Expert Systems with Applications 41 (2014) 6634–6645

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A Max–Min Ant System algorithm to solve the Software Project Scheduling Problem

Broderick Crawford [a,b], Ricardo Soto [a,d], Franklin Johnson [a,c,*], Eric Monfroy [e], Fernando Paredes [f]

[a] Pontificia Universidad Católica de Valparaíso, Avenida Brasil 2950, Valparaíso, Chile
[b] Universidad Finis Terrae, Av. Pedro de Valdivia 1509, Santiago, Chile
[c] Universidad de Playa Ancha, Av. Leopoldo Carvallo 270, Valparaíso, Chile
[d] Universidad Autónoma de Chile, Pedro de Valdivia 641, Santiago, Chile
[e] CNRS, LINA, University of Nantes, 2 rue de la Houssinière, Nantes, France
[f] Escuela de Ingeniería Industrial, Universidad Diego Portales, Manuel Rodríguez Sur 415, Santiago, Chile

ARTICLE INFO

Article history:
Available online 14 May 2014

Keywords:
Ant Colony Optimization
Project management
Software engineering
Software Project Scheduling Problem

ABSTRACT

The Software Project Scheduling Problem is a specific Project Scheduling Problem present in many industrial and academic areas. This problem consists in making the appropriate worker-task assignment in a software project so the cost and duration of the project are minimized. We present the design of a Max–Min Ant System algorithm using the Hyper-Cube framework to solve it. This framework improves the performance of the algorithm. We illustrate experimental results and compare with other techniques demonstrating the feasibility and robustness of the approach, while reaching competitive solutions.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

In project management, the manager has to initiate, schedule, execute, control, and close the project, this is a complex job for the project manager. This is the reason why we need to optimize these activities. The Software Project Scheduling Problem (SPSP) is an specific Project Scheduling Problem (PSP) (Chen, 2011; Laalaoui & Bouguila, 2014) which consists in making the appropriate worker-task assignment that minimizes cost and duration for the whole project, so the task precedence and resource constraints are satisfied (Alba & Chicano, 2007; Chang, yi Jiang, Di, Zhu, & Ge, 2008; Nan & Harter, 2009). The PSP consists in organizing multiple activities which can be assigned multiple resources, which can be renewable or non-renewable, so that meet the defined objectives. While in SPSP the main resource are the employees, who must to be assigned to project tasks according to their skills. Another difference is that PSP may have different single objectives, while in SPSP the main objective is to minimize the duration and cost of the project.

The SPSP is a variation of another NP-hard problem called Resource-Constrained Project Scheduling (RCPSP) is a very popular problem in the literature (Brucker, Drexl, Mhring, Neumann, & Pesch, 1999). RCPSP has many variations such as single-mode RCPSP, multi-mode RCPSP, RCPSP with non-regular objective function, stochastic RCPSP among others (Fahmy, Hassan, & Bassioni, 2014; Hartmann & Briskorn, 2010; Tavana, Abtahi, & Khalili-Damghani, 2014; Wu, Wan, Shukla, & Li, 2011). This has given rise to multiple jobs, which derives the SPSP.

Alba proposes the original modeling of the SPSP and its resolution by Genetic Algorithms (GA) (Alba & Chicano, 2007). Chang proposes a Time-line based model for SPSP using Genetic Algorithms (Chang et al., 2008). Later Xiao proposes a genetic algorithm and Ant Colony System (ACS) to solve SPSP (Xiao, Ao, & Tang, 2013), and Crawford proposes a new resolution by using Max–Min Ant Systems (MMAS) in Crawford, Soto, Johnson, and Monfroy (2013a). Luna makes a scalability analysis of multi-objective metaheuristics solving SPSP (Luna, González-Álvarez, Chicano, & Vega-Rodríguez, 2014). Other articles have been written about SPSP only as surveys.

SPSP is a problem where a set of employees must be assigned to a set of tasks, so all tasks are completed by employees who have the necessary skills to accomplish tasks. This assignment should minimize the cost and duration of the whole project. The employees have remuneration and several skills and they can work on multiple tasks during the workday. Most of the methods used for solving the problem belong to the class of priority-rules based methods or the class of metaheuristics.

ACO is a promising metaheuristic (Dorigo, Maniezzo, & Colorni, 1996), inspired from the behavior of real ant colonies. It is a probabilistic approach for solving computational problems which can

* Corresponding author at: Pontificia Universidad Católica de Valparaíso, Avenida Brasil 2950, Valparaíso, Chile. Tel.: +56 322273719.
E-mail address: franklin.johnson@upla.cl (F. Johnson).

http://dx.doi.org/10.1016/j.eswa.2014.05.003
0957-4174/© 2014 Elsevier Ltd. All rights reserved.

be reduced to finding good paths through graphs. This metaheuristic can solve several NP-hard combinatorial problems effectively (Christina & Miriam, 2012; Liao, Egbelu, Sarker, & Leu, 2011).

In this paper we present the model of SPSP, and the design of a Max–Min Ant System algorithm (Dorigo & Stützle, 2004; Stützle & Hoos, 2000) using the Hyper-Cube framework (HCF) (Blum & Dorigo, 2004) to solve the Software Project Scheduling Problem. MMAS is a specific ACO algorithm in which only the best ant is used to update the pheromone information.

We introduce the ACO Hyper-Cube (ACO-HC) algorithm, to solve the SPSP. This algorithm uses the HCF. HCF was originally proposed by Blum and Dorigo (Blum & Dorigo, 2004). This framework automatically handles the limits of pheromone values by modification in the update pheromone rule. This modification allows the algorithms to be more robust and easier to implement. The HCF can be applied to different ACO algorithms (Johnson, Crawford, & Palma, 2006).

We implement our proposed algorithm, and we conducted a series of tests to analyze the convergence to obtain better solutions. In addition we work in different tests to get the best parameterization. The tests were performed using different numbers of tasks, employees, and skills. The results were compared with other techniques such as Ant Colony System and Genetic Algorithms.

In this contribution we have built a solution for Software Project Scheduling Problem adapting the Max–Min Ant System metaheuristic and integrating it with the Hyper-Cube framework. That is, we have integrated two techniques for building competitive and robust solutions improving the solutions of other proposals.

This paper is organized as follows. In Section 2 we present a detailed definition of SPSP, in Section 3 is presented a description of ACO-HC. Section 4 presents the design of an ACO-HC for SPSP. In a subsection we presents the construction graph, pheromone update rules, the heuristic information, and the algorithm. Section 5 presents the experimental results. The conclusions are outlined in Section 6.

2. Description of the Software Project Scheduling Problem

The Software Project Scheduling Problem is one of the most common problems in managing software engineering projects (OZDAMAR & ULUSOY, 1995). It consists in finding a worker-task schedule for a software project. SPSP should consider remunerations and employee skills which must be assigned to project tasks according to the requirements of these tasks (Alba & Chicano, 2007; Barreto, Barros, & Werner, 2008; Xiao et al., 2013). The most important resources involved in SPSP are: the tasks, which are the job needed to complete the project, the employees who work in the task, and finally the skills. The employees have multiples skills, and the tasks required a set of skills. It should make a careful allocation according to the skills needed for the tasks and the skills of employees.

2.1. Description of skills

As mentioned above, the skills are the abilities required for completing the tasks, and the employees have all or some of these abilities. These skills can be for example, design expertise, programing expert, leadership, GUI expert. The set of all skills associated with software project is defined as $S = \{s_1, \ldots, s_{|S|}\}$, where $|S|$ is the number of skills.

2.2. Description of tasks

The tasks are all necessary activities for accomplishing the software project. These activities are for example, analysis, component design, programing, documentation, testing. The software project is a sequence of tasks with different precedence among them. Generally, we can use a graph called task-precedence-graph (TPG) to represent the precedence of these tasks. This is a non-cyclic directed graph denoted as $G(V, E)$. The set of tasks is represented by $V = \{t_1, t_2, \ldots, t_{|T|}\}$. The precedence relation of tasks is represented by a set of edges E. An edge $(t_i, t_j) \in E$, means t_i is a direct predecessor task t_j. Consequently, the set of tasks necessary for the project is defined as $T = \{t_1, \ldots, t_{|T|}\}$, where $|T|$ is the maximum number of tasks. Each task have two attributes:

- t_j^{sk} is a set of skills for the task j. It is a subset of S and corresponds to all necessary skills to complete a task j.
- t_j^{eff} is a real number and represents the workload of the task j.

2.3. Description of employees

The most relevant resource in this problem is the employees. The employees have multiple skills, commonly the employees are software engineers, and their skills are software engineering skills. The project has a set of employees and they work on the tasks. The project manager needs to assign the employees to the appropriate tasks. The problem is to create a worker-task schedule where employees are assigned to suitable tasks. The set of employees is defined as $EMP = \{e_1, \ldots, e_{|E|}\}$, where $|E|$ is the number of employees working on the project. Each employee has tree attributes:

- e_i^{sk} is a set of skills of employee i. $e_i^{sk} \subseteq S$.
- e_i^{maxd} is the maximum degree of work. it is the ratio between hours for the project and the workday. $e_i^{maxd} \in [0, 1]$, if $e_i^{maxd} = 1$ the employee has total dedication to the project, if the employee has e_i^{max} less than one, in this case is a part-time job.
- e_i^{rem} is a real number. It is the monthly remuneration of employee i.

2.4. Model description

The model for SPSP use tasks, employees, and skills previously described. We describe the elements of the model in Table 1.

The SPSP solution can be represented as a matrix $M = [E \times T]$. The size $|E| \times |T|$ is the dimension of matrix determined by the number of employees and the number of tasks. The elements of the matrix $m_{ij} \in [0, 1]$, corresponds to real numbers, which represent the degree of dedication of employee i to task j. If $m_{ij} = 0$, the employee i is not assigned to task j, that means the employee does not dedicate time for this task. If $m_{ij} = 1$, the employee i works all day in the task j. For example, if $m_{ij} = 0.5$, the employee i uses 50 percent of his time on the project.

The solutions generated in this matrix not always are feasible. That happens when all elements of a column i are 0, that means the employees are not assigned to the task i. This solution is not feasible because the task i is not finished. For this reason we define some constraints to obtain a feasible solution from the matrix M;

- First, all tasks are assigned at least one employee as is presented in Eq. (1).

$$\sum_{i=1}^{|E|} m_{ij} > 0 \quad \forall j \in \{1, \ldots, T\} \tag{1}$$

- Second, the employees assigned to the task j have all the necessary skills to carry out the task. This is presented in Eq. (2) follows by the skills needed for the task t_j are a subset of the union of the skills the employees assigned to the task.

Download English Version:

https://daneshyari.com/en/article/382800

Download Persian Version:

https://daneshyari.com/article/382800

Daneshyari.com