would become really difficult to model and almost computationally prohibitive to handle due to the size of their underlying graph.

The CSP graph representation selected in this paper was initially proposed in [6]. This representation focuses on the reduction of the graph size resulting from the modeling of the CSP as a graph. In this approach, the size of the resulting graph is drastically reduced because each variable in the problem is represented only by one node, independently of the number of values that can be assigned to this variable (as it is traditionally represented in CSP solvers). Therefore, given any problem composed by N variables whose value can be drawn from a set of M different values, the resulting graph will have only N nodes, instead of $N \cdot M$

nodes created in classical graph models. This representation was applied to the N-Queens Problem, though it can be used in other CSP-like problems such as video games [5].

The restrictions of the problem are represented in the edges of the graph. Two nodes will be connected if there is at least one restriction that involves the variables represented by the nodes. For example, given the nodes $\mathcal{N}_1$ (that represent the variable x_1) and $\mathcal{N}_2$ (correspondingly, variable x_2) there will be an edge connecting both nodes if there is at least one constraint involving the values of x_1 with the values of x_2. Using this representation, the number of edges is drastically reduced due to the decrease of the number of nodes.

This simplification in the graph size entails a change in the behavior of the ants. In classical ACO approaches ants have to select the next node to visit, because the node itself contains the value assignment. Ants only deposit a small quantity of pheromone on the graph and repeat the process until they finish their execution. When adopting the new representation the ant behavior becomes more complex because ants are in charge of selecting a specific value for the variable encoded in the node (see Algorithm 1).

Algorithm 1: Ants' behavior needed in the selected graph representation.

```
1   $\mathcal{E}val\mathcal{V}al\mathcal{L}ist \leftarrow$ getEvaluatedValues($current\mathcal{N}ode$)
2   $\mathcal{P}her\mathcal{L}ist \leftarrow$ getPheromoneInformation()
3   $selected\mathcal{V}al \leftarrow$ selectValue($\mathcal{E}val\mathcal{V}al\mathcal{L}ist$,$\mathcal{P}her\mathcal{L}ist$)
4   updatePersonalAssignment($selected\mathcal{V}al$)
5   $\mathcal{D} \leftarrow$ getPossibleNodes($current\mathcal{N}ode$)
6   if ($\mathcal{D} \neq null$) then
7       $node \leftarrow$ selectNextNode($D$)
8       $current\mathcal{N}ode \leftarrow node$
9   else
10      resetAnt()
11  end
```

In Algorithm 1 ants evaluate the different values that can be assigned to the variable encoded in the corresponding node (Line 1). This evaluation is performed by using the heuristic function defined for the specific problem. Then the pheromone information deposited in the graph is used in Line 2. Once the pheromone and the heuristic values are obtained, ants select one value for the variable encoded in the node (Line 3). Every ant updates its personal assignments, i.e. its local solution, and compute the possible nodes to visit taking into account their local solution built so far. If there is at least one possible destination, the ant selects one of them to visit in the next time step. Otherwise, the ant finishes its execution and goes back to the nest updating, at the same time, the pheromone information that has deposited through the graph (Line 10).

Another consequence of the graph reduction is the increase of the number of pheromones deposited in the graph. Pheromones are placed in the edges of the graph because the validity of a specific value in a node depends on the given values to the rest of variables in the other nodes. Thereby, the edge connecting nodes i and j stores all pheromones related to the variables and values for these nodes. Depending on the complexity of the problem being solved, the number of pheromones stored in the graph might saturate the system. The total number of different pheromones in an edge is proportional to the size of the domains of the variables involved in the constraint represented by the edge. That is, if $|D(var_s)|$ denotes the different values that the source variable can take, and $|D(var_d)|$ represents different values for the destination variable, the edge connecting source and destination node could store, a maximum of $|D(var_s)| \cdot |D(var_d)|$ different pheromones.

In order to reduce the number of pheromones stored in the graph an *Oblivion Rate* heuristic is incorporated to the system. This heuristic removes a subset of pheromones from the network. It is important to note that this heuristic must be carefully designed, because it affects directly on the system performance. Consequently, the design of this heuristic depends on the problem being addressed.

In this work, the selected *Oblivion Rate* is a dynamic function that depends on the number of pheromones created in the system to compute the number of pheromones that will be removed. This heuristic applied at step t is defined as:

$$R(t) = 1 - \frac{1}{t^{S(t)}}, \tag{2}$$

where $S(t)$ represents the number of pheromones created in the graph at step t. Equation 3 defines this function, that depends on the number of pheromones created ($P(t)$) and the maximum number of pheromones that can be created ($MaxPher$), yielding

$$S(t) = \frac{P(t)}{MaxPher}. \tag{3}$$

In order to compute the maximum number of pheromones, Expression 4 provides an upper bound value using the classical graph-based representation previously described. This upper bound is computed by estimating the number of nodes and edges that the graph would contain by using the classical representation, i.e.

$$MaxPher(j, m) = j \cdot m \cdot (j - 1) \cdot m = j \cdot m^2 \cdot (j - 1). \tag{4}$$

4 Experimental Results

The main goal of the experimental results discussed in this section is to analyze the performance of the described ACO model when tackling RCPSP problems. Performance will be measured as the quality of the solutions found by the ACO model, as well as the number of pheromones stored in the system. The dataset used in this work has been extracted from the PSPLIB library [10] by selecting those problems where the number of execution modes are greater than 1, i.e. selecting the *Multi-Mode* problems (a description about the characteristics for the selected problems can be found in Table 1).

Problem	**#Instances**	**#Activities**	**#Modes**
`j10.mm`	536	10	
`j12.mm`	547	12	
`j14.mm`	551	14	
`j16.mm`	550	16	3
`j18.mm`	552	18	
`j20.mm`	554	20	
`j30.mm`	640	30	
`m2.mm`	481		2
`m4.mm`	555	16	4
`m5.mm`	558		5

Table 1. Description of the different problems available in the RCPSP dataset.

The configuration for the ACO algorithms carried out in this work is the same for all the experiments. The colony is composed by 100 ants that are executed during 100 steps. The evaporation rate is fixed to $\rho = 0.05$ whereas the values for α and β are 1 and 2 respectively.

The first experiment carried in this work analyzes the reduction in the number of pheromones created in the system. In order to do that, the different problems have been solved using the selected ACO model without Oblivion (Normal ACO) and using the *Dynamic Oblivion*. The number of pheromones created by both ACO models and the corresponding reduction percentage are shown in Table 2.

As it can be observed in this table, there is an important reduction in the number of pheromones of, at least, 94%. This is an important reduction because each pheromone is a structure stored in the memory of the system and it could be saturated. Nevertheless, pheromones are used to guide the colony to the optimal solutions. Thus, this reduction could affect to the quality of the solutions found by the ACO model. In order to measure whether this reduction affects to the quality of the solutions found, we have computed the average minimum

Problem	**Normal ACO**	**Dynamic Oblivion**	**Reduction pct.**
j10.mm	2023	68	96.63%
j12.mm	2695	83	96.92%
j14.mm	3246	101	96.88%
j16.mm	3841	120	96.87%
j18.mm	4315	142	96.71%
j20.mm	4865	168	96.54%
j30.mm	5922	276	95.33%
m2.mm	2092	112	94.64%
m4.mm	4891	268	**94.52 %**
m5.mm	5645	121	**97.85 %**

Table 2. This table shows the maximum number of pheromones created in the system using the Dynamic Oblivion Rate, and without the Oblivion Rate (Normal ACO).

makespan obtained by the different ACO algorithms and compared it against the average best makespan published by the research community.

Dataset	**MinPubl.**	**Normal ACO**	**Dynamic Oblivion**
m2.mm	30.16 ± 6.87	31.04 ± 7.42	31.04 ± 7.4
m4.mm	22.71 ± 7.3	26.69 ± 8.61	26.68 ± 8.51
m5.mm	21.16 ± 8.14	25.73 ± 9.4	25.76 ± 9.31
j10.mm	19.04 ± 6.21	19.69 ± 6.46	19.68 ± 6.46
j12.mm	21.34 ± 6.48	22.36 ± 6.63	22.25 ± 6.54
j14.mm	23.18 ± 6.14	25.23 ± 6.73	25.31 ± 6.79
j16.mm	24.93 ± 6.02	27.67 ± 7.03	27.76 ± 7.07
j18.mm	26.57 ± 6.47	29.99 ± 7.66	30.09 ± 7.54
j20.mm	27.71 ± 6.99	32.08 ± 8.62	32.08 ± 8.76
j30.mm	28.79 ± 7.44	36.16 ± 17.39	36.16 ± 17.44

Table 3. This table shows the average minimum makespan published by the research community, the average minimum makespan obtained by the selected model without using the Oblivion Rate (Normal) and using the Dynamic Oblivion Rate for the Multi-Mode problems belonging to PSPLib.

Table 3 shows the average minimum makespan published by the research community and the average minimum makespan obtained by the selected model: without using the Oblivion Rate (Normal) and using the Dynamic Oblivion Rate for the Multi-Mode problems of the PSPLib dataset. As it can be seen in this table, the average minimum makespan obtained by our approach is really similar when the model is not using the Oblivion Rate, and when the Dynamic Oblivion Rate is used. These results are really promising if we take into account the strong reduction in the number of pheromones created in the system when the Oblivion Rate is used. The utilization of the Dynamic Oblivion Rate reduces at least the 94% of the pheromones for Multi-Mode problems, building solutions really close to the ones obtained by the system without controlling the number of pheromones, and thus having more information about the past of the algorithm. Finally, all these makespan values obtained by our approach are really close to the optimal makespan obtained by the research community. For previous reasons, we can conclude that the Oblivion Rate heuristic and the selected ACO model are a good approach for solving RCPSP problems because the system obtains solutions close to the optimal, and the number of pheromones created in the system has been extremely reduced.

5 Concluding Remarks

Constraint Satisfaction Problems (CSP) belongs to this kind of traditional NP-hard problems with a high impact in both, research and industrial domains. There are several problems that can be modelled as a CSP such as planning, scheduling, travel and car routing problems, videogames or energy, among others.

However, due to the complexity that CSP problems exhibits, researchers are forced to use heuristic algorithms for solving the problems in a reasonable time. One of the most famous heuristic algorithms is Ant Colony Optimization (ACO) algorithm, but the classical utilization of ACO algorithms build a decision graph composed by the same number of nodes as pairs $< \textit{variable}, \textit{value} >$ available in the problem. Therefore the size of the resulting graph could be unmanageable depending on the number of variables and values of the selected problem.

In order to solve this problem, a new ACO model was proposed in [6]. This model is characterized by the utilization of a reduced decision graph and by the usage of a *Oblivion Rate* heuristic for controlling the number of pheromones created in the system.

This paper studies the applicability of the novel approach to solve Multi-Mode Resource Constraint Satisfaction Problem. For evaluating the ACO model, we have selected the Multi-Mode instances belonging to the PSPLib dataset. The experimental results reveals that the ACO model is able to remove, at least, the 94% of the pheromones of the system without affecting to the quality of the solutions built by the ACO algorithm. This result reveals that the ACO algorithm is a good approach for solving RCPSP problems because it is able to guide ants to optimal, or sub-optimal solutions, maintaining in the system those pheromones created by the best solutions.

Acknowledgements

This work has been supported by the next research projects: EphemeCH (TIN2014-56494-C4-4-P) Spanish Ministry of Economy and Competitivity, CIBERDINE S2013/ICE-3095, both under the European Regional De-velopment Fund FEDER, Airbus Defence & Space (FUAM-076914 and FUAM-076915), BID3ABI (Basque Government), and RiskTrack (JUST-2015-JCOO-AG-723180).

References

1. J. E. Bell and P. R. McMullen. Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, 18(1):41–48, 2004.
2. J. Blazewicz, J.K. Lenstra, and A.H.G.Rinnooy Kan. Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5(1):11 – 24, 1983.
3. A. E. Eiben and Z. S. Ruttkay. *Constraint Satisfaction Problems.* IOP Publishing Ltd. and Oxford University Press, 1997.
4. K. Z. Gao, P.N. Suganthan, Q.K. Pan, T.J. Chua, T.X. Cai, and C.S. Chong. Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. *Information Sciences*, 289:76 – 90, 2014.
5. A. Gonzalez-Pardo and D. Camacho. Environmental influence in bio-inspired game level solver algorithms. In *Proceedings of the 7th International Symposium on Intelligent Distributed Computing - IDC 2013*, volume 511 of *Studies in Computational Intelligence*, pages 157–162. Springer Berlin Heidelberg, 2013.
6. A. Gonzalez-Pardo and D. Camacho. A new csp graph-based representation for ant colony optimization. In *2013 IEEE Conference on Evolutionary Computation*, volume 1, pages 689–696, June 20-23 2013.
7. A. Gonzalez-Pardo and D. Camacho. A new csp graph-based representation to resource-constrained project scheduling problem. In *2014 IEEE Conference on Evolutionary Computation*, pages 344–351. IEEE Xplore 2014, 2014.
8. A. Gonzalez-Pardo, J. Del Ser, and D. Camacho. *On the Applicability of Ant Colony Optimization to Non-Intrusive Load Monitoring in Smart Grids*, pages 312–321. Springer International Publishing, Cham, 2015.
9. A. Gonzalez-Pardo, F. Palero, and D. Camacho. An empirical study on collective intelligence algorithms for video games problem-solving. *Computing and Informatics*, 34(1):233–253, 2015.
10. R. Kolisch and A. Sprecher. Psplib – a project scheduling problem library. *European Journal of Operational Research*, 96:205–216, 1996.
11. V. Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI Magazine*, 13(1):32–44, 1992.
12. S. Morin, C. Gagné, and M. Gravel. Ant colony optimization with a specialized pheromone trail for the car-sequencing problem. *European Journal of Operational Research*, 197:1185 – 1191, 2009.
13. A. Schirmer. Case-based reasoning and improved adaptive search for project scheduling. *Naval Research Logistics (NRL)*, 47(3):201–222, 2000.
14. E. P. K. Tsang. *Foundations of constraint satisfaction.* Computation in cognitive science. Academic Press, 1993.