社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2005－MPS－55（5）
2005／6／28

資源制約付プロジェクトスケジューリング問題の拡張モデルに対するヒューリスティックな解法

草部 博輝[†]，中森 眞理雄[†]

[†] 東京農工大学大学院工学教育部

資源制約付プロジェクトスケジューリング問題 (RCPSP) はいくつかのスケジューリング問題の一般系モデルとして知られている．我々は本稿において，RCPSP の拡張モデルを，“RCPSP/τ+”として提案，定式化する．さらに，RCPSP/τ+ モデルに対するヒューリスティックなアルゴリズムを提案し，その性能を評価する．

A heuristic algorithm for extended model of RCPSP

Hiroaki KUSAKABE[†], Mario NAKAMORI[†]

[†] Faculty of Engineering, Tokyo A&T University

Resource-constrained project-scheduling problem (RCPSP) is a general model of the scheduling problem. In this paper, we show a extended model of RCPSP and show a fomulation of this model. We call this model *“RCPSP/τ+”* and present a metaheuristic algorithm for this model.

1 Introduction

The resource-constrained project scheduling problem (RCPSP) is well known as a general model of job-shop scheduling. Until today, several results by metaheuristic algorithms have been reported [2) 4)] . Also, some modified model of RCPSP have been suggested [1) 3)] . In this paper, we introduce *RCPSP/τ+*, an extended model of the RCPSP.

The RCPSP/τ+ is stated as follows. A project consists of n *activities*. Activities are labeled as $j = 0, \ldots, n-1$. Two activities $j = 0$ and $j = n-1$ represent the start and end of the project, respectively. For each activity, the *processing time*, the *resource requests*, and the *precedence relations* with other activities are given, whereas preemption is not allowed. For each type of resource, the availability is given. The availability of each resource is not the same in each time period. The resource requests of each activity also changes in period of its processing time. Both activities $j = 0$ and $j = n-1$ are *dummy*, whose processing time is 0. We add constraints of minimum and maximum time lags. Consider two activities i and j where i precedes j. In some cases, activity j must start within some time period after the end of the activity i. Such a constraint is called *within constraint*. In other cases, activity j can start more than some time period after the end of the activity i. Such a constraint is called *after constraint*.

A simple precedence constraint is taken as a special case of after constraint with waiting time is 0. All information on processing time, precedence relations, and resource requests and availabilities, within and after constraints are assumed to be deterministic and known in advance. The objective is to minimize the project's make span, i.e. the end time of the activity $n-1$.

In this paper, we propose the heuristic algorithm for RCPSP/τ+.

2 Problem Formulation

In this section, we show a formulation of RCPSPτ+. This can be formulated as a 0-1 integer programming problem.

minimize

$$z = \max_j \{ \sum_{t=0}^{t_{max}-1} tx_{jt} + p_j \} \quad (1)$$

subject to

$$\sum_{t=0}^{t_{max}-1} tx_{st} \leq \sum_{t=0}^{t_{max}-1} tx_{jt} + p_j + w_{js}, \quad j \in J, \ s \in S_j \quad (2)$$

$$\sum_{t=0}^{t_{max}-1} tx_{st} \geq \sum_{t=0}^{t_{max}-1} tx_{jt} + p_j + a_{js}, \quad j \in J, \ s \in S_j \quad (3)$$

$$\sum_{j=0}^{n-1} \sum_{u=0}^{\min(t,p_j-1)} d_{jru} x_{j(t-u)} \geq l_{rt}, \quad t \in T, \ s \in S_j \quad (4)$$

$$\sum_{t=0}^{t_{max}-1} x_{jt} = 1, \quad j \in J \quad (5)$$

$$x_{jt} \in \{0,1\}, \quad j \in J, \ t \in T \quad (6)$$

Here, meaning of each constants and variable are as follows:

- n : Number of activities.
- m : Number of kind of renewable resources.
- J : Set of all activities.
- t_{max} : The project term length.
- T : Set of time periods in term of project. $T = \{0, \dots t_{max} - 1\}$
- p_j : Processing time of activity j.
- S_j : Set of successors of activity j.
- w_{js} : Grace of starting of activity $s \in S_j$ after completion of activity j. The activity $s \in S_j$ must start from completion of activity j to elapse of w_{js} time periods.
- a_{js} : Waiting time of starting of activity $s \in S_j$ after completion of activity j. The activity $s \in S_j$ can start after elapse of a_{js} time periods from completion of activity j.
- d_{jru} : Activity j's request of resource r in uth period of its processing time.
- l_{rt} : Limit of renewable resource r at time t.
- x_{jt} : If the activity j starts at time t, then 1, otherwise 0.

Inequalities (2) and (3) are within, after constraint, respectively. Inequality (4) is resource constraint, and inequalities (5) and (6) are non-preemptive constraint.

3 The Algorithm

In this section, we propose an algorithm for the RCPSP/τ+. We consider two phases, *initial solution* phase and *improvement* phase.

3.1 Initial Solution

To construct the initial solution, we implement the *single path method* with *tabu list*. First, we try to dispatch activities using *single path method*. If a feasible solution is obtained, this phase terminates. Otherwise, we record predecessor of activities which has not been dispatched and its start time to the tabu list. Let J^- be the set of activities which was not dispatched. If $|J^-| \geq 2$, select one activity $j \in J^-$ which has the highest priority in J^- and all its predecessors have been dispatched. Here, we assume that activity $j^- \in J^-$ is selected. Let P_j, P_j^w and P_j^a be the set of all predecessors of activity j, which are imposed whithin constraint and which are imposed after constraint, respectively. Select one activity $k \in P_{j^-}$ according to steps below.

step 1 If $|P_{j^-}| = 0$ is satisfied, go to step 3. Otherwise if $|P_{j^-}^w| \geq 1$, select k which has the smallest value of (7), terminate.

$$\sum_{t=0}^{t_{max}-1} tx_{kt} + p_k + w_{kj^-}, k \in P_{j^-} \quad (7)$$

Otherwise, go to step 2.

step 2 If $|P_{j^-}^w| = 0$ and $|P_{j^-}^a| \geq 1$ are satisfied, select $k \in P_{j^-}^a$ which has the smallest starting time and the highest priority, terminate. Otherwise, go to step 3.

step 3 Search the activity $k \in S_0 \setminus j^-$ according to priority descending order. If $k \in S_0 \setminus j^-$ which has been imposed after constraint is found, select it. Otherwise, select one which has the highest priority. Terminate.

After recording attribute to the tabu list, iterate the single path method referring to the tabu list. If the record of activity j and its starting time t is found in tabu list, activity j is not allowed to be dispatched at starting time t. This iteration terminates when a feasible solution is found or the number of iteration exceeds the upper limit given in advance. We call this dispatching rule *Single Path Method with Tabu list 1* (SPMT1). We give the priority used in SPMT1. At the first trial in iteration of SPMT1, priority

is given exceptionally in the ascending order of value stated as follows,

$$\sum_{r=0}^{m-1}\sum_{t=0}^{p_j-1} d_{rju}. \tag{8}$$

Expression (8) represents the sum of the required resources of each activity j. From the second trial onward, we give the priority as descending order of the sum of predecessors and successors which are imposed within constraint. SPMT1 outputs feasible or infeasible solution and dispatching sequence of activities.

3.2 Improvement of Solution

Next, we try to improve the schedule obtained from SPMT1. Since a schedule which SPMT1 outputs is made from a *dispatching sequence*, we change the dispatching sequence and try to build solution. Changing dispatching sequence is implemented according as each neighborhood discussed below.

After doing neighborhood operation, we execute dispatching called SPMT2. SPMT2 is the same as SPMT1 except that it is not allowed to change the dispatching sequence in SPMT2. SPMT2 use the dispatching sequence instead of priority of activities. If SPMT2 cannot build a feasible solution using any dispatching sequence, SPMT2 outputs value of expression (9) as objective function value.

$$\sum_{j=0}^{n-1} t_{max}(1 - \sum_{t=0}^{t_{max}-1} x_{jt}) \tag{9}$$

3.3 The Neighborhood

In this subsection, we present three neighborhoods for the RCPSP/τ+. These neighborhoods are defined as changing dispatching sequence. Let $\pi(i)$ be activity whose sequence number is i, and for each $k \in J$, let $j_k^{'} := \pi(k)$.

First, let us introduce *2swap* neighborhood. We show the step of 2swap neighborhood operation below.

step 1 Select two activities whose numbers of dispatching sequences are i_1 and i_2. Let be $j_1^{'} := \pi(i_1)$ and $j_2^{'} := \pi(i_2)$. Go to step 2.

step 2 Let $\pi(i_1) = j_2^{'}$ and $\pi(i_2) = j_1^{'}$, terminate.

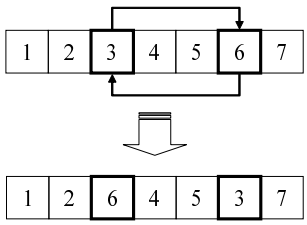

Figure 1: 2swap neighborhood

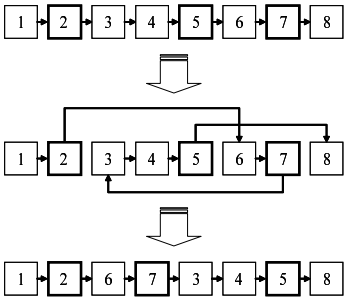

Figure 2: 3select neighborhood

An example of neighborhood operation in the case of $i_1 = 3$ and $i_2 = 6$ is shown in Figure 1.

Second, let us introduce *3select* neighborhood. We show the step of 3select neighborhood operation below.

step 1 Select three activities whose number of sequence are i_1, i_2 and i_3, where $i_1, i_2, i_3 \in J \setminus \{0, n-1\}$ Assume that $i_1 < i_2 < i_3$. Let $i := 0$. Go to step 2.

step 2 Let $\pi(i) := j_i^{'}$. If $i = n-1$, terminate. Otherwise, go to step 3.

step 3 If $i \in \{i_1, i_2, i_3\}$, go to step4. Otherwise, let $i := i + 1$ and go to step2.

step 4 If $i = i_1$, let $i = i_2 + 1$. Else if $i = i_3$, let $i = i_1 + 1$. Else if $i = i_2$, let $i = i_3 + 1$. Go to step 2.

An example of neighborhood operation in the case of $i_1 = 2$, $i_2 = 5$ and $i_3 = 7$ is shown in Figure 2.

3.4 Tabu Search

We implemented tabu search algorithm using the above three neighborhoods, i.e. shift/insertion, 2swap and 3select. For each neighborhood, attribute recorded in tabu list is different. Attribute of shift/insertion neighborhood is activity and its new number of dispatching sequence. As attribute of 2swap, we use the two activities which is exchanged number of sequence, and

that of 3select is three numbers of dispatching sequence of selected activities. If dispatching sequence which improve objective function value is found in neighborhood, these are updated immediately.

The algorithm terminates if number of iteration exceeds the upper limit given in advance.

4 Numerical Experiment

We generated 50 RCPSP/τ+ instances randomly. These instances include *dummy* activities. All 50 instances have the value of $t_{max} = 400$.

These are based on bench mark instances which are released from *PSPLIB*. Using these instances, we compared our algorithms with ILOG CPLEX 8.0. The maximum computational time of ILOG CPLEX is 50000 seconds for each instance. So, solution of ILOG CPLEX is not exactly optimal solution in some instances.

All instances have feasible solution. Algorithms examined are SPMT1 and tabu search. All tabu search used dispatching sequence of SPMT1 as initial condition. The neighborhoods implemented to tabu search are 3select after 2swap (2swap-3select TS), and 2swap after 3select (3select-2swap TS). For each algorithms, we use parameters presented as follows. Number of iterations of SPMT1 and SPMT2 is 500. Length of tabu list used in ins. TS and 2swap TS is 50, used in 3select TS is 10. Maximum iteration number of tabu search using one neighborhood is 100 and using two neighborhoods is 50 for each neighborhoods. For example, in the case of ins.-2swap TS, 2swap TS is iterated 50 times after iterating ins. TS 50 times.

ILOG CPLEX was executed on a Red Hat Linux, Pentium Ⅲ 1GHz CPU, 1GB memory. Our algorithms are implemented on Windows XP SP1, Pentium 4 2.6GHz CPU, 1GB memory with C language.

In table 1, the column "feasible" represents a number of feasible solution which each algorithms produced. The column "cpu time" represents average cpu time[sec]. The column "gap" represents the average of relative gap[%] of objective function value. In addition, ILOG CPLEX failed to give optimal solution about some instances.

Table 1: Result of each algorithm and comparison with ILOG CPLEX

algorithm	feasible	cpu time	gap
CPLEX	44	23672.96	-
SPMT1	46	0.037	11.86
2swap-3select	50	354.46	2.14
3select-2swap	50	351.97	1.66

ILOG CPLEX gave the feasible solution to 44 instances of 50. CPU time of SPMT1 is very fast. But gap is not good. 3select-2swap TS outputs a good result.

5 Conclusions

In this paper, we presented ERCPSP/τ+ which is an extended model of RCPSP and formulated this problem as a 0-1 integer programming problem. We then proposed a tabu search algorithm and presented result from computational experiments. The parameter setting and implementation of the algorithm superseding the SPMT2 are left for further research.

References

1) José A. and Daecheol Kim, Parallel machine scheduling with earliness-tardiness penalties and additional resource constraints. *Computers & Operations Research 30*, pp. 1945-1958, 2003.

2) Nonobe K. and Ibaraki, T., Formulation and tabu search algorithm for the resource constrained project scheduling problem. *Essays and Surveys in Metaheuristics (MIC'99)*, pp. 557-588, 2002.

3) S. Harthman, *Project Scheduling under Limited Resources Models, Methods, and Applications.* Springer-Verlag Berlin, Heidelberg, 1999.

4) Vicente Valls, Sacramento Quintanilla and Francisco Ballestín, Resource-constrained project scheduling: A critical activity reordering heuristic. *European Journal of Operational Research 149*, pp. 282-301, 2003.