An Extended Petri-Nets Method for Resource Constrained Multi-Project Scheduling under Certain and Uncertain Duration

Abstract

This paper proposes an extended Petri-Nets method for resource constrained Multi-project scheduling problem (RCMPSP). Several important issues about Petri-Nets on RCMPSP are discussed in this paper: (1) The most important elements in Petri-Nets on RCMPSP, including waiting place, activity place, resource place and final place, are designed. At the same time, three kinds of transitions are introduced, which are coordination transition, resource allocating transition and resource releasing transition. (2) The Petri-Nets are improved according to the varying with increase or decrease of the resource allocation number. In the extended Petri-net, the difference with the conventional Petri-Nets is to add a place and replace a releasing transition by the delaying transition and the transition rules. Extensive experiments are carried out to test the validity and performance of the proposed method. It is revealed that the extended Petri-Nets achieve a better performance than the other state-of-the-art methods. The real case studies also reveal that the proposed method deserves better ability in real case solving than the other typical methods.

Keywords: RCMPSP; Petri-Nets; duration; Modelling; Uncertain

1 Introduction

RCMPSP is widespread in engineering fields, such as job shop scheduling, flexible manufacturing systems and so on. It tries to find how to allocate resources reasonably in order to arrange the execution of each process and achieve the optimization goals of the shortest execution duration for project with timing constraints and resource constraints. In real engineering projects, there are usually multiple projects in parallel and competing resources among projects, which increase the complexity of project scheduling. The resources allocation is necessary to consider the project itself and among projects. In addition, there is competition among the executed duration of projects, which leads to use some scheduling methods of the single project problem to the multi-project very difficultly (Kurtulus et al.1982). So how to model and solve multi-project in the environment of resource constrained is the key point of this research field. This paper will focus on the modeling technology and do some research on the RCMPSP solution performance.

In order to minimize the maximum completion time of the RCMPSP, the approach based on the integer programming (IP) with Lagrange multiplier has attracted great attention where some multi-constraints in RCMPSP. Although IP is an effective method, the RCMPSP formulation is not easily made for feasible solution, since RCMPSP is inherently the IP function with dynamically changing resources constraint. So many constraint properties have been ignored in this method. Meta heuristics such as Genetic Algorithm (Hartmann 1998; Hartmann 2002; Della et al. 1995), Tabu Search (He et al.1997; Verhoeven 1998; Barnes et al.1995), Particle Swarm Optimization (Zhang et al.2005; Alcaraz et al. 2001; Hu et al.2013) and Bee Swarm Optimization (Ziarati et al. 2011) have also been used for the RCMPSP.

Although all of these algorithms have been easily formulated, they often leaded to infeasible answer or not good solutions, taking generally very long optimization searching time. These would be more remarkable when the scale of RCMPSP becomes more and more big, and more and more constrains are taken into consideration. Because it is difficult to describe some constraints of practical problem using mathematical methods, the application of mathematical programming method is gre atly limited. The system information needed for heuristics methods is limited in the engineering projects. The heuristics algorithms have defects to solve these problems. Artificial intelligent algorithms can often generate a feasible solution, and sometimes it is difficult to know whether there is a better solution.

The motivation of this work is to overcome the above difficulties. Petri-Nets are an ideal modeling tool to establish system constraints, to provide system information by mathematical analysis and provide guideline for the scheduling process. Petri-Nets can study the organization structure and dynamic behaviors of the system and focus on the relationship between the various changes that occur in the system. Modern simulation can test the virtual model to more deeply study the actual system. Therefore, the research of Petri-Nets is significant to improve resource utilization and labor productivity, enhance market competitiveness and improve operational efficiency and management level. RCMPSP is a typical complex system problem. Complex system modeling is very important before complex system solution. Petri-Nets can profoundly and succinctly describe the control system by analyzing the system's organizational structure and behaviour. In recent years, Petri-Nets were used to model and analyze various types of scheduling problems and had achieved good results. The typical researches are as follows: Wang (Wang and Wu 2004) put forward a controlled timed Petri-Nets modeling of discrete events in flexible production scheduling, which included process flow subnet, scheduling control subnet and resources subnet, and connected successfully by synchronizing changes. Gradisar (Gradišar and Mušič 2007) classified scheduling rules and established colored Petri-Nets Model for simulation of different scheduling rules. Billington (Billington et al.1991) proposed a colored timed object-oriented Petri-Nets, which added the concepts of colored token and I/O change function on the basis of original Petri-Nets. He also presented some application examples in automated manufacturing systems. For the characteristics of complex product development in multi-project scheduling, Shi (Shi eta al. 2007) proposed the specific ideas and methods of modelling multi-project scheduling by colored Petri-Nets. Chen (Chen et al.2008) proposed a timed object-oriented Petri-Nets modelling method and verified its effectiveness for resource-constrained project scheduling problem. Abdallah (Abdallah et al.2002) proposed a Petri-Nets algorithm that considered the deadlock-free scheduling duration of RCMPSP. The Objective of the Petri-Nets algorithm is to minimize the duration time and the Petri-Nets algorithm gives optimal or near optimal deadlock-free duration scheduling which considered the sequence of the transition firings. Buchholz (Buchholz 2004) proposed a novel approximate method for numerical modelling analysis of stochastic Petri-Nets, which combined numerical method and fixed point computations. In contrast with other approximation solutions, Bucholz's method is adaptive because it considered states with high probability in detail and aggregating states with small probabilities. Chen (Chen et al.2003) used colored Petri-Nets to model and to evaluate the allocation in RCMPSP. The RCMPSP was divided into subclass and decreased the complexity of the model. Zhang (Zhang and Gu 2009) presented a hybrid Petri-Nets method for RCMPSP. Both discrete and continuous RCMPSP variables are considered to use in the Petri-Nets model of RCMPSP. Chen (Chen et al. 2005) proposed a new stochastic Petri-Nets model, which is suitable for modelling and analysis of inventory system and real-time supply chains. Ding (Ding et al. 2006) proposed a novel fuzzy timed Petri-Nets model for RCMPSP, which each of duration was associated with a fuzzy number. The performance analysis of the fuzzy timed Petri-Nets model is based on the reach ability state graph. Ramirez (Ramirez et al. 2007) presented interpreted Petri-Nets to model the events and states of RCMPSP system behaviours. It proposed a scheme utilizing a solution of a programming problem for RCMPSP based on the interpreted Petri-Nets derived from an on-line methodology. Note that all papers in this topic focused on the discrete events modelling technology of RCMPSP, but not considering the resources constraint and uncertainty duration. A series of papers have been recently presented that were based on the assumption that the duration is certainty (Li et al.2003; Lee and DiCesare 1994; Baggio et al.2004).

From the summary of the above current researches, many kinds of Petri-Nets models were used to solve RCMPSP. They focused on some key technologies of RCMPSP and got better results. How to simulate the real RCMPSP system clearly is the most important goal for all researchers. There are many uncertainty duration PCPSP in actual engineering project. Some researchers used fuzzy (Ding et al.2006), probability (Ramirez et al.2007) technologies to transform uncertainty duration problem into certainty duration problem. But all of them couldn't express the detail of the states changing of all processes. In this paper, we use appropriate methods to extend Petri-Nets that are able to simulate the real system clearly according to the characteristics of practical problems. Because of the characteristics of certain duration and uncertain duration in RCMPSP, we take the extended Petri-Nets to model certain duration problem (Chen et al.2008), which divide token into logical token and resource token, and token changing represents the execution of processes and allocation of resources. Through the classification of places and transitions and adding time to places, the places are divided into activity place (AP), resource place (RP), waiting place (WP) and final place (FP). The transitions contain coordination transition (CT), resource dispatching transition (RDT) and resource releasing transition (RRT). WP and AP reflect the timing relationship among tasks by taking a balance of transaction connection. The resources competition is reflected by taking up and release for resources among tasks by RP, RDT and RRT. According to the uncertainty problem's characteristics, we add the place and transitions type, modifying rules of transition of triggering to the extended Petri-Nets. The novel strategy is presented in the extended Petri-Nets to describe the actual system of uncertainty duration problem. The main contributions of this paper are as follows.

(1) An extended Petri-Nets method is proposed to model the RCMPSP under certain and uncertain duration.

(2) Place, transition types and transition firing rules are added to the extended Petri-Nets to describe the uncertainty.

The rest parts of this paper are organized as follows. Section 2 details the RCMPSP model under certain and uncertain duration by extended Petri-Nets. Section 3 presents extensive numerical experiments to evaluate the proposed methods. Section 4 presents a real engineering case study. Section 5 draws a conclusion and perspectives.

2 Modelling

2.1 Scheduling processes of RCMPSP

There are 4 task sets in the schedule of multi-projects, which are □ Finishing task set (*Fs*); □ Going task set (*Gs*); □ Waiting task set (*Ws*) (They are the collection of tasks to satisfy the timing constraints but not satisfy the resource needs.); □ Non-executable task set (*Ds*). The task sets are shown in Fig.1.

Gs's tasks occupy some resources in task sets of multi-project scheduling process. When *Gs*'s tasks are completed they will be transferred into *Fs*. *Fs* release these resources for *Ws*'s tasks. The entire scheduling process is presented as following:

(1) In the initial stage, *Fs* contain the first virtual task. The following tasks are all in the *Ws*. the remaining tasks are in the *Ds*.

(2) Based on shared resources, the scheduling process select one or more tasks from *Ws* to execute, and give it the required resources. In the end we transfer these tasks into

the *Gs* and occupy resources.

(3) If some tasks are completed in *Gs*, we transfer these tasks into *Fs* as well as release resources.

(4) Recalculating the remaining amount of resources.

(5) When some pre-tasks of current tasks are completed, the original tasks in *Ds* will be transferred into *Ws*.

(6) We repeat steps 2-5 until *Ds* and *Ws* are empty.

(7) Finally, when all remaining tasks in the *Gs* are completed and transferred to *Fs*, the last task's completion time is the entire project's completion time.

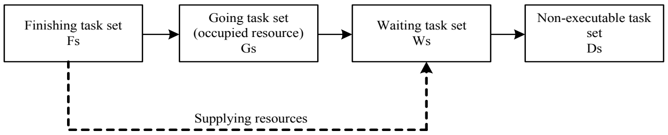


Figure1. Task sets of multi-project scheduling processes

2.2 Certain duration

2.2.1 Problem description and modelling assumptions

For RCMPSP with certain duration, it can be described as follows: there are *R* kinds of shared resources for the projects. It assumed that there are *P* projects. The implementation of task is based on occupying one resource. The daily supply of all resources is limited. The competition for resources is not only reflected among tasks in projects, but also reflected between different projects. So the goal of management is as follows: how to allocate these shared resources to different tasks in projects by effective resource scheduling in order to complete all these projects in a shortest makespan.

Before mathematical modelling, it is necessary to make a reasonable simplification for the complex practical problems. For the research convenience, we define several model assumptions as follows.

(1) Projects are not independent, which means there is no predecessor relationship among projects.

(2) In each project only one shared resource is needed to execute the task.

(3) The execution time of each task is determined when the resource supply is satisfied.

(4) All shared resources are renewable resources that mean the occupied resources will be released after the ending of task.

(5) The daily supply for each shared resource is fixed, including the type and quantity.

(6) Executing task cannot be interrupted.

2.2.2 Mathematical description of model

Symbol definitions in the description:

X represents project, X_i represents the *ith* project, the number of projects is P, M_i represents the total number of items in the *ith* project.

j represents task number in each project.

d_{ij} represents duration of *jth* task in *ith* project.

T_{iMi} represents completed time of the last task in *ith* project.

B_{ij} represents starting time of *jth* task in *ith* project.

E_{ij} represents ending time of *jth* task in *ith* project.

F_{ij} represents predecessor task set of *jth* task in *ith* project.

t represents time serial number, $G(t)$ represents execution status of tasks set in time *t*.

k represents resource serial number, K represents the total number of resources required by projects.

R_k represents *kth* resource daily supply.

r_{ijk} represents number of *kth* resource required by *jth* task in *ith* project.

The mathematical representation of decision-making goal is that the total duration to execute projects P is shortest, which is shown in equation 1.

$$\min\left\{\max_{i=1}^{P}\left\{T_{iM_i}\right\}\right\} \qquad (1)$$

Mathematical representations of the decision-making constraints are as follow.

(1) Timing constraints between the internal task executions of project is shown in equation 2.

$$B_{ih} + d_{ih} \le B_{ij}, \forall h \in F_{ij}, \forall i, j \qquad (2)$$

(2) Resource constraints, the demand for resources can't be greater than the supply of resources in every moment, which is shown in equation 3.

$$\sum_{i=1}^{P}\sum_{j=1}^{M_i} r_{ijk} \le R_k, \forall i, j \in G(t) \qquad (3)$$

2.2.3 The extended Petri-Nets model for certain duration

According to the characteristics of RCMPSP with certain duration, this paper constructs extended Petri-Nets with the object of multi-project tasks set. The extended Petri-Nets can be described as 9-tuple.

$$EXPN = (P, T, TYPEP, TYPET, A, W, ET, resourceAmount, S_0)$$

Where, $P = \left\{P_1, P_2, ..., P_m\right\}$ represents limited places set, $T = \left\{T_1, T_2, ..., T_n\right\}$ represents limited transition set. *TYPEP* and *TYPET* represent type of place and type of transition. We mainly construct three types of places and types of transition. The place type and transition type are shown in Figure 2 and 3.

An activity of project is represented by two places. One is named *WP*, the other is named *AP*. *WP* is front of *AP*. *EXPN* is also defined by two token types, which called logical token (*LT*) and resource token (*RT*). *LT* in *AP* represents that task is running. *LT* in *WP* represents that task is waiting resources. The resources will be diverted to AP once they occupy the resources.

In addition, a *RP* represents a type of resource. The *RP* token is a resource token. It represents the number of available resources. *FP* represents ending of project. *FP* has a logical token when all of projects have ended. In addition to *RP*, each place only belongs to one of the projects.

CT is used to connect to *WP*, *AP* and *FP*. *CT* is equivalent to the process of coordinating project running.

RDT is used to connect the input *RP*, *WP*, output *RP* and schedule resources. *RRT* is used to connect *AP*, *RP* and recycle resources. A represents the collection of arcs between places and transitions. *W* represents the collection of weight on these arcs and the number of tokens I/0, which is set to default 1 for simplify. When weight is 1 it is not impossible to mark weight on arcs. In the certainty duration problem, as long as the tasks in each project occupy sufficient resources, the execution time is determined. So each place in $EXPN$ is given a time, which is represented by ET . In *AP*, *ET* represents spending time from start to end in *AP*. *ET* is set to 0 in *WP*, which represents that *WP*'s logical token will be immediately transferred to the *AP* once it has the resources. *RP*'s *ET* is also set to be 0, which indicates that the resource allocation is not delay. *FP*'s *ET* is set to infinity, because once it has a logical token, all of projects have ended and then the token will not be released. *resourceAmount* represents the number of resources in RP that is the number of resource tokens. S_0 represents the initial status of $EXPN$, which is described by three-tuples $\left\langle M_0, AET_0, RA_0 \right\rangle$, M represents places' (in addition to *RP*) token status. In addition to *RP*, other places can only have one token (logical token), so the token status of each place can be represented by {0, 1}. AET represents projects' continued time. AET represents the time spent by all projects implementation when the token status of *FP* is 1. RA represents the number of resources in *RP*.

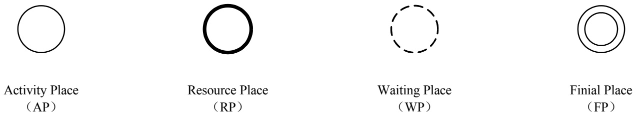


Figure2. Four kinds of places

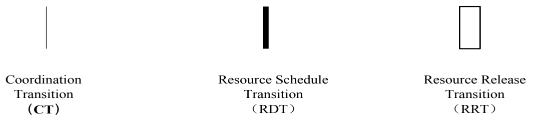


Figure3. Three kinds of transition

In RCMPSP, there is always competition for some resources among projects. The relationship is shown in Figure 4.

Figure 4 shows that the execution of task *a* and task *b* need to compete for resource RP_1 . Task *a* needs n_a resources. Task *b* needs n_b resources. The number of RP_1 is n . When WP_a and WP_b get tokens (logical token), it represents that task *a* and task *b* satisfy the predecessor relationship. When resources are satisfied, tokens of WP_a or WP_b will immediately be transferred to AP_a and AP_b . There are four situations as follows.

(1) $n \geq n_a + n_b$. It represents that resources is enough and can satisfy task *a* and task *b*. when RDT_1 is triggered, n_a tokens and n_b tokens of RP were transferred to RP_a and RP_b . Once RP_a and RP_b have tokens, T_2 and T_5 is triggered. WP_a and WP_b is transferred to AP_a and AP_b .

(2) $n \geq n_a$ but $n < n_b$; or $n \geq n_b$ but $n < n_a$. It represents only task *a* or task *b* is satisfied. It assumed that task *a* is satisfied. When RDT_1 is triggered, only n_a tokens in RP_1 is transferred to RP_a . Once RP_a has tokens, T_2 is triggered and tokens of WP_a is transferred to AP_a . Task *a* is implemented. On the contrary, if RP_b cannot get tokens, T_5 can't be triggered, and task *b* can't be implemented, so task b will be waiting resources.

(3) $n \geq n_a$ and $n \geq n_b$ but $n \leq n_a + n_b$. It represents resources can satisfy task *a* and task *b*, but can't satisfy both together. RDT_1 is triggered and takes *a* random strategy. Tokens in RP transferred randomly, so it is possible n_a tokens are transferred to RP_a or n_b tokens are transferred to RP_b . In other words, it is random to execute one task and the other task will wait for resources to be released.

(4) $n < n_a$ and $n < n_b$. It represents task *a* and *b* can't both be satisfied, so RDT_1 can't be triggered and task *a* and task *b* can't be implemented.

In addition, d_a and d_b in Figure 4 represent execution time of tokens in AP_a and AP_b . When AP_a tokens enter it and spend d_a time, AP_a ' token is transferred to other places. RRT_a is triggered. RP_1 is added n_a tokens. These situations represent task *a* is completed, and resources of task *a* are recycled. That is the same reason for AP_b .

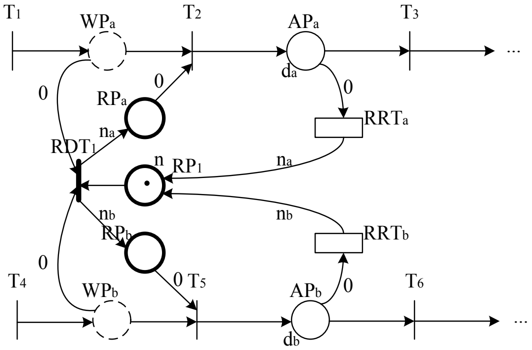


Figure4. Resource competition of certain duration

2.3 Uncertain duration

2.3.1 Problem description and assumptions in modelling

In the uncertain duration problem, the task duration of projects is uncertain, which is always related to the number of shared resources occupied. The more allocated to tasks, the less task execution duration, which tend to more realistic. In other words, when the amount of resources allocated to tasks are different, the duration of tasks is different, which is referred to the uncertainty duration. It can be described as: there are *R* kinds of shared resources, which are supplied to all projects waiting to start. It assumes that there are *P* kinds of projects, in which tasks' execution depends on occupying amount of one resource. All of resources are supplied limited. This competition for resources is not only reflected between each task in project, but also reflected between different projects. Besides, task execution duration of projects is uncertain, or variable, if a task gets more of resource, then other tasks will obtain fewer resources. Which project gets more resources will accelerate the completion, and others will postpone completion. Management objectives are as follows: under the condition of this variable duration, how to allocate these shared resources to different tasks in projects and make the completion duration of these projects be shortest by effective method of resource coordinated allocation. Compared with the certain duration, this problem adds an assumption, that is, within a certain range, the duration of tasks, which use shared resources, is inversely proportional to the number of resources.

2.3.2 Mathematical description of model

In addition to the definitions of certainty duration problem, the following symbol definitions are changed and added.

d_{ij} , is changed. It represents the execution time of someone task, which does not need shared resources.

H_{ijk} , is changed. It represents the daily maximum demand of *kth* resource for *jth* task of *ith* project.

Q_{ijk} , is changed. It represents the total demand of *kth* resource for *jth* task of *ith* project.

D_{ij} , is changed. It represents the actual execution time of *jth* task of *ith* project.

Mathematical representation of decision objective is also that the total duration for P project being executed is shortest, which is shown in equation 4.

$$\min\left\{\max_{i=1}^{P}\left\{T_{iM_i}\right\}\right\} \qquad (4)$$

(1) Mathematical representation of decision constraint condition is the relationship among task execution time, total amount demand of someone resource and daily assigned amount of this resources, which is shown as follows.

$$D_{ij} = \begin{cases} \dfrac{Q_{ijk}}{r_{ijk}} & , \textit{when task use shared resources} \\ d_{ij} & , \textit{when task don't use shared resources} \end{cases} \qquad (5)$$

(2) Timing constraints between the internal task executions of project is shown in equation 6.

$$B_{ih} + D_{ih} \le B_{ij}, \forall h \in F_{ij}, \forall i, j \qquad (6)$$

(3) Resource constraints, the demand for resources can't be greater than the supply of resources in every moment, which is shown in equation 7.

$$\sum_{i=1}^{P}\sum_{j=1}^{M_i} r_{ijk} \le R_k, \forall i, j \in G(t) \qquad (7)$$

(4) The daily demand of shared resources of tasks is limited, which is shown in equation 8.

$$0 \le r_{ijk} \le H_{ijk} \qquad (8)$$

2.3.3 The extended Petri-Nets model for uncertain duration

According to the characteristics of the uncertain project duration, it improves the extended Petri-Nets model on certain duration. It refines the repository and divides it into Global Repository Place (*GRP*) and the Local Repository Place (*LRP*). And then, the *GRP* represents a type of global resources, and the inside token (resources token) indicates the number of a certain resource that is the number of the current available resource. The *LRP* represents the current situation about the resource occupied by a task, which need share resources, and the number of token indicates the amount of a certain resource occupied by the task execution. In addition, the local repository pool contains two properties, which are the maximum amount of a certain resource can be assigned every day when task executes and the total amount of a certain resource required by the task execution. The number of transferring to the inside token is determined by the scheduling transition of the resource connected to it. Thus, five types of the place are shown in Figure 5.

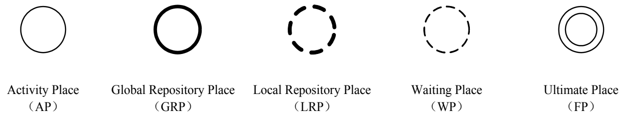


Figure5. Five types of the place

The classification of the transition is also improved. Three types of transition are shown in Figure 6.

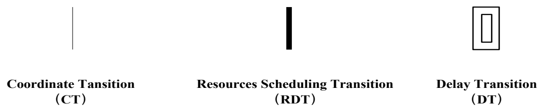


Figure6. Three types of the transition

And the way of the resource scheduling in the *RDT* is different because it schedules the resource mainly by the attributes connected to the *LRP*. The *DT* replaces the *RRT* described by the previous section. It also connects the *WP*, the *LRP*, the *AP* and the *GRP*. It contains not only the feature of releasing resources, but also the feature of delay and the logistic token transition (coordinating the project process). When the *LRP* connected to it includes tokens, it indicates that the task occupies some resources and begins to execute. At present, the delay time of the *DT* manifests the execution time that is required by the procedure that the task occupy these resources. Because of that delay time, the *WP* to which it connects forward and the token in the *LRP* can happen to transfer and they transfer to the *AP* to which it connects backward and the *GRP*. That indicates the end of the task execution.

In the RCMPSP under the environment of the uncertain project duration that kind of the uncertain project duration is reflected in the task requires sharing resources. Thus, the execution time of the task, which is not need sharing resources, is sure so that every place is given to the concept of time in the extended Petri-Nets. The *ET* manifests that time and is called execution time. If the *ET* in the Activity Place is zero, it indicates that the execution of that activity requires the Delay Transition connected to it determines sharing resources and its execution time. Otherwise, it indicates that the execution of that activity is not need sharing resources so the *ET* represents the time takes by the activity from begin to the end. In that improvement Petri net model, the competitive relation between tasks is shown in Figure 7.

Figure 7 shows four tasks *a*, *b*, *c*, *d*, where the execution of c and d does not require sharing resources and their execution time are *dc* and *dd*. On the contrary, the execution of task *a* and *b* requires the competitive resource *RP1*. And a new concept is proposed. It means that every task requires resources has a maximum amount of resources that it can obtain every day. When the task obtains over those resources, the task execution time is shortest. The Resource Scheduling Transition RDT_1 mainly responses the assignment of the RP_1 resource. As shown in Figure 7, that WP_a and WP_b obtain the tokens indicates that task *a* and *b* satisfy the tight conditions. At present, RDT_1 will take the allocation strategy that it randomly produces two values x_a and x_b ($0 < x_a \le n_a$, $0 < x_b \le n_b$) according to the attributes n_a and n_b of the Local Repository RP_a and RP_b to which it connects backward. And the attributes means the task *a* and *b* can obtain the maximum amount of the resource RP_1 every day. Then it allocates the resources according to the number of the Global Repository Place RP_1 to which it connects backward. The type of the assignment is similar to the RDT_1 shown in Figure 3 except the competitive objects x_a and x_b .

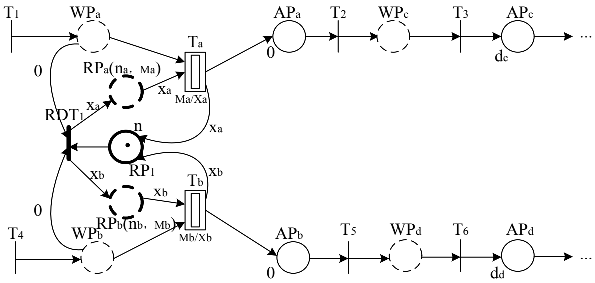


Figure7. The resource competitive relationship

3 Numerical experiments

In order to verify usefulness of the extended Petri-Net modeling method, some numerical experiments were carried out. The experiments simulate the Single Mode Sets Cases in PSPLIB (PSPLIB 2013) library; these cases consist of 30-120 activities. The experiments verified the certain duration RCMPSP are run for benchmarks with j30, j60, j90, and j120 case studies. The numbers of schedules are set as 1000, 5000 and 50,000. Because the durations of the cases in PSPLIB library are certain, in order to verify the usefulness of the extended Petri-Nets of this paper, we change the certainty durations of cases j30, j60, j90 and j120, and get the new benchmarks j'30, j'60, j'90 and j'120 which lose the duration constraints. The experiments verified the uncertain duration RCMPSP are run for benchmarks with j'30, j'60, j'90 and j'120 case studies. The numbers of schedules are also set as 1000, 5000 and 50,000.

3.1 Success rate of solution

Ziarati (Ziarati 2011) showed the success rate of BSO, ABC, BA, BS0-FBI, ABC-FBI and BA-FBI for cases of PSPLIB library. Table 1 shows the comparison of the proposed extended Petri-Nets of this paper with Ziarati (Ziarati 2011) and common Petri-Nets. Table 1 is the average rates of all cases of PSPLIB. Table 2 is average success rate of extended Petri-Nets for certain duration and uncertain duration. On the basis of these results of Table 1 and 2, we can make the following conclusions.

(1) The proposed extended Petri-Nets of this paper have high success rates on the all problems. The performance of the proposed extended Petri-Nets increase as the complexity of the case studies increases.

(2) The certain duration problems have higher success rates than uncertain duration problems. The performance of uncertain durations problems decrease with the increment of complexity the case studies.

Table1. Average success rate for all cases of PSPLIB

Solution	Reference	1000	5000	50,000
(1) the average success rate of j30				
BSO-FBI	Ziarati 2011	83.55%	90.21%	96.25%
ABC-FBI	Ziarati 2011	82.50%	90.00%	95.63%
BA-FBI	Ziarati 2011	83.96%	91.50%	97.09%
Petri-Nets		99.09%	100%	100%
Extended Petri-Nets	This study	100%	100%	100%
(2) the average success rate of j60				
BSO-FBI	Ziarati 2011	72.08%	73.34%	77.09%
ABC-FBI	Ziarati 2011	71.67%	73.34%	76.67%
BA-FBI	Ziarati 2011	72.30%	73.96%	78.13%
Petri-Nets		90.24%	96.78%	96.89%
Extended Petri-Nets	This study	99.43%	100%	100%
(3) the average success rate of j90				
BSO-FBI	Ziarati 2011	72.30%	74.17%	75.42%
ABC-FBI	Ziarati 2011	71.88%	73.96%	75.00%
BA-FBI	Ziarati 2011	72.30%	74.60%	76.25%
Petri-Nets		89.12%	91.47%	92.98%
Extended Petri-Nets	This study	97.09%	98.87%	99.81%
(4) the average success rate of j120				
BSO-FBI	Ziarati 2011	29.17%	30.84%	33.00%
ABC-FBI	Ziarati 2011	29.34%	30.34%	32.67%
BA-FBI	Ziarati 2011	29.84%	31.17%	33.84%
Petri-Nets		65.78%	77.56%	81.75%
Extended Petri-Nets	This study	90.00%	93.36%	94.46%

Table2. Average success rate

Cases	1000	5000	50000
j30	100%	100%	100%
j'30	97.76%	99.65%	100%
j60	99.43%	100%	100%
j'60	92.76%	96.77%	99.34%
j90	97.09%	98.87%	99.81%
j'90	88.80%	90.12%	90.67%
j120	90.00%	93.36%	94.46%
j'120	72.25%	77.89%	80.00%

3.2 Solution speed

This section discusses the solution speed of common Petri-Nets and extended Petri-Nets for PSPLIB. The solution speed is defined as the number of iterations a method requires to solve a PSPLIB instance. Figure 8 shows the solution speed of common Petri-Nets and extended Petri-Nets for certain duration of each case in j30, j60, j90, j120 of PSPLIB. Figure 9 shows the solution speed of common Petri-Nets and extended Petri-Nets for uncertain duration of each case in j'30, j'60, j'90 and j'120.

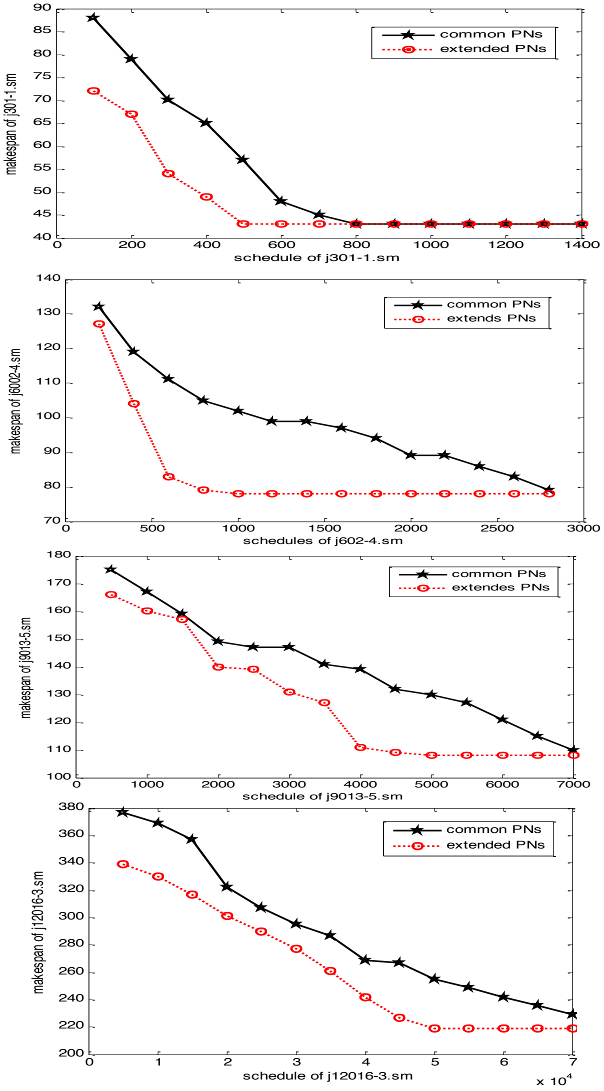
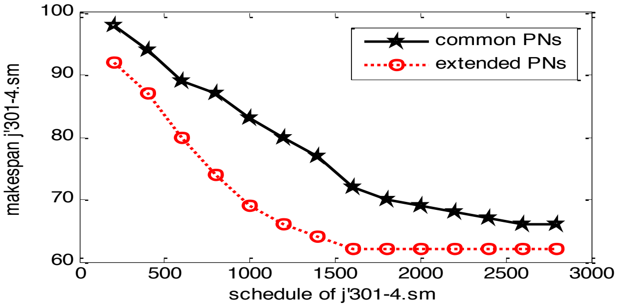


Figure8. Certainty duration solution behaviors of Petri-Nets and extends Petri-Nets for PSPLIB



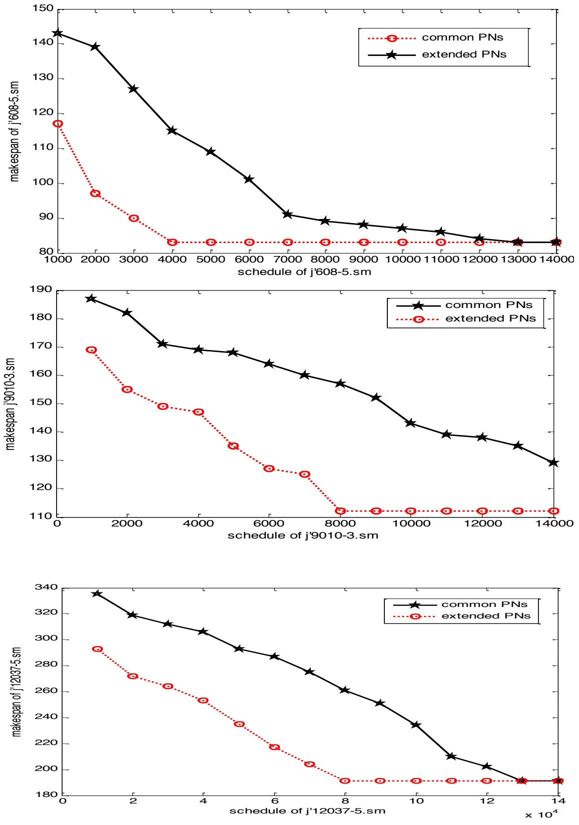


Figure9. Uncertain duration solution behaviour of Petri-Nets and extends Petri-Nets for PSPLIB

3.3 Evaluation of adaptability for unexpected change of uncertainty duration

In order to investigate adaptability for the unexpected change of the uncertainty duration for extended Petri-Nets in this paper, we set the PSPLIB cases where only half of the total resources are committed to RCMPSP after some time intervals from the beginning. We take four cases of PSPLIB (j303-5.sm, j608-9.sm, j903-5.sm, and j12010-3.sm) as testing cases. The schedules for these four cases by using extended Petri-Nets in this paper are list in table 3. We see that there is no significant difference in the solution schedules for each case. The experiment results show that extended Petri-Nets in this paper has good performance to the unexpected changes of uncertainty duration.

Table3. Evaluation of adaptability for unexpected changed of uncertainty duration

Unexpected changed	Schedule
Case of j'303-5.sm	
All processes are executed from the beginning	1479
Half of processes are executed at 400 step	1480
Half of processes are executed at 500 step	1483
Half of processes are executed at 600 step	1489
Case of j'608-9.sm	
All processes are executed from the beginning	3960
Half of processes are executed at 1000 step	3966
Half of processes are executed at 1200 step	3990
Half of processes are executed at 1400 step	4001
Case of j'903-5.sm	

All processes are executed from the beginning	7853
Half of processes are executed at 2000 step	7910
Half of processes are executed at 3000 step	7922
Half of processes are executed at 4000 step	7958
Case of j'12010-3.sm	
All processes are executed from the beginning	60129
Half of processes are executed at 10000 step	60298
Half of processes are executed at 20000 step	60332
Half of processes are executed at 30000 step	60398

3.4 Comparative study

In this subsection we compare the performance of extended Petri-Nets method with other approaches published in scientific literature. The objective of this experiment is to compare the performance to find the schedule with the least makespan. The experiments are conducted on j60 case with 500 instances, j90 case with 600 instances.

As far as this paper is concerned we have found other heuristic and meta-heuristic algorithms, including simulated annealing (SA), tabu search (TS), genetic algorithm (GA), ant colony optimization (ACO), particle swarm optimization (PSO) and bee algorithm (BA). All these methods are regarded as effective proofed and widely adopted methods to solve the RCMPSP. The performances of these methods are obtained from the original literature.

Table4. Average deviation of optimal makespan for j60

Method	Reference	1000	5000	50000
SA	Bouleimen et al.2003	12.75	11.9	---
TS	Baar et al. 1999	12.97	12.18	11.58
GA	Hartman 1998	13.30	12.74	12.26
ACOSS	Chen et al. 2010	11.75	10.98	10.67
BA-FBI	Ziarati 2011	12.55	12.04	11.16
Petri-Nets	This study	9.88	9.03	8.76
Extended Petri-Nets	This study	5.27	5.04	4.06

Table5. Average deviation of optimal makespan for j90

Method	Reference	1000	5000	50000
SA	Bouleimen et al.2003	42.81	37.68	---
TS	Nonobe et al. 2002	40.86	37.88	35.85
GA	Hartman 1998	39.37	36.74	34.03
ACOSS	Chen et al. 2010	35.19	32.48	30.56
LFT	Kolisch 1996	42.84	41.84	40.63
BA-FBI	Ziarati 2011	37.72	36.76	34.55
Petri-Nets	This study	39.14	38.66	38.05
Extended Petri-Nets	This study	29.19	27.06	24.09

Table 4-5 displays the average deviations for main methods over the j60 and j120 case study. On the basis of these results of Table 4-5, we can make the following conclusions.

(1) The extended Petri-Nets have better performance than other state-of-the-art methods.

(2) In general, the extended Petri-Nets method provides competitive results compared to other meta-heuristic methods investigated in this work.

4 Case studies

In this paper, the RCMPSP instances in the steel enterprise production line, which are proposed by the professor J.Nino Mora in the MIT Operation Research Center, will be used as the application examples (Nino 2005). For the problem containing three projects in the production line at the same time, the number of tasks that it contains can be abstracted. The optimal solution value of the instance is 142 days under the certainty duration, while it is 135 days under the uncertainty duration. For these instances, it can be modeled based on the extended Petri-Nets. The extended Petri-Nets presented in this paper and the object-oriented Petri net model optimization method and dynamic heuristic Petri net model simulation optimization method proposed by the literature (Maier and Moldt 2001; Dmitriy 2008; Dimitris 1994) will be simulated in the EM-Plant V7.0 simulation platform and the results show in Table 6. From Table 6, it can be found that the execution days for the convergence of the simulation iteration among the three methods differ not much in the certainty duration. And extended Petri-Nets proposed in this paper is slightly better. For the examples in the uncertainty duration, the makespan based on the Petri-Nets model method proposed by this paper reduce over 10-20% than the object-oriented Petri-Nets model optimization method and dynamic heuristic Petri-Nets model simulation optimization method proposed by the literature (Maier and Moldt 2001; Dmitriy 2008; Dimitris 1994).

Table 6. Simulation results for the three model methods

Makespan	The optimal results		Object-oriented Petri-Nets	
	Certain	Uncertain	Certain	Uncertain
Project 1	125	119	130	130
Project 2	141	134	154	142
Project 3	142	135	159	150
	Dynamic heuristic Petri-nets		Extended Petri-Nets	
	Certainty	Uncertainty	Certain	Uncertain
Project 1	126	135	127	121
Project 2	149	148	145	138
Project 3	155	149	145	139

5 Conclusions

The paper proposes extended Petri-Nets method to solve RCMPSP under certain and uncertain duration. Extensive experiments are conducted to evaluate the performance and the validity of the proposed method. Several conclusions can be drawn from the experiments as follows:

(1) The proposed extended Petri-Nets performances better than the other state-of-the-art methods in success rate of solution and solution speed. The performance of uncertain duration problems for the extended Petri-Nets decrease as the complexity of the case studies increases.

(2) The proposed extended Petri-Nets own better performances than the other state-of-the-art methods to the unexpected changes of uncertain duration. In general, the extended Petri-Nets method provides competitive results compared with the other meta-heuristic methods investigated in this work.

(3) The case studies show that the proposed method deserves better ability in real case solving than the other typical real case solving methods.

References

Abdallah, I. B., Elmaraghy, H. A., & Elmekkawy, T. 2002. Deadlock-free scheduling in flexible manufacturing systems using Petri nets. *International Journal of Production Research*, *40*(12), 2733-2756.

Alcaraz, J., & Maroto, C. 2001. A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operations Research*, *102*(1-4), 83-109.

Barnes, J. W., & Chambers, J. B. 1995. Solving the job shop scheduling problem with tabu search. *IIE transactions*, *27*(2), 257-263.

Billington, J., Wheeler, G. R., & Wilbur-Ham, M. C. 1991. PROTEAN: a high-level Petri net tool for the specification and verification of communication protocols. In *High-level Petri Nets*. Springer Berlin Heidelberg. 560-575.

Buchholz, P. 2004. Adaptive decomposition and approximation for the analysis of stochastic Petri nets. *Performance Evaluation*, *56*(1), 23-52.

Baggio, G., Wainer, J., & Ellis, C. 2004, March. Applying scheduling techniques to minimize the number of late jobs in workflow systems. In *Proceedings of the 2004 ACM symposium on Applied computing* (pp. 1396-1403). ACM.

Bouleimen, K. L. E. I. N., & Lecocq, H. O. U. S. N. I. (2003). A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. *European Journal of Operational Research*, *149*(2), 268-281.

Baar, T., Brucker, P., & Knust, S. 1999. *Tabu search algorithms and lower bounds for the resource-constrained project scheduling problem*.(pp. 1-18). Springer US.

Chen, Y. L., Hsu, P. Y., & Chang, Y. B. 2008. A Petri net approach to support resource assignment in project management. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *38*(3), 564-574.

Chen, F. F., & Chen, J. 2003. Performance modelling and evaluation of dynamic tool allocation in flexible manufacturing systems using coloured Petri nets: an object-oriented approach. *The International Journal of Advanced Manufacturing Technology*, *21*(2), 98-109.

Chen, H., Amodeo, L., Chu, F., & Labadi, K. 2005. Modeling and performance evaluation of supply chains using batch deterministic and stochastic Petri nets. *Automation Science and Engineering, IEEE Transactions on*, *2*(2), 132-144.

Chen, W., Shi, Y. J., Teng, H. F., Lan, X. P., & Hu, L. C. 2010. An efficient hybrid algorithm for resource-constrained project scheduling. *Information Sciences*, *180*(6), 1031-1039.

Della Croce, F., Tadei, R., & Volta, G. 1995. A genetic algorithm for the job shop problem. *Computers & Operations Research*, *22*(1), 15-24.

Ding, Z., Bunke, H., Kipersztok, O., Schneider, M., & Kandel, A. 2006. Fuzzy timed Petri nets—analysis and implementation. *Mathematical and Computer Modelling*, *43*(3), 385-400.

Dmitriy, A., 2008. Algorithmic issues in Queuing System and Combinatorial Counting Problems. *MIT Operation Research Center Doctor Degree Dissertation*.

Dimitris, B., 1994. A mathematical programming approach to stochastic and dynamic optimization problems. *MIT Operation Research Center Working Paper (http://www.mit.edu/~orc/research/reports).*

Gradišar, D., & Mušič, G. 2007. Production-process modelling based on production-management data: a Petri-net approach. *International Journal of Computer Integrated Manufacturing*, *20*(8), 794-810.

Hartmann, S. 1998. A competitive genetic algorithm for resource - constrained project scheduling. *Naval Research Logistics (NRL)*, *45*(7), 733-750.

Hartmann, S. 2002. A self - adapting genetic algorithm for project scheduling under resource constraints. *Naval Research Logistics (NRL)*, *49*(5), 433-448.

He, D. W., & Kusiak, A. 1997. Design of assembly systems for modular products. *Robotics and Automation, IEEE Transactions on*, *13*(5), 646-655.

Hu, W., Liang, H., Peng, C., Du, B., & Hu, Q. 2013. A Hybrid Chaos-Particle Swarm Optimization Algorithm for the Vehicle Routing Problem with Time Window. *Entropy*, *15*(4), 1247-1270.

Kolisch, R. 1996. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research*, *90*(2), 320-333.

Kurtulus, I. S. E. W., & Davis, E. W. 1982. Multi-project scheduling: Categorization of heuristic rules performance. *Management Science*, *28*(2), 161-172.

Li, J., Fan, Y., & Zhou, M. 2003. Timing constraint workflow nets for workflow analysis. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *33*(2), 179-193.

Lee, D. Y., & DiCesare, F. 1994. Scheduling flexible manufacturing systems using Petri nets and heuristic search. *Robotics and Automation, IEEE Transactions on*, *10*(2), 123-132.

Maier, C., & Moldt, D. 2001. Object coloured petri nets-a formal technique for object oriented modelling. In *Concurrent object-oriented programming and petri nets* (pp. 406-427). Springer Berlin Heidelberg.

Nino, J. 2005. Optimal Resource Allocation in A Dynamic and Stochastic Environment: A Mathematical Programming Approach. *MIT Operation Research Center Technical Reports* (http://www.mit.edu/~orc/research/reports), 126-165.

Nonobe, K., & Ibaraki, T. (2002). Formulation and tabu search algorithm for the resource constrained project scheduling problem. In Essays and surveys in metaheuristic (pp. 557-588). *Springer US.*

PSPLIB: Project Scheduling Problem Library, http://129.187.106.231/psplib [accessed on May 1. 2013]

Ramírez-Treviño, A., Ruiz-Beltrán, E., Rivera-Rangel, I., & López-Mellado, E. 2007. Online fault diagnosis of discrete event systems. A Petri net-based approach. *Automation Science and Engineering, IEEE Transactions on*, *4*(1), 31-39.

SHI, G. Q., LI, B. H., & CHAI, X. D. 2007. Modeling of Multi-project Scheduling for Complex Product Development Based on CPN [J]. *Journal of System Simulation*, 19*(17)*, 3869-3874

Verhoeven, M. G. A. 1998. Tabu search for resource-constrained scheduling. *European Journal of Operational Research*, *106*(2), 266-276.

Wu, L., Zhang J., 2009. Modeling technology for semiconductor wafer fabrication systems based on Agent-Oriented knowledge colored timed Petri-Nets. *Computer Integrated Manufacturing System*. 5(10),1921-1929.

Wang, X., Wu, T., 2004. Flexible job shop scheduling based on Petri-net model: ACO-GA hierarchical evolutionary optimization approach. *Journal of Zhejiang University (Engineering Science)*, 38(3), 286-291.

Zhang, H., Li, X., Li, H., & Huang, F. 2005. Particle swarm optimization-based schemes for resource-constrained project scheduling. *Automation in Construction*, *14*(3), 393-404.

Ziarati, K., Akbari, R., & Zeighami, V. 2011. On the performance of bee algorithms for resource-constrained project scheduling problem. *Applied Soft Computing*, *11*(4), 3720-3733.

Zhang, H., & Gu, M. 2009. Modeling job shop scheduling with batches and setup times by timed Petri nets. *Mathematical and Computer Modelling*, *49*(1), 286-294.