Available online at http://jnrm.srbiau.ac.ir

Vol.1, No.1, Spring 2015

Journal of New Researches in Mathematics

Science and Research Branch (IAU)

Uncertain Resource Availabilities: Proactive and Reactive Procedures for Preemptive Resource Constrained project Scheduling Problem

M.Fallah *

Department of Industrial Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran.

Received Spring 2015, Accepted Summer 2015

Abstract

Project scheduling is the part of project management that deals with determining when in time to start (and finish) which activities and with the allocation of scarce resources to the project activities. In practice, virtually all project managers are confronted with resource scarceness. In such cases, the Resource-Constrained Project Scheduling Problem (RCPSP) arises. This optimization problem has become popular over the last few decades because of its practical relevance to various industrial and research fields. Numerous procedures have been developed in the literature for finding either optimal or heuristic solutions for the RCPSP.

Resource constrained project scheduling problem under stochastic circumstances have been considered in recent decades where uncertainty is modeled by means of activities' duration. In this research we study an extension of basic stochastic RCPSP in which availability of resources is not predefined. Activities follow preempt-resume mode in case of any disruption due to resource infeasibility. A solution for this variant of RCPSP is defined in three steps by utilizing heuristic procedures for creating initial schedule, adding time and resource buffers in order to yield proactive schedule and applying reactive policies to encounter with unhandled breakdowns. Computational experiments depict that proposed combined procedure achieve significant performance gains over the use of each method separately.

Keywords: *Stochastic Resource Availabilities, Proactive Procedures, Reactive Policies*

* Email:mohammad.fallah43@yahoo.com

1 Introduction

Exploring a baseline schedule for list of activities under resource constrainted called RCPSP attracts attention of many researches in recent decades. Most of procedures including exact and heuristic ones are developed in a static and deterministic environment. While cases in which complete information regarding activities' duration, resource availabilities and resource usages are known in advance are rarely happen in practice. Nevertheless, variety of objective functions such as minimizing project make-span, maximizing project net present value, minimizing project earliness-tardiness costs and leveling resource usage over time are introduced in literature which could be recalled in stochastic environment, for a complete review we refer to Herroelenet al. (1998), Brucker et al. (1999) and Demeulemeester and Herroelen (2002).

As mentioned, even for common (repeated) projects or in low risk environment it is seldom to acquire comprehensive data about all aspects of project. So, it is vital to develop algorithms in order to reach a feasible schedule where activities and/or resources are subject to unknown risk during project execution. The inherent uncertainty in any surroundings can commence from great number of potential sources (Yu and Qi 2004; Wang 2004). Among many causes, we can mention resource unavailability, activities' completion time delay and late material arrival. Uncertainty in activities' duration is also studied in several researches (Leus2003 and Van de Vonder et al. 2005).In this paper we study the first of these possible causes. This variant of stochastic RCPSP is initially studied by Lamberachts et al. (2008) using preempt-repeat mode. Therefore, project should be protected against any disruption to pursue baseline schedule. In reality, unhandled disruptions which are not absorbed by means of protection could affect starting time of activities and impose *instability costs* to project. It is defined as absolute value of deviation between realized starting time of activity *i* with planned one weighed by corresponding penalty cost w_i. Lamberachts et al. (2008) stress that penalty costs can be seen as extra costs

for changing subcontracts to start by a given delay or for holding inventory in warehouses between delivery time and starting time of the activity.

Applying instability cost to a project gives an alternative to consider minimizing weighted earliness-tardiness as an objective function. In addition, it measures quality robustness (Herroelen and Leus 2003) of a schedule, defined as stability of solution's quality for a wide range of possible execution scenarios. Meeting deadline of project could be inspired by assign higher costs to dummy end activity. So, an optimal solution having maximum quality robustness simultaneously minimizes weighted instability cost.

2 Problem statement

Recent survey of the various methods to scheduling under uncertainty is done by Herroelen and Leus(2004, 2005).Based on finding in this field, all provided approaches could be categorized in one of Proactive, Reactive or Predictive-Reactive categories. Proactive procedures are usually proposed based on buffer insertion concept; where buffers makes baseline schedule enrich to cope with adverse effects of disruptions during activities execution time. Indeed, the aim of using proactive approaches is to create robust schedules. The robustness of a schedule can be defined as the ability to absorb small fluctuations in unknown parameters, for instance, an increase in the duration of some activities resulting from uncontrolled factors (Al Fawzan and Haouari 2005) or breakdown of resources (Lamberachts et al. 2008). Despite of content of protection added to schedule, project is subject to breakdown due to unknown risks such as bad weather, depending to the environment context. Thereupon, rather than adding protection to schedule, reactive scheduling make a decision online during project execution whenever an even occur. At any decision point, when an unexpected incident occurs, decision-making process exerts a priority rule to repair or re-optimize the schedule with regards to objective function. Hence, baseline schedule and scenario model is not needed using reactive scheduling.

Besides, some sources refer to predictive-reactive scheduling to emphasize on worth of using combined procedures; Van

de vonder et al. (2005) survey the trade-off between proactive procedures and reactive ones to reach quality and solution robustness. Generally they rely on constructing initial schedule and update it afterwards at required moments. Predictive-reactive scheduling procedures can be distinguished according to the repair mode and the way the initial baseline schedule is repaired, for instance continuously or periodically by means of local schedule adaptation (Smith et al. 1995), total rescheduling(Snoek 2001), partial rescheduling (Sabuncuoglu andBayiz 2000) or match-up rescheduling (Sakkout et al. 1997;Akturk and Gorgulu 1999).

Following the classical resource constrained project scheduling problem (RCPSP) we can find out that proactive procedures are an extension of it. Based on notation developed by Herroelen et al. (2000), conceptual model for basic RCPSP can be expressed as follows:

$$\text{Minimize} \quad s_n$$

$$\text{subject to} \quad .$$

$$s_i + d_i \le s_j, \qquad \forall i, j \in A$$

$$\sum_{i:i\in I_t} r_{ik} \le a_k, \qquad \forall t, \forall k$$

An instance of RCPSP consists of a set of activities represented as $N = 0,\ldots,n$ with known durations $d_i \in \mathrm{N}$ for each activity $i \in N$. Where 0 and n are dummy start and end activities with zero duration and resource usage. During project execution each activity $i \in N$ requires a predefined amount r_{ik} of renewable resource k whenever it is active. An activity i is called active in period t if it is under execution. The availability of each resource type $k \in K$ during each period of time is depicted by a_k. Note that in the main problem studied in this text, these availabilities are considered as random variables. A set of activity pairs $i, j \in A$ are imposed to define *finish-start* precedence constraints in project. Which means activity j can start if only its predecessor i is finished. This set A is called *precedence relation* if we assume that it is an *order relation* on N implying as irreflexive and transitive relation. As mentioned, in this study we dropped the assumption of deterministic values for resource availabilities, interpreting resource are subject to wear and tear during project execution and on-hand amount of resources is function of resource characteristics $a_{kt} = \underset{k\in K}{f}\, a_k$ at any

time point t. Furthermore, in line with Lamberachts et al. (2008) we have also utilized *minimum weighted deviation* as an objective function; the focal point of exerting such a function is that in reality project managers are looking for a schedule which deviates as least as possible from the initial baseline schedule, therefore we have:

$$[1] \qquad \text{minimize} \sum_{i \in N} w_i \left| E s_i - s_i \right|$$

A solution to basic RCPSP is a vector of activities' starting time $\mathrm{s} = s_0, s_1, \ldots, s_n$, called schedule. In any schedule s_0 and s_n represent start and completion time of project respectively. Regarding to complexity of deterministic RCPSP which is known as NP-hard, many heuristics approached are developed to deal with it in literature [Kolisch and Hartmann 1999, 2006]. However, when uncertainty comes into play solutions under deterministic circumstances are not longer valid. Therefore, numerous approaches are developed to cope with stochastic RCPSP. Most of researches are focused on inherent uncertainty of activities' duration and established *scheduling policies* [Igelmund and Radermacher 1983, Fernandez et al. 1998, Stork 2001]. Nevertheless, there are few studies on RCPSP for deal with other variant of uncertainties such as stochastic availability of resources which is an adverse effect of unknown risks. Mehta and Uzsoy (1999) have studied machines randomly breakdowns for the single machine scheduling, similarly it is done before [Mehta and Uzsoy 1998] for case of job-shop scheduling. However, in field of project scheduling, as far as we know, there is no research except for Drezet (2005) and Lamberachts et al. (2008) for the stochastic resource availability. Human resource variability and limitation such as competence, a limit on the number of hours an employee works per day, vacation periods and unavailability of employees are studied by Drezet (2005). On the other hand, Lamberachts et al. (2008) develop a threefold procedure to reach a robust solution. Time buffer, resource buffer as well as reactive policies are employed to create a proactive schedule which is empowered by reactive procedure. In this article, we study a new variant of stochastic resource constrained problem in which availability

of resources are stochastic and activities are allowed to be done in preemptive mode. Where the efforts done before infeasibility due to resource breakdowns are not wasted. So, after resolving resource conflicts task can, therefore, start by the point it is deactivated. The concept of time buffer, resource buffer and reactive procedures are adapted to use in studied scheduling problem. The focal points of this article are twofold:(1) We propose a new methods to establish and enrich initial schedule for project scheduling with stochastic resource availabilities, extending the existing work of Lamberachts et al. (2008). (2) We develop a reactive procedure to cope with unknown risks make project delays and/or deviates from baseline schedule. In the process, we underline the value of activity-based policies, achieved by generating an order list of activities in each infeasibility point and revise it consecutively in future points, if it is necessary. The remainder of this paper is organized as follows: Sect.3 is devoted to initial schedule constructing approaches. Proactive strategies in way of time and resource together with their definition are provided in Sect. 4. Section 5 outlines reactive procedure and our search method to find out a suitable order list. Extensive computational results are reported in Sect. 6. A summary and some conclusions are given in Sect. 7.

3Constructing initial schedule

The approaches in this section are firstly introduced and then tested by means of a sample project network given in Fig.1. Based on the illustrated graph, project contains 7 activities where activity 0 and 6 are dummy activities. Relating to *Activity On Node* representation of project, start of project is shown by activity 0, whereas completion time of project is given by activity 6, both have zero duration and resource usage. For ease of illustration just one renewable resource is considered for sample project with availability of 5 during each period of time. Due date (deadline) of project is 12 noted by d. Below each activity, its duration, resource usage and instability cost are indicated. Note that instability cost of dummy end activity (16 for Fig. 1) should be much larger than other activities in order to show the importance of meeting project deadline, in practice.

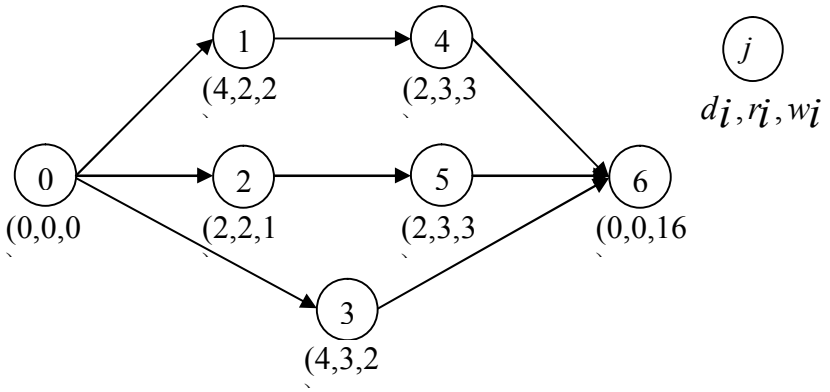


Figure.1 Sample project with 5 non-dummy activities

As discussed, a solution for RCPSP under resource uncertainties consists of three parts; in which firstly we need to have an initial schedule. In literature, Lamberachts et al. (2008) studied two main approaches, *Optimal solution* and *Cumulative Instability Weight* (CIW), in order to generate baseline schedule. Due to objective function of optimal solution which is minimal makespan, they find out that optimal solution gives a tight schedule with minimum flexibility while schedule created by CIW has more adaptable schedule. Thus, in this paper we have just keep CIW for benchmark and develop three heuristic approaches to construct initial schedule. Intensive computational results given in section 6 indicate merits of proposed methods over CIW.

3.1 Cumulative Instability Weight (CIW)

To increase robustness of schedule Lamberachts et al. (2008) suggest using instability weights as a factor to prioritize activities based on precedence relations. CIW factor for each activity is defined as follows:

$$CIW_i = w_i + \sum_{j \in SUCC_i} w_j \qquad [2]$$

Methods proposed in this paper regarding to construction of baseline schedule, have two phases. In first phase, specific factor expressing importance of activities is determined (e.g. CIW_i). A feasible order list is then constructed based on importance of activities, note that feasibility of order list means activity i can be entered to list if all of its predecessors exist in list formerly. Utilizing *serial scheduling scheme* (SGS) an order list is translated to schedule in phase two. The serial scheduling scheme

dates back to a paper by Kelley (1963). SGS adds activities sequentially to schedule until a complete schedule is obtained. In each iteration, regarding to resource and precedence constraints, a selected activity starts in the first possible point of time.

3.2Time-Resource Method (TRM)

Considering just the instability cost of an activity even with costs of its successors is a one dimension factor. In results, other characteristics of any task such as its duration and resource requirements are ignored in CIW. To overcome this shortcoming we introduce a novel approach based on combination of resource usage, duration and an instability cost as follows:

$$TR_i = w_i \times d_i \sum_{k=1}^{K} r_{ik} \qquad [3]$$

Using TRM gives an opportunity to analyst view a RCPSP problem as a puzzle; where we have to put pieces with size of $d_i \times r_{ik}$ into the free space with size of $\delta \times a_k$.

We believe that this view brings new dimensions into decision making process to construct better schedules; which is investigated in computational result. For instance, consider two activities with same CIW, which one will be scheduled according to CIW method? In general, you have to use a tie-breaker rule such as lower activity number. But, if we put into account the time-resource factor, defining by multiplying duration by resource usage, the activity with highest weighted time-resource factor will be selected to schedule earlier to avoid delay in project delivery, in advance.

3.3 Critical Resource Method (CRM)

The requirement of resources is explored as a vital factor to create an order list in TRM. Nevertheless, it has bias in computations with getting the sum of activity's usages over all types of resources. To clarify, suppose activity i and j with resource usages (5, 0) and (2, 2) over resource type I and II with availability of 30 and 2, respectively. If they have the same weight and duration, TRM computes TR factor as $TR_i = 5dw$ and $TR_j = 4dw$; and then activity I is chosen to schedule. In deterministic environment, neither activity i nor j is critical on resource type I which has availability of

30 units per period. While, activity j is crucial on resource type II requiring 2 units; and it is prone to breakdown if only one unit of resource is become out of order. Hence, realizing the importance of resource usage and availability ratio, we introduce *Critical Resource Method* (CRM) which consider maximum of normalized requirements of activities to order them. CR_i is determined as follows:

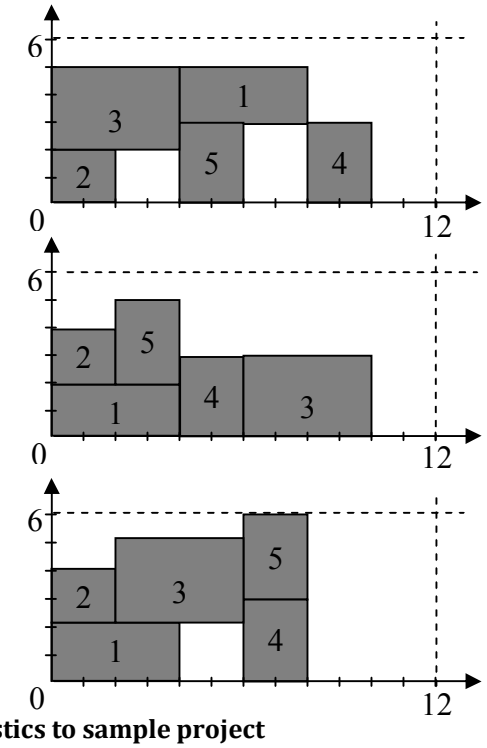
$$[4] \quad CR_i = w_i \times d_i \times \underset{K}{Max} \left[\frac{r_{ik}}{a_k} \right]$$

3.4 Cumulative Critical Resource Method (CCRM)

As a last heuristic procedure we address a cumulative variant of CRM method. In fact, there is no doubt that in graph of project regardless to resource limitation an activity criticality is corresponds to number of its successors whereof a small delay in its completion makes bullwhip effects, firstly introduced by Forrester (1961), which affect release time of project. To keep into account such a dependency of activities, *CCR* factor is defined:

$$CCR_i = w_i \times d_i \times \underset{K}{Max} \left[\frac{r_{ik}}{a_k} \right] + \sum_{j \in SUCC_i} \left[w_j \times d_j \times \underset{K}{Max} \left[\frac{r_{jk}}{a_k} \right] \right] \quad [5]$$

In way of CIW, here, for computing *CRR* factor for activity i we add its successors' *CR* factor to its own *CR* factor.



(a) CRM & TRM initial schedule

(b) CIW initial schedule

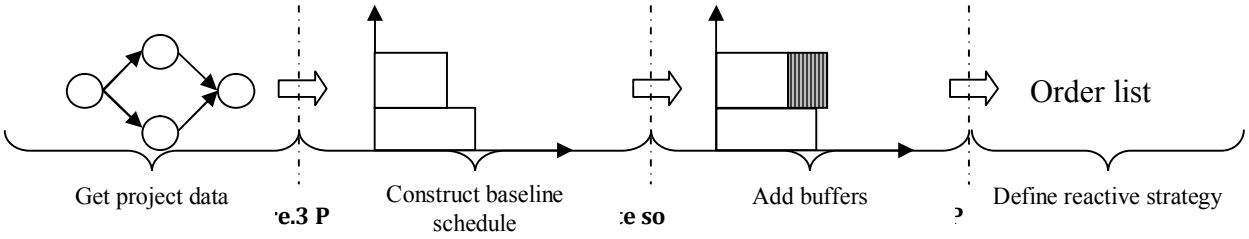
(c) CCRM initial schedule

Figure.2 Applying heuristics to sample project

In a project for which one type resource is required there is no difference between results of CRM and TRM methods; as shown in Fig. 2(a). Application of CIW and CCRM methods are also depicted in Fig. 2(b) and 2(c) respectively.

4 Proactive strategies

To reach robustness in project, proactive strategies add buffers which are commonly defined as time buffers. Proactive strategies in this paper come to play after constructing baseline schedule to assist manager encounter with uncertainties and maintain the schedule. Fig.3 demonstrates the role of such procedures in process of resolving RCPSP problem. In addition to time buffer, a new heuristic method is also developed to empower baseline schedule. Therefore, we have to compare all combination of baseline schedule, proactive procedure and reactive strategy which yields to 3×2^2 composed approaches at all to identify suitable one; three ways for constructing initial schedule and options for adding time and/or resource buffer.



4.1 Resource buffer

In deterministic environment and even in situations where uncertainty is only distinguished in activities' duration, availability of resource(s) is known before project lunch point. However, in this paper we stick to the stochastic RCPSP with uncertain resource availabilities; means each resource's unit is subject to breakdowns in each time period of project execution.

Put differently, it would lead to wrong decisions, in practice, if schedule is constructed based on deterministic resource availabilities. In such settings, using a lower amount a'_k of deterministic resource availabilities a_k for scheduling would be a simple and efficient way to cope with uncertainties. Howbeit, there is no exact procedure to examine $a'_k, k \in K$; it is suggested in researches to use expected value of resource availability

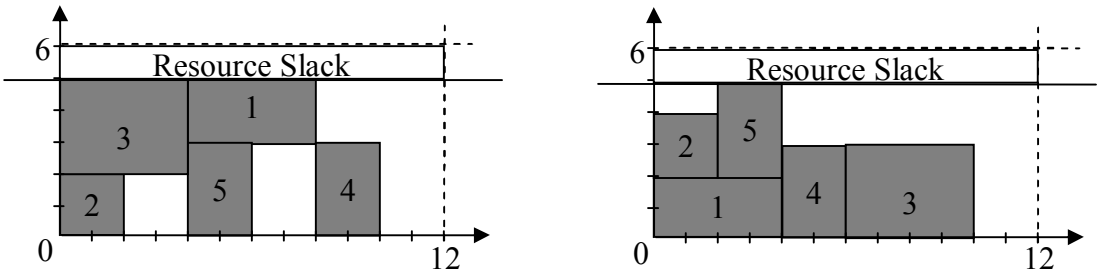
distribution as a'_k. Since expected value of a_k could be determined by given *mean time to failure* (MTTF) and *mean time to repair* (MTTR), utilizing this values would be the easiest way. Nonetheless, how often we can count on the data of MTTF and MTTR,in reality, where resources as well as project are prone to variety of uncertainties. Even, regardless to accuracy of such factors, for a symmetric distributions a probability of getting values over expected value is just 0.5. Avoiding adverse effects of inaccurate data, we express *critical-resource index* η in order to contribute a method to size the resource slack (buffer), where:

$$\eta = \frac{\sum_{i=1}^{n} d_i r_{ik}}{a_k \delta} \qquad [6]$$

Algorithm. 1 Heuristic for sizing resource buffer

1: $F = k_1,\ldots,k_K$
2: **while** $F \neq \phi$ **do**
3: **while** $s_n \leq \delta$ **do**
4: **for** $k = 1$ to K and $k \in F$ **do**
5: compute η_k
6: **end for**
7: **for** $k = 1$ to K **do**
8: $\hat{a}_k = [\eta_k a_k]$
9: **end for**
10: Re-schedule with $\hat{a}_k$
11: **end while**
12: **while** $s_n > \delta$ **do**
13: Identify Critical resource k^*
14:**if** $k^* \in F$ **then**
15: $F = F \setminus k^*$
16:**end if**
17: $\hat{a}_{k^*} = \hat{a}_{k^*} + 1$
18: Re-schedule with $\hat{a}_k$
19: **end while**
20: **end while**

Investigating critical-resource index (CRI) revealed that for any resource we have $0 < \eta_k < 1$. In an extreme point when $\eta_k = 1$, we do not have any resource slack during project execution which itself denote any disruption of resource k violates the due date of project. According to CRI a pseudo code of procedure for sizing the resource buffer is given in Algorithm.1. The procedure initialized by computing η_k for $k = 1,\ldots,K$, then modified resource availabilities are determined by $\hat{a}_k = [\eta_k a_k]$. These steps are repeated sequentially unless due date of project is not violated $s_n \le \delta$. Once the deadline is violated, critical resource will be identified and increased by one unit. This step guarantees that critical resource and its availability will be updated till we reach a feasible schedule. Through the resources, for which, one unit increasing in availability brings the highest reduction of makespan is called critical resource. This procedure continues until no resource's amount could be decreased without deadline violation. Applying the procedure for project of Fig.1 we obtain 0.5 for η which yields to $\hat{a}_k = 3$. While using $\hat{a}_k$ violates project deadline $\delta = 12$, we increase it sequentially to achieve feasible solution. Schedule becomes feasible in $\hat{a}_k = 5$ for which resulting schedule is illustrated in Fig.4. Added value of using multi dimensional procedures like CRM is pointed in Fig. 4(a) where by adding resource slack, free slack is equal to 6; while for CIW in Fig. 4(b) free slack comes down by 8 units and is equal to just 2.



(a) Resource slack on CRM schedule (b) Resource slack on CIW schedule

Figure.4 Adding Resource slack to sample project

4.2 Time buffer

As a common way to protect any schedule, time buffers are more known than other types of buffer. In case of project scheduling time buffers can be used with resource buffers or separately. They define as time slack in front of activities to absorb disruptions caused by resource breakdowns.

Although, some former researches such as theory of constraints [Goldratt 1997] locate the time buffer at the end of project and feeding chains, we opted to add time buffer buf_i before activity i start time s_i. In line with researches done for buffer sizing, we need to estimate the impact of activities' disruption imp_i in order to make proper decisions in increasing or decreasing the size of buffer. To figure out the imp_i, we need to have an image about effects of resource breakdowns on its predecessors' duration.

Lambrechts et al. (2008) express estimation for activity duration increases for *preempt-repeat* circumstances, where efforts done for activities, if they disrupted due to resource breakdowns, are not remain.

So, disrupted activities have to initiate over; whereas we adapted to *preempt-resume* mode in this paper. Following the preempt-resume state, we assume that work done on activities before disruption is not wasted and could be ran on after resource repair.

Allowing preemption for activities, let us to determine real tasks' duration as: $d_i^r = d_i + \tau_i$, where τ_i is resultant of resource breakdowns on activity's duration. For the purpose of determining d_i^r, $E\tau_i$ is essential; therefore we first challenge the assumptions addressed in this field then we depict how calculate $E\tau_i$.

Three main postulates are taken in previous researches; exponential distribution is adjusted for time to failures F_k and for repair times R_k, firstly. According to principles of *factory physics*, it can be interpreted that total amount of breakdowns $N\ t$ follow poison distribution, if we suppose several components are constitute of each resource unit; for detail review we refer to Hopp and Spearman (2001).

Pursuing poison distribution for number of events means that time between two events follows exponential distribution; time between failures in our case.

Secondly, resource allocation is considered fix, clarify that, for instance, if we allocate first unit of resource type-I to activity i, other units in free times would not be able to work on activity when assigned unit is out of order.

However, this assumption brings simplicity to calculation we utilize free allocation rather than fix.

Even, it is reasonable to consider fix allocation in high-tech projects where two units of one resource type are not necessary the same; but in others it wastes time and valuable resources.

Last but not least, we assume that resources are subject to breakdown whether they are in use or idle.

Underline preempt-resume mode, τ_{ik}, activity i duration increase regarding resource k breakdowns, is composed of aggregate repair time; which itself is related to number of disruptions.

On the basis of free resource allocation, probability of activity i preemption is calculated as:

$$\pi_{ik} = \sum_{j=r_{ik}+1}^{a_k} \binom{a_k}{j} \Pr F_k > d_i^{\,j} \times 1 - \Pr F_k > d_i^{\,a_k - j} \qquad [7]$$

In detailed, activity i with resource usage r_{ik} will be preempted certainly if $a_k - r_{ik} + 1$ units are disrupted. So,

probability of occurring x disruption(s) for activity i is:

$$\Pr X_{ik} = x = 1 - \pi_{ik}\ \pi_{ik}^{x} \qquad [8]$$

Then we can write:

$$\tau_{ik} = \sum_{x=0}^{\infty} \left[E\ \tau_{ik} \mid X_{ik} = x \right]$$

$$\Rightarrow = \sum_{x=0}^{\infty} 1 - \pi_{ik}\ \pi_{ik}^{x} \times xE\ R_k$$

Without loss of generality we suppose that repair times are also follow exponential distribution with parameter μ_k. Now we have:

$$\tau_{ik} = \frac{\pi_{ik}}{1 - \pi_{ik}\ \mu_k} \qquad [9]$$

Algorithm. 2 Heuristic for sizing time buffer

1: $E = N$, $b =$ zero vector
2: **for** $i = 1$ to n **do**
3: compute τ_i
4: **end for**
5: **for** $i = 1$ to n **do**
6:compute imp_i
7: **end for**
8:**while** $E \neq \phi$ **do**
9: select an activity i based on P_i form E
10: $b_i = b_i + 1$
11: Re-schedule by considering buffers
12: **if** $s_n > \delta$ **then**
13: $b_i = b_i - 1$
14: $E = E \setminus i$
15: **else**
16: update impact vector $\text{imp} = imp_1, \ldots, imp_{n-1}$
17: **end if**
18: **end while**

Impact of activities now can be computed by assuming τ_i as maximum value of τ_{ik} for $k = 1, \ldots, K$. While this assumption simplifies dependencies through the resources, it gives us a rough idea for resulting activities duration increase which will be employed for identify a suitable activity for buffer insertion. So, impact of activity i can be defined as weighted sum of its predecessors delay:

$$imp_i = \sum_{j \in \text{Prdec}} w_i \text{Max}\{0, s_j + d_j + \tau_j - s_i\} \quad [10]$$

Whenever we obtain input data, procedure of Algorithm.2 can be employed for buffer sizing and insertion. Heuristic procedure initiates with computing value of τ_i for all activities. The impact vector $\text{imp} = imp_1, \ldots, imp_{n-1}$ is then determined using τ_i for $i = 1, \ldots, n$. Afterwards, regret-based random sampling (RBRS), developed by Drexl (1991), is exerted to pick an activity and increase its buffer by one unit. Probabilities P_i according to RBRS are defined by:

$$P_i = \frac{\rho_i + 1^{\beta}}{\sum_{j \in N} \rho_j + 1^{\beta}} \quad [11]$$

In above relation, $\rho_i = \underset{k \in E}{Max}\, v\,k - v\,j$ in which $v\,i$ implies priority values; imp_i in our case. We have also set $\beta = 1$ in heuristic procedure whenever RBRS play role.

5 Reactive strategies

Unknown risks may cause several breakdowns in resources some of which is absorbed by means of time and/or resource buffers. While remaining ones would disrupt the project whenever resource infeasibility occurs. As assistant tool, reactive procedures serve in to maintain schedule. They would be also considered as decision making support system, while activity order list is facilitated utilizing a priority rule in disruptions point. We have engaged Genetic Algorithm, developed by Holland (1975), to explore a suitable activity list when a reactive strategy is called during the project execution. Explicitly, in each step of GA, a number of solutions are evaluated by adopting *Activity-Based policies* (AB policies) which are used in well-known sources from the literature (Ashtiani et al. 2009, Ballestín 2007). For a given order list $L = l_1,..,l_l$, AB policy insets l_j into schedule in earliest possible time considering side constraint $s_{l_i} \le s_{l_j}$ in addition to precedence and resource constraints. Side constraint imposes start-start precedence relation i, j_{ss} into project graph for any activity i appears just before activity j in L. An overview of search procedure is given in Algorithm 3. First of all, initial population is generated called ListSol; NewGen is then constructed, using CrossMut function which applies crossover and mutation operator on ListSol. For each solution L, expressed as an order list, AB policy ABP is applied to create a feasible schedule. To measure the performance of solution, fitness value is determined for it. To ensure that ElectSol contains so for explored best solution, L^* will be replaced by L if L dominates L^*. Finally, $NoIter$ shows overall number of iteration in algorithm and $NoPop$ denotes number of individuals in population. Configuration applied to GA is described in threefold: *individuals and initial population*, *crossover and mutation* and *selection.*

Algorithm. 3 Local search procedure

1:ListSol = initial population
2: ElectSol = L^*, the best solution in ListSol
3: **for** $i = 1$ to $NoIter$ **do**
4:NewGen = CrossMut (ListSol)
5: **for** $j = 1$ to $NoPop$ **do**
6: $L = \text{NewGen}(j)$
7: $S = ABP\ L$
8: Compute the fitness value of L
9: **if** L dominates the worst solution ElectSol **then**
10: ElectSol = L
11: **endif**
12: **endfor**
13: ListSol = Selection(ListSol,NewGen)
14:**end for**
15: Return ElectSol

Individuals and initial population. In GA a population consists of individuals defined in different ways corresponding to problem configuration. We express an individual as precedence feasible activity order list. Resulting schedule could be determined by AB policy which transformed an activity list L' to schedule S'. Fitness value of L' is then calculated by $\sum_{i \in N} w_i \left| s_i' - s_i \right|$. Each individual in our procedure is constructed by employing RBRS defined in 4.2 where P_i is probability of selection an eligible activity, which its predecessors are already inserted to list, and $v\,i$ is activities latest start time.

These steps *Crossover and mutation.* Harttman (1998) extends crossover operators in RCPSP where a solution is defined as list. Among them, we opt two-point cross over to combines a pair of lists into two new lists. For two selected individuals as parents (mother and father) we draw two random integers r_1 and r_2, with $1 \le r_1 < r_2 \le n$. The first r1 positions from them other (father), the positions between r_1 and r_2 are taken from the father (mother), and finally, the

remaining positions are again drawn from the mother (father) to construct daughter (son).In addition to crossover, applying a standard mutation operator (Harttman 1998) gives pportunity to reach better solution by altering activities order. In detail, for all positions $i = 1,\ldots, n-2$ the activities l_i and l_{i+1} are exchanged with a predefined probability p_{mut}.

Selection. Investigation done by Harttman (1998) unveils that simple ranking method gains better result among others such as 2-tournoment selection and proportional. Utilizing simple ranking method allows us to select $NoPop$ best lists form population and consider them as current population for next iteration of GA. In addition to show added value of using our local search procedure to find a proper reactive strategy in disruptions point, an initial order list is used for benchmark. An initial order list is created with the activities in non-increasing order of their starting time.

6Computational results

All procedures and algorithms were coded on Microsoft Visual C++ 2005 and are done on a personal computer with 2,130 MHz clock speed and 1.99 GB RAM. For the sake of evaluation, we have used problem data set J30 of well known PSBLIB (Kolisch&Sprecher 1996) data set. It contains 480 problem instances with 30 non-dummy activities and 4 resource types. While J30 instances include deterministic values, we need to extend it to stochastic environment to contain activity weights, MTTR and MTTF; considering uncertainty in resource availabilities. In line with few researches in this field, we draw activity weights from discrete triangular distribution with $P\,w_i = x = 0.21 - 0.02x$ for $x = 1,\ldots,10$. $\mathrm{MTTF_k}$ and $\mathrm{MTTR_k}$ are drawn from U(1, 5) and U($0.5\,C_{\min}$, $1.5\,C_{\min}$); where U(LB, UB) implies discrete uniform distribution between LB and UB and $C_{\min}$ is minimum makespan derived by optimal solution. Although, we have opted to use a heuristic procedure for sizing resource buffer, MTTF and MTTR are necessary in order to determine resource availability distribution. For a given MTTF and MTTR, a stationary availability of resource is calculated as: $StAvail_k = \dfrac{MTTF_k}{MTTF_k + MTTR_k}$ [12]

Therefore we have:

$$P\ a_k = j = \binom{a_k}{j} A_k^j\ 1 - A_k^{a_k - j} \qquad [13]$$

To examine the performance of a solution we use simulation; through the simulation availability of resources in each period time is calculated by [13]. Optimal solution for each instance is also determined by employing a branch and bound algorithm developed by Demeulemeester&Herroelen (1997) in order to examine objective function $\sum_{i \in N} w_i |s_i' - s_i|$. We have also done two adjustments: firstly to emphasize meeting the deadline of project, weight of end dummy activity is set to 10 times the expected value of weight distribution which is 3.85; secondly to buffer the project deadline and make it robust we set due date of project $1+\alpha$ times of $C_{\min}$, the value of α could be output of stability vs. duration analysis.

Combination of three developed heuristics, buffers and reactive strategies are compared through the simulation with 1000 replications for different values of $\alpha = 0.15, 0.3, 0.45$ and average value of objective function is reported in Tables 1, 2 and 3. Each column in Table 1 is devoted for corresponding part of final solution. Applying time and resource buffers are shown in first and second column respectively. The method for creating initial schedule is itemized is column three. Two last columns illustrate the way we maintain disrupted schedule which would be using initial order list or local search procedure addressed in Algorithm 3.

Investigating results shown in Table 1 makes worth of our local search procedure clear. As any arrangement with GA order list dominates all compositions which involve initial order list as reactive strategies. The merit of both time and resource buffers is realized when regardless of other elements last three rows of Table 1, where both types of buffer are utilized, outperform other ones. In another point of view, analyzing methods for establishment of initial schedule points out performance gains wherever CRM method is used. Understanding value of proactive strategies, Table 2 depicts comparison of methods for $\alpha = 0.3$ while results for $\alpha = 0.45$ are given in Table 3.

			Initial order list	GA order list
no time buf.	no res buf.	TRM	230.78	174.61
		CRM	201.23	158.89
		CCRM	217.58	172.36
	resbuf.	TRM	97.25	66.37
		CRM	78.13	53.62
		CCRM	90.01	61.42
timebuf.	no res buf.	TRM	155.84	136.11
		CRM	136.51	118.98
		CCRM	151.67	130.12
	resbuf.	TRM	80.06	59.02
		CRM	65.44	46.71
		CCRM	77.39	54.22

Table 1. average performance measure for $a = 0.15$

	Initial order list	GA order list
TRM	63.83	42.46
CRM	48.52	30.15
CCRM	54.09	34.38

Table 2. average performance measure for $\alpha = 0.3$

	Initial order list	GA order list
TRM	55.08	34.18
CRM	38.61	22.74
CCRM	46.59	26.69

Table 3. average performance measure for $\alpha = 0.45$

	Initial order list	GA order list
CRM	65.44	46.71
CIW	74.37	62.93

Table 4. CRM versus CIW for $\alpha = 0.15$

NoIter	*NoPop*	CRM with GA
20	40	46.71
40	20	50.96
10	80	47.83
80	10	51.09

Table 5. impact of number of iteration and population

P_{mut}	CRM with GA
0.01	48.87
0.05	46.71
0.1	49.02

Table 6. impact of mutation probability

CRM method composed with GA order list works better than other methods in both $\alpha = 0.3$ and 0.45. We mention CIW method as a benchmark in section 3; while as illustrated in Table 4, CRM gives better results over CIW method.

For whom may interests detail of our local procedure we present parameter tuning through Tables 5 and 6. We limit our effort to 8000 schedules in search process; while 10 replications are used to evaluate a list, we have to divide 800 schedules between $NoIter$ and $NoPop$.

As could be expected, devoting more effort to number of individuals rather than overall GA iteration yields better performance. Assessment of mutation probability is done in Table 6; where GA algorithm with $P_{mut} = 0.05$ shows better answer.

7. Conclusion

In this paper we have studied resource-constrained project scheduling under uncertain resource availability circumstances. We introduce three-part solution in which all possible ways to cope with uncertainty could be included. Three new heuristics for constructing initial schedule considering the resource usage and resource availabilities as major factors are developed. Novel procedures are also presented to extend proactive strategies for studied problem. For last part of solution, we have adapted to GA

algorithm in developing local search process regards to define a proper order list as reactive strategy.

An extensive computational efforts show that our three-part solution using CRM method as first part enriched by both time and resource buffer and applying GA order list as reactive strategy yields the best performance among other combinations. The result also depicts that our solution outperforms the best heuristic developed in literature called CIW.

Refrences

Akturk, M. S., andGorgulu, E. (1999).Match-up scheduling under a machine breakdown. European Journal of Operational Research,112, 81–97.

Al-Fawzan,M. A., andHaouari, M. (2005). A bi-objective model for robust resource constrained project scheduling. International Journalof Production Economics, 96, 175-187

Ashtiani, B., Leus, R. andAryanezhd, M, B. (2009).New competitive results for the stochastic resource-constrained project scheduling problem: exploring the benefits of pre-processing. Journal of Scheduling, to appear.

Ballestín, F. (2007).When it is worthwhile to work with the stochastic RCPSP?. Journal of Scheduling, 10(3), 153–166.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., andPesch, E. (1999).Resource-constrained project scheduling: notation, classification, models and methods. European Journal of Operational Research,112, 3-41.

Demeulemeester, E., andHerroelen, W. (1997).New benchmark results for the resource-constrained project scheduling problem. ManagementScience, 43, 1485–1492.

Demeulemeester, E., andHerroelen, W. (2002).International series in operations research & management science: Vol. 49. Project scheduling-a research handbook. Boston: Kluwer Academic.

Drexl, A. (1991). Scheduling of project networks by job assignment.Management Science, 37, 1590–1602.

Drezet, L.-E. (2005). Résolution d'un problème de gestion de projetssous contraintes de ressourceshumaines: de l'approchepredictive à l'approcheréactive. PhD thesis.Université François RabelaisTours, France.

Fernandez, A. A., Armacost, R. L., and Pet-Edwards, J. (1998).Understanding simulation solutions to resource constrained project scheduling problems with stochastic task durations. EngineeringManagement Journal, 10, 5–13.

Forrester, J. W. (1961). Industrial Dynamics.MIT Press.

Goldratt, E. (1997). Critical Chain, North River Press.

Hartmann, S. (1998).A competitive genetic algorithm for resource-constrained project scheduling.Naval Research Logistics, 45, 733–750.

Herroelen,W., De Reyck, B., and Demeulemeester, E. (1998).Resource-constrained scheduling: a survey of recent developments. Computersand Operations Research, 25, 279–302.

Herroelen, W., De Reyck, B., andDemeulemeester, E. (2000).On the paper "Resource-constrained project scheduling: notation, classification, models and methods" by Brucker et al. European Journalof Operational Research, 128(3), 221–230.

Herroelen, W., andLeus, R. (2003).Project scheduling under uncertainty-survey and research potentials. European Journal ofOperational Research, 165(2), 289–306.

Herroelen,W., andLeus, R. (2004).Robust and reactive project scheduling: A review and classification of procedures. International Journalof Production Research, 42(8), 1599–1620.

Herroelen, W., andLeus, R. (2005).Project scheduling under uncertainty, survey and research potentials. European Journal of OperationalResearch, 165(8), 289–306.

Holland, H. (1975).Adaptation in natural and artificial systems.Ann Arbor: University of Michigan Press.

Hopp, W., and Spearman, M. (2001).Factory physics: foundations ofmanufacturing management. New York: McGraw–Hill.

Igelmund, G., andRadermacher, F. (1983).Preselective strategies for the optimization of stochastic project networks under resource constraints.Networks, 13, 1–28.

Kelley, J. E., Jr. (1963). The critical-path method: resources, planning and scheduling. In J. F. Muth and G. L. Thompson (Eds.), Industrialscheduling (pp. 347–365). Englewood Cliffs: Prentice Hall.

Kolisch, R., and Hartmann, S. (1999). Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis. In J. Weglarz (Ed.), Project scheduling.Recent models, algorithms and applications (pp. 147–178).Dordrecht: Kluwer.

Kolisch, R., and Hartmann, S. (2006). Experimental investigation of heuristics for resource-constrained project scheduling: An update.European Journal of Operational Research, 174, 23–37.

Kolisch, R., andSprecher, A. (1996).PSPLIB—a project scheduling problem library. European Journal of Operational Research, 96,205–216.

Lamberachts, O., Demeulemeester, E., and Herroelen, W. (2008).Proactive-reactive strategies for resource-constrained project scheduling with uncertain resource availabilities. Journal of Scheduling, 11, 121-136

Leus, R. (2003). The generation of stable project plans. PhD thesis.Department of applied economics, KatholiekeUniversiteit Leuven,Belgium.

Mehta, S., andUzsoy, R. (1998).Predictive scheduling of a job shop subject to breakdowns. IEEE Transactions on Robotics and Automation,14, 365–378.

Mehta, S., andUzsoy, R. (1999).Predictable scheduling of a single machine subject to breakdowns. International Journal of ComputerIntegrated Manufacturing, 12, 15–38.

Sabuncuoglu, I., andBayiz, M. (2000).Analyse of reactive scheduling problems in a job shop environment. European Journal of OperationalResearch, 126, 567–586.

Sakkout, E., Richards, E. T., and Wallace, M. G. (1997).Unimodular probing for minimal perturbance in dynamic resource feasibility problems.In CP97 Workshop on the Theory and Practice of DynamicConstraint Satisfaction.

Smith, S. F. (1995). Reactive scheduling systems.In D. E. Brown,& W. T. Scherer (Eds), intelligent scheduling systems. Dordrecht:Kluwer Academic

Snoek, M. (2001).Anticipation optimization in dynamic job shops.In Genetic and Evolutionary Computation Conference 2001(GECCO-2001), San Francisco (pp. 43–46).

Stork, F. 2001.Stochastic resource-constrained project scheduling.Ph.D. thesis, TechnischeUniversität Berlin.

Van de Vonder, S., Demeulemeester, E., Herroelen, W., andLeus, R. (2005).The trade-off between stability and makespan in resource-constrained project scheduling. International Journal ofProduction Research, 44(2), 215–236.

Wang, J. (2004). A fuzzy robust scheduling approach for product development projects.European Journal of Operational Research,152, 180-194.

Yu, G., and Qi, X. (2004).Disruption management-framework, models and applications.Singapore: World Scientific.