Memoirs of the Faculty of Engineering, Okayama University, Vol.41, pp.44-51, January, 2007

Decomposition of Time Petri Nets for Solving Optimal Firing Sequence Problem

Ryota Maeno *, Masami Konishi, Jun Imai

Dept. of Electrical and Electronic Engineering
The Graduate School of Natural Science and Technology
Okayama University
3-1-1, Tsushima-Naka Okayama, 700-8530

(Received November 7, 2006)

Petri net model is a frequently-used versatile tool which can represent a widely discrete event system. However, when the scale of the system becomes large, the calculation time for solving optimal problem (optimal firing sequence problem) is markedly increased. In this paper, we propose an approximation method that achieves the efficiency improvement of the solution by decomposing the Petri net for solving the optimal firing sequence problem. A timed Petri Net is decomposed into several subnets in which the optimal firing sequence for each subnet is solved by Dijkstra's algorithm in polynomial computational complexity. The effectivity of the proposed method is verified by numerical experiments for the flowshop schedule problem.

1 Introduction

Recently, sophistication and complication of discrete event system such as mechanical system, communication system, and production system are highly extended. Petri net model is widely used as a strong tool to describe concisely these discrete event systems. Thus, improving performance of Petri Nets model leads to flexible treatment of the discrete event system. In addition, the efficiency improvement of operations for various kind of discrete event system should be useful from the viewpoint of the energy saving and leads to the global environment protection.

Petri net model is a directed graph that consists of two kinds of nodes called place, transition and of branch called arc which connects these nodes. The describing ability of this model is revealed high, and parallel, synchronous operations can easily be described using this model. In addition, it is applicable to a wide variety of discrete event system because of its excellence in analytical ability and the operation ability. However, the state explosion problem for complex problems is not avoidable in the application to the large scale system by the Petri net. The dynamics of Petri net is determined by the firing sequence of transitions from an initial marking (initial state) to the target marking (state of the target). And the problem to determine a legal firing sequence from initial marking to a reachable target marking is known to be NP-hard[1]. Thus, it can be said that the problem to determine a reachable firing sequence to minimize the given objective function is also NP-hard. The exact algorithms to solve legal firing sequence problem have been studied in previous literature[1]. However, these algorithms can be applied for the restricted class of Petri Nets with a simple structure such as conflict-free Petri Nets.

The research on the Petri Net model which expresses the transportation system for AGVs (Automated Guided Vehicle) is proposed by [2][3][4]. The heuristic search[5], hybrid heuristics with backtrack[6][7][8], Genetic algorithm[9] have been reported in past works for scheduling problem by using Timed Petri Nets. Conventional optimization models for Petri Nets are concentrated on its search in the entire system. However, these approaches cannot be applied for large scale systems due to the increase in combinatorial complexity. Several decomposition model for Petri Nets has been studied.

*maeno@cntr.elec.okayama-u.ac.jp

Ootsuki et al. proposed an immune systemapproach where a Petri Net is decomposed into several subnets and the firing sequence of subclass of Petri Nets are determined by immune approach[10]. The entire firing sequence is coordinated by a heuristic procedure. Thus, it may be difficult to improve the optimality of solution for the entire system by this approach.

As for the application of extended Petri Nets to transportation routing of multiple AGVs are proposed by the authers[11]. In this paper, we propose an approximation method that achieves the efficiency improvement of the solution procedure by decomposing the Timed Petri Nets for solving the optimal firing sequence problem. In the proposed method, the optimization problem for each subnet is solved by Dijkstra's algorithm in polynomial computational complexity. A novel distributed optimization method is proposed to improve the optimization performance. The performance of the proposed method is evaluated by comparing results with another optimization methods.

2 Definition of Timed Petri Nets and modeling example

In this study, The firing duration time model which is one kind of Timed Petri Nets is employed so that our decomposition scheme applies to Timed Petri Nets easily and be written as TPN. In this section, the definition of TPN is described, and modeling example of routing problem for multiple AGVs (automated guided vehicles) and of scheduling problem are indicated.

2.1 Definition of TPN

In the firing duration time model, the time parameter $\theta(t)$ which is representing the firing duration time of transition t is introduced for $\forall t \in T$. When the fireable transition t is fired, tokens are removed from each input places at the time. And tokens are atted to each output places at the same time as completing the firing. Because firing duration time is introduced as above-mentioned, the modeling of system considered the time of event occurrence becomes possible. The description of TPN, marking, firing vector, incidence matrix, firing condition and state transition rule are defined as follows. Let $\mathbb{N}$ be set of non-negative integers and $\mathbb{R}$ is set of real numbers. For a set of A and a finite set of B, A^B is defined to be a set of vectors with elements of A of whose dimension is $|B|$. For a set of A and a finite set of B and C, $A^{B\times C}$ is defined to be a set of $|B| \times |C|$ matrices with elements of A.

[Definition 1]

The firing duration time model TPN is described by $TPN = (P, T, w, M_0, \theta)$. $P = \{p_1, p_2, \cdots, p_{|P|}\}$ is set of places, $T = \{t_1, t_2, \cdots, t_{|T|}\}$ is set of transitions, $w : (P \times T) \cup (T \times P) \to \mathbb{N}$ represents connection relationship of place and transition, $M_0 : P \to \mathbb{N}$ is an initial marking, $\theta : T \to \mathbb{N} - \{0\}$ is a function of non-negative firing duration time on set of transitions. We define $w(p,t) = 0$ $(w(t,p) = 0)$ indicate that there exists no arc from place p to transition t (t to p), and $w(p,t) > 0$ $(w(t,p) > 0)$ indicates the weight of arc from place p to transition t (t to p). We call the place $p \in P$ as an input place (output place) for $t \in T$ if $w(p,t) > 0$ $(w(t,p) > 0)$, and the transition $t \in T$ as an input transition (output transition) for $p \in P$ when $w(t,p) > 0$ $(w(p,t) > 0)$.

[Definition 2]

A function $r_k : T \to \{0,1\}$ represents whether each transition is fired or not (1: fired, 0: not fired) in time $k \in \mathbb{N}$. A marking M_k: $P \to \mathbb{N}$ represents the number of tokens in each place. These variables are expressed by a column vector marking $M_k \in \mathbb{N}^P$, firing vector $r_k \in \{0,1\}^T$ defined as following.

$$(M_k)_i = M_k(p_i) \quad (\forall p_i \in P) \tag{1}$$

$$(r_k)_j = r_k(t_j) \quad (\forall t_j \in T) \tag{2}$$

[Definition 3]

For a firing duration time model $TPN = (P, T, w, M_0, \theta)$, two integer matrices $A^+_{TPN} \in \mathbb{N}^{T\times P}$, $A^-_{TPN} \in \mathbb{N}^{T\times P}$ are defined by

$$(A^+_{TPN})_{ji} = w(t_j, p_i) \tag{3}$$

$$(A^-_{TPN})_{ji} = w(p_i, t_j) \tag{4}$$

[Definition 4]

For time $k \in \mathbb{N}$, for transition $t \in T$, if the following equation

$$M_k(p) \geq w(p,t) \quad (\forall p \in P) \tag{5}$$

is satisfied and transition t is not in firing duration, then transition t is fireable. For subset $T' \subset T$, even though $\forall t \in T'$ is fireable, if the following constraints

$$M_k(p) \geq \sum_{t\in T'} w(p,t) \quad (\forall p \in P) \tag{6}$$

is not satisfied, not all transition $t \in T'$ can be firing at the same time. This situation is called as conflict. When a fireable transition $t \in T$ is fired in time k, the number of tokens $w(p,t)$ is removed from each input place p. At time $k+\theta(t)$, the number of tokens $w(t,p)$ is added to each output place p. The duration time from k to $k+\theta(t)$ is considered as firing duration of transition t.

Following the above definition, the firing condition and state equation for TPN are derived. The function c_k representing residual firing duration for transition $t \in T$ which is in firing duration is introduced. If the transition t is not in firing duration, then $c_k(t) = 0$. $c_k \in \mathbb{N}^T$ indicates a column vector comprising $c_k(t)$ for $\forall t \in T$. The firing condition for TPN is represented by the following equation.

$$M_k - (A^-_{TPN})^{\mathrm{T}} r_k \geq 0 \tag{7}$$

The condition for the completion of firing transition is written as following.

$$(c_k)^{\mathrm{T}} \cdot r_k = 0 \tag{8}$$

When a firing vector r_k satisfying (7) and (8) is given, the marking M_k is changed into M_k^+ by following state equation.

$$M_k^+ = M_k - (A^-_{TPN})^{\mathrm{T}} r_k \tag{9}$$

The residual firing duration $c_k(t)$ is allocated to the transition $\forall t \in T$ as following.

$$c_k^+(t) = \begin{cases} \theta(t) & (r_k(t) = 1) \\ c_k(t) & (\text{otherwise}) \end{cases} \tag{10}$$

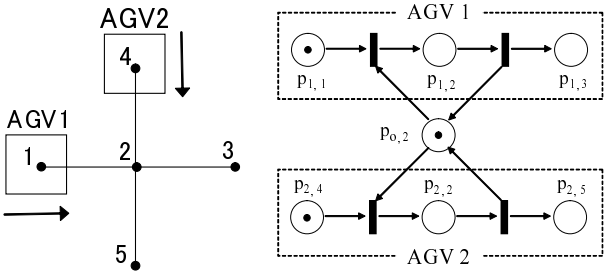
When a transition $t \in T$ is satisfied $c_k^+(t) = 1$ from firing vector until time k, the number of tokens $w(t,p)$ are added to the output place at time $k+1$. Thus, the following equation (11) is satisfied where $e_k \in \{0,1\}^T$ is column vector comprising $e_k(t)$ defined as (12).

$$M_{k+1} = M_k^+ + (A^+_{TPN})^{\mathrm{T}} e_k \tag{11}$$

$$e_k(t) = \begin{cases} 1 & (c_k^+(t) = 1) \\ 0 & (\text{otherwise}) \end{cases} \tag{12}$$

$c_k^+(t)$ is changed by the following equation at time $k+1$.

$$c_{k+1}(t) = \begin{cases} c_k^+(t) - 1 & (c_k^+(t) \geq 1) \\ 0 & (\text{otherwise}) \end{cases} \tag{13}$$



(a)AGV motion (b)Petri net model

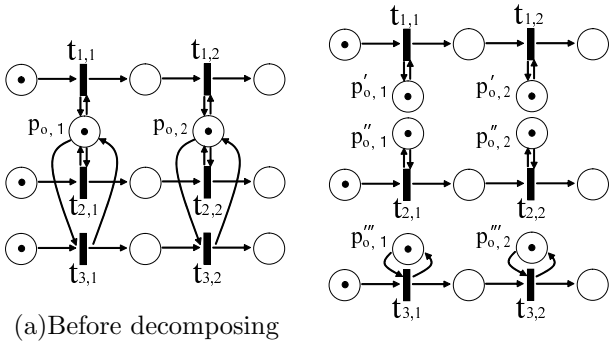
Fig. 1: AGV motion and its Petri net model

2.2 Modeling of routing problem

If AGV1 and AGV2 advance toward the direction of the arrow respectively in the route of fig.1(a), this system is expressed by fig.1(b). Here, place $p_{i,j}$ represents the state that AGVi exists in node j, and transition $t_{i,a,b}$ represents the event that AGVi moves from node a to node b ($a \neq b$). And condition of the collision avoidance at node j is satisfied by introducing resource place $p_{o,j}$.

2.3 Modeling of scheduling problem

When the production scheduling problem is targeted, the 2 machine 3 jobs flowshop problems are expressed by the fig.2(a). Here, transition $t_{i,j}$ represents processing in process j of job i, and the processing time is represented as firing duration of the transition. And the condition that two or more jobs are not treatable in process j at the same period is satisfied by introducing place $p_{o,j}$.



(a)Before decomposing (b)After decomposing

Fig. 2: Modeling of flowshop scheduling problem

3 The optimization method by decomposing Petri Nets

3.1 Optimal Firing Sequence Problem

Optimal Firing Sequence Problem for TPN is defined as following in this paper. Given $TPN = (P,\ T,\ w,\ M_0,\ \theta)$, a targeted marking $M_{ref} : P' \to \mathbb{N}$ ($P' \subset P$), time horizon $N_t \in \mathbb{N} - \{0\}$, and objective function $J : (\{0,\ 1\}^T)^{N_t} \to \mathbb{R}$, the problem is to derive a set of firing vector $(r_o,\ r_1,\ \cdots,\ r_{N_t-1}) \in (\{0,\ 1\}^T)^{N_t}$ to minimize J satisfying $c_{N_t} = 0 \wedge M_{N_t}(p) = M_{ref}(p)$ $(\forall p \in P')$. We call the problem as an optitmal firing sequence problem.

3.2 Optimization algorithm

In this section, optimization algorithm for solving optimal firing sequence problem by decomposing Petri Net is proposed. The condition of judging whether an optimal firing sequens problem can be decomposed is shown as follows.

- The entire system consists of multiple entities. The dynamics of each entity u_i $(1 \le i \le m)$is represented by each subnet described by TPN.
- The entire objective function J can be written by the sum of each objective function J_{u_i} $(r_0^{u_i},\ r_1^{u_i},\ \cdots, r_{N_t}^{u_i})$ for each subsystem u_i $(1 \le i \le m)$.
- Targeted marking $M_{ref}(p)$ is not defined for any place $p \in P_R$

A general $TPN = (P,\ T,\ w,\ M_0,\ \theta)$ is decomposed into several subnets by the following procedure. Initially, the set of transition T is described as direct sum of subset T_{u_i} which is representing the activity of each entity u_i $(1 \le i \le m)$.

$$T = T_{u_1} \sqcup T_{u_2} \sqcup \cdots \sqcup T_{u_m} = \coprod_{i=1}^{m} T_{u_i} \tag{14}$$

The set of places P is also described as direct sum of subset P_{u_i}, P_R as following equation.

$$P = P_{u_1} \sqcup P_{u_2} \sqcup \cdots \sqcup P_{u_m} \sqcup P_R = \left(\coprod_{i=1}^{m} P_{u_i} \right) \sqcup P_R \tag{15}$$

P_{u_i} and P_R are defined as follows. $OUT(p)$ is set of output transitions for place p. $IN(p)$ is set of input transitions for place p.

$$\begin{aligned} P_{u_i} &= \{p \mid IN(p) \subset T_{u_i}, OUT(p) \subset T_{u_i}\} \\ P_R &= P \setminus \left(\coprod_{i=1}^{m} P_{u_i} \right) \end{aligned}$$

$r_k^{u_i} \in \{0,1\}^{T_{u_i}}$ is a column vector with the element of $r_k(t)$ for all $t \in T_{u_i}$. $\theta_{u_i} \in \mathbb{N}^{T_{u_i}}$ is a column vector with the element of $\theta(t)$ for all $t \in T_{u_i}$. $c_k^{u_i} \in \mathbb{N}^{T_{u_i}}$ is a column vector with the element of $c_k(t)$ for all $t \in T_{u_i}$. $e_k^{u_i} \in \{0,1\}^{T_{u_i}}$ is a column vector with the element of $e_k(t)$ for all $t \in T_{u_i}$. $M_k^{u_i} \in \mathbb{N}^{P_{u_i}}$ is a column vector with the element of $M_k(p)$ for all $p \in P_{u_i}$. $M_k^R \in \mathbb{N}^{P_R}$ is a column vector with the element of $M_k(p)$ for all $p \in P_{u_i}$.

The firing condition for the TPN is written as

$$M_k^{u_i} - (A_{u_i}^-)^{\mathrm{T}} r_k^{u_i} \ge 0 \quad (1 \le i \le m) \tag{16}$$
$$M_k^R - (B^-)^{\mathrm{T}} r_k \ge 0 \tag{17}$$
$$(c_k^{u_i})^{\mathrm{T}} r_k^{u_i} = 0 \quad (1 \le i \le m) \tag{18}$$

by an incidence matrix $A_{u_i}^- \in \mathbb{N}^{T_{u_i} \times P_{u_i}}$, an incidence matrix $A_{u_i}^+ \in \mathbb{N}^{T_{u_i} \times P_{u_i}}$ and

$$B^- = [(B_{u_1}^-)^{\mathrm{T}}, \cdots, (B_{u_m}^-)^{\mathrm{T}}]^{\mathrm{T}} \quad (B_{u_i}^- \in \mathbb{N}^{T_{u_i} \times P_R})$$
$$B^+ = [(B_{u_1}^+)^{\mathrm{T}}, \cdots, (B_{u_m}^+)^{\mathrm{T}}]^{\mathrm{T}} \quad (B_{u_i}^+ \in \mathbb{N}^{T_{u_i} \times P_R})$$

The state equation for TPN can be expressed by

$$M_k^{u_i +} = M_k^{u_i} - (A_{u_i}^-)^{\mathrm{T}} r_k^{u_i} \tag{19}$$
$$M_{k+1}^{u_i} = M_k^{u_i +} + (A_{u_i}^+)^{\mathrm{T}} e_k^{u_i} \tag{20}$$
$$M_k^{R+} = M_k^R + (B^-)^{\mathrm{T}} r_k \tag{21}$$
$$M_{k+1}^R = M_k^{R+} + (B^+)^{\mathrm{T}} e_k \tag{22}$$

In the example of section 2.1, if we assume individual AGV (AGV1, AGV2) to be each element $(u_1,\ u_2)$, $T_{u_1} = \{t_{1,1,2},\ t_{1,2,3}\}$, $T_{u_2} = \{t_{2,4,2},\ t_{2,2,5}\}$, $P_{u_1} = \{p_{1,1},\ p_{1,2},\ p_{1,3}\}$, $P_{u_2} = \{p_{2,4},\ p_{2,2},\ p_{2,5}\}$, $P_R = \{p_{o,2}\}$ and the following expressions are obtained.

$$\begin{aligned} T &= T_{u_1} \sqcup T_{u_2} \\ P &= P_{u_1} \sqcup P_{u_2} \sqcup P_R \end{aligned}$$

In the example of section 2.2, if we asume individual job (Job1, Job2, Job3) to be each element $(u_1,\ u_2,\ u_3)$, $T_{u_i} = \{t_{i,1},\ t_{i,2}\}$ $(1 \le i \le 3)$, $P_R = \{p_{o,1},\ p_{o,2}\}$ and the following expressions are obtained.

$$\begin{aligned} T &= T_{u_1} \sqcup T_{u_2} \sqcup T_{u_3} \\ P &= P_{u_1} \sqcup P_{u_2} \sqcup P_{u_3} \sqcup P_R \end{aligned}$$

From the decomposition scheme for the set of places and transitions, the $TPN = (P,\ T,\ w,\ M_o,\ \theta)$ can be

decomposet into each subnet $TPN^i = (P_{u_i} \cup P_{Ru_i}, T_{u_i}, w^{u_i}, M_o^i, \theta^{u_i})$ for entity u_i. P_{Ru_i} is a set of places one-to-one corresponding to P_R, and w^{u_i} is defined by the following equation using a bijection relation $\zeta : P_{Ru_i} \rightarrow P_R$.

$$w^{u_i}(p,t) = w(p,t) \quad (\forall p \in P_{u_i}; \forall t \in T_{u_i}) \tag{23}$$
$$w^{u_i}(t,p) = w(t,p) \quad (\forall p \in P_{u_i}; \forall t \in T_{u_i}) \tag{24}$$
$$w^{u_i}(p,t) = w(\zeta(p),t) \quad (\forall p \in P_{Ru_i}; \forall t \in T_{u_i}) \tag{25}$$
$$w^{u_i}(t,p) = w(t,\zeta(p)) \quad (\forall p \in P_{Ru_i}; \forall t \in T_{u_i}) \tag{26}$$

For example, the model illustrated in Fig.1(b) is dcomposed as the model illustrated Fig.3 by duplicationg a place $p_{o,2}$ into $p'_{o,2}$, $p''_{o,2}$ where $P_{Ru_1} = \{p'_{o,2}\}$, $P_{Ru_2} = \{p''_{o,2}\}$. The model illustrated in Fig.2(a) is dcomposed as the model illustrated Fig.2(b) by duplicationg places $p_{o,1}$, $p_{o,2}$ where $P_{Ru_1} = \{p'_{o,1}, p'_{o,2}\}$, $P_{Ru_2} = \{p''_{o,1}, p''_{o,2}\}$, $P_{Ru_3} = \{p'''_{o,1}, p'''_{o,2}\}$.

M_k^i is the marking for TPN^i, $M_k^{Ru_i}$ is the column vector comprising $M_k^i(p)$ for $p \in P_{Ru_i}$. The firing condition for TPN^i is written by (16), (18), (27), and the state equation for TPN^i is written by (19), (20), (28), (29). Where the initial value of $M_0^{Ru_i}(p)$ for $\forall p \in P_{Ru_i}$ is large enough so that a firing vector violating firing condition for subnet TPN^i is not feasible for the entire TPN.

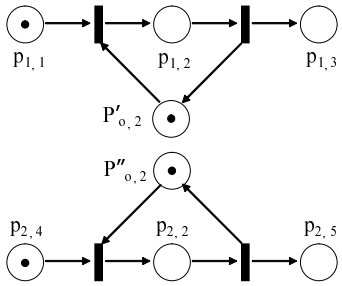


Fig. 3: An example of decomposition for AGV system

$$M_k^{Ru_i} - (B_{u_i}^-)^{\mathrm{T}} r_k^{u_i} \geq 0 \tag{27}$$
$$M_k^{Ru_i+} = M_k^{Ru_i} - (B_{u_i}^-)^{\mathrm{T}} r_k^{u_i} \tag{28}$$
$$M_{k+1}^{Ru_i} = M_k^{Ru_i+} + (B_{u_i}^+)^{\mathrm{T}} e_k^{u_i} \tag{29}$$

When a firing sequence of transition set T is feasible for all subnets TPN^i, the necessary and sufficient conditiion for the firing sequence is feasible for entire TPN is described by (17).

Optimization algorithm is explained as follows.

Step 1 The variable N representing number of iterations is set to $N := 1$. The subproblems formulated as (30) for all u_j $(1 \leq j \leq m)$ is solved. The optimal solution for (30) is regarded as a tentative solution $\hat{r}_k^{u_j}$.

$$\min_{\{r_k^{u_j}\}} J_{u_j} \tag{30}$$

Step 2 If the tentative solution $\hat{r}_k^{u_j}$ satisfies (17) and the solution has not been updated from previous solution, this algorithm is completed and the solution $\hat{r}_k^{u_j}$ is regarded as near optimal solution.

Step 3 The entity executed re-optimization in this iteration is decided. if the $u_i (i \neq m)$ is selected in the previous re-optimization, u_{i+1} is selected. if u_m is selected in the previous re-optimization, u_1 is selected. Following, the entity selected in this iteration is represented as u_j.

Step 4 The re-optimization for entity u_j selected at Step 3 is executed. The objective fuction for subproblem is written as (31) where the tentative solution $\hat{r}_k^{u_i}$ $(i \neq j)$ for u_i exept u_j is constant. The penalty for violating firing condition of (17) is added to the (30). $\omega_{p,u_j}^{(N)}$ is a weighting factor of penalty function in the N times of iterations.

$$\min_{\{r_k^{u_j}\}} \left(J_{u_j} + \sum_{k=0}^{N_t} \sum_{p \in P_{Ru_j}} \omega_{p,u_j}^{(N)} \alpha_{u_j,p,k}(r_k^{u_j}) \right) \tag{31}$$

The penalty function $\alpha_{u_j,p,k}(r_k^{u_j})$ indicates the number of tokens required for the infeasible solution to be a feasible for firing vector $r_k^{u_j}$ at place $p \in P_R$ in time k. This function is described as (32)-(36) where the marking at previous iteration derived by tentative solution $\hat{r}_k^{u_i} (1 \leq i \leq m)$ is denoted as $\hat{M}_k$.

$$\alpha_{u_j,p,k}(r_k^{u_j}) = \max\{0, \ -\alpha'_{u_j,p,k}\} \tag{32}$$
$$\alpha'_{u_j,p,k} = K_{u_j,k}(p) + M_k^{Ru_j}(p) - B_{u_j}^-(p) r_k^{u_j} \tag{33}$$
$$K_{u_j,k} = \hat{M}_k^{R+} - (B_{u_j}^+ - B_{u_j}^-)^{\mathrm{T}} \hat{\Sigma}_{k-1}^{u_j} + (B_{u_j}^-)^{\mathrm{T}} \hat{r}_k^{u_j} + (B_{u_j}^+)^{\mathrm{T}} \hat{\sigma}_k^{u_j} - M_o^{Ru_j} \tag{34}$$
$$(\hat{\sigma}_k^{u_j})_i = \begin{cases} 1 & ((\hat{c}_k^{u_j})_i \geq 1) \\ 0 & (\text{otherwise}) \end{cases} \tag{35}$$
$$\hat{\Sigma}_{k-1}^{u_j} = \hat{r}_0^{u_j} + \hat{r}_1^{u_j} + \hat{r}_2^{u_j} + \cdots + \hat{r}_{k-1}^{u_j} \quad (p \in P_{Ru_j}; \ k = 0, \cdots, N_t) \tag{36}$$

$B_{u_j}^-(p)$ indicates a row vector corresponding to place p in matrix $(B_{u_j}^-)^{\mathrm{T}}$. $K_{u_j,\,k}(p)$ is a constant as an element of column vector $K_{u_j,\,k}$ for each place p.

Step 5 The tentative solution is updated by $\hat{r}_k^{u_j} := r_k^{u_j}$ using the solution $r_k^{u_j}$ derived as Step 4, and the

value of weighting factor for u_j is updated by (37), (38). $N := N + 1$ and return to Step 2.

$$\omega_{p,u_j}^{(N+1)} = \omega_{p,u_j}^{(N)} + \Delta\omega \sum_{k=0}^{N_t} \alpha_{u_j,p,k}(\hat{r}_k^{u_j}) \quad (37)$$

$$\omega_{p,u_i}^{(N+1)} = \omega_{p,u_i}^{(N)} \quad (i \neq j) \quad (38)$$

3.3 Solving subproblem

The subproblem (30),(31) proposed in section 3.1 is the interger programing problem, and the problem can be derived by MILP solver such as ILOG CPLEX. However, if the subproblem satisfies principle of the optimality, the broblem can be effectively solved by Dijkstra's algorithm in a polynomial order. For example, the condition that the objective function J_{u_j} is the sum of total transition time from initial marking to targeted marking is considered.

Let $\mathcal{P}_s^j$ a set of states where $S(P_{u_j} \cup P_{Ru_j})$ is the set of marking for $P_{u_j} \cup P_{Ru_j}$, $R_{MC}(TPN^j, k) \subset S(P_{u_j} \cup P_{Ru_j}) \times \mathcal{N}^{T_{u_j}}$ is set of 2-tuples $(M_k^j,\ c_k^{u_j})$ consisting of reachable marking and column vector $c_k^{u_j}$ at time k for TPN^j and h_k is 0-1 binary variable indicating 1 if the marking in time k has reached the targeted marking and otherwise zero.

$$\mathcal{P}_s^j = \{(M_k^j,\ c_k^{u_j},\ k,\ h_k) | (M_k^j,\ c_k^{u_j}) \in R_{MC}(TPN^j, k);\ 0 \leq k \leq N_t;\ h_k = 0 \vee M_k^{u_j} = M_{ref}^{u_j}\} \quad (39)$$

If the $M_k^{u_j}$ is the same as the targeted marking, the state $(M_k^j, 0, k, 1)$ or $(M_k^j, 0, k, 0)$ can be selected. However, if the state becomes $(M_k^j, 0, k, 1)$, the marking cannot be selected except the state $(M_k^j, 0, k+1, 1)$ so that the marking already reached the targeted marking has never been transitioned other marking.

The cost of state transiton from $a = (M_k^j,\ c_k,\ k,\ h_k) \in \mathcal{P}_s^j$ to $b = (M_{k+1}^j,\ c_{k+1},\ k+1,\ h_{k+1}) \in \mathcal{P}_s^j$ can be regarded as constant number $d_{u_j,a,b}$ concretely decided by a and b, because the time cost and the penalty cost are determined by time k, marking M_k^j and firing vector $r_k^{u_j}$. In addition, the objective function (30), (31) is represented as sum of $d_{u_j,a,b}$. Therefore, the optimization problem (30), (31) can be solved by Dijkstra's algorithm where state set $\mathcal{P}_s^j$ is assumed as nodes set for Dijkstra's and $d_{u_j,a,b}$ is assumed as distance cost from a to b.

4 Improvement of performance for optimization algorithm

In this section, improvement of optimization performance of the proposed method is discussed. Ill-condition occurs during the optimization is explained and a novel updating method for weighting factor in the penalty optimization method is proposed.

4.1 Ill-condition for proposed method

Let us consider a situation where a TPN model shown in fig.4(a) is docomposed into subnet for $u_1,\ u_2,\ u_3$ such as fig.4(b) and the token placement of the model is changed as shown in fig.4 after the period of $k \in \mathcal{N}$. Two pieces of tokens on place p in the TPN model before decomposition are removed from initial value $M_0(p) = 1$. Thus, the number of tokens on place p is -1 at time k, and the penalty factor is $\alpha_{u_j,p,k} = 1$ for all of the entities u_i $(1 \leq i \leq 3)$ from (32)-(36).

The number of tokens on place p is maximum values $M_k(p''') = 1$, and that cannot be increased any further. Thus, it is though that the entity involving the conflict on place p at time k is $u_2,\ u_3$, and u_1 is never involving the conflict. However the ill-condition that the penalty to the violation of restriction is imposed on element u_1 is caused because the panalty function for u_1 is $\alpha_{u_1,p,k} = 1$ from (32)-(36). Morever, a needless increase of the penalty term is caused to element u_1 which doesn't involve the conflict by updating the penalty weighting factor from (37), and the optimization performance deteriorates.

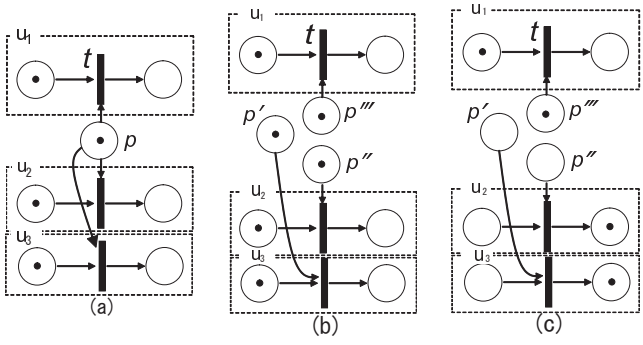


Fig. 4: Problem for optimization by decomposition of Petri Net

4.2 Improvement of penalty method

Let $x_k^{u_i}(p)$ denote the maximum number of tokens which can be placed on place $p \in P_{Ru_i}$ at time k. $x_k^{u_i}(p)$ is represented as the following equation.

$$x_k^{u_i}(p) = \max\{M_k^i(p) |\ (M_k^i,\ c_k^{u_i}) \in R_{MC}(TPN^i,\ k)\} \quad (\forall p \in P_{Ru_i}) \qquad (40)$$

In the following, the step 4 and step 5 of the algorithm explained in section 3.2 are modified to improve optimization performance. The weighting factor w_{p,u_j} for (31) indicating an unit penalty coefficient of entity u_j for violating firing condition at a time for place p is modified to $w_{u_j,u_i,p}$ indicating an unit penalty coefficient of entity u_j for violating firing condition (17) by firing vector $r_k^{u_j}$ and $\hat{r}_k^{u_i}(i \neq j)$. The following equations (41)-(45) are defined so that the unit penalty coefficient depends on the involved degree to the conflict at time k for place p.

$$\omega_{p,u_j}^{(N)} = \eta_{u_j,k,p} \sum_{i|i\neq j} h_{u_j,u_i,k,p} w_{u_j,u_i,p}^{(N)} \qquad (41)$$

$$\Delta X_k^{u_i}(p) = \begin{cases} x_k^{u_i}(p) - \hat{M}_k^{Ru_i}(p) & (i \neq j) \\ x_k^{u_j}(p) - M_k^{Ru_j}(p) & (i = j) \end{cases} \qquad (42)$$

$$h_{u_j,u_i,k,p} = \begin{cases} +\infty & (\sum_{i'|i'\neq j} \Delta X_k^{u_{i'}}(p) = 0) \\ \frac{\Delta X_k^{u_i}(p)}{\sum_{i'|i'\neq j} \Delta X_k^{u_{i'}}(p)} & (\text{otherwise}) \end{cases} \qquad (43)$$

$$\eta_{u_j,k,p} = \begin{cases} 0 & (\sum_{i|i\neq j} \Delta X_k^{u_i}(p) + \Delta X_k^{u_j}(p) = 0) \\ \frac{(\sum_i s_{u_j,u_i,k,p}) \Delta X_k^{u_j}(p)}{\sum_{i|i\neq j} \Delta X_k^{u_i}(p) + \Delta X_k^{u_j}(p)} & (\text{otherwise}) \end{cases} \qquad (44)$$

$$s_{u_j,u_i,k,p} = \begin{cases} 1 & (\Delta X_k^{u_i}(p) > 0) \\ 0 & (\text{otherwise}) \end{cases} \qquad (45)$$

(41) indicates that the weighting factor of entity u_j for place p is represented by the product of weighting $w_{u_j,u_i,p}$ and rations $h_{u_j,u_i,k,p}$ and $\eta_{u_j,k,p}$. $\Delta X_k^{u_i}(p)$ represents the difference between maximum number of tokens which can be placed on $p \in P_{Ru_i}$ at time k in subnet TPN^i and the number of tokens actually placed on $p \in P_{Ru_i}$ by firing vector $\hat{r}_0^{u_i}, \hat{r}_1^{u_i}, \cdots, \hat{r}_k^{u_i}$ ($r_0^{u_j}, r_1^{u_j}, \cdots, r_k^{u_j}$ if $i = j$). $h_{u_j,u_i,k,p}$ is the ratio between $\Delta X_k^{u_i}(p)$ and the total sum of $\Delta X_k^{u_{i'}}(p)$ for all entities $u_{i'}(i' \neq j)$. $\eta_{u_j,k,p}$ is the ratio between $\Delta X_k^{u_j}(p)$ and the average of $\Delta X_k^{u_i}(p)$ for all entities u_i written as $(\sum_{i|i\neq j} \Delta X_k^{u_i}(p) + \Delta X_k^{u_j}(p))/\sum_i s_{u_j,u_i,k,p}$. By using the above equations, the unit penalty coefficient depens on the ratio of difference between the number of tokens and the maximum numbers of tokens on place $p \in P_{Ru_i}$ for each entity u_i. The updating method for weighting function $\omega_{p,u_j}^{(N)}$ at step 5 is modified to curb the needless increase of weighting factors. The weighting factor is updated by following equation only when the value of $\sum_{k=0}^{N_t} h_{u_j,u_i,k,p}\eta_{u_j,k,p}\alpha_{u_j,p,k}(r_k^{u_j})$ is positive number and the value is larger than the previous value of $\sum_{k=0}^{N_t} h_{u_j,u_i,k,p}\eta_{u_j,k,p}\alpha_{u_j,p,k}(\hat{r}_k^{u_j})$.

$$\omega_{u_j,u_i,p}^{(N+1)} = \omega_{u_j,u_i,p}^{(N)} + \Delta\omega \qquad (46)$$

5 Numerical Experiments

The optimal solution cannot be obtained for flowshop scheduling problem to minimize total tardiness penalties due to the fact that the problem is NP-complete. In order to evaluate the performance of the proposed method (DPN), we developed a simulated annealing method to solve especially for flowshop scheduling problems. The simulated annealing method is constructed so that processing order of operations at each machine is successively improved by repeating the generation of neighborhood solution. The neighborhood solution is created by inserting a randomly selected operation to a randomly selected position. The detail of the algorithm for simulated annealing method for flowshop scheduling problem is described in [12]. The parameters used for the experiments are shown in Table 1. The total number of search time at a temperature N_S is determined so that the total computation time for SA is almost the same as that of average computation time for the proposed method (DPN). Table 2 shows the average of objective value, computation time, and ratio DPN/SA when 20 kinds of example for 40Job and 80Job are solved respectively. A Pentium IV 3.2GHz with 1Gbyte memory is used for computation. From the results of Table 2, the performance ratio between the DPN and SA method is 1.02 when the number of jobs are 40. The performance of SA method is slightly better than that of the proposed method. However, when the number of jobs is 80, the performance of the proposed method is better than that of the SA method. This is because the proposed method can effectively generate near optimal solutions by decomposing the original problem into several subproblems. For the SA method, it becomes extremely difficult to solve large-sized problems because the search space is extremely larger than the small-sized problems. The performance between the proposed method and the SA method is almost the same even though the proposed method can be applied for wide variety problems.

Table. 1: Parameters for SA method

T_{init}: Initial temperature factor
h: Annealing ratio
N_A: Annealing times
N_S: Search times at the same temperature
40Job: T_{init}=0.5, h=0.97, N_A=2000, N_S= 27000
80Job: T_{init}=0.5, h=0.97, N_A=2000, N_S= 25000

Table. 2: Comparison between DPN and SA

Job	DPN		SA		
	Obj.	Time	Obj.	Time	Ratio
40	39	603	38	600	1.02
80	14.25	1162	51.35	1168	0.52

6 Conclusions

In this paper, we have proposed decomposition and optimization method for the large scale system represented by Timed Petri Nets. A penalty based optimization method is proposed to solve an optimal firing sequence problem. A new method for updationg weighing factor for penalty coefficients is proposed to improve optimization performance. Through the numerical experiments, the performance of the proposed method is equal or better than that of the SA method specialized for the flowshop scheduling problem even though the proposed method can be applied for wide variety problems. Reduction of the calculation time for subproblem and application to various discrete event system are the future works.

Bibliography

[1] M.Yamaguchi, S.Nakamura and T.Watanabe: An Approximation Algorithm for the Legal Firing Sequence Problem of Petri Nets, Research Report for Information Processing Society of Japan, *Algorithm*, No. 33-3, pp. 17-24 (1993)

[2] H. Shiizuka and M. Suzuki: Modeling of AGV Networks in Flexible Manufacturing Systems; *Comp. Ind. Engng.*, Vol. 27, Nos. 1-4, pp. 81-86 (1994)

[3] N. Wu and M. Zhou: Modeling and Deadlock Control of Automated Guided Vehicle Systems; *IEEE/ASME Trans. Mechatronics*, Vol. 9, No. 1, pp. 50–57 (2004)

[4] N. Wu and M. Zhou: Resource-Oriented Petri Nets in Deadlock Avoidance of AGV Systems; *Proc. IEEE Conf. Robotics and Automation*, pp. 64–69 (2001)

[5] D.Y. Lee, F. DiCesare: Scheduling Flexible Manufacturing Systems Using Petri Nets and Heuristic Search, *IEEE Trans. Robot. Automat.*, **10**-2, 123/132 (1994)

[6] A.R. Moro, H. Yu and G. Kellehr: Hybrid Heuristic Search for the Scheduling of Flexible Manufacturing Systems Using Petri Nets, *IEEE Trans. Robot. Automat.*, **18**-2, 240/244 (2002)

[7] M.D. Jeng, S.C. Chen: Heuristic Search Based on Petri Net Structures for FMS Scheduling, *IEEE Trans. Ind. Appl.*, **35**-1, 196/202 (1999)

[8] H.H. Xiong and M. Zhou: Scheduling of Semiconductor Test Facility via Petri Nets and Hybrid Heuristic Search, *IEEE Trans. Semicon. Manuf.*, **11**-3, 384/393 (1998)

[9] G. Cavory, R. Dupas, and G. Goncalves: A Genetic Approach to Solving the Problem of Cyclic Job shop Scheduling with Linear Constraints, *Eur. J. Oper. Res.*, 161, 73/85 (2005)

[10] J.T. Ootsuki, Y. Fujii, H. Mizutani, T. Sekiguchi: Immune System Derived Approach for Finding Sequence of a Sub-Class of Petri Nets, *Trans. IEEJ*, **188**-C, 419/427 (1998)

[11] R. Maeno, T. Nishi, M. Konishi: An Optimization Method for Routing Problems for Multiple AGVs by Decomposition of Petri Nets, *Transaction of the Institute of Systems, Control and Information Engineers*, **50**-11 (to appear) (2006)

[12] T. Nishi, M. Konishi, S. Hasebe and I. Hashimoto: Machine-Oriented Decentralized Scheduling Method using Lagrangian Decomposition and Coordination Technique, *Proc. Int. Conf. Robot. Automat*, pp. 4173-4178 (2002)