論 文
40~3~8

Safe Petri Net의 상태천이행렬식에 의한 비연속시스템의 점화순서 결정 알고리즘

An Algorithm for Determining Firing Sequence of Safe Petri Net Using its Matrix Equation

黃 祉 善*·李 載 晩**
(Chang-Sun Hwang · Jae-Man Lee)

Abstract- This paper deals with the determination of a firing sequence of transitions in the reachability problem of Safe Petri Net. The determination problem of a firing sequence is very important from the point of practical view, especially in the control of the discrete systems modelled by Safe Petri Net. The determination method of a firing sequence of transitions by means of the matrix equation for the discrete systems modelled by Safe Petri Net is proposed. First, a construction method of the incidence matrix and the firing rule for Safe Petri Net with self-loop are presented by defining the permissive arc in place of self-loop. Next, we develop a method that can find the enable transitions of Safe Petri Net by means of the matrix equation of Safe Petri Net. Finally, by using this method, we propose an algorithm that determines the firing sequence of transitions of Safe Petri Net.

1. 서 론

비연속시스템(시퀜스제어시스템)은 condition -event system이라고도 하며 조건(condition)과 사건(event)의 상호접속으로 동작하고 있는 비연속시스템이라 볼 수 있고 비동기성, 병렬성, 순서성 및 경합 등의 특성을 갖고 있다[1~7]. 이러한 시스템에 대한 해석 및 설계법은 종래 논리수학 및 순서회로론을 기초로한 방법에 불과하고 대부분의 경우 현장주도형의 경험과 know-how에 의한 해석 및 설계법으로서 한계성이 있고 컴퓨터를 중심으로한 해석 및 설계법의 이론화가 절실히 요망되고 있다[4]. 이러한 시스템을 모델링함에 있어서 (1) 시스템의 필요한 성질을 정확히 표현할 수 있고, (2) 개념레벨에서 상세레벨까지 정확히 표현할 수 있고, (3) 관계자사이의 정보교환에 유효하고, (4) 해석 및 설계에 유효한 수학적 표현이 가능한 표현법이 필요하다[4]. 이러한 조건을 만족하는 모델이 Petri Net이다. 최근에 와서 비연속 시스템을 모델링, 제어함에 있어서 Petri Net를 도입하여 그래프 이론적인 관점에서 해석 및 설계법의 이론화를 기하고 있지만 그 해석이론

*正 會 員 : 釜山大 工大 電氣工學科 敎授 · 工博
**正 會 員 : 釜山大 大學院 電氣工學科 博士課程
接受日字 : 1990年 7月 16日
1次修正 : 1990年 12月 13日
2次修正 : 1991年 2月 26日

은 아직 충분히 개발되어 있지않는 상태이다. 비연속시스템의 규모가 커지고 복잡할 경우에는 충돌과 deadlock이 일어날 가능성이 있다.

Petri Net에 의해서 표현된 비연속시스템의 해석문제중에서 활성(liveness)문제, 도달성(reachability)문제 등 많은 해석문제가 도달성문제에 귀착되므로[1~3] 실제적인 관점에서 도달성문제는 가장 중요한 해석문제이며, 특히 비연속시스템의 제어문제에 있어서 중요하다.

도달성문제를 해석하는 기본적인 수단으로서 도달성나무(reachability tree)에 의한 방법과 상태천이행렬식에 의한 방법이 있다[1~3]. 그러나 이러한 해석방법을 사용하여도 Petri Net의 도달성문제의 일반적 해법은 아직 밝혀지지 않고 있다[1~3]. 그래서 응용하는데 맞는 제약조건을 해석의 대상이 되는 Petri Net에 부과하여 문제를 정한 후에 해석하는 경우가 많다[4, 6, 7]. 도달성나무를 사용한 도달성문제의 해석시 유계(bounded)이면 구조에 한정을 가하지 않는 Petri Net에 대하여도 필요충분한 해를 구할 수 있지만 막대한 계산량이 필요하게 되고 도달성나무에 있어서 ω가 출현할 경우에는 일반적으로 어떤 점화순서가 가능한가를 규정하여 결정하는 문제(reachability problem)를 풀 수 없다[1, 2, 9, 10]. 상태천이행렬식을 사용한 도달성문제의 해석에 있어서는 상태천이행렬식의 비부정수해가 존재하지 않을 때에는 점화순서를 갖지않는 것은 알지만, 비부정수해가 얻어졌을 경우에도 일반적으로 점화순서가 존재하는가를 알 수가 없다. 점화순서가 존재한다고 하더라도 얻어진 비부정수해는 트랜지션의 점화회수에 대한 정보만을 알 수 있고 트랜지션의 점화순서에 관한 정보는 알 수가 없으므로[1, 2] 비연속시스템제어를 위한 트랜지션의 점화순서를 찾을 수 있는 방법을 개발하는 것이 대단히 중요하다.

본 논문에서는 비연속시스템이 비동기성, 병렬성, 순서성 및 경합 등의 성질을 갖는 condition-event system인 것에 착안하여 이러한 시스템의 동작모델인 Petri Net의 부분집합인 Safe Petri Net[1~4]를 도입하여 비연속시스템을 Safe Petri Net로서 표현하고 Safe Petri Net의 상태천이행렬식에 의한 비연속시스템의 도달성문제를 해석하고 트랜지션의 점화순서를 결정하는 문제에 대하여 논한다. 2장에서는 비연속시스템의 모델링도구로서 Safe Petri Net를 정의하고 점화법칙에 대하여 기술한다. 3장에서는 Safe Petri Net의 상태천이행렬식에 의한 비연속시스템의 도달성문제를 해석하고 점화가능트랜지션을 찾는 방법을 제안하고 이 제안된 방법에 의해서 Safe Petri Net의 트랜지션의 점화순서를 결정하는 알고리즘을 제안한다. 끝으로 4장에서는 본 논문에서 제안한 알고리즘을 간단한 Safe Petri Net에 적용하여 그 유용성을 보인다.

2. Safe Petri Net

2.1 Petri Net

Petri Net는 다음의 4개의 원소로 정의된다[1~3].

$$C=(P,\ T,\ I,\ O) \tag{1}$$

P : 플레이스(place)의 집합
T : 트랜지션(transition)의 집합
I : $P\rightarrow T$, 입력함수(input function)
O : $T\rightarrow P$, 출력함수(output function)

Petri Net의 각 플레이스에 토큰(token)을 적당히 배치함으로서 Marked Petri Net를 얻을 수 있으며 다음과 같이 정의한다.

$$G=(P,\ T,\ I,\ O,\ M) \tag{2}$$

식 (2)의 M은 마킹함수(marking function)이라 하여 토큰의 배치를 나타내는 벡터로서 다음과 같이 정의된다.

$$M=(m_1,\ m_2,\cdots,m_n)^T$$

여기서 n는 플레이스의 수이고 m_i는 플레이스 p_i안에 있는 토큰의 수 N(비부정수)이다.

2.2 Safe Petri Net

비연속시스템에서는 일반적으로 플레이스들은 모델링하려는 장비의 동작상태를 의미하므로 플레이스안에 복수개의 토큰이 존재하게 됨은 의미가 없다. 따라서 Petri Net에 safe라는 구속조건(플레이스안에는 토큰의 수를 최대 1개만을 허용)을 부

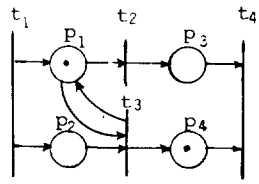


그림 1 Safe Petri Net
Fig. 1 Safe Petri Net

가하여 Safe Petri Net[1~4]라 하고 비연속시스템의 모델링도구로서 이용한다. 그러므로 마킹함수 M의 원소인 플레이스 p_i안에 있는 수인 비부정수 N는 다음과 같이 된다. 그리고 그림 1은 Safe Petri Net의 예를 보인다.

$$0 \leq N \leq 1 \tag{4}$$

2.3 Safe Petri Net의 점화법칙

2.3.1 자기루프(self-loop)를 갖지 않는 Safe Petri Net의 점화법칙

트랜지션 t_i가 점화가능(enable)하기 위한 조건은 다음과 같다[2~4](그림 2 참조).

1) 입력측 모든 플레이스에 토큰이 존재하여야 하고
2) 출력측 모든 플레이스에 토큰이 존재하지 않아야 한다.

트랜지션 t_i의 점화후의 마킹상태는 다음과 같이 변화된다.

1) 입력측 모든 플레이스의 토큰이 소멸되고
2) 출력측 모든 플레이스에 토큰이 생성된다.

자기루프가 존재할 경우에는 위의 점화법칙은 성립되지 아니한다. 자기루프가 존재할 때에는 그림 3과 같이 자기루프대신 플레이스에서 트렌지션에 아크의 화살표 대신 작은 흑점으로 표시하여 허가아크(permissive arc)라 부르고 자기루프의 등가넷트(equivalent net)로서 정의한다. 자기루프대신 허가아크를 도입함으로서 자기루프를 갖는 Safe Petri Net의 점화법칙은 다음과 같이 수정된다.

2.3.2 자기루프를 갖는 Safe Petri Net의 점화법칙

트랜지션 t_i가 점화가능(enable)하기 위한 조건은 다음과 같다(그림 4 참조).

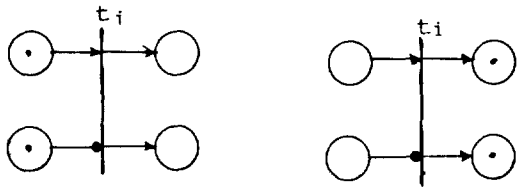
1) 허가아크를 포함한 입력측 모든 플레이스에 토큰이 존재하여야 하고
2) 출력측 모든 플레이스에 토큰이 존재하지 않아야 한다.

트랜지션 t_i의 점화후의 마킹상태는 다음과 같이 변화된다.

1) 입력측 모든 플레이스의 토큰은 소멸되지만 허가아크인 경우는 소멸되지 않고
2) 출력측 모든 플레이스에 토큰이 생성된다.

2.3.3. 경합의 조정

경합관계에 있는 트랜지션중에서 복수개가 동시

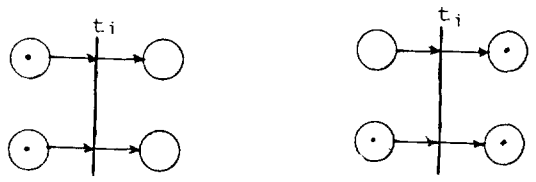


(a) 점화가능 (b) 점화후
(a) Enable (b) After firing

그림 2 Safe Petri Net의 점화법칙
Fig. 2 Firing rule of Safe Petri Net

(a) Safe Petri Net with self-loop
(a) 자기루프를 갖는 Safe Petri Net
(b) Equivalent net with permissive arc
(b) 허가아크에 의한 등가넷트

그림 3 자기루프의 등가넷트
Fig. 3 Equivalent net of self-loop



(a) 점화기능 (b) 점화후
(a) Enable (b) After firing

그림 4 허가이크를 갖는 Safe Petri Net의 점화법칙
Fig. 4 Firing rule of Safe Petri Net with permissive arc

(a) 경합 플레이스 (b) t_2보다 t_1우선
(a) Conflict place (b) t_1 prior to t_2

그림 5 경합의 조정
Fig. 5 Adjustment of conflict

에 점화할려고 할 때 임의의 하나만이 점화되며 나머지는 점화할 수 없게 된다[4, 8]. 경합관계에 있는 2개의 트랜지션의 점화순서를 정하기 위하여는 그림 5와 같이 더미 플레이스(dummy place)를 추가함으로서 우선관계를 결정할 수 있고 3개 이상인 경우에도 같은 방법으로 우선순위를 결정할 수 있다.

3. 점화순서를 결정하는 알고리즘

비연속시스템의 동작을 해석하기 위하여 Safe Petri Net의 상태천이행렬식에 의하여 도달성문제를 논하고 점화가능트랜지션을 찾는 방법과 이를 이용하여 트랜지션의 점화순서를 결정하는 알고리즘을 구하고자 한다.

시각 k에 있어서의 마킹상태를 표시하는 마킹벡터 $M(k)$를 다음과 같이 정의한다.

$$M(k)=[p_1(k)p_2(k)\cdots p_n(k)]^T \tag{5}$$

여기서 $p_j(k)$는 시각 k에서 플레이스 p_j의 마킹상태이다.

시각 k에 있어서의 점화벡터 $U(k)$를 다음과 같이 정의한다.

$$U(k)=[t_1(k)t_2(k)\cdots\cdots t_n(k)]^T \tag{6}$$

여기서 시각 k에서 트랜지션 t_i가 점화하면 $t_i(k)$는 1이고, 점화하지 않으면 0이다.

임의의 마킹상태에서 어떤 트랜지션이 점화한 후의 마킹상태를 방정식으로 나타낸 것이 다음과 같이 상태천이행렬식이다[1~3]. 즉

$$M(k+1)=M(k)+[B^+-B^-]\times U(k) \tag{7}$$

$M(k+1)$: 점화벡터 $U(k)$가 점화한 후의 마킹함수

$[B^+-B^-]$: 접속행렬로서 B^+는 트랜지션에서 플레이스에로, B^-는 플레이스에서 트랜지션에로의 접속행렬

단, 천이가 행하여지기 위한 제약으로서 다음 조건을 만족하는 것만이 적용가능하다[1~3].

$$M(k)\geq B^-\times U(k) \tag{8}$$

만일 트랜지션들이 계열로서 점화한 경우 점화회수벡터를 $f(\sigma)$라 하면 식 (7)은 식(9)와 같이 된다.

$$M(k+1)=M(k)+[B^+-B^-]\times f(\sigma) \tag{9}$$

여기서 $f(\sigma)=\sum_k U(K)$로서 비부정수를 원소로 갖는다.

주어진 초기 마킹상태 $M(I)$에서 최종 마킹상태 $M(F)$에 천이시키는 점화순서가 존재하기 위해서는 식 (9)에 의해

$$M(F)-M(I)=[B^+-B^-]\times f(\sigma) \tag{10}$$

에서 비부정수해 $f(\sigma)$가 존재하면 도달가능하다고 한다[1~3]. $M(F)-M(I)=0$, 즉 $[B^+-B^-]\times f(\sigma)=0$를 만족하는 양의정수해 $f(\sigma)$가 존재하면, 즉 $f(\sigma)$의 각 요소가 한번이상 점화한다면 live하다고 하고, 그렇지 않으면 deadlock이라고 한다.

점화순서가 존재한다고 하더라도 얻어진 비부정수해는 트랜지션의 점화회수에 대한 정보만을 알 수 있고 트랜지션의 점화순서에 대한 정보는 알 수가 없으므로 그것을 결정하는 문제가 남는다[1~3]. 다음에 비연속시스템제어를 위하여 트랜지션의 점화순서를 찾는 방법을 제안한다.

3.1 Safe Petri Net의 접속행렬

Safe Petri Net의 상태천이행렬식에 의한 비연속시스템을 해석하기 위하여 먼저 Safe Petri Net의 접속행렬을 구하여야 한다.

Safe Petri Net의 접속행렬 B는 트랜지션의 출력측의 관계를 표시하는 B^+접속행렬과 입력측의 관계를 표시하는 B^-접속행렬에 의하여 다음과 같이 구성된다[1~3, 5].

$$B=B^+-B^- \tag{11}$$
$$B^-(i,\ j)=\#(p_i,\ O(t_j))\leq 1$$
$$B^-(i,\ j)=\#(p_i,\ I(t_j))\leq 1$$

여기서 $\#(p_i,\ O(t_j))$는 트랜지션 t_j에서 출력측 플레이스 p_i로의 아크의 수이고 $\#(p_i,\ I(t_j))$는 입력플레이스 p_i에서 트랜지션 t_j로의 아크의 수이다.

자기루프 대신 허가아크를 갖는 Safe Petri Net인 경우에는 B^+접속행렬과 B^-접속행렬을 다음과 같이 구할 수 있다.

1) B^-행렬을 구성할 때 허가아크는 P로 표시하고
2) B^-행렬의 P의 위치에 해당하는 B^+행렬의 위치의 요소를 1로 바꾸고
3) B^-행렬의 P를 1로 바꾼다.

또 역으로 허가아크를 자기루프로 등가변환하여 구하여도 같은 결과를 얻는다.

3.2 자기루프 존재여부를 나타내는 행렬

Safe Petri Net안에 자기루프 존재여부를 나타내는 행렬을 SL행렬이라 정의하고 SL행렬은 B^+접속행렬과 B^-접속행렬 모두에 1인 원소만을 1로

하고 그 외는 0으로 하는 행렬로서 다음과 같이 정의된다.

$$SL(i,\ j)=\begin{cases}1 & \text{if}B^{+}(i,\ j)+B^{-}(i,\ j)=2\\ 0 & \text{if}B^{+}(i,\ j)+B^{-}(i,\ j)\neq 2\end{cases} \quad (12)$$

3.3 점화가능트랜지션 및 점화순서를 구하는 알고리즘

먼저 몇가지 정의를 정의한다.

[정의 1]

⊕연산을 다음과 같이 정의한다.

$$X=[x_1x_2\cdots\cdots x_n]$$
$$Y=[y_1y_2\cdots\cdots y_n]^T$$
$$X\oplus Y=(x_1\oplus y_1)\times(x_2\oplus y_2)\times\cdots\times(x_n\oplus y_n)$$
$$=\prod_{i=1}^{n}(x_i\oplus y_i) \quad (13)$$

단, $x_i\oplus y_i=\begin{cases}1 & x_i\leq y_i\\ 0 & x_i>y_i\end{cases}$

[정의 2]

Θ연산을 다음과 같이 정의한다.

$$X=[x_1x_2\cdots\cdots x_n]^T$$
$$Y=[y_1y_2\cdots\cdots y_n]^T$$
$$X\Theta Y=[x_1x_2\cdots x_n]^T\Theta[y_1y_2\cdots y_n]^T \quad (14)$$
$$=[x_1\Theta y_1x_2\Theta y_2\cdots x_n\Theta y_n]^T$$

단, $x_i\Theta y_i=\begin{cases}0 & x_i\leq y_i\\ 1 & x_i>y_i\end{cases}$

[정의 3]

함수 $L(X)$를 다음과 같이 정의한다.

$$X=[x_1x_2\cdots x_n]^T$$
$$L(x_i)=\begin{cases}1 & x_i\geq 1\\ 0 & x_i<0\end{cases} \quad (15)$$

다음에 몇가지 벡터함수를 정의한다.

시각 k에서 트랜지션의 출력측 플레이스의 마킹벡터 $MO(k)$를 다음과 같이 정의한다.

$$MO(k)=[M(O(t_1))M(O(t_2))\cdots M(O(t_n))]^T \quad (16)$$

여기서 $M(O(t_j))$는 트랜지션 t_j의 출력측에 있는 모든 플레이스의 토큰의 합이고 $MO(k)$는 다음 식 (17)으로 구할 수 있다.

$$MO(k)=(B^{+})^T\times M(k) \quad (17)$$

예로서 그림 1의 $MO(k)$를 식 (17)에 의하여 다음과 같이 구할 수 있다.

$$MO(k)=(B^{+})^T\times M(k)$$
$$=\begin{bmatrix}1&1&0&0\\0&0&1&0\\1&0&0&1\\0&0&0&0\end{bmatrix}\times\begin{bmatrix}1\\0\\0\\1\end{bmatrix}=\begin{bmatrix}1\\0\\2\\0\end{bmatrix}$$

시각 k에서 트랜지션의 출력측에 있는 자기루프를 갖는 마킹벡터 $S(k)$를 다음과 같이 정의한다.

$$S(k)=[SM(0(t_1)),\ SM(0(t_2)),\cdots,$$
$$SM(0(t_n))]^T \quad (18)$$

여기서 $SM(0(t_i))$는 트랜지션 t_i와 자기루프를 이루고 있는 모든 플레이스의 토큰의 합이고 $S(k)$는 다음 식 (19)으로 구할 수 있다.

$$S(k)=(SL)^T\times M(k) \quad (19)$$

시각 k에서 트랜지션의 출력측 마킹에서 자기루프의 마킹을 제외한 마킹벡터 $RM(k)$는 다음과 같이 정의한다.

$$RM(k)=MO(k)-S(k) \quad (20)$$

예로서 그림 1의 $S(k)$와 $RM(k)$를 식 (19)와 식 (20)에 의하여 다음과 같이 구할 수 있다.

$$S(k)=(SL)^T\times M(k)$$
$$=\begin{bmatrix}0&0&0&0\\0&0&0&0\\1&0&0&0\\0&0&0&0\end{bmatrix}\times\begin{bmatrix}1\\0\\0\\1\end{bmatrix}=\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$$
$$RM(k)=MO(k)-S(k)$$
$$\begin{bmatrix}1\\0\\2\\0\end{bmatrix}-\begin{bmatrix}0\\0\\1\\0\end{bmatrix}=\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

3.3.1 입력측만을 고려한 점화가능벡터

출력측 플레이스를 고려하지 않고 입력측의 플레이스만을 고려한 점화가능벡터 $\bar{U}(k)$는 트랜지션을 원소로 하는 벡터이며 각 트랜지션의 입력측의 모든 플레이스에 토큰이 있으면 1, 그렇지 않으면 0이다.

[정리 1]

$\bar{U}(k)$는 식 (21)로 구할 수 있다.

$$\bar{U}(k)=(B^{-})^T\oplus M(k) \quad (21)$$

(증명) : j번째 트랜지션 t_j가 점화가능하기 위해서는 t_j의 모든 입력측 플레이스에 토큰이 존재하여야 함으로

$$t_j=\prod_{i=1}^{n}(\#(p_i,\ I(t_j))\oplus p_i(k))=1$$

이 만족되어야 하며 모든 트랜지션에 대하여 벡터로 나타내면 식 (21)과 같다. Q.E.D.

예로서 그림 1의 $\bar{U}(k)$를 [정리 1]에 의하여 다음과 같이 구할 수 있다.

$$\bar{U}(k)=(B^-)^T\oplus M(k)$$

$$=\begin{bmatrix}0&0&0&0\\1&0&0&0\\1&1&0&0\\0&0&1&1\end{bmatrix}\oplus\begin{bmatrix}1\\0\\0\\1\end{bmatrix}=\begin{bmatrix}1\\1\\0\\0\end{bmatrix}$$

3.3.2 점화가능벡터

Safe Petri Net의 점화가능벡터를 $U^*(k)$로 정의한다.

[정리 2]

Safe Petri Net의 점화가능벡터 $U^*(k)$는 식 (22)로 구할 수 있다.

$$U^*(k)=\bar{U}(k)\Theta L(MO(k)) \tag{22}$$

(증명) : Safe Petri Net의 점화가능벡터는 입력측만 고려한 점화가능벡터에서 출력측에 토큰이 있는 트랜지션을 제외하면 된다. 즉,

$$U^*(k)=\bar{U}(k)\Theta L(MO(k))$$

와 같으며 점화가능벡터 $U^*(k)$에서 1의 값을 가지는 트랜지션이 점화가능트랜지션이다. Q.E.D.

예로서 그림 1의 $U^*(k)$를 [정리 2]에 의하여 다음과 같이 구할 수 있고 t_2가 점화가능함을 알 수 있다.

$$U^*(k)=\bar{U}(k)\Theta L(MO(k))$$

$$=\begin{bmatrix}1\\1\\0\\0\end{bmatrix}\Theta\begin{bmatrix}1\\0\\1\\0\end{bmatrix}=\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$$

이 $U^*(k)$를 식 (7)에 대입하면 다음 마킹상태를 구할 수 있다.

$$M(k+1)=M(k)+[B^+-B^-]\times U(k)$$

$$=\begin{bmatrix}1\\0\\0\\1\end{bmatrix}+\begin{bmatrix}1&-1&0&0\\1&0&-1&0\\0&1&0&-1\\0&0&1&-1\end{bmatrix}\times\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$$

$$=\begin{bmatrix}1\\0\\0\\1\end{bmatrix}+\begin{bmatrix}-1\\0\\1\\0\end{bmatrix}=\begin{bmatrix}0\\0\\1\\1\end{bmatrix}$$

3.3.3 점화순서를 결정하는 알고리즘 및 흐름선도

Safe Petri Net의 동작을 표시하는 트랜지션의 점화순서를 결정하는 알고리즘은 다음과 같고 흐름선도로 나타내면 그림 6과 같이 된다.

[단계 1] B^+행렬, B^-행렬을 구성하고 이로부터 SL행렬을 구한다.

[단계 2] 시각 k에서 $M(k)$의 초기마킹벡터 $M(1)$를 구한다.

[단계 3] $\bar{U}(k)$와 $U^*(k)$를 구한다(정리 1, 정리 2).

[단계 4] [단계 3]의 $U^*(k)$를 점화하고 점화한 벡터를 상태천이방정식의 $U(k)$에 대입하여 마킹벡터 $M(k+1)$를 구한다.

[단계 5] [단계 3]으로 돌아가 $M(k+1)$을 $M(k)$에 대치하여 원하는 최종마킹상태 $M(F)$가 될 때까지 되풀이 한다.

[단계 6] 점화한 트랜지션을 순서대로 나열하면 점화순서가 구해진다.

이상에서 제안한 Safe Petri Net의 상태천이행렬식에 의한 점화순서를 결정하는 알고리즘은 도

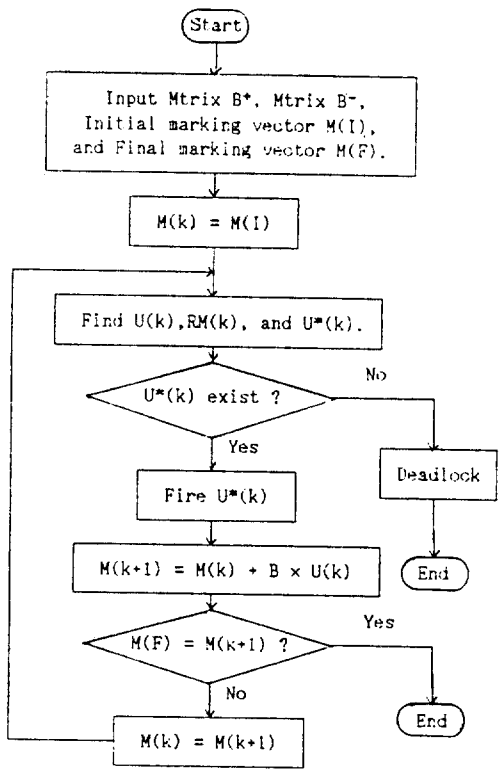


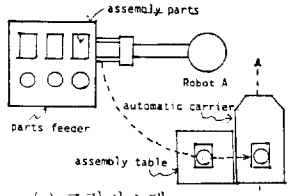
그림 6 흐름선도
Fig. 6 Flow chart

달성나무의 방법에 비하여 다음과 같은 점에서 개량되었다고 할 수 있다. 도달성나무의 방법에서는 Safe Petri Net의 구조가 크게되면 주어진 초기마킹에 대하여 모든 점화순서를 구하여야 하므로 복잡하게 되는 난점이 있다. 또 초기마킹에서 최종마킹으로 천이할 때 어떤 시각에 있어서 동시에 복수개의 점화가능트랜지션이 존재할 수 있다. 이 경우 도달성나무에 의한 방법에서는 최악의 경우 이러한 복수개의 동시점화가능트랜지션 전부의 조합에 대하여 부분나무를 작성하여야 한다[1, 2, 9, 10]. 그러나 본 논문에서 제안한 방법에서는 이러한 문제에 대하여 탐색하는 수를 대폭 감소시킬 수 있고 효율적으로 점화순서를 구할 수 있다. 따라서 도달성나무에 의한 방법에 비하여 계산량을 대폭 감소시킬 수 있으며 최다수행규칙에 의한 도달성나무의 생성시와 같은 결과를 얻을 수 있다.

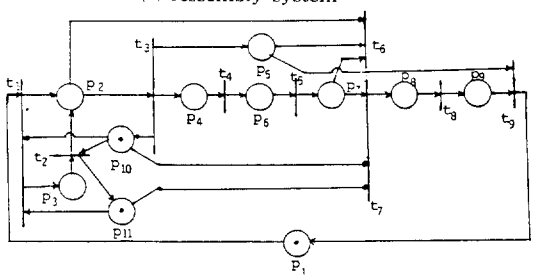
4. 적용예 : 로보트에 의한 조립시스템

적용예로서 이상의 점화순서 결정 알고리즘을 간단한 조립시스템에 적용해 본다. 그림 7(a)에서는 로보트 A가 부품공급기에서 두 종류의 조립부품을 2회에 걸쳐서 작업대에 집어 옮겨서 조립한 후 콘베이어에 옮기는 장치를 나타낸 것이다. 이 조립시스템을 Safe Petri Net로서 표현한 예가 그림 7(b)이다. 이와같이 비연속시스템을 Safe Petri Net로서 표현한 시스템에 대하여 그 시스템의 동작의 검증 혹은 해석이 필요하게 된다. 이 경우 점화순서를 확인함으로서 Safe Petri Net에 의해 모델링된 비연속시스템의 도달성문제, 즉 live상태와 deadlock상태를 해석할 수 있다.

그림 7의 Safe Petri Net에 대하여 점화법칙에 의하여 t_1이 점화하면 P_1, P_{10}, P_{11}의 마크가 소멸되고 P_2와 P_3에 동시에 마크가 들어가며 부품이 반입중임을 나타내고 이 부분Safe Petri Net는 마크를 외부로 하나씩만 빼낼 수 있다. 다음에 t_3의 점화로서 P_2의 마크가 소멸되고 P_4, P_5, P_{10}에 마크가 들어간다. t_2가 점화가능하게 되고 t_2가 점화하면 P_2, P_{11}에 마크가 들어가고 P_2의 마크가 소멸된다. 동시에 t_4가 점화하여 P_6에 마크가 들어가고 t_5가 점화하면 P_7에 마크가 들어감으로서 t_6가 점화하게 된다.(한 부품의 작업대 이동완료). 두번째 부품인 경우도 같은 방법으로 t_3의 점화에서 P_7까지 마크를 이동시키면 t_7이 점화가능하여 P_8에 마크가 들어가고 t_8가 점화하면 P_9에 마크가 들어가서 t_9이 점화하면 초기상태로 돌아가게 되고 이



(a) 조립시스템
(a) Assembly system



P_1, P_{10}, P_{11} : 제어동작
P_2 : 두 종류의 조립부품의 반입동작
P_3 : 두 종류의 조립부품의 반입동작
P_4 : 부품을 잡는 동작
P_5 : 로보트동작
P_6 : 부품들어올리는 동작
P_7 : 부품을 작업대위에 놓는 동작
P_8 : 작업대에서 조립동작
P_9 : 조립품을 콘베이어에 옮기는 동작
t_1 : 작업개시
t_2 : 반입대기종료
t_3 : 반입종료
t_4 : 잡는 동작종료
t_5 : 들어올리는 동작종료
t_6 : 작업대위에 놓는 동작종료
t_7 : 2번째부품을 작업대위에 놓는동작종료
t_8 : 작업대에서 조립동작 종료
t_9 : 조립품의 콘베이어에 옮기는 동작종료

(b) (a)의 Sate Petri Net모델
(b) Safe Petri Net model of

그림 7 조립시스템의 예
Fig. 7 An example of an assembly system

와같은 동작을 되풀이하게 된다.

이와같이 간단한 조립시스템에 대하여는 수작업으로 비연속시스템의 동작의 검증 혹은 해석을 할 수 있지만 규모가 크고 복잡한 비연속시스템에 대하여는 할 수 없다.

비연속시스템을 Safe Petri Net로서 표현한 시스템에 대하여 동작의 검증 혹은 해석을 하기 위하여 본 논문에서 개발한 상태천이행렬식을 이용한 알고리즘을 위의 조립시스템의 Safe Petri Net

```
Initial Marking
  P 1  P10  P11
Final Marking
  P 1  P10  P11

B- MARIX
Pl. Tr. |  1  2  3  4  5  6  7  8  9
--------------------------------------
  1     |  1  0  0  0  0  0  0  0  0
  2     |  0  0  1  0  0  1  0  0  0
  3     |  0  1  0  0  0  0  0  0  0
  4     |  0  0  0  1  0  0  0  0  0
  5     |  0  0  0  0  0  1  0  0  1
  6     |  0  0  0  0  1  0  0  0  0
  7     |  0  0  0  0  0  1  1  0  0
  8     |  0  0  0  0  0  0  0  1  0
  9     |  0  0  0  0  0  0  0  0  1
 10     |  1  1  0  0  0  0  1  0  0
 11     |  1  0  0  0  0  0  1  0  0

B+ MARIX
Pl. Tr. |  1  2  3  4  5  6  7  8  9
--------------------------------------
  1     |  0  0  0  0  0  0  0  0  1
  2     |  1  1  0  0  0  1  0  0  0
  3     |  1  0  0  0  0  0  0  0  0
  4     |  0  0  1  0  0  0  0  0  0
  5     |  0  0  1  0  0  0  0  0  0
  6     |  0  0  0  1  0  0  0  0  0
  7     |  0  0  0  0  1  0  0  0  0
  8     |  0  0  0  0  0  0  1  0  0
  9     |  0  0  0  0  0  0  0  1  0
 10     |  0  0  1  0  0  0  1  0  0
 11     |  0  1  0  0  0  0  1  0  0

RESULT OF SIMULATION

No. |  MARKING STATE      : FIRING TR.(SEQUENCE)
-----------------------------------------------
# 1 : P1 P10 P11                :  T1
# 2 : P2 P3                     :  T3
# 3 : P3 P4 P5 P10              :  T2  T4
# 4 : P2 P5 P6 P11              :  T5
# 5 : P2 P5 P7 P11              :  T6
# 6 : P2 P11                    :  T3
# 7 : P4 P5 P10 P11             :  T4
# 8 : P5 P6 P10 P11             :  T5
# 9 : P5 P7 P10 P11             :  T7
#10 : P5 P8 P10 P11             :  T8
#11 : P5 P9 P10 P11             :  T9
#12 : P1 P10 P11                :  T1

          LIVE !

f = [  1  1  2  2  2  1  1  1  1  ]
```

그림 8 시뮬레이션 결과
Fig. 8 Simulation result

에 대하여 적용하여 보았다. 즉 초기마킹, 최종마킹및 접속행렬을 컴퓨터에 입력하여 시뮬레이션한 결과 그림 8과 같이 만족한 결과를 확인 할 수 있었다.

5. 결 론

본 논문에서는 비동기성, 병렬성, 순서성 및 경합 등의 특성을 갖는 비연속시스템의 동작모델인 Petri Net의 부분집합인 Safe Petri Net를 도입하여 비연속시스템을 Safe Petri Net로서 표현하고 Safe Petri Net의 상태천이행렬식을 이용하여 비연속시스템제어를 위한 트랜지션의 점화순서를 결정하는 알고리즘을 제안하였다. 자기루프를 갖는 경우 Safe Petri Net의 점화법칙을 수정하여 이 알고리즘을 컴퓨터로서 실행하였다. 컴퓨터의 시뮬레이션 결과 점화 가능 트랜지션과 점화 순서를 정확히 알 수 있었으며 Safe Petri Net에 의해 모델링된 비연속시스템의 도달성문제, 즉 live상태와 deadlock상태를 해석할 수 있었다.

본 알고리즘은 주어진 Safe Petri Net의 상태천이방정식을 이용한 점화순서를 결정하는 문제의 한가지 해를 이 알고리즘에 의해서 얻었지만 비연속시스템의 설계법을 Safe Petri Net에 의해서 개발하여 이 알고리즘을 공식적(formal)으로 적용하는 문제, 개발된 시스템의 실시간제어문제, 비연속시스템의 이상을 검지하여 정상복귀(fault tolerace and recovery)까지의 Safe Petri Net에 의한 모델링 등에 대하여는 다음 기회에 미루기로 한다.

본 연구는 한국과학재단(891-0707-010-2)의 지원으로 수행되었으며 관계자 여러분께 감사드립니다.

참 고 문 헌

[1] J.L. Peterson : Petri Net Theory and the Modelling of Systems, Prentice-Hall, (1981).

[2] A. Ichikawa and S. Kobayashi : Representation and Control of Event-Driven System, Journal of SICE, 21-10, 929/938, (1982).

[3] Tadao Murata : Petri Nets : Properties, Analysis and Applications, Proc. IEEE, 77~4, 541~580(1989).

[4] K. Hasegawa, K. Takahashi, R. Masuda and H. Ohno : Proposal of Mark Flow Graph for Discrete System Control, Trans. of the Society of Instrument and Control Engineers, 20~2, 122/128(1984).

[5] Wolfgang Reising : Petri Nets, Springer Verlag, (1985).

[6] T. Murata, N. Komoda, K. Matsumoto and K. Haruna : A Petri Net Based Controller for Flexible and Maintainable Sequence Control and its Applications in Factory Automation,

IEEE Trans. of Industrial Electronics. IE -33-1, 1/8(1986).

[7] M. Kayama and H. Nagase : A High Speed Control Method for Automated Machine Systems Based on Token Transition of Petri Net Theory, T. IEE Japan, 109-D-7, 463/469(1989).

[8] K. Takahashi, K. Hasegawa and Zbigniew Banaszsk : A Synthesis Method for Petri Net with Prescribed Firing Sequence, Trans. of the Society of Intrument and Control Engineers, 21~3, (1985).

[9] E.W. Mayer : An Algorithm for the General Petri Net Reachability Problem. Proc. of the 13th Annual ACM Symposium on Theory of Computing(1981).

[10] A. Ichikawa and K. Hiraishi : Necessary and Sufficient Condition for a Class of Reachability in Petri Nets. Trans. of the Society of Instrument and Control Engineers, 20~8, (1984).

[11] C.S. Hwang and J.M. Lee : Analysis of Matrix Equation Based on Petri Net for Discrete System Control. Proc. of SICE '90, 693/696, International Session, Tokyo, Japan, July 1990.