Neurocomputing 148 (2015) 122–128

Contents lists available at ScienceDirect

Neurocomputing

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

Hybrid particle swarm optimization for preemptive resource-constrained project scheduling

Yongyi Shou *, Ying Li, Changtao Lai

School of Management, Zhejiang University, Hangzhou 310058, PR China

ARTICLE INFO

Article history:
Received 7 April 2012
Received in revised form
21 July 2012
Accepted 25 July 2012
Available online 28 July 2014

Keywords:
Project scheduling
Preemption
Particle swarm optimization
Peak crossover

ABSTRACT

In this paper a hybrid particle swarm optimization procedure is proposed to solve the preemptive resource-constrained project scheduling problem in which a maximum of one interruption per activity is allowed. Four types of particle representations are designed and two schedule generation schemes are adopted to decode the particle representations. Particle-updating mechanisms based on the peak crossover operator are designed for all particle representations. Computational experiments have been carried out on standard project scheduling problem sets. Analysis of the computational results has confirmed that introduction of preemption helps to reduce project duration and the proposed particle swarm optimization procedures are effective for preemptive resource-constrained project scheduling.

1. Introduction

The resource-constrained project scheduling problem (RCPSP), together with some of its extensions, has been widely studied during the last decades. The classic RCPSP is to schedule a project so as to minimize its duration [1], where n activities have to be processed in order subject to both precedence relations and resource constraints [2]. The RCPSP has been proven to be an NP-hard problem [3]. A variety of optimization procedures have been developed to solve the problem. Exact methods have been proposed, such as branch and bound algorithms [4,5] and zero-one programming approaches [6,7]. However, exact methods are not so capable of large-scale projects with 60 or more activities [1]. The proposition and development of heuristic and meta-heuristic algorithms, especially genetic algorithms (GAs) [1], ant colony optimization (ACO) algorithms [8] and particle swarm optimization (PSO) algorithms [9], improve the solution quality and computing speed significantly. Of obvious practical importance, the RCPSP and corresponding procedures are widely studied and applied [10].

The classic RCPSP is based on some assumptions, among which non-preemption requires that an activity cannot be interrupted once it has started [11]. However, interruption happens frequently in real-world projects. As a result, the majority of project management software packages also allow preemption. With the demands of project management practice, more researchers begin to relax the non-preemption assumption. The RCPSP without non-preemption assumption is called the preemptive resource-constrained project scheduling problem (PRCPSP).

Kaplan [12] is the first to study the PRCPSP systematically. In the PRCPSP, activities are assumed to be interrupted at an integer time instant and restarted later with no additional cost. Demeulemeester and Herroelen [13] had an in-depth discussion about the PRCPSP, and proposed a branch-and-bound procedure to solve the problem. According to their research it was argued that given a constant resource availability the introduction of preemption in project scheduling has little effect on project duration.

Sorli and Pilar [14] categorized three modes of preemption, viz., (a) non-preemption, (b) a maximum of one preemption per activity (at an integer time instant), and (c) any number of preemptions (at integer time instants). The problem defined by Demeulemeester and Herroelen [13] is equivalent to the mode (c) of Sorli and Pilar [14], and is the most generalized extension of the RCPSP in regards to preemption. Ballestín et al. [15] studied the PRCPSP and suggested to use the term m_PRCPSP, where m is a non-negative integer. The m_PRCPSP can be defined as an RCPSP in which any activity can be preempted at most m times and each preemption happens at a non-negative integer time instant. For a specific activity, m shall not exceed its duration. Ballestin et al. [15] argued that 1_PRCPSP, i.e., m_PRCPSP with $m=1$, has practical and theoretical significance. Their computational experiments indicated that preemption helps to decrease project duration.

Similar to the RCPSP, many methods for the PRCPSP have also been proposed in the literature, including exact methods, heuristics and meta-heuristics. Exact methods (e.g., [12,13,16,17]) are capable to find the optimal solutions. However, for large-scale project scheduling problems or other more complex problems, exact methods are not efficient enough. Since large-scale projects are very common in

* Corresponding author. Tel.: +86 571 88206548; fax: +86 571 88206812.
E-mail address: yshou@zju.edu.cn (Y. Shou).

http://dx.doi.org/10.1016/j.neucom.2012.07.059
0925-2312/

practice, researchers started to pay more attention to heuristic or meta-heuristic algorithms for the PRCPSP to improve the solution efficiency. For example, Ballestín et al. [15] adapted the serial schedule generation scheme (SGS) to transform an activity list to a feasible schedule with one interruption and named this scheme as 1_serial SGS or 1_SSGS. A genetic algorithm using 1_SSGS as the decoding scheme was then designed for the 1_PRCPSP [15]. Particle swarm optimization was also applied to solve the PRCPSP [18], in which a particle-updating mechanism was designed based on the partially mapped crossover operator. The computational experiment on an example project demonstrated that the PSO algorithm is effective for preemptive project scheduling.

The objective of this paper is to study the 1_PRCPSP and to extend the PSO algorithm mentioned in [15]. To adhere to the basic structure of the PSO algorithm, we fully summarize four kinds of particle solution presentations and adopt the peak crossover operator [19] to improve the particle-updating mechanism [18].

The rest of the paper is as follows. In Section 2, we describe the 1_PRCPSP in detail. Section 3 introduces the PSO procedures we have developed for the problem, including the particle solution representations, schedule generation schemes, the fitness function and particle-updating mechanisms. Computational experiments were carried out and the results are shown in Section 4. Section 5 discusses the impacts of project parameters. Finally, Section 6 briefly summarizes the findings of this paper.

2. Problem descriptions

A project can be represented by an activity-on-node (AON) network $G=(V, E)$, where V is the set of nodes representing the activities, and E is the set of arcs denoting the precedence relations. The project consists of a set of $n+2$ activities $j=0, 1, \ldots, n, n+1$, where activities 0 and $n+1$ represent dummy start and end activities respectively. The duration of activity j is denoted by a non-negative integer d_j, and $d_0=d_{n+1}=0$. A set of K renewable resources are available for the project. The capacity of resource k is denoted as R_k, where $k=1, 2,\ldots, K$. The demand on resource k by activity j is r_{jk}, where $j=0, 1,\ldots, n, n+1, k=1, 2, \ldots, K$, and r_{jk} is a non-negative integer.

Activities in progress are allowed to be interrupted for a maximum of once, and preemption occurs at an integer time instant. If preempted, the activity is to start again after a period of time with no additional cost. If preemption occurs exactly at the start or end of an activity, or if the activity immediately restarts after preemption, then in either situation the activity is not recognized as having been preempted.

For the preemptive project scheduling problem, it is required to determine whether an activity shall be preempted, when the preemption shall occur, and when the remainder part of the preempted activity shall be restarted, so as to minimize the project duration under the precedence relations and resource constraints.

To formulate the 1_PRCPSP, some extra symbols and definitions are given as follows: s_j denotes the start time of activity j; P_j denotes the set of immediate predecessors of activity j; t denotes the time instant and T denotes the due date of the project. Activity j can be split into two parts j_a and j_b, and their start times are $s_{j,a}$ and $s_{j,b}$, respectively. Their durations are non-negative integer numbers $d_{j,a}$ and $d_{j,b}$, where $d_{j,a}+d_{j,b}=d_j$. Moreover, the set of ongoing activity at time instant t is denoted by A_t. Hence, the 1_PRCPSP is formulated as follows [13,20]:

$$\min s_{n+1} \quad (1)$$

s.t.

$$s_{ib}+d_{ib} \le s_{ja}, \quad i \in P_j, \ j \in V \quad (2)$$

$$s_{ja}+d_{ja} \le s_{jb}, \quad j \in V \quad (3)$$

$$d_{ja}+d_{jb} = d_j, \quad j \in V/\{0, n+1\} \quad (4)$$

$$s_0 = 0 \quad (5)$$

$$\sum_{j \in A_t} r_{jk} \le R_k \quad \forall k, \quad \forall t \in [0, T] \quad (6)$$

$$d_{ja}, d_{jb} \in Z/Z^- \quad (7)$$

The objective function (1) is to minimize the start time of dummy end activity $n+1$, which is equal to the project duration. Constraints (2) and (3) ensure that there is no violation of precedence relations. Constraint (4) ensures that the sum of the durations of the two parts is equal to the duration of the whole activity. Constraint (5) is the initial condition. Constraint (6) enforces resource requirements. Constraint (7) indicates that the preemption of an activity occurs at a non-negative integer time instant.

3. Particle swarm optimization

Particle swarm optimization is one of the well-recognized swarm intelligence algorithms [19,21,22]. Similar to the genetic algorithm and ant colony optimization algorithm, PSO starts with an initial population (i.e., a swarm) of individuals (i.e., particles) and updates them iteratively. Each particle is represented as a point in an M-dimension space and characterized by its position and velocity. The particles are updated according to its own flying experience and its companions' flying experience [21,23].

In this paper, a hybrid particle swarm optimization procedure is proposed to solve the 1_PRCPSP. Four forms of encoding methods are introduced, viz., permutation-based representation, priority-based representation, permutation and preemption point-based representation, and priority and preemption point-based representation. The first two are decoded by the 1_SSGS scheme in [15]. And for the latter two, we propose the serial SGS with preemption points (PP_SSGS) which will be further discussed in the second sub-section.

3.1. Particle solution representation

In general, two forms of solution representations, i.e., permutation-based (or activity list) representation [2,24] and priority-based (or random key) representation [1,19,25,26] are adopted widely for the classic RCPSP. In this paper, we design another two forms to represent activities preempted in the 1_PRCPSP, i.e., permutation and preemption point-based representation, and priority and preemption point-based representation. For each representation, we take a project with five non-dummy activities (see Fig. 1) to illustrate their corresponding structures.

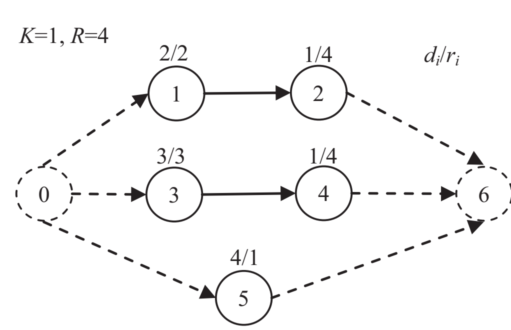

Fig. 1. A project with five non-dummy activities.