IUST International Journal of Engineering Science, Vol. **18***, No.***3-4, 2007***, Page* **7-18**
Industrial & Mechanical Engineering Special Issue

A GENETIC ALGORITHM AND A MODEL FOR THE RESOURCE CONSTRAINED PROJECT SCHEDULING PROBLEM WITH MULTIPLE CRUSHABLE MODES

S. M. Seyed-Hosseini, Majid Sabzehparvar, and Siamac Nouri

Abstract: *This paper presents an exact model and a genetic algorithm for the multi-mode resource constrained project scheduling problem with generalized precedence relations in which the duration of an activity is determined by the mode selection and the duration reduction (crashing) applied within the selected mode. All resources considered are renewable. The objective is to determine a mode, the amount of continuous crashing, and a start time for each activity so that all constraints are obeyed and the project duration is minimized. Project scheduling of this type occurs in many fields; for instance, predicting the resources and duration of activities in software development projects. A key feature of the model is that none of the typical models can cope with the continuous resource constraints. Computational results with a set of 100 generated instances have been reported and the efficiency of the proposed model has been analyzed.*

Keywords: *Crashable modes; Multi-mode; Time/resource trade-off; Time windows.*

1. Introduction

For many real-life applications of project scheduling, it is possible to perform the individual activities in alternative ways (modes). These modes are different in processing time, time lags to other activities, and resource requirements. They reflect time/cost, time/resource and resource/resource trade-offs [10]. Such real-life projects can be modeled as instances of the Multi-Mode Resource-Constrained Project Scheduling Problem or briefly ***MRCPSP*** which is denoted by $MPS|prec|C_{\max}$. *MRCPSP* in the situations involving minimum and maximum time lags or Generalized Precedence Relations called ***MRCPSP-GPR*** [7: Page 512] and denoted by $MPS|temp|C_{\max}$ [16: Page 160, 9: Ch 6 Page 103]. *MRCPSP* is a generalized version of the standard well-known Resource-Constrained Project Scheduling Problem or briefly ***RCPSP*** ($PS|prec|C_{\max}$) which is in *GPR* denoted by ***RCPSP-GPR*** or $PS|temp|C_{\max}$ [16, Page 22].

There is a general case of *MRCPSP-GPR* in which the duration/cost of an activity is treated as a function of both the resource requirements (mode selection) and the amount of crashing (duration reduction), applied within the selected mode. This complex case has been introduced by Ahn and Erenguc [1] and called Resource-Constrained Project Scheduling Problem with Multiple Crash able Mode or briefly ***RCPSPMCM***. They provide this example: Activity i can be done by "worker A using machine X (mode 1)" or by "worker B using machine Y (mode 2)". Assuming 8 hours of work per day, worker A, using machine X can finish activity i in 10 working days at a price of $400 and worker B using machine Y can complete the activity in 8 working days at the price of $500. Furthermore, workers A and B can shorten the activity duration by working additional hours each day. For example, worker A can finish the activity in 8 days by working 10 hours a day. Duration reduction, i.e. crashing, can be done in various ways: using overtime or additional shift(s), or allocating more resources that might be acquired easily by incurring additional expenditures.

Their objective function involves the minimization of the total project cost which is the sum of the activity execution cost and the tardiness cost. In the absence of resource constraints, the problem reduces to the Time/Cost Trade-off Problem or briefly ***TCTP***. In the absence of crashing within a mode, the problem reduces to the *MRCPSP*.

The complexity of the problem is increased when the crashing applied within a selected mode is continuous and minimum and maximum time lags (time windows) between a pair of activities are considered. In such a case denoted by ***RCPSPMCM-GPR***, due to the continuous crashing, no exact solution method has been reported [1: Page 255].

Paper first received May. 03, 2007 and in revised form Sep. 22, 2007.
S.Mohammad Seyed-Hosseini, Majid Sabzehparvar, Siamac Nouri, Department of Industrial Engineering, Iran University of Science & Technology, seyedhoseini@yahoo.com, msabzeh@gmail.com,

The problem is strongly *NP-hard*. Moreover, the problem *RCPSP-GPR* is also *NP-hard*, and even the question whether a problem instance has a feasible solution is *NP-complete* in strong sense [3]. As a generalization of the problem *RCPSP-GPR*, the problem *MRCPSP-GPR* and its corresponding feasibility problem belong to the same complexity class. [9: Page 8, 16: Page 165].

The remainder of this paper is organized as follows: In Section 2, we discuss the relevant literature review. Section 3 gives a conceptual formulation based on the previous works. A brief reason why this specific formulation is tackled by the authors is discussed in Section 4. New mathematical formulation for *RCPSP-GPR*, *MRCPSP-GPR*, *MRCPSP-GPR with Mode-Dependent time lags* and *RCPSPMCM-GPR* as the contributions of this paper are presented in Subsections 5.1, 5.2, 5.3 and 5.4 respectively. Genetic algorithm implementation as another contribution is discussed in Section 6. Computational results are reported in Section 7. Section 8 is reserved for our conclusions.

2. Relevant Literature Review

MRCPSP without maximum time lag has been treated by several authors since the early eighties [16, P. 160]. *RCPSPMCM* without *GPR* in which resource/resource, time/resource and time/cost trade-off are considered, has been introduced by Ahn and Erenguc [1]. They have presented a heuristic procedure for *RCPSPMCM* but no exact solution method has been reported. A linear approximation of duration function has been carried out by Deckro and Hebert [6] called modeling diminishing returns but they have not considered resource constraints in their model.

Exact algorithms for the case of *MRCPSP* have been reviewed and their performance has been tested by Hartmann and Drexl [12].

The most efficient method for solving *MRCPSP* known thus far is the branch-and-bound algorithm of Sprecher and Drexl [19]. The best heuristic procedure for *MRCPSP* at present is a genetic algorithm published by Hartmann [16: Page 160].

Since we are going to present a mathematical formulation for *RCPSPMCM*, the focus of remaining literature review will be on the exact procedures especially mathematical formulations.

In order to be able to specify the resource constraints in the correct and solvable form, linear programming based approaches for *RCPSP* (not *MRCPSP*) have been presented by several authors [7].

In formulation by Pritsker et al. [17], the binary decision variable x_{it} is defined to be 1 if activity i finishes at time instant t, and to be 0 otherwise.

For the *MRCPSP* without *GPR* ($MPS \mid prec \mid C_{\max}$), the best $0-1$ programming model based on an extension of the formulation by Pritsker et al., has been presented by Talbot [20]. The model of Talbot have been developed by Reyck and Herroelen [8] for the case of *GPR* ($MPS \mid temp \mid C_{\max}$).

Since lower bound calculations play the most important role in branch-and-bound procedures, it has been addressed by recent papers [2].

The best known lower bound for *RCPSP* and *MRCPSP* is currently based on the resolution of several large linear programs [4, 13]. Computational experience by Mothering et al. [16: Page 75] has shown that already for medium-sized projects of *RCPSP* with 100 activities, the resulting linear program cannot be solved in an acceptable amount of computational time. Broker and Knits [4] strengthen one of this relaxation by taking into account time windows for the activities and use column generation to deal with the large number of variables.

3. Conceptual Formulation for *RCPSPMCM-GPR*

Let:

v_{im_i} binary decision variable; *1* if activity i is performed in mode m_i, *0* otherwise.

s_i / f_i starting/finishing time of activity i

d_i duration of activity i

$ss_{ij}^{\min}$ / $ss_{ij}^{\max}$ minimal/maximal time lag between start to start times of activities i and j

$sf_{ij}^{\min}$ / $sf_{ij}^{\max}$ minimal/maximal time lag between start to finish times of activities i and j.

$fs_{ij}^{\min}$ / $fs_{ij}^{\max}$ minimal/maximal time lag between finish to start times of activities i and j.

$ff_{ij}^{\min}$ / $ff_{ij}^{\max}$ minimal/maximal time lag between finish to finish times of activities i and j.

$ss_{im_i, jm_j}^{\min}$ elements of transformed matrix of minimal time lags, if activities i and j are performed in modes m_i and m_j respectively.

R_k availability or the maximum number of the K^{th} resource type

r_{im_ik} resource requirement of type k for activity i in mode m_i

K number of resource types required for the project.

$\overline{T}$ upper bound on the shortest project duration, where $\overline{T} \geq es_{n+1}$ (earliest start).

$\overline{D}$ project due date

$\overline{P}$ per period penalty cost incurred if the project is delayed beyond $\overline{D}$

c_{im_i} marginal crashing cost of activity i using mode m_i

$\underline{d}_{im_i} / \overline{d}_{im_i}$ rashed (minimum) / normal (maximum) duration of activity i in mode m_i

$\overline{c}_{im_i} / \underline{c}_{im_i}$ crashed (maximum) / normal (minimum) cost of activity i in mode m_i

Assuming an *AoN* network *N(v)* in standardized form with minimal start to start precedence relations using the transformation rules [3], the problem can be modeled conceptually as a mathematical programming model in the following way [1, 16] :

$$Min \quad \sum_{i=1}^{n}\sum_{m_i}^{M_i}\{\underline{c}_{im_i} + c_{im_i} \times (\overline{d}_{im_i} - d_i)\} \times v_{im_i} + \overline{P}\max\{0, S_{n+1} - \overline{D}\} \quad (1)$$

Subject to:

$$\sum_{m_i=1}^{M_i} v_{im_i} = 1 \qquad i = 1,...,n \quad (2)$$

$$s_j - s_i \geq \sum_{m_i=1}^{M_i}\sum_{m_j=1}^{M_j} ss_{im_i jm_j}{}^{\min} v_{im_i}.v_{jm_j} \qquad i \prec j,\ \langle i, j\rangle \in E_{ss^{\min}} \quad (3)$$

$$\sum_{m_i=1}^{M_i} \underline{d}_{im_i} v_{im_i} \leq d_i \sum_{m_i=1}^{M_i} v_{im_i} \leq \sum_{m_i=1}^{M_i} \overline{d}_{im_i} v_{im_i} \qquad i = 1,...,n \quad (4)$$

$$\sum_{i\in A(s,t,v)}\sum_{m_i=1}^{M_i} r_{im_i k} v_{im_i} \leq R_k \qquad k = 1,...,K,\ 0 \leq t \leq \overline{T} \quad (5)$$

$$s_i \geq 0 \quad (6)$$

$$s_0 = 0 \quad (7)$$

$$v_{im_i} \in \{0,1\} \qquad (i \in V,\ m_i = 1,...,M_i) \quad (8)$$

Eq. (1) minimizes the project cost including crashing cost within each mode and penalty cost incurred if the project is delayed beyond tardiness of the project, computed as $\max\{0, s_{n+1} - \overline{D}\}$. s_0 and s_{n+1} are the starting times of the non-real first and end activities respectively and s_i, d_i and v_{im_i} are decision variables to be determined. Eq. (2) ensures that only one of the modes is selected. Eq. (3) denotes the *GPRs* in standardized form.

The duration variable of the activity, is bounded between $\underline{d}_{im_i}$ and $\overline{d}_{im_i}$ by Eq. (4) if activity i is performed in m_i. Eq. (5) which is a conceptual statement of the resource constraints expresses that at no time instant of t, during the project horizon between 0 and $\overline{T}$ the resource availability may be violated. Moreover, we define:

$$A(s,t,v) := \{i \in V | s_i \leq t < s_i + d_i\}, (t \geq 0) \quad (9)$$

which is the set of real activities in progress at time t, depending on starting time s and assignment v.

4. Difficulties of Modeling Resource Constraints

The mathematical program above cannot be solved directly because it is necessary to translate the set $A(s,t,v)$, that is used in Eq. (5) into the solvable constraints. Hence 0-1 programming formulations have to be used in order to be able to specify the resource constraints in the correct and solvable form [7: P 208].

In formulation by Talbot [20] which is for the case of *MRCPSP*, the binary decision variable $v_{im_i t}$ is defined to be 1 if activity i is performed in mode m_i and started at time t, and to be 0 otherwise. Thus the Eq. (5) can be stated as follows:

$$\sum_{i=1}^{n}\sum_{m_i=1}^{M_i} r_{imk} \sum_{s=\max\{t-d_{im},es_i\}}^{\min\{t-1,ls_i\}} v_{im_i s} \leq R_k \quad k = 1,2,...,K; \quad t = 1,2,...,\overline{T} \quad (10)$$

The variable v_{imt} can only be defined over the interval between the earliest (es_i) and latest (ls_i) starting time of the activity in question. The difficulty increases when dealing with different and smaller duration units (e.g., hour, minute, second) is necessary. Consequently, the number of decision variables will be increased exponentially.

5. New Mathematical Formulation

The formulation which is presented here has been inspired by one of the rectangle packing problems models [5].

5.1. Mathematical Formulation for *RCPSP-GPR*

5.1.1. Main Idea

In *RCPSP-GPR* we have a single mode so m_i must be omitted from the notations defined above. For visualizing the problem in three dimensions, imagine the number of resource types required for the project are two (K=2).

It can, however, be beyond the confines of two, i.e. any integer value (K=1, 2, 3, ...). Moreover, it must be noted that the geometrical model is applicable in the situation involving the uniformly distributed resource needs over processing times which is not the case about nonrenewable resources. Thus renewable resources which usually have this property are considered.

In such a case, there is a certain correspondence between boxes to be packed, and activities to be

scheduled. Each box would correspond to an activity, with a duration equal to the length and a resource request of type k (k=1, 2) equal to width and height respectively.

An empty box B_0 of width W_0 equal to time horizon T, length L_0 equal to R_1, the resource capacity

available of type 1 and height H_0 equal to R_2, the resource capacity available of type 2 is given. There is a series of boxes B_i (or Activities A_i) ($i = 1,...,n$), of width $w_i = d_i$, length $l_i = r_{i1}$ and height $h_i = r_{i2}$ to be packed in which m_i has been omitted from both d_{im_i} and r_{im_ik} for the case of single mode.

Furthermore, the constraint that activity preemption is not allowed corresponds to the natural requirement that boxes must be packed as a whole.

The bottom left behind corner of the big empty box B_0 is supposed to be at point $(0,0,0)$ so the top right front corner is (T, R_1, R_2). Let:

(x_i, y_i, z_i): The bottom left behind coordinates of activity i,

$(x_i + d_i, y_i + r_{i1}, z_i + r_{i2})$: The top right front coordinates of activity i.

5.1.2. GPRs Constraints

The x-coordinate of the bottom left behind corner of activity i is given by the activity starting time:

$$x_i = s_i \qquad n = 1,2,...,n \tag{11}$$

and is the most important decision variable to be determined. Thus, *GPRs* can be formulated as follows:

$$x_i + ss_{ij}^{\min} \le x_j \le x_i + ss_{ij}^{\max} \quad \langle i, j \rangle \in E_{ss}, \tag{12}$$

$$x_i + sf_{ij}^{\min} \le x_j + d_j \le x_i + sf_{ij}^{\max} \quad \langle i, j \rangle \in E_{sf}, \tag{13}$$

$$x_i + d_i + fs_{ij}^{\min} \le x_j \le x_i + d_i + fs_{ij}^{\max} \quad \langle i, j \rangle \in E_{fs}, \tag{14}$$

$$x_i + d_i + ff_{ij}^{\min} \le x_j + d_j \le x_i + d_i + ff_{ij}^{\max} \quad \langle i, j \rangle \in E_{ff}, \tag{15}$$

in which $f_i = x_i + d_i$ and $f_j = x_j + d_j$ are the finishing time of activity i and j respectively.

5.1.3. Constraint of Makespan

The finishing time of end activities $f_i = x_i + d_i$ should not be exceeded from *T*, i.e.,

$$T - x_i - d_i \ge 0. \quad i \in \textit{end activities} \tag{16}$$

5.1.4. The Objective Function

The minimization of makespan for *RCPSP-GPR* which is linear and the most popular objective in the project scheduling problems can be used as follows:

$$\textit{Minimize} \quad \boldsymbol{T}. \tag{17}$$

5.1.5. No Overlapping Constraints

The constraints for packing boxes are as follows [5]:

1. Since the activity boxes may not be rotated, each edge of an activity box should be parallel to a specific edge of the main box B_0.

2. There should be no overlapping for any two small boxes, i.e., the overlapping area is zero.

In the situations involving single type renewable resources in project (K=1), the problem can be formulated as the same as rectangle packing problem. The difficulties start when the number of resource types are two or more $(K \ge 2)$. In this case which is one of the contributions of this paper, the problem must be formulated totally different from packing problem. In packing problem, boxes must be packed to a container in which no overlapping between a pair of boxes coordinates is permitted, i.e., one of y-z, x-z or x-y overlapping is allowed at a time.

The no overlapping constraints for project scheduling must be changed as: There should be no overlapping for any two boxes between x- and y-coordinates as well as x- and z-coordinates, i.e., it is not important to have overlapping between y- and z-coordinates.

For *RCPSP-GPR* with single type renewable resource in which no precedence relation of type $fs_{ij}^{\min} \ge 0$ between two activities i and j exists, one of the following constraints must be held:

$$\begin{aligned}&(x_j \ge x_i + d_i) \vee (x_i \ge x_j + d_j) \vee \\ &(y_j \ge y_i + r_i^1) \vee (y_i \ge y_j + r_j^1)\end{aligned} \tag{18}$$

Let:

t_{xij}: 0-1 integer variable; 0 if activity i to be located at the left hand side of activity j (activity i precedes activity j) without any overlapping between them, 1 otherwise.

t_{xji}: 0-1 integer variable; 0 if activity j to be located at the left hand side of activity i (activity j precedes activity i) without any overlapping between them, 1 otherwise.

We use the same definitions for notations t_{yij} and t_{yji} toward y-coordinate, t_{zij} and t_{zji} toward z-coordinate.

Using the binary decision variables above, these constraints can be stated as follows:

$$\begin{aligned}&x_j - x_i - d_i + M * t_{xij} \ge 0 \; i < j, \; i = 1,....,n-1, \\ &j = 2,....,n\end{aligned} \tag{19}$$

$$x_i - x_j - d_j + M * t_{xji} \geq 0 \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{20}$$

$$y_j - y_i - r_{i1} + M * t_{yij} \geq 0 \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{21}$$

$$y_i - y_j - r_{j1} + M * t_{yji} \geq 0 \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{22}$$

$$t_{xij} + t_{xji} + t_{yij} + t_{yji} = 3 \; , i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{23}$$

where *M* is a big constant. Equations above ensure that there should be no overlapping for any two boxes, between x- and y-coordinates. No overlapping constraints for x- and z-coordinates which are applicable in double types of renewable resources, are defined as follows:

$$z_j - z_i - r_{i2} + M * t_{zij} \geq 0 \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{24}$$

$$z_i - z_j - r_{j2} + M * t_{zji} \geq 0 \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{25}$$

$$t_{xij} + t_{xji} + t_{zij} + t_{zji} = 3, \; i < j, \; i = 1,....,n-1, \quad j = 2,....,n \tag{26}$$

The same equations as (24), (25) and (26) can be developed in situations involving $K > 2$.

5.1.6. Resource Constraints

The y- and z-coordinates of activities should not exceed from R_1 and R_2 respectively. This can be stated in the form of constraints as follows:

$$R_1 - y_i - r_{i1} \geq 0 \quad i = 1,....,n \tag{27}$$

$$R_2 - z_i - r_{i2} \geq 0 \quad i = 1,....,n \tag{28}$$

5.2. Mathematical Formulation for *MRCPSP-GPR*

In the case of *MRCPSP-GPR* individual activities can be executed in alternative ways (modes). Activity i $(i = 1,...,n)$ when performed in mode m_i $(m_i = 1,...,M_i)$ has a duration d_{im_i} and requires r_{im_ik}, a constant amount of resource k over duration. To describe the mathematical formulation, Let:

$$d_i = \sum_{m_i=1}^{M_i} d_{im_i} . v_{im_i} \qquad i = 1,...,n \tag{29}$$

$$r_{ik} = \sum_{m_i=1}^{M_i} r_{im_ik} . v_{im_i} \qquad i = 1,...,n \quad k = 1,...,K \tag{30}$$

MRCPSP-GPR can be formulated by replacing Eq. (29), (30) and (2) into the *RCPSP-GPR* model.

5.4. Mathematical Formulation for *RCPSPMCM-GPR*

The *RCPSPMCM-GPR* involves *MRCPSP-GPR* plus the continuous crashing in each mode.

Let:

K_m number of resources types required for each mode different from other modes

$d_{im_i}(r_{im_i1},...,r_{im_iK_m})$ duration of activity i in mode m_i as a function of resources requirements

$\underline{r}_{im_ik} / \bar{r}_{im_ik}$ lower/upper bound on resource requirement of type k for activity i in mode m_i

Δr_{im_ik} resource increment of type k above the lower bound in mode m_i for Activity i

c_k procurement cost per unit of resource of type k

β_i regression coefficient

All other notations remain as previously defined.

As illustrated in Fig. 1, an approximation of an activity duration in each mode is made by the function defined over the range of possible values of the upper and lower bounds on the resources requirements of any type in that mode as follows:

$$d_{im_i} = d_{im_i}(r_{im_i1},...,r_{im_iK_m}) \; \underline{r}_{im_ik} \leq r_{im_ik} \leq \bar{r}_{im_ik} \, , \underline{r}_{im_ik} > 0 \tag{33}$$

So $\bar{d}_{im_i}$ and $\underline{d}_{im_i}$ are constants and can be calculated by Eqs. (32) and (33) respectively.

$$\bar{d}_{im_i} = d_{im_i}(\underline{r}_{im_i1},...,\underline{r}_{im_iK_m}) \tag{34}$$

$$\underline{d}_{im_i} = d_{im_i}(\bar{r}_{im_i1},...,\bar{r}_{im_iK_m}) \tag{35}$$

The duration reduction (crashing) for each activity between $\bar{d}_{im_i}$ and $\underline{d}_{im_i}$ is allowed by increasing the amount of resources committed to it. $d_{im_i}(r_{im_i1},...,r_{im_iK_m})$ may be a constant, a linear expression, or a non-linear function such as hyperbolic function and can be different from activity to activity.

Eq. (33) can be replaced by Eq. (36) as follows:

$$d_{im_i} = d_{im_i}(\underline{r}_{im_i1} + \Delta r_{im_i1},...,\underline{r}_{im_iK} + \Delta r_{im_iK_m}) \; i = 1,....,n \quad m_i = 1,...,M_i \tag{36}$$

where:

$$r_{im_ik} = \underline{r}_{im_ik} + \Delta r_{im_ik} \qquad i = 1,....,n \qquad m_i = 1,...,M_i \quad k = 1,...,K_m \tag{37}$$

$$0 \leq \Delta r_{im_ik} \leq \bar{r}_{im_ik} - \underline{r}_{im_ik} \; i = 1,....,n \; m_i = 1,...,M_i \quad k = 1,...,K_m \tag{38}$$

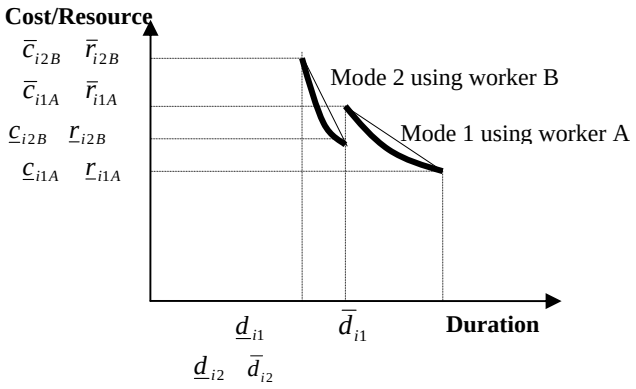

Fig 1. Continuous time/resource trade-off in each mode of an activity

$d_{im_i}(\underline{r}_{im_i1}+\Delta r_{im_i1},...,\underline{r}_{im_iK}+\Delta r_{im_iK_m})$ may be a linear or a non-linear function and can be obtained by regression as follows:

$$d_{im_i}=\bar{d}_{im_i}+(-\beta_{im_i})\Delta r_{im_ik} \quad . \qquad i=1,....,n \; m_i=1,...,M_i \; k\in m_i \tag{39}$$

In which β_{im_i} is the non-positive slope of the linear approximation of activity duration i in mode m_i, for resource type k and can be calculated simply in the following form as well.

$$\beta_{im_i}=\frac{\bar{d}_{im_i}-\underline{d}_{im_i}}{\bar{r}_{im_ik}-\underline{r}_{im_ik}}. \qquad i=1,....,n \qquad m_i=1,...,M_i \; k\in m_i \tag{40}$$

In the situation involving two types resources different from the other types in each mode ($k_m=2$), multiple regression with two independent variables can be used in the following form:

$$d_{im_i}=\bar{d}_{im_i}+(-\beta_{im_i1})\Delta r_{im_i1}+(-\beta_{im_i2})\Delta r_{im_i2}, \tag{41}$$

in which β_{im_i1} and β_{im_i2} can be found easily using *MATLAB* and having enough observations.

5.4.3. The Objective Function

The objective function can be stated in detail by incorporating concerning items as follows:

$$Minimize \quad \sum_{i=1}^{n}\sum_{m_i}^{M_i}\{\sum_{k\in m_i} c_k\underline{r}_{im_ik}+c_k\Delta r_{im_ik}\}\times v_{im_i}+\bar{P}\max\{0,T-\bar{D}\} \tag{42}$$

where

$$\underline{c}_{im_i}=\sum_{k\in m_i} c_k\underline{r}_{im_ik}, \tag{43}$$

$$\sum_{k\in m_i} c_k\Delta r_{im_ik}=c_{im_i}\times(\bar{d}_{im_i}-d_i)\cdot \tag{44}$$

In which Eq. (1) has been modified by replacing the duration costs with the resource requirements costs. As a special case, when $c_k=\bar{D}=0$ and $\bar{P}=1$, Eq. (42) is converted to Eq. (17). The model is a non-linear programming due to $\Delta r_{im_ik}v_{im_i}$ but in the absence of discrete multi-mode, and using linear regressions such as Eqs. (39) and (41), the problem is converted to a linear programming. In the absence of crashing within a mode the problem reduces to the *MRCPSP-GPR* and the linear model as well.

5.4.4. An Example

Fig.2 shows an example in the form of an *AoN* network for a project with eight real activities, each of which can be executed in two discrete mode and a single type of resources required for each mode. Crashing in each mode can be performed using Eq. (39) by increasing Δr_{im_ik}.

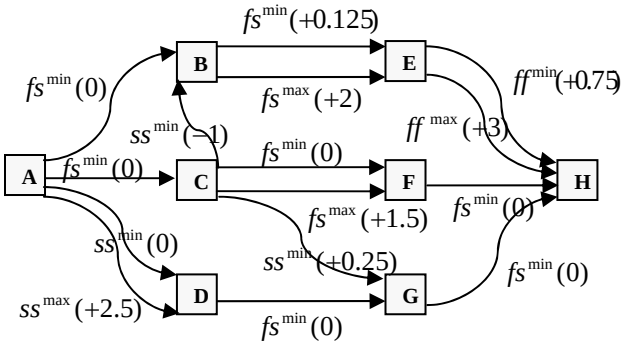

Fig. 2. An example in the form of an AON network

Relevant data of Fig. 2 are shown in Tables 1 and 2.

Table 1. Relevant data of Fig. 2 for mode 1and resource type $k=1$

Activity i	$\underline{r}_{i11}$	$\bar{r}_{i11}$	Resource required r_{i11}	Resource increment $0\le\Delta r_{i11}\le\bar{r}_{i11}-\underline{r}_{i11}$	$\bar{d}_{i1}$	$\underline{d}_{i1}$	Linear approximation $d_{i1}(\Delta r_{i11})=\bar{d}_{i11}-\beta_{i1}\Delta r_{i11}$
A	1	5	$1+\Delta r_{A11}$	$0\le\Delta r_{A11}\le 4$	9	4	$9-1.25\Delta r_{A11}$
B	1	2	$1+\Delta r_{B11}$	$0\le\Delta r_{B11}\le 1$	4	2.4	$4-1.60\Delta r_{B11}$
C	1	1	$1+\Delta r_{C11}$	$\Delta r_{C11}=0$	3	3	$3-0.00\Delta r_{C11}$
D	1	3	$1+\Delta r_{D11}$	$0\le\Delta r_{D11}\le 2$	4	2	$4-1.00\Delta r_{D11}$
E	1	3	$1+\Delta r_{E11}$	$0\le\Delta r_{E11}\le 2$	9	1	$9-4.00\Delta r_{E11}$
F	1	2	$1+\Delta r_{F11}$	$0\le\Delta r_{F11}\le 1$	4	3	$4-1.00\Delta r_{F11}$
G	1	2	$1+\Delta r_{G11}$	$0\le\Delta r_{G11}\le 1$	4	1	$4-3.00\Delta r_{G11}$
H	1	6	$1+\Delta r_{H11}$	$0\le\Delta r_{H11}\le 5$	9	1.5	$9-1.50\Delta r_{H11}$

Table 2. Relevant data of Fig. 2 for mode 2 and resource type $k = 2$

Activity i	$\underline{r}_{i22}$	$\bar{r}_{i22}$	Resource required r_{i22}	Resource increment $0 \le \Delta r_{i22} \le \bar{r}_{i22} - \underline{r}_{i22}$	$\bar{d}_{i2}$	$\underline{d}_{i2}$	Linear approximation $d_{i2}(\Delta r_{i22}) = \bar{d}_{i2} - \beta_{i2}\Delta r_{i22}$
A	1	3	$1+\Delta r_{A22}$	$0 \le \Delta r_{A22} \le 2$	6	2.45	$6-1.78\Delta r_{A22}$
B	0.5	1.5	$0.5+\Delta r_{B22}$	$0 \le \Delta r_{B22} \le 1$	2	1	$2-1.00\Delta r_{B22}$
C	0.75	2	$0.75+\Delta r_{C22}$	$0 \le \Delta r_{C22} \le 1.25$	4	1.5	$4-2.00\Delta r_{C22}$
D	1	3.25	$1+\Delta r_{D22}$	$0 \le \Delta r_{D22} \le 2.25$	6	1.85	$6-1.85\Delta r_{D22}$
E	0.25	1.75	$0.25+\Delta r_{E22}$	$0 \le \Delta r_{E22} \le 1.5$	12	1.71	$12-6.85\Delta r_{E22}$
F	1.5	2	$1.5+\Delta r_{F22}$	$0 \le \Delta r_{F22} \le 0.5$	2.5	2	$2.5-1.00\Delta r_{F22}$
G	1	1.5	$1+\Delta r_{G22}$	$0 \le \Delta r_{G22} \le 0.5$	5	3.83	$5-2.34\Delta r_{G22}$
H	1	3.25	$1+\Delta r_{H22}$	$0 \le \Delta r_{H22} \le 2.25$	5	1.54	$5-1.54\Delta r_{H22}$

shows one resource will be working half-time addition to 5 full-time. Moreover, assume $c_1 = c_2 = 1, \bar{D} = 0$ and $\bar{P} = 1000$.The problem has been solved using *LINGO 8*. Table 3 shows the amount of important decision variables. Assume the availability of the renewable resource of type 1 and 2 are $R_1 = 5.5$ and $R_2 = 3.25$ respectively, in which decimal part of the number, means the level of part-time effort as a percentage that this resource will be working on the project in a day. Accordingly, 0.50 or 50% (out of 5.5)

Table 3. Optimum solution of example 1

Activity i	v_{i1}	v_{i2}	x_i	d_i	y_i	r_{i11}	Δr_{i11}	z_i	r_{i22}	Δr_{i22}
A	0	1	0.0000	2.4500	------	------	------	0.0000	3.0000	2.0000
B	1	0	2.4500	3.0769	3.9230	1.5769	0.5769	------	------	------
C	1	0	2.5269	3.0000	2.9230	1.0000	0.0000	------	------	------
D	1	0	2.4500	2.0769	0.0000	2.9230	1.9230	------	------	------
E	0	1	5.6519	2.2907	------	------	------	1.5842	1.6657	1.4157
F	0	1	5.5269	2.4157	------	------	------	0.0000	1.5842	0.0842
G	1	0	4.5269	1.0000	0.9230	2.0000	1.0000	------	------	------
H	0	1	7.9427	1.5400	------	------	------	0.0000	3.2500	2.2500

$T = 9.48$

6. Genetic Algorithm Implementation

6.1. Chromosome Representation

The first step in the proposed GA is to consider a chromosome representation or solution structure as shown in Figure 3. This structure is severely depends on nature of model decision variable(s) and constraints. According to the proposed model, each solution is represented by a matrix of size $(K+1) \times n$, where *n* is number of activities. The associated chromosome for the example project is shown in Figure 4. For example, activity "E" must be down by mode 1 and $\Delta r_{E1} = 2$. Therefore, the activity duration "E" is equal to 1.

	1	2	3	…	*n*
Mode, m_i	m_1	m_2	m_3	⋯	m_n
r_{im_i1} or Δr_{im_i1}	r_{1m_11}	r_{2m_21}	r_{3m_31}	⋯	r_{nm_n1}
:	:	:	:	:	:
r_{im_iK} or Δr_{im_iK}	r_{1m_1K}	r_{2m_2K}	r_{3m_3K}	⋯	r_{nm_nK}

Fig. 3. Chromosome representation

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
1	1	2	2	1	1	2	2
2.5	0.3	0	1.2	2	0.5	0.7	5
2	1	0.7	2	0.5	0.3	0.4	2

Fig. 4. A typical Chromosome for example project given in Figure 2

6.2. Genetic Operators

For exploring of solution space, we use the single and double point crossovers and two specialized versions of mutation named 'Mode Mutation' (MM) and 'Resource Mutation' (RM). Mutation operators enable the proposed GA to explore the search space which can not be reachable by the crossover operators [11].

6.2.1. Single Point Crossover

It is the same classical crossover in which two selected parents are recombined by a single cross point. An instance of single point crossover implementation for the example of Figure 2 is shown in Figure 5. The bold line is cross point.

6.2.2. Double Points Crossover

It is an extended version of single point crossover in which two selected parents are recombined by two cross points. Figure 6 shows an instance of double point crossover implementation in which offspring 1 and 2 are created when information related to activities "D", "E", and "F" is swapped between parents 1 and 2.

Parent 1

A	B	C	D	E	F	G	H
1	1	2	2	1	1	2	2
2.5	0.3	0	1.2	2	0.5	0.7	5
2	1	0.7	2	0.5	0.3	0.4	2

Parent 2

A	B	C	D	E	F	G	H
2	2	1	1	2	2	1	1
4	1	0	2	1.5	1	0.3	3
1.5	0.5	1.2	1.5	1.1	0.4	0.2	0.7

Offspring 1

A	B	C	D	E	F	G	H
1	1	2	***1***	***2***	***2***	***1***	***1***
2.5	0.3	0	***2***	***1.5***	***1***	***0.3***	***3***
2	1	0.7	***1.5***	***1.1***	***0.4***	***0.2***	***0.7***

Offspring 2

A	B	C	D	E	F	G	H
2	2	1	***2***	***1***	***1***	***2***	***2***
4	1	0	***1.2***	***2***	***0.5***	***0.7***	***5***
1.5	0.5	1.2	***2***	***0.5***	***0.3***	***0.4***	***2***

Fig 5. Implementing single point crossover

Parent 1

A	B	C	D	E	F	G	H
1	1	2	2	1	1	2	2
2.5	0.3	0	1.2	2	0.5	0.7	5
2	1	0.7	2	0.5	0.3	0.4	2

Parent 2

A	B	C	D	E	F	G	H
2	2	1	1	2	2	1	1
4	1	0	2	1.5	1	0.3	3
1.5	0.5	1.2	1.5	1.1	0.4	0.2	0.7

Offspring 1

A	B	C	D	E	F	G	H
1	1	2	1	2	2	2	2
2.5	0.3	0	2	1.5	1	0.7	5
2	1	0.7	1.5	1.1	0.4	0.4	2

Offspring 2

A	B	C	D	E	F	G	H
2	2	1	2	1	1	1	1
4	1	0	1.2	2	0.5	0.3	3
1.5	0.5	1.2	2	0.5	0.3	0.2	0.7

Fig 6. Implementing double point crossover

6.2.3. Mode Mutation

It is implemented on the first row of chromosome as follows: an activity is randomly selected then the current mode is randomly changed into other mode (i.e., $1,\ldots,M_i$). In Figure 7, the mode assigned to activity "D" is changed into 2.

6.2.4. Resource Mutation

It is implemented on the other rows of chromosome as follows: an activity is randomly selected then $\Delta r_{im_i k}$ in the current solution is randomly changed between $0 \le \Delta r_{im_i k} \le \bar{r}_{im_i k} - \underline{r}_{im_i k}$. In Figure 7, Δr_{D11} and Δr_{D22} are changed.

Mode Mutation

A	B	C	D	E	F	G	H
1	1	2	2	1	1	2	2
2.5	0.3	0	1.2	2	0.5	0.7	5
2	1	0.7	2	0.5	0.3	0.4	2

Offspring

A	B	C	D	E	F	G	H
1	1	2	***1***	1	1	2	2
2.5	0.3	0	1.2	2	0.5	0.7	5
2	1	0.7	2	0.5	0.3	0.4	2

Resource Mutation

A	B	C	D	E	F	G	H
2	2	1	1	2	2	1	1
4	1	0	2	1.5	1	0.3	3
1.5	0.5	1.2	1.5	1.1	0.4	0.2	0.7

Offspring

A	B	C	D	E	F	G	H
2	2	1	1	2	2	1	1
4	1	0	1	1.5	1	0.3	3
1.5	0.5	1.2	1.5	1.1	0.1	0.2	0.7

Fig 7. Implementing mutation

6.3. Selection Strategy

The selection strategy means that how to choose the individuals in the current population that will create offspring for the next generation. Generally, it is better that best solutions in the current generation selected as parents for creating offspring. The most common method for the selection mechanism is the "roulette wheel" sampling, in which each individual is assigned a slice of a circular "roulette wheel" and the size of the slice is proportional to the individual's fitness.
The wheel is spun *Pop_Size* times, where *Pop_Size* is the number of individuals in the population. On each spin, the individual under the wheel's marker is selected to be in the pool of parents for the next generation. This method can be performed as follows:

a. Let *CF* be the sum of the *Cost Function* of all solutions in the current population as follows:

$$CF = \sum_{p=1}^{Pop\_Size} (cf_p) \tag{45}$$

b. Where cf_p is the cost function or fitness values of solution *p* and *Pop_Size* is equal to the number of chromosomes in the population.

c. Let ρ_p be the relative probability related to chromosome *p* as follows:

$$\rho_p = \frac{cf_p}{CF} \qquad p = 1,..., Pop\_Size \tag{46}$$

d. Let P_p be the cumulative probability related to chromosome *p* as follows:

$$P_p = \sum_{j=1}^{p} \rho_j, \qquad p = 1,..., Pop\_Size \tag{47}$$

e. Generate a random number, say *r*, in the range of [0, 1]. If $r < P_1$, then the first chromosome is selected. Otherwise, the p^{th} chromosome is selected where,

$$P_{p-1} < r < P_p, \qquad 2 \le p \le Pop\_Size \tag{48}$$

The fitness of each solution is directly obtained by Eq. (42). The initial population is randomly created in terms of a continuous uniform distribution.

6.4. Stoppage Conditions

We use three criteria for stopping the proposed GA as follows: 1) the maximum number of the established generation, 2) an arbitrary minimum value of variance of the last generation, and 3) the maximum run time. The algorithm is finished if one of the three mentioned criteria satisfied.
The variance of each generation can be calculated as follow:

$$\sigma_g^{\,2} = \frac{1}{Pop\_Size} \sum_{p=1}^{Pop\_Size} (CF_{gp} - \overline{CF}_g)^2 \tag{49}$$

where CF_{gp} is the fitness of p^{th} chromosome in generation *g*. $\overline{CF}_g$ is average fitness of all chromosomes in generation *g* that is calculated as follow:

$$\overline{CF}_g = \frac{1}{Pop\_Size} \sum_{p=1}^{Pop\_Size} CF_{gp} \tag{50}$$

7. Computational Results

In order to show that the model serves to solve instances of practical size, ProGen/max [14] is used to generate 100 *RCPSPMCM-GPR* instances in 20 categories. Table 4 shows the control parameters used for ProGen/max to generate instances of the *MRCPSP-GPR* and Table 5 gives the necessary adaptations for obtaining instances of *RCPSPMCM-GPR.*
The *Order Strength* ***OS*** is a [0, 1]-normalized measure defined as the number of precedence relations, which is minimum for parallel and maximum for series digraphs [14]. The *Resource Factor* ***RF*** reflects the average portion of resources requested by each activity [14].
Setting ***RF*** at 1 leads to the most complex resource-constrained project scheduling problem instances. The *Resource Strength* ***RS*** measures the scarcity of the resource availability to the respective requirements [14].

Table 4. The parameter settings for instances of *MRCPSP-GPR*

Symbol	Important Control Parameter	Value
N	number of non-dummy activities	10,20,30,40,50
M_i	number of Modes per activity	2, 3
d_{im_i}	duration of each mode	[5, 10]
	number of initial and terminal activities	[2, 3]
.	maximum number of predecessors and successors	3
OS	order strength	0.5
	percentage of backward arcs (maximal time lags)	20%
	degree of redundency	0.0
	number of cycle structures	[1, 3]
	number of activities per cycle structure	[1, 5]
	coefficient of cycle structure density	0.3
	deviations of minimal time lags from duration	0.5
	tightness of maximal time lags	0.5
	slack factor	0.0
K	number of renewable resource types	2, 3, 4, 6
$r_{im.k}$	renewable resource demand	[1, 5]

The *degree of redundancy* is computed by dividing the number of redundant arcs in the network by their theoretical maximal value.
The *coefficient of cycle structure density* is a measure of the amount of precedence relations in a cycle structure. The *tightness of maximal time lags* determines how the values of the maximal time lags relate to their theoretical minimal value which results in time- and resource feasibility.
The *slack factor* determines how the minimal time lags depend on the activity modes [14]. Upper bound for resource demand of type k can be obtained using the following equation:

$$\bar{r}_{im_ik} = \underline{r}_{im_ik} + ran_i \quad (58)$$

In order to find $\underline{d}_{im_i}$, Let: w_{im_i} be the amount of total work required to complete activity i in mode m_i and can be stated in the following form when *Microsoft office Project* is used [15].

$$w_{im_i} = d_{im_i} \sum_{k=1}^{K_m} r_{im_ik} \quad (59)$$

Table 5. Adaptations of the data to obtain instances of *RCPSPMCM-GPR*

Symbol	Important Control Parameter	Value
K_m	number of resources types required for each mode different from other modes	1, 2
$\bar{d}_{im_i} = d_{im_i}$	normal/maximum duration of each mode	[5, 10]
$\underline{r}_{im_ik} = r_{im_ik}$	lower bound renewable resource demand	[1, 5]
ran_i	a random real number for range between lower and upper bound of resource demand	[0, 3]
$\bar{D}$	due date of project for instances	0
$\bar{P}$	per period penalty cost of project	100

Assume w_{im_i} is a fixed constant obtained by replacing d_{im_i} and r_{im_ik} with $\bar{d}_{im_i}$ and $\underline{r}_{im_i}$ respectively. Accordingly, $\underline{d}_{im_i}$ can be calculated by the following equation replacing r_{im_ik} with $\bar{r}_{im_ik}$.

$$d_{im_i} = w_{im_i} \Big/ \sum_{k=1}^{K_m} r_{im_ik} \quad (60)$$

For the instances of $K_m = 1$, β_{im_i} is calculated simply by Eq. (40). In the situations involving $K_m \geq 2$ which mandate more than two observations, the Sufficient observations can be obtained by generating a few random numbers for r_{im_ik}, between $\underline{r}_{im_i}$ and $\bar{r}_{im_ik}$, and then calculate d_{im_i} by Eq. (60). At this time a software such as *MINITAB* or *MATLAB* can be used to obtain the relevant regression equations. In this paper we have used Eq. (41) for $K_m = 2$.

Since the CPU time for solving a mathematical model depends on the number of constraints and variables, to reduce the size of the experiment, we report only on important parameters influencing solution time.
We fixed the other parameters which usually influence on the feasible area of the model at specific values rather than varying them over the entire range of complexity. Consequently, the instances are categorized according to the combinations of N *={10,20,30,40,50}* and (M_i, K_m, K) *={(2,1,2), (3,1,3), (2,2,4), (3,2,6)}.*
For each category (out of 20), 5 instances have been generated for a total of 100. The proposed GA has been coded in the MATLAB using fmincon(…) and ga(…) functions.
In order to have some benchmarks, the instances have been optimally solved by the Lingo 8.0 software using branch-and-bound (*B&B*) method as well. Each problem is allowed a maximum of 1000 seconds of CPU time. All the computational experiments have been carried out on an intel® Celeron® mobile 1.3 GHz Personal Computer with 512 Mb RAM. Tables 6 and 7 summarize our findings as average CPU times for B&B and GA.

Table 6. The average CPU time for solving five instances in each category

Number of	$(M_i = 2, K_m = 1, K = 2)$			$(M_i = 3, K_m = 1, K = 3)$		
Activities	B&B	GA	O.F. Gap %	B&B	GA	O.F. Gap %
10	59.75	17.21	13%	76.95	18.91	19%
20	385.14	51.82	11%	435.38	50.24	23%
30	>1000	167.60	--	>1000	196.32	--
40	>1000	383.04	--	>1000	372.75	--
50	>1000	925.22	--	>1000	951.30	--

Table 7. The average CPU time for solving five instances in each category

Number of	$(M_i = 2, K_m = 2, K = 4)$			$(M_i = 3, K_m = 2, K = 6)$		
Activities	B&B	GA	O.F. Gap %	B&B	GA	O.F. Gap %
10	143.19	26.76	17%	311.45	31.08	28%
20	>1000	98.91	--	>1000	123.31	--
30	>1000	268.04	--	>1000	259.70	--
40	>1000	681.38	--	>1000	701.15	--
50	>1000	>1000	--	>1000	>1000	--

The values below the columns of B&B and GA are the CPU time in second. The quality of GA in compare to B&B has been measured by O.F. Gap% which is the difference between objective functions of two methods. By increasing the number of activities, B&B can not reach to the optimal solution in an acceptable amount of time. The proposed GA could reach to near optimal solutions in shorter time respect to B&B method. As indicated in Tables 6 and 7, the solution time of proposed model is very sensitive to K_m ,*K* and *N*.

8. Conclusions

In this paper we considered the multi-mode resource constrained project scheduling problem with generalized precedence relations in which the duration of an activity is determined by a discrete mode selection and a *continuous* function of the resource requirements within the selected mode. This problem called *RCPSPMCM-GPR* is general case of the problem *MRCPSP-GPR*. In the absence of discrete multi-mode, the problem reduces to the continuous *Time/Cost* or *Time/Resource* Trade-off Problem. In the absence of crashing within a mode, the problem reduces to the *MRCPSP-GPR*.

Although, considering real numbers for durations and resources increases the complexity of the problem, it allows flexibility to achieve an active pattern of resource usage over time as well as a shorter make span. Firstly, we presented an exact model for the *RCPSP-GPR* and *MRCPSP-GPR*. Secondly, the model was developed for optimally solving the problem *RCPSPMCM-GPR*. Thirdly, the genetic algorithm was used to solve the proposed nonlinear model. We modified the objective function by replacing the duration costs with the resource requirements costs.

The proposed model has no need for a feasible solution to startup with. No formulation as an exact solution has been reported for the *RCPSPMCM-GPR* except the one presented in this paper. Furthermore, time horizon can be continuous in this model thus dealing with different processing time units (e.g., hours, days, weeks, and so on), is possible.

Future efforts will be devoted to the application of the other meta heuristic approaches for solving the nonlinear model in the situations involving instances of practical size (more than 100 activities) in a reasonable amount of time.

References

[1] Ahn, T., & Erenguc, SS., *"The resource constrained project scheduling problem with multiple crash able modes"*: A heuristic procedure. European Journal of Operational Research, 1998, 107: PP. 250-259.

[2] Baptiste, P., & Demassey, S., *"Tight LP bounds for resource constrained project scheduling*". OR Spectrum 26, 2004, PP. 251-262.

[3] Bartuch, M., Möhring, R.H., & Radermacher, F.J, *"Scheduling project networks with resource constraints and time windows*". Annals of Operations Research 16, 1988, PP. 201-240.

[4] Brucker, P., & Knust, S., *"A linear programming and constraint propagation-based lower bound for RCPSP"*. European Journal of Operational Research 127, 1988, PP. 355-362.

[5] Chen, H D., & Xu, R., *"A new heuristic algorithm for rectangle packing"*, Computers & Operation Research, 2006, Article in press.

[6] Deckro, R.F., & Hebert, J.E., "*Modeling diminishing returns in project resource planning*". Computers and Industrial Planning 44, 2002, PP. 19-33.

[7] Demeulemeester, E.L., & Herroelen, W.S., **"***Project Scheduling, A Research Handbook"*. Kluwer Academic Publishers, 2002.

[8] De Reyck, B., & Herroelen, W., *"The multi-mode resource-constrained project scheduling problem with generalized precedence relations*". European Journal of Operational Research 119, 1999, PP. 538-556

[9] Dorndorf, U., *"Project scheduling with time windows*: *from theory to application"*. Physica-Verlag Heidelberg New York, 2002.

[10] Elmaghraby, S.E., *"Activity nets: A guided tour through some recent developments*". European Journal of Operational Research 82, 1995, PP. 383-408.

[11] Haupt, R.L., & Haupt, S.E., "*Practical genetic*

algorithms, John Wiley & Sons, 2004.

[12] Hartmann, S., & Drexl, A., *"Project scheduling with multiple modes: A comparison of exact algorithms"*. Networks 32, 1998, PP. 283-298.

[13] Heilmann, R., *"A branch-and-bound procedure for the multi-mode resource constrained project scheduling problem with minimum and maximum time lags*". European Journal of Operational Research 144, 2003, PP. 348-365.

[14] Kolisch, R., Sprecher, A., & Drexl, A., *"Characterization and generation of a general class of resource-constrained project scheduling problems"*. Management Science 41, 1995, PP. 1693-1703.

[15] Marmel, E., *Microsoft office project 2003 Bible*, Wiley Publishing, Inc, 2004.

[16] Neumann, K., Schwindt, C., & Zimmermann, J., "*Project scheduling with time windows and scarce resources"*. Springer, Berlin, 2003.

[17] Pritsker, A.A.B., Watters, L.J., & Wolfe, P.M., "*Multi-project scheduling with limited resources: A zero-one programming approach"*. Management Science 16, 1969, PP. 93-108.

[18] Seyed-Hosseini, S.M., & Sabzehparvar, M., *"A geometrical model for the multi-mode resource constrained project scheduling problem"*. Business Research Yearbook of conference for International Academy of Business Disciplines Florida USA Vol. XIV 1, 2007, PP. 149-155.

[19] Sprecher, A., Drexl, A., *Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm*. European journal of operational research 107, 1998, PP. 431–450.

[20] Talbot, F.B., *"Resource-constrained project scheduling problem with time-resource trade-offs: The non preemptive case"*. Management Science 28, 1982, PP. 1197-1210.