アルゴリズム 33－3
（1993．5．28）

# ペトリネットの発火系列問題に対する近似アルゴリズム

山内 雅弘　中村 伸治　渡辺 敏正
広島大学 工学部 第二類 回路システム工学講座
(724)東広島市鏡山一丁目４－１
(電話)0824-24-7662 (ファクシミリ)0824-22-7195
(電子メール)watanabe@huis.hiroshima-u.ac.jp

**概要：** 本論文はペトリネットの発火系列問題（LFSと略記）に対する性能の良い近似アルゴリズム *YWLFS* を提案する．ペトリネットの発火系列問題とは，ペトリネットPN，初期マーキングＭと発火回数ベクトルＸ（その要素X(t)が各トランジションｔの発火回数を表す）が与えられたとき，各トランジションｔがその中に合計で丁度X(t)回現れ，且つＭから順に発火可能であるトランジションの系列が存在するか否かを判定し，存在するならばその様な一つの系列δを求める問題である．この問題は非常に簡単な構造を持つペトリネットに対してさえNP-完全であることが知られている．実験では，これまでの当研究室での関連研究において既にδの存在が既知である2181個の入力例に対して*YWLFS*を適用した．その結果，*YWLFS*は2050個(94%)の入力例について実際にδを見つけ，δの発見に不成功であった131個の入力例については既知のδの長さの平均72%までの長さの発火可能な系列を求めている．このことから*YWLFS*は高い能力を持つものと思われる．

# An Approximation Algorithm for the Legal Firing Sequence Problem of Petri Nets

Masahiro Yamauchi, Shinji Nakamura and Toshimasa Watanabe

*Department of Circuits and Systems, Faculty of Engineering, Hiroshima University,*
*4-1 Kagamiyama, 1 Chome, Higashi-Hiroshima, 724 Japan*
*Phone: +81-824-24-7662 (Watanabe) Facsimile: +81-824-22-7195*
*E-mail: watanabe@huis.hiroshima-u.ac.jp*

**Key words:** firing sequences, Petri nets, schedulings, approximation algorithms, computational complexity

**Abstract:** The paper proposes an approximation algorithm *YWLFS* for the *legal firing sequence problem*, LFS for short, of Petri nets: " Given a Petri net, an initial marking M and a firing count vector X (with each component X(t) denoting the prescribed total firing number of a transition t), find a firing sequence δ which is legal on M with respect to X, where δ is a sequence of transitions and is called *legal* on M *with respect to* X if and only if the first transition of δ is firable on M, the rest can be fired one by one subsequently and each transition t appears exactly X(t) times in δ." Experimental results show that *YWLFS* is a promising algorithm: when it is applied to 2181 total test problems for each of which existence of a solution is guaranteed, it finds solutions to 2050 problems (94%), and, for each of 131 problems to which *YWLFS* failed to get solutions, average length of δ found by *YWLFS* is 72% of the total length of a corresponding solution.

## 1. Introduction

An approximation algorithm *YWLFS* for the *legal firing sequence problem* (LFS for short) of Petri nets is proposed and experimental evaluation is given. LFS is defined as follows:

> (LFS) Given a Petri net PN=(P,T,E,$\alpha$,$\beta$), an initial marking M and a firing count vector X, with each component X(t) denoting the prescribed total firing number of a transition t, find a firing sequence $\delta$ which is legal on M with respect to X, where $\delta$ is a sequence of transitions and is called *legal* on M *with respect to* X if and only if the first transition of $\delta$ is firable on M, the rest can be fired one by one subsequently and each transition t appears exactly X(t) times in $\delta$.

LFS is fundamental in the sense that it is included, as a subproblem, in various basic problems of Petri net theory. For example,

(1) classical scheduling problems [1,4,5] or cyclic scheduling problems that can be formulated as LFS for timed Petri nets ( Petri nets in which transitions have delays that is integers)[16,18];

(2) the minimum initial resource allocation problem (called the *minimum initial marking problem*, MIM for short) for Petri nets PN (see [12,13,14,17,19] for the definition and approximation algorithms with experimental results);

(3) the well-known marking reachability problem, asking for a firing sequence $\delta$ whose firing changes a given initial marking to a specified target marking [8,9].

In [12,13,15], LFS is introduced and its time complexity is formally analyzed. Unfortunately the recognition version (that is, asking "yes" or "no" on the existence of solutions ) of LFS is shown to be intractable (NP-complete) for a restricted class of problems. Polynomial or pseudo-polynomial time solvability of LFS for some classes of Petri nets having simple structure is also given. The details will be explained in Section 3. Even though intractability of LFS has been shown, it does not seem that any approximation algorithm for LFS has ever been proposed. A firing sequence $\delta$ which is legal on M with respect to X is called a *solution* to LFS, and let $\bar{\delta}$ denote the firing count vector such that $\bar{\delta}$(t) is equal to the total occurrence of t in $\delta$. We consider a firing sequence $\delta$ as an approximate solution to LFS in the following sense:

- (i) $\delta$ is legal on M;
- (ii) $\bar{\delta}(t) \le X(t)$ for any transition $t \in T$ of PN;
- (iii) the total length $|\delta|$ of $\delta$ is as large as possible among those firing sequences $\delta$' satisfying (i) and (ii).

The paper proposes an approximation algorithm *YWLFS* for LFS. Time complexity of *YWLFS* is $O(\overline{X} \cdot |X|^2 \cdot |T| \cdot |E|)$, where $\overline{X}=\max\{X(t)|t \in T\}$ and |X| denotes the total sum of X(t), $\forall t \in T$. Note that *YWLFS* is a pseudo-polynomial time algorithm. Experimental results show that *YWLFS* is a promising algorithm: when it is applied to 2181 test problems for each of which existence of a solution is guaranteed by the results given in [16-19], it finds solutions to 2050 problems (94%) of them, and, for each of 131 problems to which *YWLFS* failed to get solutions, average length of $\delta$ found by *YWLFS* is 72% of the total length of a corresponding solution. More precisely, *YWLFS* finds solutions to 1765 problems (98%) of 1800 test problems with Petri nets having $5 \le |P| \le 90$, $9 \le |T| \le 99$, $34 \le |E| \le 554$, $9 \le |X| \le 297$ and state machines as underlying Petri nets (see Section 2 for the definition); to 285 problems (75%) of 381 problems with those having $15 \le |P| \le 97$, $13 \le |T| \le 97$, $67 \le |E| \le 536$, $13 \le |X| \le 291$ and general Petri nets as underlying ones, where |P|, |T| and |E| are the numbers of places, transitions and edges, respectively. For each of 35 problems having state machines (96 problems having general Petri nets, respectively) as underlying Petri nets such that *YWLFS* failed to get solutions, average length of $\delta$ found by *YWLFS* is 94% (64%) of the total length of a corresponding solution.

## 2. Preliminaries

Technical terms or notations whose definitions are not given in this paper can be identified in [10,11]. We assume that the readers are familiar with graph theory terminologies (see [2] for example). A *Petri net* is a bipartite digraph PN=(P,T,E,$\alpha$,$\beta$), where P is the set of *places*, T is that of *transitions* such that $P \cap T = \phi$, and $E=E_{in} \cup E_{out}$ is an edge set such that $E_{in}$={edges from T to P} and $E_{out}$={edges from P to T} with weight functions $\alpha: E_{out} \to Z^+$ (non-negative integers) and $\beta: E_{in} \to Z^+$. We always consider PN to be a simple directed digraph unless otherwise stated. PN is a *marked graph* if any $p \in P$ has $|{}^*p| \le 1$ and $|p^*| \le 1$. PN is a *state machine* if any $t \in T$ has $|{}^*t| \le 1$ and $|t^*| \le 1$. PN is a *free choice net* if, for each arc $(p,t) \in E_{out}$, either $p^*=\{t\}$ or ${}^*t=\{p\}$ holds. PN is a *marked graph* if any $p \in P$ has $|{}^*p| \le 1$ and $|p^*| \le 1$. PN is a *state machine* if $(\forall t \in T)$ $|{}^*t|$, $|t^*| \le 1$. PN is a *free choice net* if, for each arc $(p,t) \in E_{out}$, either $p^*=\{t\}$ or ${}^*t=\{p\}$ holds. PN is a *forward* or *backward conflict-free* Petri net if $|p^*|=1$ or $|p^*|=1$ for $\forall p \in P$, respectively. PN with an initial marking M is called *persistent* if, for any marking M' that is reachable from M, $t_1$ is firable on M'[$t_2$> whenever both $t_1$ and $t_2$ are firable on M'. It is shown in [7] that conflict -free Petri nets are persistent. (See [6] for the related results.)

Let $A=A^+-A^-=[a_{ij}^+]-[a_{ij}^-]$ denote a $|P| \times |T|$ matrix, called the *place-transition incidence matrix* of PN, which is defined by

$$a_{ij}^+=\begin{cases}\beta(t_j,p_i) & \text{if } (t_j,p_i) \in E,\\ 0 & \text{otherwise,}\end{cases} \qquad a_{ij}^-=\begin{cases}\alpha(p_i,t_j) & \text{if } (p_i,t_j) \in E,\\ 0 & \text{otherwise.}\end{cases}$$

A *marking* M for PN is a function $M: P \to Z^+$, and |M| denotes the total sum of M(p) over all $p \in P$. A transition t is *firable* on a marking M if $M(p) \ge \alpha(p,t)$ for $\forall p \in {}^*t$. *Firing* such t on M is to define a marking M' such that, for $\forall p \in P$, we have $M'(p)=M(p)+\beta(t,p)$ if $p \in t^*-{}^*t$, $M'(p)=M(p)-\alpha(p,t)$ if $p \in {}^*t-t^*$, $M'(p)=M(p)-\alpha(p,t)+\beta(t,p)$ if $p \in {}^*t \cap t^*$ and M'(p)=M(p) otherwise. We denote as M'=M[t>. If no transition is firable on a marking M then M is called a *dead marking*. Let $\delta=t_{i1}...t_{is}$ be a sequence of transitions, called a *firing sequence*, and $\bar{\delta}(t)$ be the total number of occurrences of t in $\delta$. $\bar{\delta}=[\bar{\delta}(t_1)...\bar{\delta}(t_n)]^{tr}$ (n=|T|) is called the *firing count vector* of $\delta$. Let $|\delta|$ denote the sum of $\bar{\delta}(t)$ over all $t \in T$. $\delta$ is called *single-round* if each transition t in $\delta$ occurs $\bar{\delta}(t)$ times consecutively, and is called *multi-round* otherwise. For a marking M and an n-dimensional vector $X=[x_1 ... x_n]^{tr}$, $\delta$ is *legal* on M if and only if $t_{ij}$ is firable on $M_{j-1}$ for $j=1,...,s$, where $M_0=M$ and $M_j=M_{j-1}[t_{ij}>$. We denote $M[\delta>=M_s$. $\delta$ is *legal* on M *with respect to* X if and only if $\delta$ is legal on M and $\bar{\delta}=X$. An n-dimensional vector X with each component being a nonnegative integer is called a *T-invariant* of PN if and only if $X \neq \mathbf{0}$ and $AX=\mathbf{0}$. Let $m=|P|$. An m-dimensional vector Y with each component being a nonnegative integer is called a *P-invariant* of PN if and only if $Y \neq \mathbf{0}$ and $Y^{tr}A=\mathbf{0}$. (A T-invariant or a P-invariant plays an important role in Petri net theory. See [10, 11].) Let |X| denotes the total sum of X(t), $\forall t \in T$, and let $\overline{X}=\max\{X(t)|t \in T\}$. The support ||X|| of a vector X is the set of transitions whose X components are nonzero. PN is *consistent* if PN has a T-invariant X with ||X||=T. A T-invariant X of PN is *elementary* if there is no T-invariant X' of PN such that $||X'|| \subset ||X||$ ( a proper inclusion ). See [3] for the details of *NP*-completeness or *NP*-hardness. For any nonnegative real number x, let $\lfloor x \rfloor$ denote the maximum integer not greater than x.

Given a Petri net $PN=(P,T,E,\alpha,\beta)$, we may add a set L of places, called *processor pools*, and associate each $p \in L$ with a set $N(p) \subseteq T$ by adding two edges (p,t),(t,p) for any $t \in N(p)$, where $N(p) \cap N(p')$ may be nonempty even if $p \neq p'$. This is often the case with timed Petri nets, where such a transition $t \in N(p)$ represents a task to be processed by a processor denoted by a place p. (See [16-19] for timed Petri nets and related discussions concerning scheduling problems.) In this case the original Petri net PN is often called the *underlying Petri net* of the resulting one.

### 3. The Legal Firing Sequence Problem LFS

We explain intractability of LFS and the basic idea of the approximation algorithm *YWLFS*, by using an example. Also summarized are known results on LFS.

First a simple example of LFS is given.

**Example 1.** Consider the Petri net PN shown in Fig.1, and suppose that we are given a firing count vector X and an initial marking $M_0$ as

$X=[X(t_1),X(t_2),X(t_3),X(t_4)]^{tr}=[1,1,1,1]^{tr}$,

$M_0=[M_0(p_1),M_0(p_2),M_0(p_3),M_0(p_4)]^{tr}=[1,0,0,0]^{tr}$.

There is a firing sequence $\delta=t_1t_2t_3t_4$ which is legal on $M_0$ with respect to X. There also exist a firing sequence $\delta'=t_1t_4$ which is legal on this marking $M_0$. Clearly $\bar{\delta'}$ is not equal to X. If we unfortunately select $\delta'$ then backtracking is required. For example, if we choose $\delta'$ then we reach

$M_0'=M_0[\delta'>=[0,0,0,1]^{tr}$,

which is a dead marking. Let M and $X_{rest}$ denote the current marking and the current firing count vector, and initially we set $M \leftarrow M_0$ and $X_{rest} \leftarrow X$. We also consider a set of transitions:

$F=\{t \in T | X_{rest}(t)>0 \text{ and } (\forall p \in {}^*t) M(p) \geq \alpha(p,t)\}$.

If F={t} then the only possible choice is the transitin t, and we concatenate t at the end of the current firing sequence $\delta$ as $\delta \leftarrow \delta \cdot t$. The point is how we handle the case where $|F| \geq 2$. In this example consider the marking $M=[0,1,0,0]^{tr}=M_0[t_1>$. Then we have $F=\{t_2,t_4\}$, and the one avoiding reachability to dead markings is to be selected. What we are requiring here is a certain measure showing that $t_2$ is the one to be fired next. The approximation algorithm *YWLFS* to be proposed in this paper computes a value effect(t) for every transition t in F, and choose a transition $t_f$ with $\text{effect}(t_f)=\max\{\text{effect}(t)|t \in F\}$ as the one to be fired next. The details of computing effect(t) will be given later in Section 4. Here we give only the values effect(t), $t \in \{t_2,t_4\}$:

$\text{effect}(t_2)=3$ and $\text{effect}(t_4)=0$.

Hence $t_2$ is selected as desired. Intuitively speaking, these values in this example mean the following: if $t_2$ is fired once then the produced token is used in making three transitions firable, while firing $t_4$ has no such transition. In fact, $t_3$ is made firable by firing $t_2$, and then both $t_2$ and $t_4$ become firable after firing $t_3$. (It should be noted that effect(t) does not always denote the number of such transitions but shows possibility of their existence.) *YWLFS* repeats the three processes in this order: computing effect(t) for all $t \in T$, finding $t_f$ with $\text{effect}(t_f)=\max\{\text{effect}(t)|t \in T\}$ and then fire $t_f$. ♦

The known results concerning LFS are summarized in the following theorems.

**Theorem 1** [12,13,15]. LFS is *NP*-complete even if PN is a consistent free choice net with $|^*p|>0$, $0<|p^*| \leq 2$ for $\forall p \in P$ and $|^*t|>0$, $|t^*|>0$ for $\forall t \in T$, |M|=1 and X=1 is a T-invariant with ||X||=T.♦

**Theorem 2** [12,13,15]. LFS is *NP*-complete even if PN, X and M are restricted to a *consistent free choice net*, an *elementary T-invariant* with ||X||=T and a marking with |M|=1, respectively, satisfying one of (i) through (iii) for $\forall p \in P$ and $\forall t \in T$:

(i) $|^*p|>0$, $0<|p^*| \leq 2$, $|^*t|>0$, $|t^*|>0$ and X=1.

(ii) All edge weights are equal to 1, $|^*p|>0$, $|p^*|>0$, $|^*t|>0$, $|t^*|>0$ and X=1.

(iii) All edge weights are equal to 1, $|^*p|>0$, $|p^*|>0$, $|^*p|+|p^*| \leq 3$, $|^*t|>0$, $|t^*|>0$.♦

**Theorem 3** [12,13,15]. LFS is NP-complete even if PN, X and M are restricted to a consistent state machine, T-invariant and a marking with |M|=1, respectively, satisfying either (i) or (ii) for $\forall p \in P$:

(i) $|^*p|>0$, $|p^*|>0$, some edge weights are greater than 1 and X=1;

(ii) $|^*p|>0$, $|p^*|>0$, $|p^*|+|^*p| \leq 3$ and all edge weights are equal to 1.♦

For polynomial or pseudo-polynomial time solvability of LFS, we have the following theorems. The results on persistent nets by [6] is essentially used.

**Theorem 4** [12,13,15]. LFS for a persistent Petri net in the multi-round firing (for a conflict-free PN in the single-round firing, respectively) can be solved in $O(|P||X|)$ ($O(|P||T|)$) time, where X is a T-invariant of PN if PN is backward conflict-free.♦

**Theorem 5** [15]. Given a state machine PN with all edge weights equal to 1, an initial marking M and a firing vector X, the recognition version of LFS can be answered in $O(|X|)$ time, and if the answer is "yes" then there is an $O(|X|^2)$ algorithm for finding a solutin to LFS. ♦

**Remark 1.** It should be noted that, with the above notations of time complexities using |X| in Theorems 4 and 5, they appear to be bounded by a polynomial function of |X|. However this is not the case. Since each X(t) takes size proportional to $log_2X(t)$ bits in the input, |X| has size proportional to $\Delta=\sum_{t\in T} log_2X(t)$ bits. We have

$$|X|/\Delta \geq |X|/(|T|log_2\bar{X}) \geq |X|/(|T|log_2|X|),$$

and the last term is not bounded by any polynomial function of $|T|log_2|X|$. That is, |X| is not polynomially bounded by the size Δ of input. Nevertheless we use such representation as above for notational simplicity. Clearly if |X| is bounded by |P|, |T|, |E| or a constant then |X|/Δ is bounded by a polynomial function of such one of them.

## 4. Approximation Algorithm *YWLFS*

The algorithm *YWLFS* consists of three procedures,

*SEARCH_LFS*($t_f$,$M_v$,visit,max),
*COMP_EFFECT*(M,$X_{rest}$,F,effect),
*FIRE*(M,$X_{rest}$).

*YWLFS* constructs a firing sequence δ such that $\bar{\delta}$ is as close to $X_{rest}$ (which is initially set to X) as possible by repeating procedure *FIRE*(M,$X_{rest}$): the procedure finds a transition $t_f$ such that firing $t_f$ once has possibility of making many other transitions firable (that is, making occurrence of subsequent dead markings less possible).

The value effect(t) is a measure to be used in *FIRE*(M,$X_{rest}$), that is, a transition t with the maximum effect(t) is selected as the one to be fired next. Intuitively speaking, if effect(t) is large then the tokens produced by firing t once will necessarily be used in making many other transitions firable (that is, these transitions cannot fire without them). Hence we may expect that firing such t will avoid occurrence of subsequent dead markings. For each t∈T, the value effect(t) is computed in *COMP_EFFECT*(M,$X_{rest}$,F,effect). The procedure first computes two values max(p) for each p∈P and visit(t) for the transition t in *SEARCH_LFS*(t,$M_v$,visit,max), where $M_v$=M[t> and *SEARCH_LFS*(t,$M_v$,visit,max) is a depth-first search tracing edges in their direction. It also gives a maximal set of transitions t such that tokens produced by firing $t_f$ and subsequent firing of other transitions are used in making t firable. The value visit(t) denotes maximum possible number of firing of t starting from $M_v$ if $t\neq t_f$ or from M if $t=t_f$. The value max(p) denotes maximum possible number of tokens that can be brought into p after firing of $t_f$ starting from M. Computing effect(t) requires two more values supply(p) for p∈P and rate(t') for t'∈T. The value supply(p) is given by

$$supply(p)=(\sum_{t\in {}^*p} \beta(t,p)\cdot visit(t)) / (\sum_{t\in {}^*p} \beta(t,p)\cdot X_{rest}(t)),$$

where both the numerator and the denominator denote the number of tokens brought into p through firing each t∈*p with $X_{rest}(t)>0$ by visit(t) times (which is maximum possible from $M_v$) in the former and $X_{rest}(t')$ times in the latter. The other value rate(t') for t'∈T with $X_{rest}(t')>0$ is given by

$$rate(t')=(\sum_{p\in {}^*t'} \alpha(p,t')\cdot count'\cdot supply(p)) / (\sum_{p\in {}^*t'} \alpha(p,t')),$$

where $count'=\min\{\lfloor(max(p)+M_v(p))/\alpha(p,t')\rfloor, visit(t')\}$, both the numerator and the denominator are the total number of tokens to be deleted by firing t' count' times in the former and once in the latter, and the numerator is expected to represent the number of tokens deleted among those brought into places p∈*t after created by firing t' count' times. Now the value effect(t) is defined by

$$effect(t)=\sum_{t'\in T} rate(t').$$

We can expect that if t has large effect(t) then firing t creates tokens that will be necessary in making many other transitions firable subsequently. Hence *FIRE*(M,$X_{rest}$) selects t having maximum effect(t) as the one to be fired next.

We give Example 2 showing compution of these values in the problem of Example 1.

**Example 2.** If $M=M_0$ and $X_{rest}=X$ then $F=\{t_1\}$ and we immediately obtain

$M=[0,1,0,0]^{tr}$ and $\delta=t_1$.

In the next step,

$F=\{t_2,t_4\}$, $X_{rest}=[0,1,1,1]^{tr}$, $M_v=[0,0,0,0]^{tr}$

and we get the following values as given below. If $t_f=t_2$ then

| | $p_1$ | $p_2$ | $p_3$ | $p_4$ | | $t_1$ | $t_2$ | $t_3$ | $t_4$ |
|---|---|---|---|---|---|---|---|---|---|
| max | 0 | 1 | 1 | 1 | visit | 0 | 1 | 1 | 1 |
| $\beta_{sum}$ | 0 | 1 | 1 | 1 | $\alpha_{sum}$ | 0 | 1 | 1 | 1 |
| β' | 0 | 1 | 1 | 1 | α' | 0 | 1 | 1 | 1 |
| supply | 0 | 1 | 1 | 1 | rate | 0 | 1 | 1 | 1 |
| | | | | | count' | 0 | 1 | 1 | 1 |

and

effect($t_2$)=3.

On the other hand if $t_f=t_4$ then

| | $p_1$ | $p_2$ | $p_3$ | $p_4$ | | $t_1$ | $t_2$ | $t_3$ | $t_4$ |
|---|---|---|---|---|---|---|---|---|---|
| max | 0 | 0 | 0 | 1 | visit | 0 | 0 | 0 | 1 |
| $\beta_{sum}$ | 0 | 1 | 1 | 1 | $\alpha_{sum}$ | 0 | 1 | 1 | 1 |
| β' | 0 | 0 | 0 | 1 | α' | 0 | 0 | 0 | 0 |
| supply | 0 | 0 | 0 | 1 | rate | 0 | 0 | 0 | 0 |
| | | | | | count' | 0 | 0 | 0 | 0 |

and

effect($t_4$)=0.

Hence $t_2$ is fired next and we obtain $\delta=t_1t_2$. Similarly the desired firing sequence $\delta=t_1t_2t_3t_4$ is obtained. ♦

The formal description of *YWLFS* is as follows.

**procedure** *SEARCH_LFS*($t_f$,$M_v$,visit,max);

```
/* computes max(p) for each p and visit(t_f): max(p) denotes
maximum possible number of tokens that can be brought into p
after firing of t_f on M; visit(t) does maximum possible number
of firing of t starting from M_v if t≠t_f or from M if t=t_f */
   begin
1. for each p'∈t_f* do
      begin
2.    temp←visit(t_f)·β(t_f,p');
      /* computing the value max(p') */
3.    if (max(p') < temp) then max(p')←temp;
                    /* #tokens brought into p' */
4.    for each t'∈p'* do
         begin
5.       temp'←α(p',t')·visit(t');
         /* #tokens deleted from p' */
6.       count←X_rest(t') - visit(t');
         /* count>0 if and only if X_rest(t')>0 and
                                  visit(t')<X_rest(t') */
7.       if (count>0)∧
            (M_v(p')<temp'+α(p',t')≤max(p')+M_v(p'))
         /* tokens produced by firing t_f visit(t_f) times are
         required in making t' firable */
         then
            begin /*updating visit(t') and repeat
                  SEARCH_LFS starting from t'*/
8.          k←⌊(max(p') + M_v(p') - temp') / α(p',t')⌋;
9.          if (count > k) then count←k;
10.         visit(t')←visit(t') + count;
11.         if (count > 0) then
               SEARCH_LFS(t',M_v,visit,max)
            end
         end
      end
   end;

procedure COMP_EFFECT(M,X_rest,F,effect);
   /* computes effect(t) */
   begin
1. for each t_f∈F do
      begin
2.    for each p∈P do
         begin
         max(p)←0; supply(p)←0; β_sum(p)←0; β'(p)←0
         end;
3.    for each t∈T do
         begin
         visit(t)←0; rate(t)←0;
         α_sum(t)←0; α'(t)←0
         end;
4.    count←X_rest(t_f);
5.    for each p∈*t_f do
         if (count > ⌊M(p)/α(p,t_f)⌋) then
            count←⌊M(p)/α(p,t_f)⌋;
6.    visit(t_f)←count; /* maximum possible number
                      of firing of t_f starting from M */
7.    for each p∈P do /* defining M_v */
         if p∈*t_f then M_v(p)←M'(p)-α(p,t_f)
         else M_v(p)←M'(p);
8.    SEARCH_LFS(t_f,M_v,visit,max);
9.    for each p∈P do  /* computing supply(p) */
         begin
10.      for each t∈*p with X_rest(t)>0 do
            begin
11.         β_sum(p)←β_sum(p)+β(t,p)·X_rest(t);
12.         β'(p)←β'(p)+β(t,p)·visit(t)
            end;
13.      supply(p)←β'(p)/β_sum(p)
         end;
14.   for each t∈T with X_rest(t)>0 do
         /* computing rate(t) */
         begin
15.      for each p∈*t do
            begin
16.         count'←max{⌊(max(p)+M_v(p))/α(p,t)⌋,
                                                visit(t)};
17.         α_sum(t)←α_sum(t)+α(p,t);
18.         α'(t)←α'(t)+α(p,t)·count'·supply(p)
            end;
19.      rate(t)←α'(t) / α_sum(t)
         end;
20.   for each t∈T do effect(t_f)←effect(t_f)+rate(t)
      end
   end;

procedure FIRE(M,X_rest);
   /* repeats finding a transition t with
   effect(t)=max{effect(t')|t'∈T}, and executes firing
   of t on a current marking M */
   begin
1. F←{t∈T|X_rest(t)>0 and (∀p∈*t)M(p)≥α(p,t)};
2. while (|F| ≠ 0) do
      begin
      if |F| = 1 then next_t←t∈F;
3.    else
         begin
4.       for each t∈T do effect(t)←0;
5.       COMP_EFFECT(M,X_rest,F,effect);
         /* computes effect(t) for all t∈T */
6.       effect_max←0;
         /* finding max{effexct(t)|t∈T} */
7.       for each t∈T do
8.          if (effect_max < effect(t)) then
9.             begin effect_max←effect(t); next_t←t
               end
         end;
      /* next_t has effect_max=max{effect(t)|t∈T} */
10.   X_rest(next_t)←X_rest(next_t) - 1;
11.   M←M[next_t>; δ←δ·next_t; /*concatenation*/
12.   F←{t∈T|X_rest(t)>0 and (∀p∈*t)M(p)≥α(p,t)}
      end
   end;

algorithm YWLFS;
/* input: PN, M_0 and X */
/* output: δ and X_rest" */
   begin
1. δ←∅;
2. for each p∈P do M(p)←M_0(p);
3. for each t∈T do X_rest(t)←X(t);
4. while (X_rest(t) ≠ 0 for some t∈T) do
```

5. $FIRE(M, X_{rest})$;
6. Output δ and $X_{rest}$;
/* if $X_{rest}(t) \neq 0$ for some t then finding a solution is failed */
**end.**

It is clear that *YWLFS* finds a firing sequence δ that is legal on $M_0$, since procedure $FIRE(M, X_{rest})$ chooses a transition next_t that is firable on a current marking M with $X_{rest}(next\_t)>0$. Time complexity of the procedures are summarized as follows:

| | |
|---|---|
| *SEARCH_LFS* | $O(\overline{X} \cdot |E|)$, |
| *COMP_EFFECT* | $O(\overline{X} \cdot |X| \cdot |E|)$, |
| *FIRE* | $O(\overline{X} \cdot |X|^2 \cdot |E|)$, |

where $\overline{X}=\max\{X(t) | t \in T\}$ and $|X|$ denotes the total sum of $X(t)$, $\forall t \in T$. Hence time complexity of *YWLFS* is

$O(\overline{X} \cdot |X|^2 \cdot |T| \cdot |E|)$.

Note that *YWLFS* is a pseudo-polynomial time algorithm.

## 5. Experimental Results

We have implemented *YWLFS* on a workstation SUN SPARC station by using the C programming code. All the test problems are taken from those which are constructed and have been used in our research such as [16-19], where it is described how they are generated. The underlying Petri nets are either state machines or general Petri nets. It should be noted that existence of a solution to each of these problems is guaranteed. Hence capability of *YWLFS* can be shown by means of results obtained by aplying *YWLFS* to these problems. Experimental results show that *YWLFS* is a promising algorithm.

We first summarize the number and sizes of test problems as well as some statistical data.

(1) The number of test problems:

| **underlying Petri nets(upn)** | **#test problems** |
|---|---|
| state machines(sm) | 1800 |
| general Petri nets(gn) | 381 |
| **total** | 2181 |

(2) Sizes of Petri nets:

| **upn** | \|P\| | \|T\| | \|E\| | \|X\| |
|---|---|---|---|---|
| sm | $5 \le |P| \le 90$ | $9 \le |T| \le 99$ | $34 \le |E| \le 554$ | $9 \le |X| \le 297$ |
| gn | $15 \le |P| \le 97$ | $13 \le |T| \le 97$ | $67 \le |E| \le 536$ | $13 \le |X| \le 291$ |

(3) Firing count vectors X:

$X=kX'$ for $k=1,2,3$, and $X'=\mathbf{1}$ ($X'(t)=1$ for $\forall t \in T$).

(4) Successful cases (where *YWLFS* finds solutions):

| **upn** | **#cases** | **ratio** (= #cases / 2181) |
|---|---|---|
| sm | 1765 | 98% |
| gn | 285 | 75% |
| **total** | 2050 | 87% |

(5) Unsuccessful cases:

| **upn** | **#cases** | **average of ratio** $|\bar{\delta}| / |X|$ |
|---|---|---|
| sm | 35 | 94% |
| gn | 96 | 64% |
| **total** | 131 | 72% |

A part of other experimental results are shown in Table 1. Other statistical data are given in Figs. 2 through 5, and in Table 2. shows in the column "Success" the total number of successful cases out of 600 (out of 127, respectively) test problems, each having state machines (general Petri nets) as underlying Petri nets, for each value of k, k=1,2,3. The column "Ave. ratio" denotes average ratios ($=|\bar{\delta}|/|X|$) over all 600 (127) test problems. The column "Success" is schematically shown in Figs. 2 and 3. Fig. 4 (Fig. 5, respectively) shows average ratios ($=|\bar{\delta}|/|X|$) as well as the total number of unsuccessful cases out of 600 (out of 127) test problems, each having state machines (general Petri nets) as underlying Petri nets, for values of $|X|$.

## 6. Concluding Remarks

Experimental results show that *YWLFS* is a very promising approximation algorithm: it finds solutions to 2050 problems (94%) of 2181 test problems for which existence of solutions are guaranteed. Theoretical estimate of worst approximation by *YWLFS*, as well as providing more experimental results, is left for future research.


## Acknowledgements

The research of T. Watanabe was partly supported by the Telecommunications Advancement Foundation (TAF), Tokyo, Japan; and is partly supported by The Okawa Institute of Information and Telecommunication, Tokyo, Japan.


## References


[1] E. G. Coffman, Computer and Job-Shop Scheduling Theory, John Wiley & Sons, NY, 1976.

[2] S. Even: Graph Algorithms, Pitman, London, UK, 1978.

[3] M. R. Garey and D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-completeness, Freeman, San Francisco, CA, 1978.

[4] R. L. Graham, Bounds on Multiprocessing Timing Anomalies, SIAM J. Appl. Math., Vol.17, No.2(1969), pp.416-429.

[5] R. L. Graham, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey, Annals of Discrete Mathematics, Vol.5(1979), pp.287-326.

[6] R. M. Keller: A fundamental theorem of asynchronous parallel computation, Proc. Sagamore Comput. Cof. (Lecture Notes in Computer Science, Parallel processing (T-Y. Feng (Ed.)), Springer-Verlag, Berlin, pp.102-112(1975).

[7] L. H. Landweber and E. L. Robertson: Properties of conflict-free and persistent Petri nets, J. Assoc. Comput. Mach., Vol.25, No.3(1978), pp.352-364.

[8] E. W. Mayr, An Algorithm for the General Petri Net Reachability Problem, Proc. 13th Annual ACM Symposium on Theory of Computing(1981), pp.238-246.

[9] T. Murata: Circuit theoretic analysis and synthesis of marked graphs, IEEE Trans. Circuits and Systems, CAS-24-7(1977),400-405.

[10] J. L. Peterson: Petri Net Theory and the Modeling of Systems, Prentice-Hall Inc., Englewood Cliffs, NJ, 1981.

[11] W. Reisig: Petri Nets/An introduction, Springer-Verlag, Berlin, 1982.

[12] T. Watanabe, Y. Mizobata and K. Onaga, Legal firing sequences and minimum initial markings for Petri nets, Proc. 1989 IEEE International Symposium on Circuits and Systems, pp.323-326 (May 1989).

[13] T. Watanabe, Y. Mizobata, and K. Onaga, Legal firing sequence and

related problems of Petri nets, Proc. the 3rd International Workshop on Petri Nets and Performance Models(PNPM89), IEEE Computer Society Press, pp.277-286(1989).

[14] T. Watanabe, Y. Mizobata and K. Onaga, Minimum initail marking problems of Petri nets, Trans. IEICE of Japan, Vol.E72, No.12(1989), pp.1390-1399.

[15] T. Watanabe, Y. Mizobata and K. Onaga, Time complexity of legal firing sequence and related problems of Petri nets, Trans. IEICE of Japan, Vol.E72, No.12(1989), pp.1400-1409.

[16] T. Tanida, T. Watanabe, M. Yamauchi and K. Onaga, Priority-list scheduling in timed Petri nets, Trans. IEICE of Japan, Vol.E75-A, No.10(1992), pp.1394-1406.

[17] T. Watanabe, T. Tanida, M. Yamauchi and K. Onaga, The minimum initial marking problem for scheduling in timed Petri nets, Trans. IEICE of Japan, Vol.E75-A, No.10(1992), pp.1407-1421.

[18] T. Watanabe and M. Yamauchi, New priority-lists for scheduling in timed Petri nets, Proc. 14th International Conference on Application and Theory of Petri nets (Lecture Notes in Computer Science, Springer-Verlag) (June 1993), to appear.

[19] M. Yamauchi and T. Watanabe, A new approximation algorithm for the minimum initial marking problem of Petri nets, Proc. 6th Karuizawa Workshop on Circuits and Systems, IEICE of Japan(April 1993), 279-284.

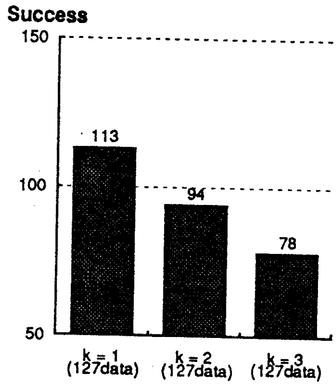


Fig. 3. A bar-graph representation of the column "Success" of Table 2 for the cases having general Petri nets as underlying ones.

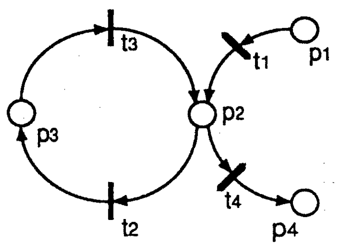


Fig. 1. An example of Petri net PN.

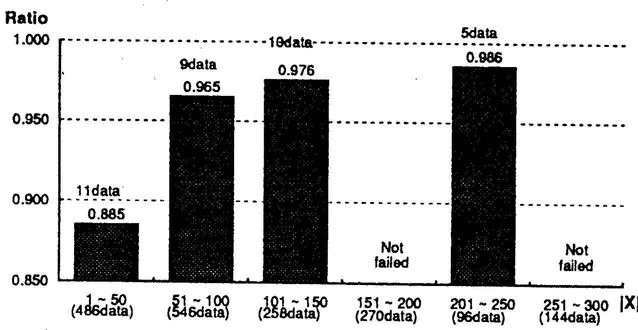


Fig. 4. Average ratios (=|δ|/|X|) as well as the total number of unsuccessful cases out of 600 test problems, each having state machines as underlying Petri nets, for values of |X|.

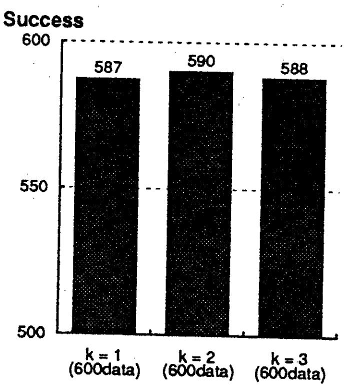


Fig. 2. A bar-graph representation of the column "Success" of Table 2 for the cases having state machines as underlying Petri nets.

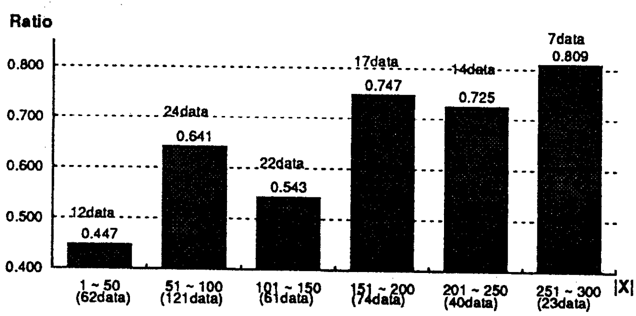


Fig. 5. Average ratios (=|δ|/|X|) as well as the total number of unsuccessful cases out of 127 test problems, each having general Petri nets as underlying ones, for values of |X|.

Table 1. A part of our experimental results. The columns "#DATA", "$l$", "Ratio" and "Time" denote data identification, the length $|\bar{\delta}|$ of firing sequences $\delta$ found by *YWLFS*, the ratio $|\bar{\delta}|/|X|$ and CPU time in 1/60 second, respectively, and the other columns are self-explanatory.

| #DATA | \|P\| | \|T\| | \|E\| | \|X\| | k | $l$ | Ratio | Time |
|---|---|---|---|---|---|---|---|---|
| gn1.16.pn | 49 | 52 | 210 | 52 | 1 | 50 | 0.962 | 2416 |
| gn1.19.pn | 66 | 70 | 278 | 140 | 2 | 140 | 1.000 | 9877 |
| gn1.26.pn | 87 | 80 | 330 | 240 | 3 | 152 | 0.633 | 41886 |
| gn1.31.pn | 94 | 94 | 376 | 94 | 1 | 94 | 1.000 | 12383 |
| gn1.36.pn | 90 | 91 | 370 | 273 | 3 | 273 | 1.000 | 85127 |
| gn1.43.pn | 32 | 37 | 149 | 74 | 2 | 64 | 0.865 | 1563 |
| gn1.50.pn | 50 | 54 | 214 | 54 | 1 | 54 | 1.000 | 1472 |
| gn2.18.pn | 53 | 55 | 250 | 110 | 2 | 110 | 1.000 | 914 |
| gn2.25.pn | 71 | 70 | 320 | 140 | 2 | 140 | 1.000 | 2914 |
| gn2.28.pn | 67 | 70 | 316 | 210 | 3 | 206 | 0.981 | 5123 |
| gn2.31.pn | 86 | 83 | 390 | 83 | 1 | 83 | 1.000 | 1587 |
| gn2.34.pn | 89 | 79 | 372 | 158 | 2 | 158 | 1.000 | 4830 |
| gn2.39.pn | 96 | 93 | 422 | 93 | 1 | 93 | 1.000 | 2658 |
| gn2.40.pn | 96 | 93 | 422 | 186 | 2 | 186 | 1.000 | 6394 |
| gn3.1.pn | 18 | 13 | 77 | 13 | 1 | 5 | 0.385 | 2 |
| gn3.13.pn | 51 | 54 | 302 | 54 | 1 | 54 | 1.000 | 499 |
| gn3.14.pn | 51 | 53 | 286 | 106 | 2 | 106 | 1.000 | 1133 |
| gn3.23.pn | 65 | 68 | 376 | 68 | 1 | 68 | 1.000 | 989 |
| gn3.26.pn | 89 | 80 | 450 | 160 | 2 | 124 | 0.775 | 4100 |
| gn3.36.pn | 92 | 91 | 514 | 273 | 3 | 222 | 0.813 | 16568 |
| gn3.52.pn | 50 | 54 | 288 | 162 | 3 | 162 | 1.000 | 3426 |
| gn3.61.pn | 80 | 78 | 420 | 78 | 1 | 78 | 1.000 | 1774 |
| gn3.62.pn | 79 | 77 | 428 | 231 | 3 | 231 | 1.000 | 11367 |
| gn3.69.pn | 90 | 95 | 536 | 285 | 3 | 285 | 1.000 | 26027 |
| gn3.72.pn | 87 | 95 | 526 | 95 | 1 | 95 | 1.000 | 3555 |
| gn3.77.pn | 16 | 13 | 71 | 26 | 2 | 26 | 1.000 | 18 |
| gn3.80.pn | 28 | 33 | 170 | 99 | 3 | 99 | 1.000 | 642 |
| gn3.91.pn | 60 | 47 | 263 | 94 | 2 | 94 | 1.000 | 793 |
| gn3.93.pn | 58 | 46 | 272 | 138 | 3 | 97 | 0.703 | 1498 |
| gn3.99.pn | 76 | 81 | 448 | 243 | 3 | 243 | 1.000 | 13462 |
| sm1.1.pn | 14 | 16 | 64 | 16 | 1 | 16 | 1.000 | 26 |
| sm1.103.pn | 84 | 95 | 378 | 285 | 3 | 285 | 1.000 | 89518 |
| sm1.105.pn | 81 | 95 | 378 | 95 | 1 | 95 | 1.000 | 12238 |
| sm1.142.pn | 83 | 99 | 396 | 198 | 2 | 198 | 1.000 | 37896 |
| sm1.144.pn | 79 | 99 | 396 | 99 | 1 | 99 | 1.000 | 14896 |
| sm1.156.pn | 13 | 24 | 96 | 24 | 1 | 24 | 1.000 | 150 |
| sm1.175.pn | 88 | 99 | 396 | 297 | 3 | 297 | 1.000 | 146404 |
| sm1.180.pn | 80 | 99 | 394 | 99 | 1 | 99 | 1.000 | 15350 |
| sm1.193.pn | 48 | 58 | 232 | 116 | 2 | 116 | 1.000 | 8280 |
| sm1.199.pn | 60 | 71 | 284 | 71 | 1 | 71 | 1.000 | 7874 |
| sm1.23.pn | 55 | 70 | 278 | 140 | 2 | 140 | 1.000 | 14697 |
| sm1.35.pn | 76 | 94 | 376 | 94 | 1 | 94 | 1.000 | 10624 |
| sm1.36.pn | 74 | 94 | 374 | 282 | 3 | 282 | 1.000 | 83678 |
| sm1.52.pn | 41 | 54 | 214 | 108 | 2 | 108 | 1.000 | 7493 |
| sm1.67.pn | 83 | 94 | 374 | 94 | 1 | 94 | 1.000 | 12747 |
| sm1.68.pn | 81 | 94 | 374 | 188 | 2 | 188 | 1.000 | 32855 |
| sm1.94.pn | 43 | 54 | 216 | 54 | 1 | 54 | 1.000 | 3308 |
| sm1.97.pn | 69 | 79 | 316 | 158 | 2 | 158 | 1.000 | 16691 |
| sm2.108.pn | 78 | 95 | 434 | 95 | 1 | 95 | 1.000 | 3234 |
| sm2.110.pn | 13 | 16 | 72 | 32 | 2 | 28 | 0.875 | 25 |
| sm2.142.pn | 84 | 99 | 446 | 297 | 3 | 297 | 1.000 | 15494 |
| sm2.147.pn | 11 | 15 | 64 | 15 | 1 | 12 | 0.800 | 7 |
| sm2.175.pn | 89 | 99 | 450 | 99 | 1 | 99 | 1.000 | 3279 |
| sm2.180.pn | 81 | 99 | 446 | 198 | 2 | 198 | 1.000 | 12922 |
| sm2.19.pn | 61 | 70 | 316 | 70 | 1 | 68 | 0.971 | 1041 |
| sm2.26.pn | 74 | 85 | 378 | 85 | 1 | 85 | 1.000 | 1555 |
| sm2.36.pn | 75 | 94 | 424 | 282 | 3 | 282 | 1.000 | 19387 |
| sm2.55.pn | 59 | 69 | 316 | 138 | 2 | 132 | 0.957 | 2182 |
| sm2.59.pn | 54 | 69 | 306 | 69 | 1 | 66 | 0.957 | 824 |
| sm3.43.pn | 32 | 38 | 212 | 38 | 1 | 38 | 1.000 | 140 |
| sm3.43.pn | 32 | 38 | 212 | 38 | 1 | 38 | 1.000 | 140 |
| sm3.49.pn | 48 | 54 | 300 | 162 | 3 | 162 | 1.000 | 2566 |
| sm3.86.pn | 35 | 45 | 248 | 90 | 2 | 90 | 1.000 | 1324 |

Table 2. Other statistical data. The column "Success" shows the total number (and its ratio in parenthesis) of successful cases out of 600 (out of 127, respectively) test problems, each having state machines (general Petri nets) as underlying Petri nets, for each value of k, k=1,2,3. The column "Ave. ratio" denotes average ratios (=$|\bar{\delta}|/|X|$) over all 600 (127) test problems.

| | State machine (600data) | | General net (127data) | |
|---|---|---|---|---|
| k | Success | Ave. ratio | Success | Ave. ratio |
| 1 | 587 (97.8%) | 0.998 | 113 (89.0%) | 0.970 |
| 2 | 590 (98.3%) | 0.999 | 94 (74.0%) | 0.911 |
| 3 | 588 (98.0%) | 0.999 | 78 (61.4%) | 0.846 |