Journal of the Operations Research
Society of Japan
Vol. 40, No. 1, March 1997

A RESOURCE CONSTRAINED PROJECT SCHEDULING PROBLEM WITH REATTEMPT AT FAILURE: A HEURISTIC APPROACH

Masao Mori Ching-Chih Tseng
Tokyo Institute of Technology

(Received May 22, 1995; Revised September 21, 1995)

Abstract This article considers a new type of scheduling problem for a research and development (R& D) project, which consists of more than one activity with uncertainty. When an activity is finished, we take a functional test to justify whether it is successful or not. If unsuccessful, the activity will be reattempted once more in one of the different combinations of resources and duration depending on failure type. In our problem, we assume that each activity may be performed in one of the various resource-duration modes. The objective of this paper is to propose a heuristic approach, including two dispatching rules which contain six policies, and find the best policy with which the quasi-minimum expected project duration of our problem can be obtained. A statistical procedure is used to choose the policy with the smallest expected project duration. Furthermore, we use a posterior analysis to evaluate the proposed algorithm with the selected policy.

1. Introduction

A traditional resource constrained project scheduling problem is considered to be an activity which is subject to technological precedence constraints (i.e. an activity can start only if all its predecessor activities have been completed) and which cannot be interrupted once begun (i.e. no preemption is allowed). In this problem, an activity can be performed in exactly one or more than one combination of duration and resource requirements. For example, an activity can be completed within 5 days with 3 persons required per day by selecting one mode. For any activity, once initialized in a specific mode, it must be fixed without changing its mode until it is completed. Resources are available per period in a constant amount. The problem described above can be called a **single-mode** or **multi-mode** resource constrained project scheduling problem depending on an activity performed in exactly one or more ways. Interested readers should examine single-mode works [1,2,4,7,11,13] and multi-mode ones [6,9,10,12,14].

The following example illustrates the above-mentioned multi-mode one. In Figure 1 and Table 1 (ignore the column of "reattempt after test" temporarily), the project is shown to consist of six activities (including one dummy activity), each of which is successfully accomplished in one of two modes with probability one. Four resources are required by each activity simultaneously. The resources available in each period of the project duration are 1,2,6,8 for resource type one to four, respectively.

Up to now, a traditional resource constrained project scheduling problem is considered to be unique in advance, in which every activity is completed successfully with probability one. However, many real world systems do not meet the above condition, such as an R&D project. Consider, for example, an R&D project in which an activity must be reattempted once more, if previously unsuccessful, before its successors can start. In this paper we are

concerned with a new type of scheduling problem for a multi-mode resource constrained R&D project in which an activity with reattempt at failure is considered. This type of the problem is common in practice, especially in a software system development project or new weapon system research and development project. As far, however, the open studies have no concern with this problem.

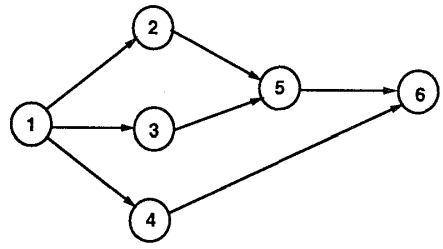

Figure 1: A 6-activity project

Table 1: The data of resource-duration modes for the project

activity no.	mode	do before test					reattempt after test						
		dura.	required resource				fail. type	fail. pro.	dura.	required resource			
i	j	d_{ij}	q_{ij1}	q_{ij2}	q_{ij3}	q_{ij4}	k	p_{ijk}	$d_{i'jk}$	$q_{i'jk1}$	$q_{i'jk2}$	$q_{i'jk3}$	$q_{i'jk4}$
1	1	2	1	0	2	1	0	0.6	0	0	0	0	0
							1	0.3	3	1	1	0	1
							2	0.1	1	1	0	1	1
	2	3	0	0	2	1	0	0.5	0	0	0	0	0
							1	0.3	2	0	2	1	2
							2	0.2	3	0	1	1	2
2	1	1	1	0	3	2	0	0.7	0	0	0	0	0
							1	0.2	2	0	2	1	2
							2	0.1	1	0	0	2	1
	2	3	0	0	3	2	0	0.8	0	0	0	0	0
							1	0.2	2	0	1	1	2
3	1	3	1	0	1	4	0	0.7	0	0	0	0	0
							1	0.2	2	1	0	2	1
							2	0.1	3	1	1	0	1
	2	4	0	1	1	4	0	0.5	0	0	0	0	0
							1	0.3	3	0	0	1	4
							2	0.2	2	0	1	0	3
4	1	5	1	0	1	3	0	0.6	0	0	0	0	0
							1	0.2	4	1	0	1	1
							2	0.2	3	0	0	1	2
	2	7	0	1	1	3	0	0.7	0	0	0	0	0
							1	0.2	5	0	0	1	1
							2	0.1	2	0	1	0	1
5	1	4	1	0	2	2	0	0.8	0	0	0	0	0
							1	0.2	3	1	0	2	1
	2	6	0	1	2	2	0	0.7	0	0	0	0	0
							1	0.3	4	1	0	1	1
6	1	0	0	0	0	0	0	1.0	0	0	0	0	0

This paper is organized as follows. Section 2 describes the new type of scheduling problem we consider. Section 3 presents a heuristic approach which includes two dispatching rules which contains six policies to find the quasi-minimum project duration of our problem. Section 4 introduces a procedure whose goal is to compare these policies and to select one of the policies as being the best one and then provides a posterior analysis which is used to evaluate the proposed algorithm with the selected policy.

The notations used in this paper are summarized in Table 2.

Table 2: Notation

Symbol	Definition
N	the number of the activities of the project,
R	the number of the resource types,
M_i	the number of the resource-duration modes of activity i,
K_{ij}	the number of failure type of activity i performed in mode j
d_{ij}	the duration of activity i which is performed in mode j,
d_{ijk}	the duration of reattempt activity i, performed in mode j and failed in type k,
q_{ijr}	the amount of resource type r, required by activity i performed in mode j,
q_{ijkr}	the amount of resource type r required by reattempt activity i performed in mode j and failed in type k,
p_{ijk}	the probability of the event associated with activity i, performed in mode j and failed in type k,
Q_r	the available amount of resource type r in each time period, $r = 1, 2, \cdots, R$,
LF_i	the latest finish time of activity i,
$\langle a, b\rangle$	the precedence relation of activities a and b, a precedes b,
$\mathbf{S}_i$	a set of immediate successor activities of activity i,
U	a set of unscheduled activities,
EJ	a set of activities which are eligible to schedule at this time,
A	a set of activities which are active at this time.

2. Problem statement

We consider a new type of scheduling problem for a multi-mode resource constrained R&D project which consists N activities. Nth is the dummy activity which means the completion point of project. Activity i may be performed in one of the modes $j = 1, \cdots, M_i$. Each job, once initiated in a specific mode, it must be fixed without changing its mode until it is completed. Scheduling activity i in mode j takes d_{ij} time units (duration). Activity preemption is not allowable. There are $r = 1, \cdots, R$ resources, where resource r is available with Q_r at each time period. Scheduling activity i in mode j uses q_{ijr} resources units per period for resource r. When a activity i is completed, a functional test must be taken. If previously unsuccessful, a reattempt activity i' is generated, break the existed precedence relations $\langle i, h\rangle, h \in \mathbf{S}_i$ and new precedence relations $\langle i, i'\rangle$, $\langle i', h\rangle$ are added. This reattempt activity i' must be scheduled only once (we assume) in one of the different combinations of the required resources $q_{i'jkr}$ and the duration $d_{i'jk}$ corresponding to its failure type with probability p_{ijk}. Each activity may bring about one or more failure types according to the mode in which the activity is performed. A heuristic approach is proposed to find the quasi-minimum expected project duration.

Certainly, in actual project operating aspect, a key activity may possibly be reattempted more than one time. But based on the following two reasons, we assume that an activity is reattempted only once. For the first reason we know by experience that the later adjustment done at reattempt often brings a satisfying and permissive result. The other one is the problem will become very complicate for more than one reattempt and we can view this assumption, at most one reattempt, as an approximate estimate of one or more than one reattempt.

To give a more description of our problem, we present an example as Section 1. Instead of being successfully completed with probability one, every activity may fall in one of the failure types with corresponding probability (refer to "reattempt after test" column of Table 1). For instance, in Figure 2 (part of Figure 1), activity 3 may be performed in mode 1.

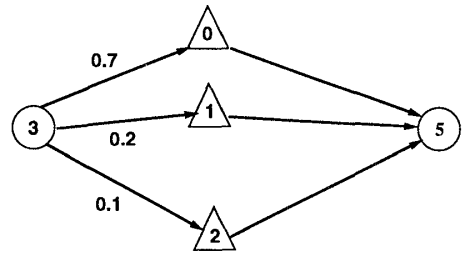

Figure 2: Relation of two dependent activities

It may succeed with probability 0.7 (triangle node 0 denotes a dummy reattempt activity). Otherwise it may also fall into failure type 1/type 2 with probability 0.2/0.1 which means that we must reattempt activity 3 with the combination of duration 2/3 and the required amounts are 1,0,2,1/1,1,0,1 for resource type 1 to 4 respectively. The failure type 1 is depicted as triangle node 1 in Figure 2. Activity 5 may not start until activities 2 and 3 are all successfully completed.

Similarly to our problem, Tseng et al. [14] proposed a problem that an R&D project, which is composed of a collection of activities, is scheduled. After the completion of the project, a test will be taken. They assumed that each activity can be performed in one of the various resource-duration modes, corresponding with different success probabilities. Their objective is to decide the mode in which each activity should be performed, so that the maximal probability of success for project test contributed by all its activities can be found, under general precedence, resource and project duration constraints.

3. A scheduling approach

For a traditional resource constrained project scheduling problem, because the amounts of available resources are not always sufficient to satisfy demands of concurrent activities, sequencing decisions are required whenever any activity finishes processing. Furthermore for an R&D project due to the uncertainties of activities, the possibility of being generated reattempt activity increases the complexity of problem.

3.1. Two dispatching rules containing six policies

At first, from the resulting precedence network, the latest finish time of (reattempt) activity i, LF_i is defined as

$$LF_i = \begin{cases} \bar{T} & \text{if } i = N, \\ \min_j\{LF_j - \min_k\{d_{jk} | k = 1, \cdots, M_j\} : \langle i, j \rangle, j \in \mathbf{S}_i\} & \text{otherwise,} \end{cases}$$

where $\mathbf{S}_i$ denotes the set of immediate successor activities of activity i and $\bar{T}$ denotes an upper bound of project duration determined by

$$\bar{T} = \sum_{i=1}^{N} \max_{j,k}\{d_{ij} + d_{ijk} | j = 1, \cdots, M_i, k = 1, \cdots, K_{ij}\}.$$

Similarly, what we are concerned about is how to decide which activity in the set of eligible activities **EJ** should be scheduled first and in which mode it should be performed. To solve it, many different ways have been proposed. Now we adopt two dispatching rules containing six policies, which behaved best in traditional multi-mode resource constrained project scheduling problem, in our algorithm. One is the *deterministic dispatching rule*, MINLF. Talbot [12] presented and compared eight heuristic dispatching rules, and then

the rule MINLF was shown to behave best regarding the average quality of solution. Once an activity i in **EJ** has been selected according to MINLF to be scheduled, a quite greedy way of assigning an activity mode is to take the mode j with the *smallest* duration and to schedule activity i as early as possible.

The other one is the *stochastic dispatching rule*, however, Drexl [5] as well as Drexl and Gruenewald [6] proposed a stochastic construction method (a weight random selection technique) to solve assignment-type project scheduling problems. The stochastic nature of this method emerges from using some criteria measuring the impacts of activity selection and mode assignment in a probabilistic way. Drexl and Gruenewald [6] found that the stochastic construction method outperforms MINLF in a traditional multi-mode resource constrained project scheduling problem. For convenience they called this stochastic activity selection and mode assignment procedure STOCOM (STOchastic COnstruction Method).

Analogously to [6], we calculate

$$\gamma_i = (\max\{LF_k | k \in \mathbf{EJ}\} - LF_i + \epsilon)^\alpha, i \in \mathbf{EJ} \tag{3.1}$$

for selecting (reattempt) activity $i \in \mathbf{EJ}$ randomly with probabilities proportional to γ_i for all $i \in \mathbf{EJ}$. Equation (3.1) measures the worst-case consequence of *not* to dispatch activity i with respect to the latest finish time; $\epsilon > 0$ makes γ_i to be positive; $\alpha \geq 0$ transforms the term (.) in an exponential way, thus diminishing or enforcing the difference between the mode-dependent activity durations for $\alpha < 1$ or $\alpha > 1$ respectively.

Suppose that the (reattempt) activity π is selected, then mode j to activity π is assigned at random with probabilities proportional to $\lambda_{\pi j}$. $\lambda_{\pi j}$ is defined as

$$\lambda_{\pi j} = (\max\{d_{\pi\eta} | \eta = 1, \cdots, M_\pi\} - d_{\pi j} + \epsilon)^\alpha, j = 1, \cdots, M_\pi, \tag{3.2}$$

where M_π =1 if activity π is a reattempt one.

We can find the five results derived from this rule by tuning the parameter α to five values (Note each value of parameter α represents one policy in our algorithm.) of α =0.0,0.5,1.0,1.5, and 2.0. Note that MINLF corresponds to the case of $\alpha = \infty$.

3.2. Algorithm statement

As described in Section 2, suppose an activity a in mode m was completed, whether or not its reattempt activity a' is generated depends on the probability, p_{amk}. Once this reattempt activity a' was generated , we put a' immediately after a and update the relative precedence relations; then we add a' into the set of unscheduled activities **U** and calculate the latest finish time, $LF_{a'}$.

The algorithm to find feasible solutions to our problem is described in brief as follows. Initially (time $t_{now} = 0$), put all activities i in **U** . At time $t_{now} = 0$, or any subsequent time when an (a reattempt) activity finishes processing, that in **U** the activities whose all predecessor activities have completed are put in **EJ** and removed from **U**. In **EJ** an activity is selected and mode is assigned using a policy of two heuristic dispatching rules. The selected activities are placed in the active set **A** and are removed from **EJ** and the job dispatching process is repeated. If resource conflict occurs or **EJ** is empty, time t_{now} is advanced to the earliest finish time of jobs currently in the active set **A**, the finishing activity is removed from **A** and a reattempt activity is generated randomly according to its corresponding probabilities if the finished activity is not a reattempt one. Put this generated reattempt activity immediately after its previous one and revise the relative precedence relations and a new **EJ** is formed. A schedule is completed when all activities are removed the unscheduled list **U**.

4. Computational results

4.1. Generation of a problem instance

Some researchers ([6],[10]) have proposed the algorithm of generating a problem instance for a traditional resource constrained project scheduling problem. In this paper, the problem instance generated for comparative purposes may be characterized as follows (project summary measures).

- The problem size, in term of the number of activities N, is the first project summary measure.
- The network complexity C that denotes the value of dividing the number of arcs (precedence relations) by the number of nodes (activities), affects the performance of scheduling procedures.
- Generate an activity network by using *A Random Activity Network Generator* proposed by Demeulemeester et al.[4] when N and C are given.
- The number of modes of activity i is generated randomly such that $2 \leq M_i \leq 4$.
- The failure type k of activity i being scheduled in mode j is fixed at K_{ij} =1.
- The probability p_{ijk} of failure type k is generated at random, and let $\sum_{k=0}^{K_{ij}} p_{ijk} = 1$.
- The number of resource type $R = 4$ is fixed.
- The (integer) duration d_{ij} of activity i being scheduled in mode j is generated at random such that $5 \leq d_{ij} \leq 10$.
- The (integer) duration d_{ijk} of reattempt activity i being scheduled in mode j and failed in type k is generated at random such that $1 \leq d_{ijk} \leq 5$.
- Activity-mode-dependent (integer) resource demands (requirements) are randomly generated such that $0 \leq q_{ijr} \leq 5$ for all resources.
- The (integer) resource amount, which is depended on failure type and required by reattempt activity, is randomly generated such that $0 \leq q_{ijkr} \leq 3$ for all resources.
- The availability Q_r of resource type r is determined by multiplying the peak resource requirement

$$Q_r' = \max\{q_{ijr}, q_{ijkr} | i = 1, \cdots, N, j = 1, \cdots, M_i, k = 1, \cdots, K_{ij}\}$$

with θ thus getting

$$Q_r = Q_r' \cdot \theta, \qquad r = 1, \cdots, R.$$

4.2. Results of the proposed algorithm

The main factors of study were three resource availability levels (θ =1.5, 2.0, 2.5). For each level, nine problem instances were generated with N =20 to 100. Once a problem instance is erected, we want to compare the output data of the ρ=6 different policies and select the best one which can obtain the quasi-minimum expected project duration. One of the six policies is a deterministic dispatching rule MINLF (with the minimum duration mode) and the other five policies are the stochastic dispatching rule STOCOM, with different values of α =0.0, 0.5, 1.0, 1.5, 2.0. An immediate problem encountered is that the simulation output data are stochastic, so how many replications of each policy we should simulate to get a reliable result.

Let T_{ij} be the random variable of project duration from the jth replication of the ith policy, and $\mu_i = T_{ij}$ denotes the expected project duration of policy i, $i = 1, \cdots, \rho$. Let μ_{i_l} be the lth smallest of the μ_i's, so that $\mu_{i_1} \leq \mu_{i_2} \leq \cdots \leq \mu_{i_\rho}$. Now we want to select a policy

with the *smallest* expected one, μ_{i_1}.

Described as in Law and Kelton[8] (pp. 596-597), the statistical procedure stated below has the nice property that, with probability at least P^*, the expected response of the *selected* policy will be no larger than $\mu_{i_1} + d^*$, where d^* denotes the indifference amount. Thus, we are protected (with probability at least P^*) against selecting a policy with mean that is more than d^* worse than that of the best policy.

The statistical procedure involves "two-stage" sampling from each of the six policies. In the first stage we make a fixed number of replications of each policy, then use the resulting variance estimates to determine how many more replications from each policy are necessary in a second stage of sampling in order to reach a decision. It must be assumed that the T_{ij}'s are normally distributed, but (importantly) we need *not* assume that the values of $\sigma_i^2 = \mathrm{Var}(T_{ij})$ are known. The procedure's performance should be robust to departures from the normality assumption, especially if the T_{ij}'s are averages.

Table 3: Selecting the best of the six policies (N =50, C =4)

Policy i	$\bar{T}_i^{(1)}(20)$	$S_i^2(20)$	N_i	$\bar{T}_i^{(2)}(N_i-20)$	W_{i1}	W_{i2}	$\tilde{T}_i(N_i)$
			θ =1.5				
α = 0.0	268.87	34.54	284	269.09	0.08	0.92	269.08
α = 0.5	262.17	9.70	79	262.82	0.30	0.70	262.62
α = 1.0	262.38	13.05	107	262.08	0.21	0.79	262.14
α = 1.5	261.97	11.19	92	261.02	0.24	0.76	261.24
α = 2.0	261.68	4.19	34	261.74	0.65	0.35	261.70
MINLF	260.60	11.07	91	261.32	0.24	0.76	261.14
			θ =2.0				
α = 0.0	185.19	5.30	43	185.35	0.52	0.48	185.26
α = 0.5	177.41	2.59	21	174.50	0.98	0.02	177.35
α = 1.0	173.79	3.88	31	173.97	0.73	0.27	173.84
α = 1.5	172.36	2.73	22	174.00	0.95	0.05	172.43
α = 2.0	171.48	2.62	21	171.10	0.99	0.01	171.48
MINLF	171.95	0.94	21	174.30	1.23	-0.23	171.40
			θ =2.5				
α = 0.0	158.38	7.05	58	158.99	0.36	0.64	158.77
α = 0.5	149.37	4.14	34	149.57	0.62	0.38	149.45
α = 1.0	145.63	5.48	45	145.26	0.47	0.53	145.43
α = 1.5	144.21	1.73	21	142.50	1.10	-0.10	144.38
α = 2.0	142.65	1.76	21	142.60	1.10	-0.10	142.65
MINLF	142.29	1.49	21	141.30	1.13	-0.13	142.43

Table 4: Selecting the best of the six policies (N =100, C =6)

Policy i	$\bar{T}_i^{(1)}(20)$	$S_i^2(20)$	N_i	$\bar{T}_i^{(2)}(N_i-20)$	W_{i1}	W_{i2}	$\tilde{T}_i(N_i)$
			θ =1.5				
α = 0.0	506.45	52.48	432	508.79	0.05	0.95	508.68
α = 0.5	498.23	21.23	174	497.04	0.14	0.86	497.20
α = 1.0	494.39	28.90	238	493.78	0.09	0.91	493.83
α = 1.5	491.11	35.38	291	491.62	0.08	0.92	491.58
α = 2.0	490.14	15.94	131	489.01	0.17	0.83	489.20
MINLF	487.45	23.25	191	487.65	0.12	0.88	487.63
			θ =2.0				
α = 0.0	341.18	9.99	82	341.86	0.27	0.73	341.68
α = 0.5	329.29	5.17	42	329.85	0.53	0.47	329.55
α = 1.0	324.86	6.74	55	325.18	0.41	0.59	325.05
α = 1.5	322.40	4.58	37	321.14	0.61	0.39	321.91
α = 2.0	320.84	4.19	34	319.54	0.65	0.35	320.38
MINLF	319.79	2.87	23	318.97	0.93	0.07	319.73
			θ =2.5				
α = 0.0	283.67	9.65	79	283.72	0.29	0.71	283.70
α = 0.5	276.98	10.50	86	276.77	0.26	0.74	276.82
α = 1.0	273.36	6.30	51	272.38	0.46	0.54	272.84
α = 1.5	269.48	3.26	26	270.98	0.85	0.15	269.71
α = 2.0	268.92	3.03	24	267.55	0.91	0.09	268.79
MINLF	266.88	3.86	31	267.13	0.72	0.28	266.95

First-stage We make $n_0 \geq 2$ replications of each the i policies and obtain the first-stage sample means $\bar{T}_i^{(1)}(n_0)$ and variances $S_i^2(n_0)$ for $i = 1, 2, \cdots, \rho$. Then we compute the total sample size N_i needed for policy i as

$$N_i = \max\left\{n_0 + 1, \left\lceil \frac{\tau^2 S_i^2(n_0)}{(d^*)^2} \right\rceil\right\},$$

where $\lceil x \rceil$ is the smallest integer that is greater than or equal to the real number x, and τ (which depends on ρ, P^*, and n_0) is a constant that can be obtained Law and Kelton[8].

Second-stage Next, we make $N_i - n_0$ *more* replications of policy i $(i = 1, 2, \cdots, \rho)$ and obtain the second-stage sample means

$$\bar{T}_i^{(2)}(N_i - n_0) = \frac{\sum_{j=n_0+1}^{N_i} T_{ij}}{N_i - n_0}$$

Then define the weights

$$W_{i1} = \frac{n_0}{N_i}\left[1 + \sqrt{1 - \frac{N_i}{n_0}\left(1 - \frac{(N_i - n_0)(d^*)^2}{\tau^2 S_i^2(n_0)}\right)}\right]$$

and $W_{i2} = 1 - W_{i1}$, for $i = 1, 2, \cdots, \rho$. Finally, define the weighted sample means

$$\tilde{T}_i(N_i) = W_{i1}\bar{T}_i^{(1)}(n_0) + W_{i2}\bar{T}_i^{(2)}(N_i - n_0)$$

and select the policy with the smallest $\tilde{T}_i(N_i)$.

Table 5: The weighted sample means by using STOCOM and MINLF

Policy	N =20 C =2	N =30 C =3	N =40 C =3	N =50 C =4	N =60 C =4	N =70 C =5	N =80 C =6	N =90 C =6	N =100 C =6
				θ =1.5					
$\alpha = 0.0$	100.85	160.24	225.52	269.08	309.94	351.85	400.43	452.33	508.68
$\alpha = 0.5$	97.77	151.80	220.82	262.62	303.71	345.14	394.01	437.34	497.20
$\alpha = 1.0$	97.47	148.34	217.77	262.14	300.72	343.52	392.01	432.19	493.83
$\alpha = 1.5$	96.26	146.51	216.67	261.24	298.96	341.63	390.76	431.61	491.58
$\alpha = 2.0$	96.08	146.88	215.59	261.70	296.55	343.12	390.18	431.32	489.20
MINLF	95.20	146.45	215.54	261.14	294.83	344.58	388.68	428.92	487.62
Best policy	MINLF	MINLF	MINLF	MINLF	MINLF	$\alpha = 1.5$	MINLF	MINLF	MINLF
				θ =2.0					
$\alpha = 0.0$	81.88	122.53	171.94	185.26	217.36	250.04	270.01	312.95	341.68
$\alpha = 0.5$	77.63	117.97	165.82	177.35	209.86	242.80	263.34	303.58	329.55
$\alpha = 1.0$	75.18	113.70	161.54	173.84	205.36	240.44	260.20	300.75	325.05
$\alpha = 1.5$	73.79	113.64	161.35	172.43	203.41	239.28	259.82	301.06	321.91
$\alpha = 2.0$	73.60	112.21	160.48	171.48	202.20	238.66	259.11	300.58	320.38
MINLF	73.00	111.43	160.33	171.40	201.66	238.23	258.70	300.92	319.73
Best policy	MINLF	MINLF	MINLF	MINLF	MINLF	MINLF	MINLF	$\alpha = 2.0$	MINLF
				θ =2.5					
$\alpha = 0.0$	72.81	107.75	156.95	158.77	183.46	216.50	225.43	262.04	283.70
$\alpha = 0.5$	70.46	105.42	151.96	149.45	177.28	210.04	216.47	255.21	276.82
$\alpha = 1.0$	70.43	104.19	150.61	145.43	174.93	206.58	213.47	253.58	272.82
$\alpha = 1.5$	70.24	104.06	149.81	144.38	172.07	204.60	212.77	253.37	269.71
$\alpha = 2.0$	69.82	103.51	149.32	142.65	171.30	204.01	212.05	252.36	268.79
MINLF	70.58	103.23	149.43	142.43	170.45	202.29	212.34	252.38	266.95
Best policy	$\alpha = 2.0$	MINLF	$\alpha = 2.0$	MINLF	MINLF	MINLF	$\alpha = 2.0$	$\alpha = 2.0$	MINLF

· N denotes the number of activities.
· C denotes the complexity of network.
· Each cell denotes a weighted sample mean.

Our goal is to select a policy with the smallest μ_i and to be $100P^*$=90 percent sure that we have made the correct selection provided that $\mu_{i_2} - \mu_{i_1} \geq d^*$=1. For each problem instance, we first made n_0=20 initial independent replications (Note that each replication represents the average of 10 project durations.) of each policy, so that τ=2.870 (see Law and Kelton [8] Table 10.11, p.606). The results of the first-sampling, $\bar{T}_i^{(1)}(20)$ and $S_i^2(20)$, can be obtained. From the $S_i^2(20)$'s, τ, and d^*, we next computed the total sample size N_i for each policy. Then we made $N_i - 20$ additional replications for each policy, i.e., 264 more replications for policy α =0.0, 59 more for policy α =0.5, etc., for the case θ=1.5 of N =50, C =4, in Table 3, and computed the second-stage sample means $\bar{T}_i^{(2)}(N_i - 20)$. Finally,

we calculated the weights of W_{i1} and W_{i2} for each policy and the weighted sample means $\tilde{T}_i(N_i)$. Here we show two examples, the generated problem instances for N =50 C =4 and for N =100 C =6, as in Table 3 and Table 4 respectively. The overall results of the weighted sample mean and the smallest weighted sample mean are shown in Table 5 . From their weighted sample means, we can find $\tilde{T}_{\text{MINLF}} - \tilde{T}_{\alpha=2.0} < 1$ in some cases. It means that the quasi-minimum expected project durations obtained using MINLF and STOCOM with α =2.0 respectively are probably very close together. From Table 5, we find that MINLF which has the smallest weighted sample mean occurs with a considerably high frequency. So we can make a conclusion that the algorithm with policy MINLF is the most likely to produce the quasi-minimum expected project duration.

Furthermore we wish to investigate the additional project duration increased by resource conflict for different levels of resource availability. That is to say, we want to make a comparison of the weighted sample means which are obtained under enough resource case of θ =10.0 and under other insufficient case of θ for the same problem instance. In Table 6 the upper element of each cell is the weighted sample means obtained with policy MINLF. And the lower one is the percentage deviation of the weighted sample mean obtained under each θ from the mean derived under θ=10.0.

Table 6: The weighted sample means (policy MINLF) under different θ level

θ	N =20 C =2	N =30 C =3	N =40 C =3	N =50 C =4	N =60 C =4	N =70 C =5	N =80 C =6	N =90 C =6	N =100 C =6
θ=1.5	95.20	146.45	215.54	261.14	294.83	344.58	388.68	428.92	487.62
	(35.67)	(43.28)	(44.62)	(105.75)	(98.94)	(88.55)	(145.35)	(101.40)	(116.64)
θ=2.0	73.00	111.43	160.33	171.40	201.66	238.23	258.70	300.92	319.73
	(4.03)	(9.02)	(7.58)	(35.05)	(36.07)	(30.36)	(63.30)	(41.30)	(42.05)
θ=2.5	70.58	103.23	149.43	142.43	170.45	202.29	212.34	252.38	266.95
	(0.05)	(0.10)	(0.02)	(12.22)	(15.01)	(10.69)	(34.04)	(18.50)	(18.60)
θ=10.0	70.17	102.21	149.04	126.92	148.20	182.75	158.42	212.97	225.08

4.3. A posterior analysis of the proposed heuristic algorithm with the best policy

Let ϕ be a problem instance, h be an arbitrary scheduling policy. ω denotes a realization that the failure type of each activity is disclosed after the project was scheduled through the execution of our algorithm. We wish to find the best policy β which minimizes the expected project duration, i.e. satisfying

$$ET(\phi,\beta) = \min_h ET(\phi,h) = \sum T(\phi,h,\omega)P(\omega)$$

for any problem instance ϕ, where $T(\phi,h,\omega)$ is the project duration for a problem instance ϕ and a realization ω under a scheduling policy h, and $P(\omega)$ is the probability measure for causing failure. However β may not exist. Even if β exists, it is almost impossible to be found out. Instead, in this paper our goal is to find the policy h^* with which quasi-minimum expected project duration can be gotten for the most problem instances.

From the results of last subsection, MINLF is considered as the most favorable candidate of the policy h^*. In order to know the quality of the solution found using the proposed algorithm with MINLF, we want to evaluate the value of

$$\frac{[ET(\phi,\text{MIN}LF) - ET(\phi,\beta)]}{ET(\phi,\beta)}.$$

Since $ET(\phi,\beta)$ is unknown, we try to evaluate it in another way.

Since

$$T(\phi,\text{MIN}LF,\omega) \geq T(\phi,\beta',\omega) = T(\phi,CPL,\omega) + T_{conflict}(\phi,\omega),$$

where $T(\phi,\beta',\omega)$, $T(\phi,CPL,\omega)$ and $T_{conflict}(\phi,\omega)$ denote the minimum duration, the critical path length and the additional duration increased due to resource conflict, of the "actual" project respectively. $T(\phi,\beta',\omega)$ can be obtained by the posteriorly best algorithm β' of the "actual" project which we have known after finding out a realization which type of failure has happened for each activity through the execution of our algorithm with the best policy MINLF. We know the more the available amount of resources is given, the smaller $T_{conflict}(\phi,\omega)$ will be. If an infinite number of resources availability level are given, $T(\phi,\beta',\omega)$ must be equivalent to $T(\phi,CPL,\omega)$.

However, there is still no optimal procedure β' which is considered to be computationally feasible for the large and complex projects which occur in practice. Furthermore, McGinnis states that the results given by the optimization algorithms have been uniformly discouraging and there is no algorithm that can solve problems with fifty activities or more with reasonable computational effort [7]. Hence we use $T(\phi,CPL,\omega)$ as a lower bound of $T(\phi,\beta',\omega)$. That is, we estimate

$$\frac{[ET(\phi,\text{MIN}LF)-ET(\phi,CPL)]}{ET(\phi,CPL)},$$

which gives rigorous evaluation measure for the performance of MINLF.

Once a problem instance ϕ is generated, we first obtain the quasi-minimum project duration by our proposed algorithm with MINLF for θ =1.5, 2.0, 2.5, 3.0, 4.0,and 10.0, then find the posteriorly critical path length of the project which we have known after finding out what type of failure has happened for each activity through the execution of our algorithm. We simulate 50 iterations for each problem instance, for different resource availability levels from θ =1.5 to 10.0. The numerical results are shown in Table 7 and each cell denotes the percentage value of estimate.

Table 7: The values of estimate obtained with MINLF and CPL

θ	N =10 C =2	N =20 C =2	N =30 C =3	N =40 C =3	N =50 C =4	N =60 C =4	N =70 C =5	N =80 C =6	N =90 C =6	N =100 C =6	N =120 C =8	N =150 C =8
θ=1.5	9.06	37.84	44.63	40.03	90.08	102.42	88.91	143.22	95.33	116.99	158.87	151.86
θ=2.0	0.00	3.70	6.87	5.93	34.34	30.24	30.13	60.88	36.05	42.47	71.61	64.93
θ=2.5	0.00	0.12	0.28	0.25	9.10	13.47	6.96	30.91	16.55	15.09	38.23	30.59
θ=3.0	0.00	0.00	0.02	0.00	2.31	2.07	1.48	11.96	5.14	4.07	19.62	12.80
θ=4.0	0.00	0.00	0.00	0.00	0.72	0.23	0.23	1.85	0.54	0.87	4.44	0.74
θ=10.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

From Table 7, some results can be found:

1. The percentage values in Table 7 are near to those in Table 6 on the same problem instance for each resource availability level.
2. For θ =10.0, when sufficient resources are supplied, the values of estimate are all equal to zero. That is to say, we can find that the minimum project duration is equivalent to the critical path length if sufficient resources are given.
3. For θ = 4.0, when a considerably high resource availability level is given, the critical path length is very close to the minimum project duration. The values of estimate are very small, at most 4.44%.
4. For θ = 3.0, when a larger amount of available resources is given, we find that the values of estimate fall below 20%.
5. For a rare case of resource availability, the values of estimate become very large. It can not be denied that the main factor to make the values of estimate increase sharply is $T_{conflict}(\phi,\omega)$ for small θ.

From the above-mentioned results, we can make a conclusion that the quasi-minimum expected project duration obtained by our algorithm with policy MINLF is close to the posteriorly minimum expected project duration at least for large θ.

5. Conclusion

This paper considers the problem of scheduling an R&D project which consists of more than one activity. We assume that (1) each activity may be performed in one of a variety of resource-duration modes, and that (2) an unsuccessful activity will be reattempted once more in one of the different combinations of resources and duration depending on failure type. Our goal is to find the quasi-minimum expected project duration subject to limited amount of resources and precedence relations among activities under uncertainty.

In this paper we propose a heuristic approach which includes two dispatching rules that decide which activity in an eligible set should be scheduled first and in which mode it should be performed. The method of generation a problem instance a project have been presented and used to evaluate our proposed algorithm. Then a two-stage sampling statistical procedure is used to choose the policy with the smallest expected project duration. From the results, we find that MINLF, the best policy with the smallest weighted sample mean, occurs with a considerably high frequency. So we can make a conclusion that the algorithm with the best policy MINLF is the most likely to produce the minimum expected project duration. Furthermore, in order to evaluate the quality of solution obtained by our proposed algorithm, we used a posterior analysis and calculated the values of estimate in twelve problem instances. From the analysis, we can make a conclusion that the quasi-minimum expected project duration obtained by our algorithm with MINLF policy is close to the posteriorly minimum expected project duration.

Acknowledgements The authors would like to thank the referees sincerely for valuable comments, especially for an essential suggestion which supported us to improve the contents of Section 4 substantially.

References

[1] Christofides, N., Alvarez-Valdes, R. and Tamarit, J. M., Project scheduling with resource constraints: a branch and bound approach, *European Journal of Operational Research*, Vol. 29(1987)262-273.

[2] Davis, E. W. and Heidorn,G. E., Optimal project scheduling under multiple resource constraints, *Management Science*, Vol. 17(1971)B803-B816.

[3] Demeulemeester, E., Dodin, B. and Herroelen, W., A random activity network generator, *Operations Research*, Vol. 41(1993)972-980.

[4] Demeulemeester, E. and Herroelen, W., A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, *Management Science*, Vol. 38(1992) 1803-1818.

[5] Drexl, A., Scheduling of project networks by job assignment, *Management Science*, Vol. 37(1991)1590-1602.

[6] Drexl, A., and Gruenewald, J., Nonpreemptive multi-mode resource-constrained project scheduling, *IIE Transactions*, Vol. 25(1993)74-81.

[7] Elsayed, E.A. and Nasr, N.Z., Heuristic for resource-constrained scheduling, *International journal of production research*, Vol. 24(1986)299-310.

[8] Law,A.M. and Kelton,W.D., Simulation modeling and analysis (2nd ed.), MacGraw-Hill Book Co., Singapore, 1991.

[9] Patterson, J. H., Talbot, F. B., Slowinski, R., and Weglarz, J., Computational experience with a backtracking algorithm for solving a general class of precedence and resource-constrained scheduling problems, *European Journal of Operational Research*, Vol. 49(1990)68-79.

[10] Sprecher, A., *Resource-constrained project scheduling: exact methods for the multi-mode case*,Springer-Verlag, Berlin Heidelberg, 1994.

[11] Stinson, J. P., David, E. W. and Khumawala, B. M., Multiple resource-constrained scheduling using branch and bound, *AIIE Transactions*, Vol. 10(1978)252-259.

[12] Talbot, F. B., Resource-constrained project scheduling with time-resource tradeoffs: the nonpreemptive case, *Management Science*, Vol. 28(1982)1197-1210.

[13] Talbot, F. B. and Patterson, J. H., An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems, *Management Science*, Vol. 24(1978)1163-1174.

[14] Tseng, C.C., Mori, M., and Yajima, Y., A project scheduling model considering the success probability, in:M. Fushimi and K. Tone (eds), *Proceedings of APORS'94*. (World Scientific Publishing Company, 1995)399-406.

Masao Mori
Department of Industrial Engineering and Management,
Graduate School of Decision Science and Technology,
Tokyo Institute of Technology.
2-12-1 Oh-okayama, Meguro-ku,
Tokyo 152, Japan.
E-mail: mori@me.titech.ac.jp