Transaction E: Industrial Engineering Vol. 17, No. 1, pp. 25-35 © Sharif University of Technology, June 2010



## Resource-Constrained Project Scheduling Problem with Flexible Work Profiles: A Genetic Algorithm Approach

M. Ranjbar<sup>1,\*</sup> and F. Kianfar<sup>2</sup>

**Abstract.** This paper deals with the resource-constrained project scheduling problem with flexible work profiles. In this problem, a project contains activities interrelated by finish-start-type precedence constraints with a time lag of zero. In many real-life projects, however, it often occurs that only one renewable bottleneck resource is available and that activities do not have a fixed prespecified duration and associated resource requirement, but a total work content, which essentially indicates how much work has to be performed on them. Based on this work content, all feasible work profiles have to be specified for the activities, each characterized by a fixed duration and a resource requirement profile. The task of the problem is to find the optimum work profile and start time of each activity in order to minimize the project makespan. We propose a procedure to find all feasible work profiles of each activity and we use a genetic algorithm with a new crossover operator to schedule the activities. Computational results on a randomly generated problem set are presented.

Keywords: Project scheduling; Heuristic; Genetic algorithm.

## INTRODUCTION

This research considers the Resource Constrained Project Scheduling Problem with Flexible Work Profiles (RCPSP\_FWP). This problem was proposed by Kolisch [1] in the field of pharmaceutical R&D projects and deals with the lead optimization phase of pharmaceutical research where a number of leads (molecules as a basis for potential drugs) are processed by different departments in order to optimize their biochemical characteristics. The RCPSP\_FWP can be stated as follows. A single project consists of a set of activities that are interrelated by finish-start-type precedence relations with a time lag of zero. There is a single constrained renewable resource (laboratory) in which all activities have to be processed. For each activity, instead of a fixed duration and resource requirement, the total work content is given, which essentially indicates how much work has to be performed. In other words, activity duration and resource usage at each time period of their execution are unknown. The way each activity is processed, i.e. its work profile, is not predetermined but limited by the following five constraints:

- a) No preemption is allowed.
- b) The resource usage of each activity in a processing period has to be within a resource-dependent interval.
- c) The summation over used resource units for all periods of execution of each activity should be equal to the work content of the activity.
- d) During processing, the resource usage of each activity has to be constant for a resource-dependent period.

Each work profile for an activity is considered as an activity mode and all feasible work profiles for an activity constitute its set of modes.

The constraints (a)-(d) assure that the processing of activities is done in an efficient way by reducing the number of setups due to constraints (a) and (d) and by forbidding the case of too few or to many resources assigned to an activity within one period

<sup>1.</sup> Department of Industrial Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, P.O. Box 91775-1111, Iran.

<sup>2.</sup> Department of Industrial Engineering, Sharif University of Technology, Tehran, P.O. Box 11155-9414, Iran.

<sup>\*.</sup> Corresponding author. E-mail:  $m_ranjbar@ferdowsi.um.ac.ir$ 

Received 8 June 2008; received in revised form 24 January 2009; accepted 8 June 2009

26

by constraint (b). Since the number of feasible work profiles of each activity in the general form and on the basis of conditions (a)-(d) may be very large, we limit ourselves to only smooth (SM), Non-Increasing (NI), Non-Decreasing (ND) or triangular (TR) types of figure for work profiles. Figures 1 to 4 illustrate some possible work profiles of an activity with work content of 15.

The RCPSP\_FWP is a different version of the well-known Resource Constrained Project Scheduling Problem (RCPSP) in which a single renewable resource is available and activity duration and resource usage to a single renewable resource are known constants. Also, the RCPSP\_FWP is a generalization of the discrete time/resource trade-offs (DTRTP), introduced by De Reyck et al. [2]. In the DTRTP, only smooth work



Figure 1. Smooth work profile.



Figure 2. Non-decreasing work profile.



Figure 3. Non-increasing work profile.



Figure 4. Triangular work profile.

profiles are considered for each activity, and constraints (b), (c) and (d) are ignored.

To the best of our knowledge, literature on the RCPSP\_FWP, as defined in this paper, is virtually void. Research efforts have been concentrated on two related problems, RCPSP and DTRTP. Salewski et al. [3], Brucker et al. [4], Kolisch and Padman [5] and Kis [6] include some noteworthy literature survey research for the RCPSP. Also, numerous exact and heuristic procedures are presented in the literature for the RCPSP. Chapter 6 of the project scheduling research handbook of Demeulemeester and Herroelen [7] gives an extensive literature overview of the RCPSP and, hence it is not repeated here. In addition, Kolisch and Hartmann [8] present a classification and performance evaluation of different heuristic and metaheuristic algorithms for the RCPSP. De Reyck et al. [2] present several heuristic procedures for the DTRTP based on the Tabu Search (TS) and some local searches. These heuristic procedures are based on the decomposition of the problem into a mode assignment phase and a resource-constrained project scheduling phase with fixed mode assignments. Demeulemeester et al. [9] have developed a Branchand-Bound algorithm (B&B) for the DTRTP based on the concept of activity-mode combinations, i.e. subsets of activities executed in a specific mode. Also, Ranjbar et al. [10] present an efficient hybrid metaheuristic, based on the scatter search for the DTRTP and RCPSP.

The NP-hardness of the RCPSP was shown by Blazewicz et al. [11] as a generalization of the job shop scheduling problem. Furthermore, Demeulemeester et al. [9] show that the DTRTP, as a generalization of the parallel machine problem, is strongly NP-hard. Therefore, the RCPSP-FWP as a generalization of the DTRTP is strongly NP-hard.

Since the RCPSP\_FWP is a very complex problem, we use a Genetic Algorithm (GA) as our solution approach. Two famous GAs applied to project scheduling problems are Mori and Tseng [12] and Hartmann [13].

The contribution of this paper is twofold. First, we present a linear model for the RCPSP\_FWP and also an enumeration scheme to generate feasible work profiles. Second, we propose a metaheuristic solution procedure based on GA to solve RCPSP\_FWP.

The outline of the paper is as follows. First, the problem formulation and notation are described and generation of feasible work profiles of a single activity is discussed. After that, schedule representation and the generation scheme are explained and the general structure of the genetic algorithms and its operators are described. In the section following these computational results are reported. Finally, overall conclusions and suggestions for future research are drawn.

# PROBLEM FORMULATION AND NOTATION

The objective of the RCPSP\_FWP is to schedule each activity of a project in one of its defined modes subject to precedence and resource constraints, in order to minimize the project makespan. The parameters used are defined in Table 1.

The dummy activities, 0 and n + 1, represent the start and completion of the project, respectively, with  $w_o = w_{n+1} = 0$ . The RCPSP\_FWP can be formulated by introducing the following decision variables:

$$s_{it} = \begin{cases} 1; & \text{if activity } i \text{ is started at time instant } t \\ 0; & \text{otherwise} \end{cases}$$

$$f_{it} = \begin{cases} 1; & \text{if activity } i \text{ is finished at time instant } t \\ 0; & \text{otherwise} \end{cases}$$

$$h_{it} = \begin{cases} 1; & \text{if } r_{i(t-1)} \neq r_{i(t)} \\ 0; & \text{otherwise} \end{cases}$$

In the following, we represent an MIP formulation for the RCPSP\_FWP.

$$\min Z = \sum_{t=ef_{n+1}}^{lf_{n+1}} tf_{(n+1)t},\tag{1}$$

subject to:

$$\sum_{t=es_i}^{ls_i} s_{it} = 1; \qquad i \in N,$$
(2)

$$\sum_{t=ef_i}^{lf_i} f_{it} = 1; \qquad i \in N,$$
(3)

$N = \{0, \cdots, n, n+1\}$	Set of activities with index $i$		
$E = \{(i, j); i, j \in N\}$	Set of precedence relations		
P <sub>i</sub>	Set of all predecessors of activity $i; i \in N$		
$S_i$	Set of all successors of activity $i; i \in N$		
a	Constant availability of the single renewable resource		
$w_i$	Work content of activity $i; i \in N$		
$\underline{r}_i, \overline{r}_i$	Lower and upper bounds of resource usage of activity $i$ in each		
	time period; $i \in N$		
<u>t</u>	Minimum number of consecutive periods where all activities		
	should have a constant resource usage		
	Duration of activity $i; i \in N$		
$\underline{d}_i = \begin{bmatrix} \underline{w}_i \\ \overline{\tau}_i \end{bmatrix}$	Lower bound for duration of activity $i; i \in N$		
$\bar{d_i} = \left\lfloor \frac{w_i}{x_i} \right\rfloor$	Upper bound for duration of activity $i; i \in N$		
$d_i^{\min}$	Minimum possible duration for a feasible work profile of		
	activity $i; i \in N$		
$d'_{i}^{\min}$	Minimum possible duration for an infeasible work profile of		
	activity i with minimum violation; $i \in N$		
$T = \sum_{i=1}^{n} \overline{d}_i$	An upper bound on the project makespan		
$es_{i} = \max\left\{ \left( \frac{\sum\limits_{j \in P_{i}} w_{j}}{a} \right), \left( \max_{j \in P_{i}} \left\{ es_{j} + d_{j}^{\min} \right\} \right) \right\}$	Earliest start time of activity $i; i \in N$		
$ef_i = es_i + \underline{d}_i$	Earliest finish time of activity $i; i \in N$		
$lf_i = \min\left\{ \left( T - \frac{\sum\limits_{j \in S_i} w_j}{a} \right), \left( T - \min_{j \in S_i} \{ ls_j \} \right) \right\}$	Latest finish time of activity $i; i \in N$		
$ls_i = lf_i - \underline{d}_i$	Latest start time of activity $i; i \in N$		
(t) = (t - 1, t]	Time period $(t); t = 1, 2, \cdots, T$		
$r_{i(t)}$	Resource usage of activity $i$ at time period $t; i \in N$ ,		
	$t = es_i + 1, \cdots, lf_i$		
$wp_i = (r_{i(t)}); t = es_i + 1, \cdots, lf_i$	Work profile of activity $i; i \in N$		

27

28

$$\sum_{t=es_{i}+1}^{lf_{i}} r_{i(t)} = w_{i}; \qquad i \in N,$$
(4)

$$\sum_{t=ef_i}^{lf_i} tf_{it} \le \sum_{t=es_j}^{ls_j} ts_{jt}; \qquad \forall (i,j) \in E,$$
(5)

$$\sum_{i=1}^{n} r_{i(t)} \le a; \qquad t = 1, \cdots, T,$$
(6)

$$r_{i(0)} = 0; \qquad i \in N,\tag{7}$$

$$\sum_{\tau=t}^{t+\underline{t}-1} h_{i\tau} \le 1; \qquad i \in N, t = es_i + 1, \cdots, lf_i, \qquad (8)$$

$$r_{i(t+1)} \le \bar{r}_i(r_{i(t)} + s_{it}); \quad i \in N, t = 1, \cdots, T, \quad (9)$$

- $r_{i(t)} \leq \bar{r}_i(r_{i(t+1)} + f_{it}); \qquad i \in N, t = 1, \cdots, T,$ (10)
- $r_{i(t+1)} r_{i(t)} \le \bar{r}_i \cdot h_{i,t+1}; \quad i \in N, t = 1, \cdots, T,$ (11)
- $r_{i(t)} r_{i(t+1)} \le \bar{r}_i \cdot h_{i,t+1}; \quad i \in N, t = 1, \cdots, T,$ (12)

$$s_{it}, f_{it}, h_{it} \in \{0, 1\}; \qquad i \in N, t = 1, \cdots, T,$$
 (13)

 $r_{i(t)} \in \{0\} \cup \{\underline{r}_i, \underline{r}_i + 1, \cdots, \overline{r}_i\};$ 

$$i \in N, t = 1, \cdots, T. \tag{14}$$

Objective Function 1 minimizes the project makespan. Constraints 2 and 3 ensure that exactly one start and finish time are assigned to each activity, respectively. Constraint 4 assures that used resource units by each activity are equal to the activity work content. Constraints 5 and 6 indicate the precedence and resource constraints, respectively. Constraint 7 indicates that the resource is available after time instant zero. Constraint 8 corresponds to the constraints (d) of the work profile constraints. Also, Constraints 9-12 ensure that every change in the level of resource usage for each activity is less than its resource usage upper bound. Finally, Constraints 13 and 14 show the value domains of the decision variables.

## ASSIGNMENT OF FLEXIBLE WORK PROFILES

In this section, we consider the problem of generating feasible work profiles of a single activity within the RCPSP\_FWP. We first propose a model that generates the work profile with minimum durations. Next, we present an enumeration procedure for generating the NI, ND and SM work profiles. We further discuss the generation of TR work profiles based on NI and ND work profiles.

## Generation of Work Profiles with Minimum Duration

Based on the defined notations in Table 1 and the definition of the two following decision variables, we present a model for finding a feasible work profile with minimum duration for each single activity,  $i \in N$ .

$$\begin{aligned} x_{ri} &= \text{ number of periods with a resource usage of} \\ r \text{ units during execution of activity} \\ i \in N(r = \underline{r}_i, \underline{r}_i + 1, \cdots, \overline{r}_i), \end{aligned}$$

$$y_{ri} = \begin{cases} 1; & \text{if there is at least one period with} \\ & \text{resource usage of } r \text{ units during} \\ & \text{execution of activity } i \\ 0; & \text{otherwise} \end{cases}$$

$$d_i^{\min} = \min \sum_{r=\underline{r}_i}^{\bar{r}_i} x_{ri},\tag{15}$$

subject to:

$$\sum_{r=\underline{r}_i}^{\bar{r}_i} r x_{ri} = w_i,\tag{16}$$

$$\underline{t}y_{ri} \le x_{ri} \le \bar{d}_i y_{ri},\tag{17}$$

$$x_{ri} \in \{0, 1, \cdots\}, \qquad r \in [\underline{r}_i, \overline{r}_i],$$
(18)

$$y_{ri} \in \{0, 1\}, \qquad r \in [\underline{r}_i, \bar{r}_i].$$
 (19)

Objective Function 15 minimizes the number of periods where the activity has a positive resource usage and, thus the duration of the activity. Constraint 16 takes care that the total used resource units equals the activity work content. Constraint 17 couple  $x_{ri}$ variable with the  $y_{ri}$ -variable; if  $y_{ri} = 0$ , then  $x_{ri}$  is set to zero, otherwise, if  $y_{ri} = 1$ ,  $\underline{t} \leq x_{ri} \leq \overline{d_i}$ . Finally, Constraints 18 and 19 show the value domains of the decision variables. We call Constraints 16 to 19 the Feasible Work Profile Space (FWPS<sub>i</sub>) for activity *i*. If FWPS<sub>i</sub> =  $\emptyset$ , there is not any feasible work profile for activity *i*. In this case, we consider the work profiles with minimum violation.

## Generation of Work Profiles with Minimum Violation

In the case there is no feasible work profile for an activity, we consider only one work profile for the activity which has a minimum violation from work profile constraints. To this purpose, we alter the previous model in order to find an infeasible work

profile that minimizes the number of periods that are short of the minimum number of periods with constant duration  $\underline{t}$ . We define decision variable  $v_{ri} \geq 0$  as the number of periods with resource usage r short of the minimum number of periods regarding to activity i. Note that  $v_{ri}$  is formally defined as  $v_{ri} = \max(0, \underline{t} - x_{ri})$ and the new model is as follows:

$${d'}_i^{\min} = \min \sum_{r=\underline{r}_i}^{\bar{r}_i} v_{ri},\tag{20}$$

subject to:

$$\sum_{r=\underline{r}_i}^{\bar{r}_i} r x_{ri} = w_i,\tag{21}$$

$$\underline{t}y_{ri} - v_{ri} \le x_{ri} \le \overline{d}_i y_{ri}, \qquad r \in [\underline{r}_i, \overline{r}_i], \tag{22}$$

$$x_{ri} \in \{0, 1, \cdots\}, \qquad r \in [\underline{r}_i, \overline{r}_i], \tag{23}$$

$$y_{ri} \in \{0, 1\}, \qquad r \in [\underline{r}_i, \overline{r}_i], \tag{24}$$

$$v_{ri} \ge 0, \qquad r \in [\underline{r}_i, \bar{r}_i]. \tag{25}$$

Objective Function 20 minimizes the number of periods that are short of the minimum number of periods with a constant resource usage. Altered Constraint 22 sets the variable,  $v_{ri}$ . Other constraints are taken from the previous model.

### An Enumeration Procedure for the ND Work Profiles

In order to enumerate all ND work profiles of an activity, i, we use an enumeration tree in which level  $l(l = 1, \dots, \bar{r}_i - \underline{r}_i + 1)$  corresponds to all possible values of  $x_{ri}$ , defined as the number of periods with a resource usage of  $r = l - 1 + \underline{r}_i$  units. In each node of level  $l, \hat{w}_i$ indicates the remaining work content of activity i, i.e. the work content of activity i minus the total assigned resource units in nodes located on the path from the start node (l = 0) to the current node. At each level, l > 0, there are  $\overline{t} - t + 1$  possible descendants for a parent node, located at level l-1 where  $\overline{t} = |\hat{w}_i/r|$ . These descendants are generated by assigning different values to  $x_{ri}$  from set  $\{0, \underline{t}, \underline{t}+1, \cdots, \overline{t}\}$ . The remaining work content is updated as  $\hat{w}_i := \hat{w}_i - rx_{ri}$  after assignment. We define a block in a work profile as a number of consecutive periods with equal levels of resource usage and  $\mathbf{x}_i = (x_{\underline{r}_i}, x_{\underline{r}_i+1}, \cdots, x_{\overline{r}_i})$  as the block representation of the work profile of activity iin which  $x_r$  represents the length of the block with a resource usage of r units where  $r \in [\underline{r}_i, \overline{r}_i]$ . In each node of the enumeration tree wherein we face one of the following conditions, we fathom the node.

- 1. Finding a feasible work profile, i.e.  $\hat{w}_i = 0$ . In this case, the feasible work profile is obtained by following the path between the start node and the current node.
- 2. Finding a positive remaining work content that is not enough for the extension of more blocks, i.e. in a level,  $l = r + 1 \underline{r}_i$ ,  $0 < \hat{w}_i < \underline{t}(r+1)$ .
- 3. Reaching an infeasible work profile. If in each node of level  $l = r + 1 \underline{r}_i$  we replace  $\hat{w}_i$  with  $w_i$  and  $\underline{r}_i$  with r in the model for FWPS<sub>i</sub>, and we find FWPS<sub>i</sub> =  $\emptyset$ , then all descendants of this node will be infeasible.
- 4. Reaching the last level of the enumeration tree, i.e.  $l = \bar{r}_i \underline{r}_i + 1.$

To generate all NI work profiles, we correspond each NI work profile  $\mathbf{x}_i = (x_{\underline{r}_i}, x_{\underline{r}_i+1}, \cdots, x_{\overline{r}_i})$  to the ND work profile  $\mathbf{x} = (x_{\overline{r}_i}, x_{\overline{r}_i-1}, x_{\underline{r}_i})$ .

## Generation and Enumeration of the TR Work Profiles

All feasible TR work profiles of each activity can be generated from its ND or NI work profiles. It should be noted that ND and NI work profiles are two special types of TR work profile. Also, SM work profiles are an especial type of ND and NI work profiles. Thus, we can conclude  $SM \subseteq NI \subseteq TR$  and  $SM \subseteq ND \subseteq TR$ . Each ND work profile has a unique corresponding NI work profile and possibly some other TR work profiles constituting a set called a *neighborhood*. To generate all feasible TR neighbors of a feasible ND work profile, we do as follows. Assume the feasible ND work profile includes  $\overline{b}$  different resource usage levels, as  $\mathbf{x} =$  $(x_{r_1}, x_{r_2}, \cdots, x_{r_{\bar{b}}})$ , where  $\underline{r} \leq r_1 < r_2 < \cdots < r_{\bar{b}} \leq \bar{r}$ and block  $r_{\bar{b}}$  is the vertex of each generated TR work profile. We may use block  $r_b$  for  $1 \le b < b$  in both left and right legs of each generated TR work profile with durations  $x_{r_b}^{\bar{t}}$  and  $x_{r_b}^{r}$ , respectively. If the duration of a single block,  $r_b$ , is  $x_{r_b} \geq 2\underline{t}$ , we can distribute  $x_{r_b}$ between  $x_{r_b}^l$  and  $x_{r_b}^r$ . The number of different ways of this work is equal to the number of common solutions of the following equations:

$$x_{r_b}^l + x_{r_b}^r = x_{r_b}, (26)$$

$$x_{r_b}^l, x_{r_b}^r \in \{0, \underline{t}, \cdots, x_{r_b}\}.$$
 (27)

If  $x_{r_b} \ge 2\underline{t}$ , the number of solutions of Equations 26 and 27 is equal to  $x_{r_b} - 2\underline{t} + 3$ , including:

$$\{(x_{r_b}^l = 0, x_{r_b}^r = x_{r_b}), (x_{r_b}^l = \underline{t}, x_{r_b}^r = x_{r_b} - \underline{t}), \\ (x_{r_b}^l = \underline{t} + 1, x_{r_b}^r = x_{r_b} - \underline{t} - 1), \cdots, \\ (x_{r_b}^l = x_{r_b} - \underline{t}, x_{r_b}^r = \underline{t}), (x_{r_b}^l = x_{r_b}, x_{r_b}^r = 0)\},$$

www.SID.ir

we will have only two possible values where |x|

$$\{(x_{r_b}^l = x_{r_b}, x_{r_b}^l = 0), (x_{r_b}^l = 0, x_{r_b}^l = x_{r_b})\}.$$

If we define function f for integer numbers as:

$$f(a) = \begin{cases} a; & a \ge 0\\ -1; & a < 0 \end{cases}$$

we generally have  $f(x_{r_b} - 2\underline{t}) + 3$  different solutions for Equations 26 and 27. Thus, the number of total different feasible TR work profiles that can be obtained from a feasible ND work profile,  $\mathbf{x} = (x_{r_1}, x_{r_2}, \cdots, x_{r_b})$ , is equal to  $\prod_{b=1}^{\bar{b}-1} (f(x_{r_b} - 2\underline{t}) + 3)$ .

It can be seen that the FWPS for each activity is constituted of different neighborhoods. At the core of each neighborhood, there are two feasible work profiles, NI and ND, and other neighbors are possibly other feasible TR work profiles. Each feasible TR work profile can be obtained from only one core and, hence, is located in only one neighborhood. Therefore, FWPS is partitioned by this neighborhood definition. Figure 5 shows one ND, one NI and two TR work profiles that constitute a neighborhood.

## Upper and Lower Bounds of the Project Makespan

Regarding Equations 15 to 19, we obtain the upper bound UB for the project makespan as UB=  $\sum_{i=1}^{n} d_i^{\min}$ . The project makespan lower bound LB is the maximization of a critical path-based lower bound LB<sub>0</sub> and a resource-based lower bound LB<sub>r</sub>. The lower bound LB<sub>0</sub> is obtained by calculating the critical path in the activity network, where each activity, *i*, is assigned its shortest feasible work profile with duration  $d_i^{\min}$ , taking into account the resource availability, *a*. The resource-based lower bound LB<sub>r</sub> is computed as:



Figure 5. An example of four neighbor work profiles.

where  $\lfloor x \rfloor$  denotes the largest integer equal to, or less than, x.

## SCHEDULE REPRESENTATION AND GENERATION SCHEME

Our constructive heuristic algorithm uses a schedule representation to encode a project schedule and a Schedule Generation Scheme (SGS) to translate the schedule representation to a real schedule. The SGS determines how a feasible schedule is constructed by assigning starting times to the activities whereby the relative priority and activity work profiles are determined by the schedule representation in our algorithm. There are various approaches for both the representation and generation of a schedule.

We represent a schedule, S, of the RCPSP\_FWP by a double list  $(\rho, \mathbf{wp})$ . The first list is a vector,  $\rho =$  $(\rho_1, \cdots, \rho_n)$ , such that positive real number  $\rho_i \in \mathbf{R}^+$ represents the priority value of activity i. This list was introduced by Kolisch and Hartmann [14] as Random Key (RK) representation for the RCPSP. The second list is a work profile list  $\mathbf{wp} = (wp_1, \cdots, wp_n)$  where  $wp_i$  represents the work profile chosen for activity *i*. We utilize a modified version of the RK representation developed by Debels et al. [4] in which the Topological Order (TO) concept is used in the construction of the RK, which we denote as TORK. Each TORK is constructed by scheduling the activities using a SGS, obtaining schedule S and considering priority values as  $\rho_i = s_i(S); \forall i \in N \text{ where } s_i(S) \text{ indicates the start time}$ of activity i in schedule S. For a detailed discussion of the advantages of the topological order representation, we refer to [15].

Besides different schedule representations, there also exist three different types of SGSs in the literature: the Serial SGS (SSGS), the parallel SGS and the bidirectional SGS [16]. As it is sometimes impossible to reach an optimal solution with parallel and bidirectional SGSs, we opt for serial SGS. At each iteration of the SSGS, the activity with the highest priority is chosen and assigned the first possible starting time such that no precedence or resource constraint is violated. Although any generated schedule of the RCPSP by the SSGS belongs to the set of active schedules [17], it does not necessarily hold for the RCPSP\_FWP.

Figure 6 illustrates an example project with five non-dummy activities and one renewable resource with a = 6 and  $\underline{t} = 2$ . The schedule of Figure 7 is obtained, based on the random priority list,  $\boldsymbol{\rho} = (1, 2, 3, 4, 5)$ , and SSGS. The corresponding TORK is  $\boldsymbol{\rho} = (0, 0, 8, 8, 15)$ .

### **GENETIC ALGORITHM**

We propose a genetic algorithm to schedule activities in a RCPSP\_FWP. The general structure of our GA





Figure 7. A schedule and its TORK.

1) Build an initial population $P$ of direct schedules
while (termination criterion is not satisfied) $\mathbf{do}$
2) $NewP = \emptyset$
for $[e = 1, to psize]$ do
3) Select a mate randomly for element $e$
4) Apply crossover operator to build $S_s$ and $S_d$
5) Apply mutation operator to $S_s$ and $S_d$ with
probability $p_{mut}$
6) Apply SGS to both $S_s$ and $S_d$
7) Select $S_c$ and
8) Apply local search operator to $S_c$ with probability $p_{ls}$
9) Add $S_c$ to the $NewP$
end for
10) $\boldsymbol{P} = \boldsymbol{NewP}$
end while

 ${\bf Figure \ 8.}\ {\rm General\ structure\ of\ GA}.$ 

is shown in Figure 8. The procedure starts with the random generation of an initial population, P, with size of *psize* of schedules. In order to generate each element,  $e = (\rho, \mathbf{wp})$ , of initial population, P, we generate a random priority list,  $\rho$ , randomly select a work profile for each activity from its set of feasible work profiles and decode element, e, to a real schedule using SSGS. Then, the procedure is followed by an iterative process that continues until the termination criterion is satisfied. The iterative process consecutively applies crossover, mutation and local search operators in order to generate new solutions with higher quality. In the second step, the *NewP*, representing a new population, is initialized as an empty set and will include the generated schedules in each iteration. In order to construct the new elements, each element,  $e = 1, \cdots, psize$  of P, considered as the father schedule  $(S_f)$ , is mated randomly with element  $c \ (c \neq e \text{ and } 1 \leq c \leq psize) \text{ of } P, \text{ considered as the}$ mother schedule  $(S_m)$ , to constitute a couple. Each couple generates two children schedules considered as the son schedule  $(S_s)$  and the daughter schedule  $(S_d)$ . The children generation process is performed using two operators, i.e. crossover and mutation where mutation is applied with probability  $p_{mut}$ . The child with the better makespan obtained using SSGS is chosen as the selected child schedule  $(S_c)$ . To improve  $S_c$ , a local search procedure is applied to it with probability  $p_{ls}$ . Then, P is updated by transferring all elements of NewP to it.

### Crossover

After the selection of parents, a crossover operation combines the  $S_f$  and  $S_m$  to generate  $S_s$  and  $S_d$ . We use a two-point crossover operator, based on the idea of replacement of weak sub-schedules with better subschedules, in which the crossover points are determined, based on the resource profile for the given schedule. The crossover point,  $cp_1$ , is defined as an activity's start time and the crossover point,  $cp_2$ , is defined as an activity's finish time. All activities executed totally between these two crossover points are called a sub-schedule. We define the Resource Utilization Ratio (RUR) as follows. Let  $A[t_1, t_2] = \{i | s_i \ge t_1 \text{ and } f_i \le t_2\}$  as the set of activities that are executed totally in interval  $[t_1, t_2]$ . The RUR for schedule S in interval  $[t_1, t_2]$  is defined as  $\operatorname{RUR}^{S}[t_{1}, t_{2}] = \sum_{t=t_{1}+1}^{t_{2}} \sum_{i \in A} [t_{1}, t_{2}](r_{it}/a).$ In order to determine crossover points, we look for subschedules including at least |n/3| and, at most, [2n/3]activities giving the maximum RUR. Assume  $cp_1 = s_i$ and  $cp_2 = f_j$ , where  $i, j \in \{1, \dots, n\}$  and subject to the two following conditions:

- I)  $s_j \geq s_i$ ,
- II)  $f_j > f_i$  or  $f_j = f_i$  where  $j \neq i$ .

For all real activities i and j, we consider all possible intervals  $\lfloor t_1 = s_i, t_2 = f_j \rfloor$  regarding Conditions I and II as the candidates for the cross points. The two points,  $t_1^*$  and  $t_2^*$ , are considered as the crossover points,  $cp_1$  and  $cp_2$ , respectively, for which the maximum resource utilization ratio is obtained. In other words, the cross points are determined based on the following condition:

$$\operatorname{RUR}^{S}[t_{1}^{*}, t_{2}^{*}] \geq \operatorname{RUR}^{S}[t_{1}, t_{2}],$$
$$\forall t_{1} = s_{i},$$

and:

$$t_2 = f_j, \qquad i, j = 1, \cdots, n,$$

and holding Conditions I and II.

Each activity, i, for which  $s_i \ge cp_1$  and  $f_i \le cp_2$ , constitutes the subset  $N_{sub}$ . In the following, we describe how the son schedule,  $S_s$ , is built from  $S_f$  and  $S_m$ . First, we determine  $cp_1(S_f)$  and  $cp_2(S_f)$  using the above mentioned two-point crossover operator. To generate the vectors,  $\rho_s$  and  $\mathbf{X}_s$ , of the  $S_s$ , for every activity,  $i \in N$ , we define the following combination rule. If in schedule  $S_f$  activity  $i \in N_{sub}$ , let  $\rho_{si} = \rho_{fi}$ and  $wp_{si} = wp_{fi}$ . Otherwise, if  $\rho_{fi} < cp_1(S_f)$ , let  $\rho_{si} = \rho_{mi} - b$  and  $wp_{si} = wp_{mi}$  and if  $\rho_{fi} > cp_1(S_f)$ , let  $\rho_{si} = \rho_{mi} + b$  and  $wp_{si} = wp_{mi}$  where b is a very big constant.

This combination rule allows the crossover operator to place the good sub-schedules with the high resource utilization of the  $S_f$  in  $S_s$  and replace the weaker sub-schedules of the  $S_f$  with the lower resource utilization by corresponding sub-schedules of the  $S_m$ . The big constant b is used in order to prevent the relative priority structure between the activities of a case being mixed with the priority values of activities of another case in the combination rule. By applying a SSGS on the obtained  $\rho_s$  and  $\mathbf{wp}_s$ ,  $S_s$  is obtained. If we assume  $S_c = S_s$ , by obtaining the TORK corresponding to  $S_c$  and applying the SSGS to it, the generation of  $S_c$  is finished. By exchanging the role of  $S_f$  and  $S_m$  in generation of the child schedule,  $S_d$  can be easily obtained. Based on the lower makespan,  $S_s$ or  $S_d$  is selected as the child schedule,  $S_c$ . Schedules represented in Figures 9 to 11 are related to the example network of Figure 6 and illustrate the crossover operator in which son schedule  $S_s$  is considered as the selected child schedule  $S_c$ .



**Figure 9.** Father schedule  $(S_f)$ .



**Figure 10.** Mother schedule  $(S_m)$ .

M. Ranjbar and F. Kianfar



**Figure 11.** Child schedule  $(S_c)$ .

#### Mutation

After applying the crossover operator, each generated schedule is muted with a probability of  $p_{\text{mut}}$  using a mutation operator that works as follows. First, it selects, randomly, nrm out of n non-dummy activities as the chosen activities for mutation. Then, for each selected activity i, it chooses, randomly, a feasible work profile from the non-neighbors work profiles of its current work profile,  $wp_i$ . Finally, for the selected activity i, it changes the priority value as  $\rho_i = \rho_i + r_i$ , where  $r_i \in [1/3f, 2/3f]$  is a random integer.

#### Local Search

A local search procedure is applied over each generated schedule with probability  $P_{ls}$ , described by a Pseudocode in Figure 12. In this procedure, given a schedule S, f(S) denote the makespan of schedule S, Improvement is a Boolean variable and SA indicates the set of selected activities for a change in the work profile. This procedure tries to decrease f(S) by alternating the work profile of each activity in the neighborhood of the activity work profile.

This procedure includes a *While* loop, in which all activities are selected based on their priority values, re-

1) Build the priority list $\boldsymbol{\rho} = (\rho_1, \cdots, \rho_n)$ based on schedule
$S$ where $\rho_i = s_i$ .
2) Let $Improvement = true$ .
$\mathbf{While} \ (improvement = true) \ \mathbf{do}$
3) Let <i>improvement</i> = false and $SA = \emptyset$
for $(i = 1 \text{ to } n)$ do
4) Select activity j for which $\rho_j = \min(\rho_1, \cdots, \rho_n)$
and $j \neq SA$ .
5) Add activity $j$ to $SA$ .
6) Change the work profile of activity $j$ in its
defined neighborhood, apply SSGS for each
change in work profile and select the best
one resulting in schedule $S'$ .
7) If $f(S') < f(S)$ , let $S = (S')$ $(\boldsymbol{\rho} = \boldsymbol{\rho'} \cdot \boldsymbol{w} \boldsymbol{p} = \boldsymbol{w} \boldsymbol{p'})$ ,
f(S') = f(S) and <i>improvement=true</i> .
and for
end while
8) If $(improvement = true)$ go to 1, otherwise stop.

Figure 12. Local search procedure.

spectively. In the main step (Step 6) of this procedure, for each selected activity j, all neighbors of its current work profile are replaced one by one as the work profile of activity j, while the other activities' work profiles and priority values are not changed. Each generated schedule in Step 6 is evaluated using its makespan and, finally, the best work profile is selected for activity j, resulting in schedule S'. If f(S') < f(S), schedule S' is replaced with schedule S and *Improvement* is changed to *true*. In the last step, the procedure is stopped if no improvement is obtained; otherwise, it goes to the first step and updates the priority values based on the best found schedule of the loop.

## COMPUTATIONAL RESULTS

## Benchmark Problem Set

We have coded the procedure in Visual C++ 6.0 and performed all computational experiments on a PC Pentium IV 3GHz processor with 1024 MB of internal memory. We have validated the procedure on two sets of instances developed by Kolisch [1]; one with 30 non-dummy activities and one with 60 non-dummy activities (additionally, each instance has a dummy start and a dummy end activity, each with a duration of zero and a work content of zero). These instances have been generated by transforming a subset of instances from the 30- and 60-activity sets of the instances as available in the PSPLIB library (http://www.bwl.unikiel.de/bwlinstitute/Prod/psplib/datasm.htm. Only one renewable resource is considered in the generation of these instances.

Each instance is characterized by the following four parameters: rurf,  $t \min$ , rsf and NC, which are detailed below.

- a) rurf stands for the resource usage range factor. It determines for each activity *i* the range of  $[\underline{r}_i, \bar{r}_i]$ for the per period resource usage of this activity.  $rurf \in [0, 1]$  where 0 defines a range with  $\underline{r}_i = \bar{r}_i$ and 1 defines a large range. In these two sets of instances, rurf is set to 0.1 and 0.3, respectively. Based on the rurf value, for each activity *i*,  $\underline{r}_i$  and  $\bar{r}_i$  are calculated as  $\underline{r}_i = \lfloor w_i(1 - rurf) \rfloor$  and  $\bar{r}_i = \lceil w_i(1 + rurf) \rceil$ .
- b)  $t \min$  stands for the minimum number of consecutive periods where activity i has to have a constant resource usage.  $t \min$  has been set to 1 and 3 for all activities, respectively.
- c) rsf stands for the resource strength factor. rsf determines the scarcity of the resources. For rsf = 0, the resources will be as scarce as possible to allow a feasible solution. For rsf = 1, the amount of capacity is such that the resource constraints are

not binding any more. In our problem instance, rsf has been set to 0.2 and 0.5, respectively.

d) NC stands for the network complexity. It measures the average number of non-redundant arcs per activity in the network. NC has been set to 1.5, 1.8 and 2.1, respectively.

Using a full factorial design,  $2^*2^*2^*3 = 24$  instances have been generated with 30 and with 60 activities, respectively. Also, for each activity *i*, the work content,  $w_i$ , is a random integer from interval [1, 100].

#### **Parameter Settings**

To test our procedure, we predefine the settings of the parameters of the genetic algorithm. The number of children, *nrc*, is fixed at 2, the number of mutated activities in the mutation step is set equal to n/10. Using fine tuning, we obtained the values of  $p_{\rm mut}$  and  $p_{ls}$  parameters as  $p_{\rm mut} = 0.04$ ,  $p_{ls} = 0.1$ . Also, the fined tuned values of *psize* related to the stop condition are considered as three CPU-time-limits 1, 10 and 50 seconds, and the size of the problem-instances are shown in Table 2.

### Number of Feasible Work Profiles

Table 3 indicates the average number of feasible NI, ND and TR work profiles for each activity based on the different values of parameters t and *rurf*. It can be seen that the number of feasible work profiles is an increasing function of  $1/\underline{t}$  and *rurf*.

#### **Detailed Computational Results**

The performance of the GA for the RCPSP\_FWP on the 24 instances is summarized in Table 4. We report the total sum of the makespan of each obtained solution

Table 2. Tuned values of *psize*.

Instances	CPU-Time-Limit (s)		
$\mathbf{Set}$	1	10	50
J 30	50	110	425
J60	30	75	400

Table 3. Average number of feasible work profiles.

	Type of Work Profile		
Values of Parameter	SM	ND (NI)	$\mathbf{TR}$
$\underline{t} = 1, \ rurf = 0.1$	4.83	36.04	294.08
$\underline{t} = 1, \ rurf = 0.3$	6.15	50.12	381.42
$\underline{t} = 3, \ rurf = 0.1$	1.56	6.33	9.58
$\underline{t} = 3, \ rurf = 0.3$	2.62	10.16	15.72

	1		
Problem Set	CPU-Time Limit (s)	Data Set	
		<b>J</b> 30	J60
	5	1410	2310
Sum of the makespans	10	1319	2221
	50	1027	1783
	5	35.03%	38.61%
Average deviation from the LB	10	32.74%	37.11%
	50	25.49%	29.75%

Table 4. Detailed computational results.

and the average deviation from the lower bound LB, described previously.

Table 5 shows the effect of the work profile types. The results are presented as the average deviation from the LB for 1, 10 and 50 seconds CPU-time-limit and for four types of work profile: SM, ND, NI and TR. The results related to TR and SM work profile are the best and the worst, respectively. This result is in line with our expectations because  $SM\subseteq NI\subseteq TR$  and  $SM\subseteq ND\subseteq TR$ .

## Impact of the Mutation and the Local Search

Figures 13 and 14 indicate the impact of the mutation and the local search on the RCPSP\_FWP results by changing  $p_{\rm mut}$  and  $p_{ls}$ , respectively. The results are obtained based on the average percent deviation from

	CPU-Time Limit (s)			
Work Profile	5	10	50	
$_{\rm SM}$	44.16%	42.87%	41.59%	
ND	37.64%	36.28%	29.47%	
NI	37.69%	36.27%	29.45%	
TB	35 22%	35.06%	26.48%	

Table 5. Effect of the work profile types.



Figure 13. Impact of the mutation operator.



Figure 14. Impact of the local search operator.

the lower bound LB when the allowed CPU-time limit is 10 seconds. The results reveal that the best values for the probability of applying these two operators on each generated schedule are as  $p_{\rm mut} = 0.04$  and  $p_{ls} = 0.1$ .

## CONCLUSION

In this paper, we introduced the Resource-Constrained Project Scheduling Problem with a Flexible Work Profile (RCPSP\_FWP). We developed a linear model for the problem, an enumeration procedure for generation of feasible work profiles and a metaheuristic, based on the Genetic Algorithm (GA), for solving the problem. As the crossover operator, the GA uses a two-point crossover based on the idea of the replacement of weak sub-schedules with better ones. To that purpose, we used the resource utilization ratio to evaluate different sub-schedules. We also developed a local search incorporated with GA to improve the solutions' quality. We performed detailed computational results on two problem sets, J30 and J60, each including 24 problem instances.

Our future research efforts will focus on two extensions. First, we focus on developing an exact solution approach for the RCPSP\_FWP and also better metaheuristics. Second, we focus on considering work profiles in general forms and developing a procedure to generate them.

#### REFERENCES

- Kolisch, R. "Selection and scheduling of pharmaceutical research projects", *Perspectives in Modern Project Scheduling*, Chapter 13, Springer-Verlag (2006).
- De Reyck, B., Demeulemeester, E. and Herroelen, W. "Local search methods for the discrete time/resource trade-off problem in project networks", *Naval Research Logistic Quarterly*, 45, pp. 553-578 (1998).
- Salewski, F., Schirmer, A. and Drexl, A. "Project scheduling under resource and mode identity: model, complexity, methods, and application", *European Journal of Operational Research*, **72**, pp. 88-110 (1997).
- Brucker, P., Drexl, A., Möhring, R., Neumann, K. and Pesch, E. "Resource-constrained project scheduling: notation, classification, models and methods", *European Journal of Operational Research*, **112**, pp. 3-41 (1999).
- Kolisch, R. and Padman, R. "An integrated survey of deterministic project scheduling", *Omega*, 29, pp. 249-272 (2001).
- Kis, T. "Project scheduling: A review of recent books", *Operations Research Letters*, **33**, pp. 105-110 (2005).
- Demeulemeester, E. and Herroelen, W., Project Scheduling: A Research Handbook, Kluwer Academic Publishers (2002).
- 8. Kolisch, R. and Hartmann, S. "Experimental investigation of Heuristics for resource-constrained project scheduling: an update", *European Journal of Operational Research*, **174**, pp. 23-37 (2006).
- Demeulemeester, E., De Reyck, B. and Herroelen, W. "The discrete time/resource trade-off problem in project networks: A branch and bound approach", *IIE Transaction*, **32**, pp. 1059-1069 (2000).
- Ranjbar, M., De Reyck, B. and Kianfar, F. "A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling", *European Journal of Operational Research*, **100**, pp. 35-48 (2009).
- Blazewicz, J., Lenstra, J.K. and Rinnooy Kan, A.H.G. "Scheduling subject to resource constraints: classification and complexity", *Discrete Applied Mathematics*, 5, pp. 11-24 (1983).
- Mori, M. and Tseng, C.C. "A genetic algorithm for multi-mode resource-constrained project scheduling problem", *European Journal of Operational Research*, **100**, pp. 134-141 (1997).

- 13. Hartmann, S. "Project scheduling with multiple modes: A genetic algorithm", Annals of Operations Research, **102**, pp. 111-135 (2001).
- Kolisch, R. and Hartmann, S. "Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis", in *Project Scheduling-Recent Models, Algorithms and Applications*, Weglarz, J., Eds., Kluwer Academic Publishers, Boston, pp. 147-178 (1999).
- Debels, D., De Reyck, B., Leus, R. and Vanhoucke, M. "A hybrid scatter search/electromagnetism metaheuristic for project scheduling", *European Journal of Operational Research*, 169, pp. 638-653 (2006).
- Klein, R. "Bidirectional planning: Improving priority rule-based heuristics for scheduling resource constrained projects", *European Journal of Operational Research*, **127**, pp. 619-638 (2000).
- Kolisch, R. "Serial and parallel resource-constrained project scheduling methods revisited: theory and computation", *European Journal of Operational Research*, 43, pp. 23-40 (1996).

#### BIOGRAPHIES

Mohammad Ranjbar was born on January 9th, 1978 in Mashad, Iran. He obtained his BS and PhD from the Department of Industrial Engineering at Sharif University of Technology in 2000 and 2007, respectively, receiving his MS from the Faculty of Engineering at Tehran University. From October 2005 to March 2006, he was a visiting PhD student at the Department of Operations Management of the London Business School. In February 2008, he joined Ferdowsi University in Mashhad and, presently, is assistant professor of Industrial Engineering at this university. Dr. Ranjbar is the author of six papers published in international journals, one of which was presented at an international conference.

Ferydoon Kianfar was born on May 17th, 1942 in Tehran, Iran, later receiving his BS in Mechanical Engineering from Tehran Polytechnic. He obtained his MS and PhD in Industrial Engineering from Illinois Institute of Technology (IIT) in Chicago, USA, where, from 1969 to 1970, he was employed as an assistant professor. Since 1970 he has been working at Sharif University of Technology in Tehran, Iran, and is now a full professor of Industrial Engineering.