추이적 행렬을 이용한 패트리 넷 모델의 분석방법에 대한 연구

A study on the analysis method of Petri Net Models Using the Transitive Matrix

송유진*, 이종근*
Yu Jin Song, Jong kun Lee

Abstract

We propose a divide-conquer method of Petri nets under the condition of one -boundedness for all the Petri nets. We introduce the P-invariant transitive matrix of Petri nets and relationship between them. The feature of the P-invariant transitive matrix is that each element stands for the transitive relationship between input place and output place through the firing of the enable transition.

* 창원대학교 정보시스템연구실/컴퓨터공학과

1. 서론

가시적이며 쉽게 이해할 수 있는 표현법을 가진 패트리 넷은 제어의 흐름을 보이기 위해 토큰을 사용하는 수학적 모델링 도구이다. 또한, 안전성(Safeness), 유한성(Boundedness), 보존성(Conservation), 생동성(Liveness), 도달가능성(Reachability), 가역성(Reversibility)등의 성질을 분석함으로써 하드웨어뿐만 아니라, 병행적이고 비동기적이며 비결정적인 혹은 확률적인 특징을 갖는 정보처리 시스템을 묘사하고 연구하는데 있어서 유용한 도구이다

패트리 넷 분석 방법에는 다음과 같이 세 부분으로 나누어 볼 수 있다[14]. 1) 도달성 트리 방법 [1][4][10][12], 2) 행렬식 방법 [6][8][18]과, 3) 축조나 합성 방법[2][3][9].

첫번째 방법은 모든 도달성 트리를 생성하는 계산 방법으로서 넷의 모든 부분에 적용할 수 있지만, 상태 폭발의 복잡성 때문에 작은 넷으로 제한되는 단점이 있다. 한편, 행렬식 방법이나 축조/합성방식은 강력한 분석 방법이지만 많은 경우에 있어서 패트리 넷의 특별한 상태에만 적용할 수 있다. 특히 축조/합성방식 같은 변형 기법은 모델을 변형하고 분석하는데 많은 노력과 시간이 필요하다. 또한 경우에 따라 적용되어지는 규약들을 제시하여야 하므로 일괄적으로 자동화하는데 약점이 있다. 행렬식 방법은 자동화하고 정량화 하는데 장점이 있는 반면에 모델이 크면 클수록 행렬이 거대하여지는 문제로 분석 이해도가 어려워지는 단점이 있다.

본 연구는 이러한 문제를 개선하기 위해 패트리 넷의 플레이스 기반 추이적 행렬을 이용한 서브넷과 그들 사이의 관계를 소개하고, 이를 이용해 각 플레이스에 마킹된 토큰의 숫자가 하나 이하인 패트리 넷 모델에서의 분할방법을 제안하는데, 여기서 플레이스 기반 추이적 행렬의 각 요소는 점화 가능한 트랜지션의 점화 과정을 통한 입력 플레이스와 출력 플레이스 사이의 추이적 관계를 나타낸다. 결론적으로 본 연구에서는 분석이 가능한 축조되어진 모델을 추이적 행렬을 이용하여 조성하는 것이다.

본 논문의 구성은 다음과 같다. 먼저 2장과 3장에서는 본 논문에서 필요한 정의와 기존 연구들에 대해 설명하고, 4장에서 각 플레이스에 마킹된 토큰의 숫자가 하나 이하인 패트리 넷 모델에서의 divide-conquer 방법을 제안하며, 5장의 예제를 통해 제안 방법을 검증하고, 6장에서 앞으로의 연구 방향과 함께 결론을 맺는다.

2. 패트리 넷

패트리 넷의 기본 정의와 표기법[7, 14]은 다음과 같다.

정의 2.1 : 패트리 넷 즉, PN은 5개의 항목으로 구성된다.

$PN = (P,T,I,O,M_0)$

1) P는 플레이스의 유한집합.
2) T는 트랜지션의 유한집합.
3) $I : P \times T \rightarrow N$는 입력함수.
4) $O : T \times P \rightarrow N$는 출력함수.
5) $M_0 : P \rightarrow N$ 는 초기 토큰 상태.
6) $P \cup T = \phi$

여기서, 플레이스는 시스템의 상태(state) 혹은 조건(condition)을 나타내며 그림에서 원으로 표시한다. 트랜지션은 시스템의 상태를 변화시키는 동작(event)을 나타내며 그림에서 선분으로 표시한다. 아크(arc)는 흐름을 나타내며 화살표로 표시하고, 토큰은 플레이스의 조건의 진위, 또는 시스템의 가용자원을 나타낸다. 토큰은 시스템의 동적이며, 병행적인 동작의 특성을 나타내기 위해 사용된다. 동작이 일어나도록 하는데 필요한 조건을 만족할 경우 플레이스에 토큰을 위치시킴으로써 표현한다. 트랜지션은 자신에게로 입력되는 모든 플레이스가 토큰을 보유하고 있어야 점화될 수 있다. 트랜지션이 점화되면 자신의 각 입력 플레이스로부터 토큰을 하나씩 제거하고 각 출력 플레이스에 토큰을 하나씩 첨가한다. 따라서 토큰의 수와 위치는 패트리 넷을 실행하는 동안 바뀌게 된다.

정의 2.2 : 패트리 넷 모델의 속성

1) 안전성(Safeness)

플레이스에서 토큰의 수가 1개를 초과하지 않으면, 패트리 넷의 플레이스는 안전하다. 트랜지션은 모든 출력 플레이스가 비어 있지 않으면 점화할 수 없으므로 패트리 넷은 안전하며, 또한 넷의 플레이스들이 안전하면 패트리 넷은 안전하다.

2) 유한성(Boundedness)

플레이스에서 토큰의 수가 k개를 초과하지 않으면 플레이스는 k-유한하다. 패트리 넷이 유한하거나 안전하면 어떤 점화 순서를 선택하여도 버퍼나 레지스터에서 오버플로우가 발생하지 않는다는 것이 증명된다.

3) 보존성(Conservation)

패트리 넷에서 항상 일정한 수의 토큰이 흐르면 보존하다고 한다.

4) 생동성(Liveness)

생동성의 개념은 운영체제에서 교착상태가 발생하지 않는다는 것과 밀접한 관계가 있다. 초기 마킹 M_0로부터 어떤 점화 순서에 따라 계속 트랜지션을 점화 하는 것이 가능하면 패트리 넷은 생동적이라고 본다. 생동적인 패트리 넷은 어떤 점화 순서를 선택하여도 교착상태가 발생하지 않는 연산을 수행한다.

5) 도달가능성(Reachability)

도달가능성은 시스템의 동적 특성을 연구하는 기본이다. M0로부터 Mn으로 전환하는 점화 순서가 존재한다면 마킹 M_n은 M_0로부터 도달 가능하다고 본다.

6) 가역성(Reversibility)

각각의 마킹 M에 대하여 M_0가 M으로부터 도달가능하면 가역적이라고 한다. 가역적인 패트리 넷은 항상 초기 마킹이나 상태로 돌아갈 수 있다.

3. 패트리 넷 모델의 분할에 관한 연구들

패트리 넷은 간단한 의미를 지니고 풍부한 분석 방법을 제공하므로 비교적 소규모 시스템 모델링 및 분석에 많이 이용되어 왔다. 하지만 시스템이 복잡해질수록 모델의 이해 및 분석이 어려운 단점이 있다. 따라서 패트리 넷의 분석력을 그대로 유지시키면서 간결성 및 편리성의 표현적 요소를 더욱 보강하여 쉽게 시스템을 작성하고 이해 할 수 있도록 하기 위해 많은 연구가 이루어져 왔다. 이들 중 도달성 분석시에 시스템 상태가 급증하는 상태 폭발(state explosion)의 관점에서 상태폭발을 줄이기 위한 연구들이 있는데 그 대표적인 방법이 패트리 넷 모델의 분할 방법들이다. 모듈화 패트리 넷[13][15]에서는 모델 작성시 병행적인 단위로 시스템을 분할하여 작성할 수 있으므로 분석시 합성적 분석방법[16]을 응용하여 상태 폭발의 가능성을 줄일 수 있다. 한편 합성적 분석[16]은 시스템 모델이 여러 부시스템으로 나뉘어 기술된 경우에 적용 가능하며 우선 각각의 부시스템에 대해 도달성 그래프를 생성하여 국부적인 특성에 대한 분석을 수행하고 각 도달성 그래프를 축약한 후 전체 시스템으로 합성하여 전역적인 특성에 대한 분석을 수행한다. Notomi[10]는 모듈화 패트리 넷의 하나인 Ada-net에서 계층적인 도달성 그래프 생성에 관한 연구를 수행하였다. 그러나 이 연구는 모델 작성시에 작성자의 임의의 기준에 의해 시스템이 분할되므로 다양한 모델링 편의와 작성이 어렵다는 단점이 있다. 또 다른 방법으로는 패트리 넷 슬라이스 방법[17]을 들 수 있다. 슬라이스 기법은 모듈화 패트리 넷 방법과 유사한 결과를 얻거나 경우에 따라서 좀 더 나은 결과를 얻을 수 있지만 패트리 넷 모델의 상태를 단지 병행적인 관계를 없애는 관점에서만 분할을 시도했다.

본 연구에서는 지금까지의 연구들이 병행적인 관점에서만 이루어진 데 반해 일반적인 모델의 성질에 충실하여 패트리 넷 모델을 분할하고자 한다.

먼저, 분할 알고리즘에 의해 분할되기 전의 원래 모델과 분할된 모델이 동일한 행동임을 보이기 위해 행동일치 개념을 정의한다. 행동일치는 내부구조는 무시하고 외부적으로 나타나는 행동만을 고려한다.

정의 3.1[11] : 행동일치

두 패트리 넷 모델 P와 Q의 도달성 그래프들 간의 일대일 매핑이 가능할 때 P와 Q는 행동일치라 한다. 두 패트리 넷 모델이 동일한 행동을 수행함을 보이기 위해서는 우선 두 모델의 트랜지션간의 매핑 관계를 정의하고 매핑되는 트랜지션들의 수행 전후 조건이 동일함을 보여야 한다. 분할 알고리즘은 새로운 트랜지션을 생성하지 않고 기존의 트랜지션은 여러 서브 넷에 나타날 수 있지만 동일한 레벨을 부여하여 하나로 인식됨으로 두 모델의 트랜지션들은 일대일로 매핑된다. 그래서 분할 알고리즘은 플레이스의 입출력 정보를 유지시킴으로써 여러 서브 넷이 나타난 동일한 플레이스들은 원래 모델의 플레이스와 동일한 정보를 유지함을 증명함으로써 원래의 넷과 분할된 서브넷이 동일함을 보일 수 있다.

4. 추이적 행렬 분할 방법

4.1 플레이스 기반 추이적 행렬

정의 4.1 : 입력함수와 출력함수에 대한 행렬

PN구조의 행렬 정의 C는 다음과 같다.

$$C = (P, T, B^-, B^+)$$

$$B^-[i,j] = \#(p_i, I(t_j)) \quad (1)$$

$$B^+[i,j] = \#(p_i, O(t_j)) \quad (2)$$

여기서 B^-와 B^+ $B = B^+ - B^-$는 incidence 행렬이라 한다. 또한 $B = B^+ - B^-$는 각각 입력과 출력함수에 대한 행렬이다.

정의 4.2 : 표식화 플레이스 기반 추이적 행렬 (Labeled Place Transitive Matrix) L_{BP}.

L_{BP}는 표식화 플레이스 기반 추이적 행렬이라고 하며 다음과 같이 정의한다 [8].

$$L_{BP} = B^- diag(t_1, t_2, ..., t_n)(B^+)^T \quad (3)$$

여기서, $t_i (i = 1,2,...,n)$ 다음과 같다.

$|t_i| =$ 1이면 t_i는 fire한다. (4)

0이면 t_i는 fire 하지 않는다.

L_{BP} 요소는 하나 혹은 그 이상의 트랜지션들을 통해 한 플레이스로부터 다른 플레이스로의 이동을 나타낸다. 여기서 패트리 넷의 기본적인 구성 요소와 표식화 플레이스 기반 추이적 행렬과의 기본적인 관계를 알 수 있다.

Table 1. Relationship between PNs and labeled place transitive matrices

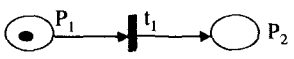
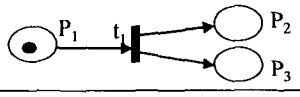
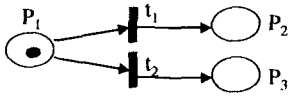
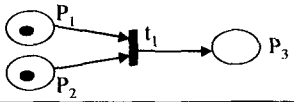
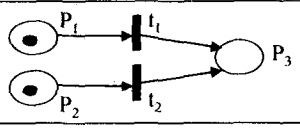
(a)		$L_{BP} = \begin{pmatrix} 0 & t_1 \\ 0 & 0 \end{pmatrix}$
(b)		$L_{BP} = \begin{pmatrix} 0 & t_1 & t_1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
(c)		$L_{BP} = \begin{pmatrix} 0 & t_1 & t_2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
(d)		$L_{BP} = \begin{pmatrix} 0 & 0 & t_1 \\ 0 & 0 & t_1 \\ 0 & 0 & 0 \end{pmatrix}$
(e)		$L_{BP} = \begin{pmatrix} 0 & 0 & t_1 \\ 0 & 0 & t_2 \\ 0 & 0 & 0 \end{pmatrix}$

Table 1은 패트리 넷과, 그들과 연관된 표식화 플레이스 추이적 행렬 L_{BP}의 예들을 보여준다. 이는 패트리 넷의 트랜지션 점화 법칙으로부터 얻어질 수 있다.

정의 4.3 : 가중적 플레이스 기반 추이적 행렬 (Weighted Place Transitive Matrix) L_{BP}^{*}

L_{BP}^{*}는 m ×m 가중적 플레이스 기반 추이적 행렬이라고 정의한다. 만약 트랜지션 t_k가 L_{BP}의 같은 행에 s번 나타난다면 L_{BP}에 있는 t_k를 L_{BP}^{*}에서는 t_k/s로 표시한다. 즉 위와 같이 하면 Table 1 (d)의 L_{BP}^{*}는 다음과 같이 얻을 수 있다.

$$L_{BP}^{*}=\begin{bmatrix}0 & 0 & t_1/2\\ 0 & 0 & t_1/2\\ 0 & 0 & 0\end{bmatrix}$$

정의 4.4 : 방정식 정의.

L_{BP}^{*}를 적용하여 만든 플레이스와 트랜지션의 상관관계 테이블에서, 만약 각각의 행이나 열에 있는 같은 트랜지션들의 합이 1과 같으면 점화 조건이 성립된다. 그러므로 다음과 같이 정의된다.

$$\sum\frac{t_k}{d}=1 \qquad (5)$$

여기에서, t_k는 트랜지션, d는 L_{BP}에 있는 동일한 트랜지션 t_k의 개수를 나타내는 분모이다.

4.2 서브 넷 알고리즘

모델의 분할 과정을 통한 서브 넷의 생성은 다음과 같이 이루어진다.

1) 플레이스 기반 추이적 행렬에서 L_{BP}^{*}를 정의한다.

2) L_{BP}^{*}를 이용한 플레이스와 트랜지션의 상관관계 테이블을 정의한다.

3) 토큰을 가$P_i^{\bullet}$ 의 형태로 하여 작성한다.지고 있는 플레이스를

4) 테이블에서 $P_i^{\bullet}$의 형태인 것만 서브 넷의 시작 플레이스로 고려한다.

5) 해당 행의 플레이스를 각각의 서브 넷 구성원으로 만들며, 이 때 그 행의 플레이스와 같은 플레이스를 테이블의 열에서 찾아 계속 연결시킨다. 마찬가지로 해당 열의 플레이스를 각각의 서브 넷 구성원으로 만들며, 이 때 그 열의 플레이스와 같은 플레이스를 테이블의 행에서 찾아 계속 연결시킨다.

6) 위5번의 과정을 해당 열과 행의 플레이스에서 트랜지션의 점화 조건에 해당하는 (5)의 수식을 만족할 때까지 계속한다.

7) 단, 플레이스 기반 추이적 행렬의 경우가 table 1(a)의 경우일 때는, 플레이스가 트랜지션 점화 이후의 과정에도 계속 단독으로 영향을 미치므로 table 1(a)의 경우가 아닐 때까지 계속 위의 5), 6)번 과정을 수행한다.

8) 또한 플레이스 기반 추이적 행렬의 경우가 table 1(b)나 table1(c)의 경우일 때에도 마찬가지로 플레이스가 트랜지션 점화 이후의 과정에 계속 단독으로 영향을 미치므로 table 1(b)나 table1(c)의 경우가 아닐 때까지 계속 위의 5), 6)번 과정을 수행한다.

9) 위의 과정을 통해 나온 각각의 서브 넷 중에서 어떤 서브 넷이 또 다른 서브 넷에 포함된다면 포함되는 서브 넷은 제거해도 무방하다. 이는 포함하고 있는 서브 넷의 분석으로 포함된 서브 넷의 분석이 가능하기 때문이다.

위와 같은 과정을 알고리즘으로 표현하면 다음과 같다.

```
char Psubnet[ ], Tsubnet[ ]; /* array of the set
of place and the set of transition */
void search() {
  for (during exist the transition in columns
  table of the place) {
    read transition;
    while (during exist the transition in the
    same room) {
```

```
      put the transition in Tsubnet[ ];
      if (there is nothing columns place of the
      transition in Psubnet[ ]) {
        put the columns place of the
        transition in Psubnet[ ];
      } /* if */
      read the next transition of the same
      room;
    } /* while */
    to move next room;
  } /* for */
  for (during exist the place in Psubnet[ ]) {
    read columns transition of the place in
    Psubnet[ ];
    put the transition in Tsubnet[ ];
  } /* for */
} /* search() */
main()
{
  read values of table and rows place;
  for (during exist place in row of table) {
    read the rows place;
    if (the rows place is initial marking place) {
      put the place in Psubnet[ ];
      search();
    } /* if */
    if (∑ t_k / d ≠ 1) {    */  t_k is k transition in
    Tsubnet[ ]     /*
    read columns place of the transition(t_k);
    search();
    } /* if */
    printf (Psubnet[ ], Tsubnet[ ]);
  } /* for */
} /* main */
```

5. 예제 패트리 넷 모델

이 장에서는, 위에서 제안된 방법을 다음과 같은 통신모델에 적용하여 분석하고자 한다.

<Fig 1> 패트리 넷 모델의 플레이스와 트랜지션 각각에 해당하는 기능을 <Table 2>에 제시하였다.

<Fig 1>은 통신 모델에서 기본적인 호출 프로세스와 전송 사이의 상호작용에 대한 패트리 넷 모델이다. 여기에서, o-, t-, cf- 는 각각 생성(originating), 종료(terminating), 전송(forwarding)을 표시한다. P2는 처음 생성되는 호출프로세스의 시작을 의미하는 토큰을 가지고 있고, P3은 호출하는 과정의 프로세스를 진행하는 토큰을 가지고 있으며, P4는 호출에 대한 응답 과정의 프로세스를 진행하기 위한 토큰을 가지고 있다. 또한 P15는 라우팅이나 전송과정에서의 에러 유무를 표시하는 토큰을 가지고 이 프로세스를 수행한다. 이 모델은 이해하기가 매우 어려울 뿐만 아니라, 변화에 민감하다. 또한 선택적인 모의 실험을 수행하기가 어렵다. 따라서 이 모델을 분석함으로써 본 연구에서 제안하고자 하는 분할 분석방법의 효율성을 제시하고자 한다.

Table 2. Significance table of place and transition

Place	Significance	transition	Significance
P1	Busy	t1	o-abandon
P2	Not-busy	t2	o-end-talk
P3	C-idle	t3	on-hook
P4	R-idle	t4	start-call
P5	Talking	t5	t-busy-tone
P6	Busy-tone	t6	t-routing
P7	Hang-up	t7	t-end-talk
P8	Arrival	t8	o-activate
P9	Talk	t9	Dialing
P10	Ringing-tone	t10	t-abandon
P11	Number	t11	t-ringing
P12	Ringing	t12	t-activate
P13	Routing	t13	o-busy-tone
P14	CF	t14	cf-routing
P15	not-CF	t15	o-ringing
		t16	o-routing
		t17	Unsubscribe-cf
		t18	Subscribe-cf

<Fig1>의 패트리 넷모델을 먼저 플레이스 기반 추이적 행렬을 이용한 L_{BP}^{*}를 적용하여 정의한다.

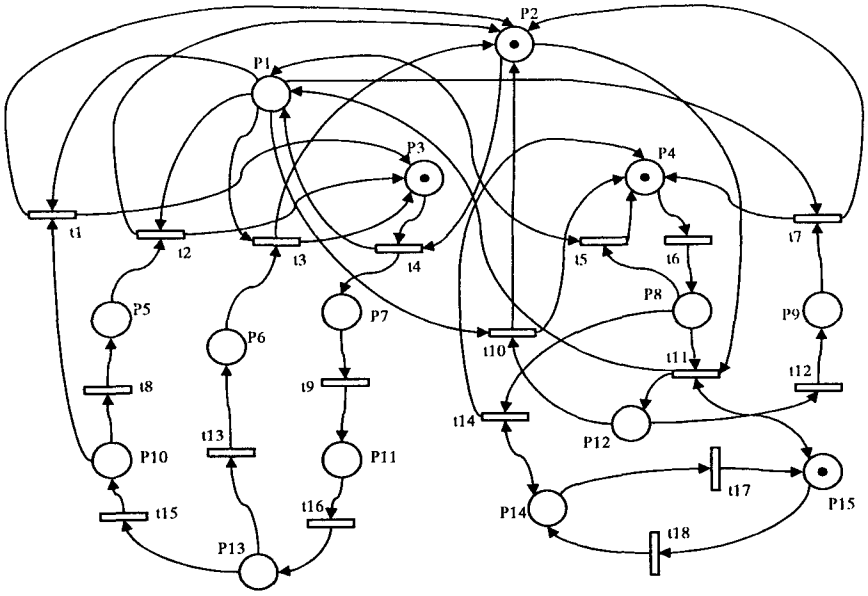


Fig 1. P/T net for basic call processing and call forwarding.

그리고 3장에서 제시한 모델의 분할 과정 1)~3)과 같은 방법으로 <Table 3>를 작성한다. <Table 3>을 이용하여 서브 넷 알고리즘을 적용하면 다음과 같은 분석으로 총 3개의 서브 넷을 구할 수 있다.

먼저, P2를 서브 넷의 시작 플레이스로 한다. P2에서 (5)번의 수식을 만족할 때까지 해당 플레이스와 트랜지션을 찾아 서브 넷으로 연결시키면 결국 P1-P2-P5-P6-P7- P9-P10-P11-P12-P13, t1-t2-t3-t4-t5-t7-t8-t9-t10-t11-t12-t13-t15-t16이 된다. 마찬가지로, P3, P4, P15를 시작 플레이스로 하여 (5)번 수식이 만족할 때까지 연결시키면 된다. 단, P4의 경우, t6에 단독으로 영향을 미치는 table 1(a)의 경우와 같으므로 t6의 영향을 받는 P8로 연결시켜 서브 넷을 생성한다. 이와 같은 과정을 거치면 <Fig 2>, <Fig 3>, <Fig4>, <Fig5>와 같이 4개의 서브 넷을 생성하게 된다.

$$I_{BP}^{*} = \begin{bmatrix} t_5/2 & \begin{matrix} t_1/2, \\ t_2/2, \\ t_3/2, \\ t_7/2, \\ t_{10}/2 \end{matrix} & \begin{matrix} t_1/2, \\ t_2/2, \\ t_3/2, \\ t_7/2, \\ t_3/2 \end{matrix} & \begin{matrix} t_5/2, \\ t_2/2, \\ t_7/2, \\ t_{10}/2 \end{matrix} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ t_4/2, t_{11}/2 & 0 & 0 & 0 & 0 & 0 & t_4/2 & 0 & 0 & 0 & 0 & t_{11}/2 & 0 & 0 & 0 \\ t_4/2 & 0 & 0 & 0 & 0 & 0 & t_4/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & t_2/2 & t_2/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & t_3/2 & t_3/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_9 & 0 & 0 & 0 & 0 \\ t_5/2, t_{11}/2 & 0 & 0 & \begin{matrix} t_5/2, \\ t_{14}/2 \end{matrix} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{11}/2 & 0 & t_{14}/2 & 0 \\ 0 & t_7/2 & 0 & t_7/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & t_1/2 & t_1/2 & 0 & t_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{16} & 0 & 0 \\ 0 & t_{10}/2 & 0 & t_{10}/2 & 0 & 0 & 0 & 0 & t_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & t_{13} & 0 & 0 & 0 & t_{15} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & t_{14}/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{14}/2 & t_{17} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{18} & 0 \end{bmatrix}$$

그러나 <Fig5>는 <Fig4>에 포함되므로 제거해도 무방하다. 결국, 본 논문에서 예시된 패트리 넷 모델은 3개의 서브 넷으로 나누어진다.

이렇게 나누어진 3개의 서브 넷<Fig1> <Fig2> <Fig 3>은 정의 3.1에서의 행동일치 정의, 즉 분할 알고리즘은 새로운 트랜지션을 생성하지 않고 기존의 트랜지션은 여러 서브 넷에 나타날 수 있지만 동일한 레벨을 부여하여 하나로 인식됨으로 두 모델의 트랜지션들은 일대일로 매핑된다. 그래서 분할 알고리즘은 "플레이스의 입출력 정보를 유지시킴으로써 여러 서브 넷이 나

Table 3. Correlation table of Place and Transition of <Fig 1>

	P_1	P_2•	P_3•	P_4•	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}•
P_1	$t_5/2$	$t_1/2$ $t_2/2$ $t_3/2$ $t_7/2$ $t_{10}/2$	$t_1/2$ $t_2/2$ $t_3/2$	$t_5/2$ $t_7/2$ $t_{10}/2$											
P_2•	$t_4/2$ $t_{11}/2$						$t_4/2$					$t_{11}/2$			
P_3•	$t_4/2$						$t_4/2$								
P_4•								t_6							
P_5		$t_2/2$	$t_2/2$												
P_6		$t_3/2$	$t_3/2$												
P_7											t_9				
P_8	$t_5/2$ $t_{11}/2$			$t_5/2$ $t_{14}/2$								$t_{11}/2$		$t_{14}/2$	
P_9		$t_7/2$		$t_7/2$											
P_{10}		$t_1/2$	$t_1/2$		t_8										
P_{11}													t_{16}		
P_{12}		$t_{10}/2$		$t_{10}/2$					t_{12}						
P_{13}						t_{13}				t_{15}					
P_{14}				$t_{14}/2$										$t_{14}/2$	t_{17}
P_{15}•														t_{18}	

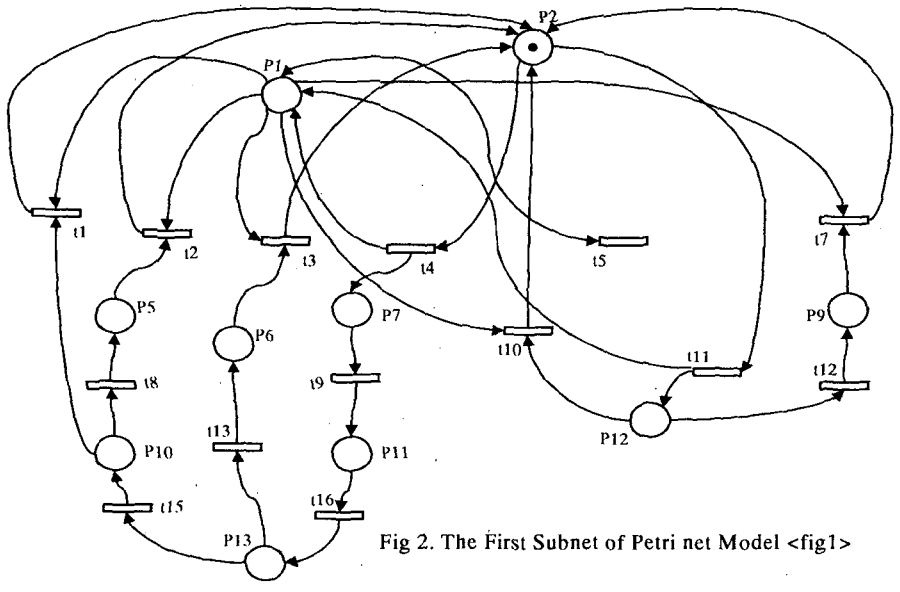


Fig 2. The First Subnet of Petri net Model <fig1>

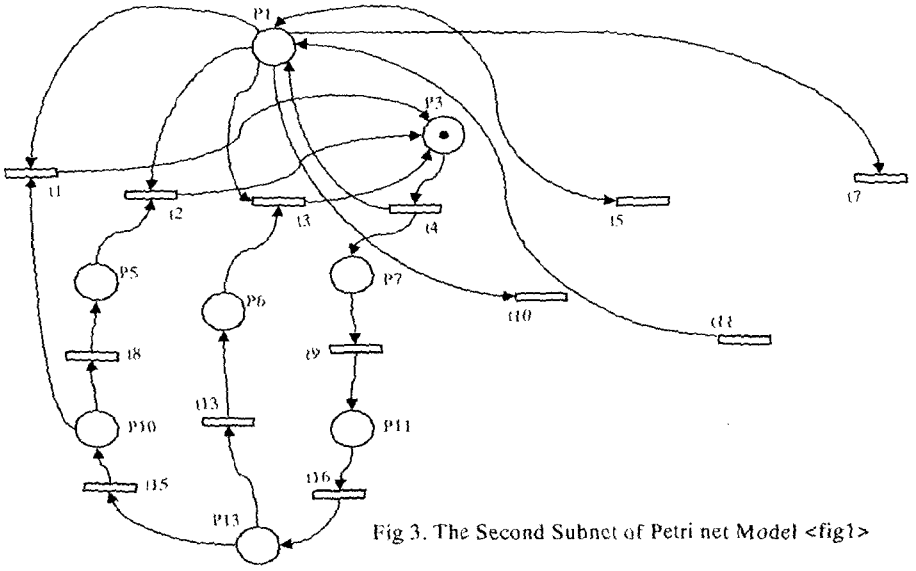


Fig 3. The Second Subnet of Petri net Model <fig1>

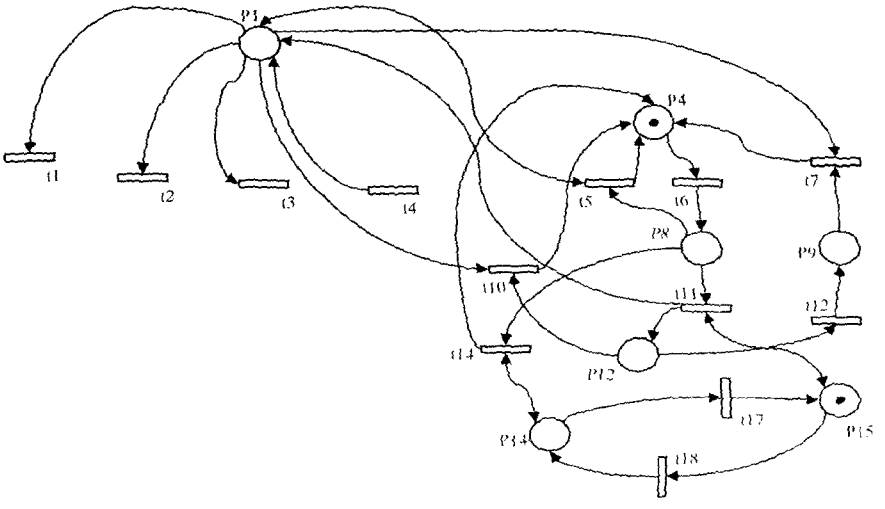


Fig 4. The Third Subnet of Petri net Model <fig1>

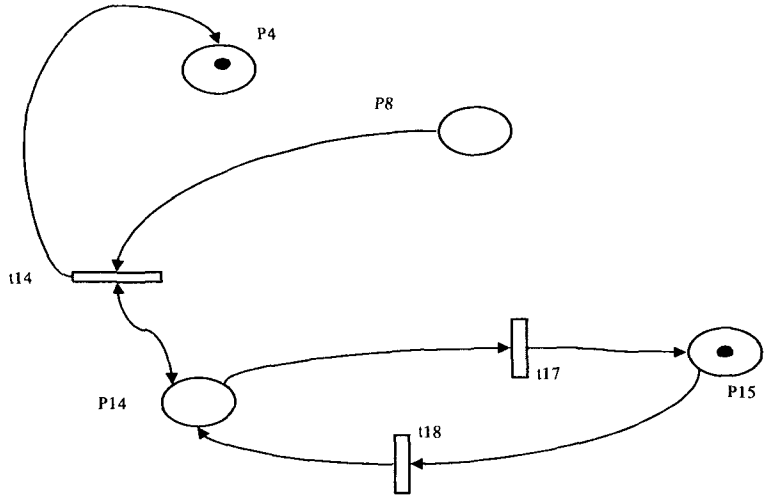


Fig 5. The Fourth Subnet of Petri net Model <fig1>

타난 동일한 플레이스들은 원래 모델의 플레이스와 동일한 정보를 유지함을 증명함으로써 원래의 넷과 분할된 서브넷이 동일함을 보일 수 있다" 는 정의에 의해 원래의 넷<Fig 1>과 동일함이 증명된다.

또한, 각각의 서브 넷은 다음과 같은 행동들로 구분되어 보여질 수 있다.

전체 모델	Fig 1	통신 모델의 기본적인 호출 프로세스와 전송 프로세스
서브 넷	Fig 2 (서브 넷 1)	다른 곳과의 연결이 없을 경우 처음 시작되는 호출 생성 프로세스
	Fig 3 (서브 넷 2)	상대방과의 연결을 시도하고자 하는 호출 프로세스
	Fig 4 (서브 넷 3)	자신에게 요구되는 호출에 대한 응답 프로세스

따라서, 이해하기 어려운 원래의 모델을 각각의 토큰이 미치는 일련의 프로세스로 분할함으로써, 각 서브 넷의 기능이 도출됨과 동시에 전체 모델의 수행 동작을 이해하기가 용이하게 된다.

한편, 각 서브 넷의 입력 트랜지션은 외부의 영향을 받는 것을 의미하기 때문에 전체 넷의 성질을 정확히 분석하기 위해서는 서브 넷 각각의 입력 트랜지션이 고려되어야 한다. 따라서 입력 트랜지션에서 각각 1개의 토큰이 입력되는 것을 고려한다. 이는 본 연구가 각 플레이스의 초기 마킹의 수를 1보다 작거나 같은 넷의 분석으로 한정하기 때문이다. 이 때 서브 넷의 출력 트랜지션으로, 입력된 토큰의 개수보다 작거나 같은 만큼 출력되었을 때 그 서브 넷이 생동성, 유한성, 가역성 등의 성질을 만족한다면, 넷이 생동성, 유한성, 가역성 등의 성질을 만족한다고 말할 수 있다. 여기서 입력된 토큰의 개수보다 작거나 같은 만큼 출력된다는 정의를 내리는 이유는 외부의 영향을 받은 조건들이 그 서브 넷 자체의 프로시져들에 영향을 줄 수도 있고, 혹은 외부 서브 넷에 영향을 줄 수도 있으므로 작거나 같은 만큼의 조건이 성립되어야 하는 것이다.

위와 같은 관점에서 각 서브 넷의 성질을 보면 세 개의 서브 넷 모두 각각 생동성, 유한성, 가역성 등의 성질을 만족함을 알 수 있다.

따라서 본 논문에서 제시한 추이적 행렬 분할 방법을 이용한 패트리 넷 모델 분할방법, 즉 패트리 넷 전체의 넷 모델을 분석하는 것보다 넷

모델을 나누어 분석하는 것이 모델을 이해하는데 용이하고, 훨씬 효율적임을 알 수 있었다.

6. 결론 및 앞으로의 연구 방향

우리는 병렬성, 비동기성, 분산, 동시성, 비결정성 등과 같은 성질을 가진 시스템들의 패트리 넷 모델을 분석하고자 할 때, 너무 규모가 크거나 복잡할 경우 분석 자체가 힘들어질 경우가 있다. 이러한 점을 해결하기 위해 큰 규모의 패트리 넷 모델을 각각 관련 있는 서브 넷으로 나누어 분석할 경우 좀 더 효율적으로 분석할 수 있다. 이를 위해 본 논문에서는 플레이스 기반 추이적 행렬을 이용하여 큰 규모의 패트리 넷을 자체의 성질을 변화시키지 않는 범위 내에서 서브 넷으로 분할하는 방법을 제안하였다. 그러나 이 방법은 각 플레이스에 마킹된 토큰의 숫자가 하나 이하인 패트리 넷 모델에서의 분할 방법이라는 제한점이 있다. 따라서 앞으로의 연구방향으로는 이러한 각 플레이스에 마킹된 토큰의 수에 제한 받지 않는 다양한 형태의 패트리 넷 모델에서도 적용 가능한 분할 방법을 개발하고자 한다. 또한 제안된 방법을 직접 적용할 수 있는 도구의 개발도 함께 병행해야 할 것이다.

참고문헌

[1] A. Valmari, A stubborn attack on state explosion, Computer-Aided Verification90, in R. P. Krushan and E. M. Clarke, Eds., AMS DIMACS Series in Discrete Math and Theoretical Computer Science, Vol. 3, pp 25-41, 1990.

[2] A. Valmary, Compositional State Space Generation, Advances in Petri Nets 93, 1993.

[3] B. Saha and S. Bandyopadhyay, Computer-based reduction technique for Petri Nets, Internation. J. SCI., Vol. 22, No. 1, pp. 49-59, 1991.

[4] E. T. Morgan and R. R. Razouk, Interactive state-space analysis of concurrent systems, IEEE Trans. Software Eng., Vol. 13, No. 10, pp.1080-1091, Oct. 1987.

[5] G.J.Holzmann, Design and Validation of Computer Protocols, Englewood Cliffs, NJ: Prentice Hall, 1991.

[6] I. Miyazawa, H. Tanaka and T. Sekiguchi, Classification of solutions of matrix equation related to parallel structure of a Petri Net, in Proceedings of ETFA96, pp. 446-452, 1996.

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

[8] J. Liu, Y. Itoh, I. Miyazawa, T. Sekiguchi, A Research on Petri Net Properties using Transitive Matrix, IEEE International Conference on Systems, Man, and Cybernetics, 1999.

[9] K. H. Lee and J. Favrel, Hierarchical Reduction Method for Analysis and Decomposition of Petri Nets, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 2, 1985.

[10] I. Miyazawa, H. Tanaka and T. Sekiguchi, Classification of solutions of matrix equation related to parallel structure of a Petri Net, in Proceedings of ETFA96, pp. 446-452, 1996.

[11] R. Miler, Communication and Concurrency, Prentice Hall, 1989.

[12] R. Janicki and M. Kounty, Optimal simulations, nets and reachability graphs, MacMaster Univ., Hamilton, ON, Canada, Tech. Rep. 1991.

[13] S. Christensen and L. Petrucci, Modular State Space Analysis of Colored Petri

Nets, Application and Theory of Petri Nets95(LNCS #935), 1995.

[14] T. Murata, Petri Nets: Properties, analysis and applications, Proc. IEEE, vol. 77, pp. 541-580, 1989.

[15] W. Damm, G. Dohmen, V. Gerstner and B. Josko, Modular Verification of Petri Nets, LNCS #430, 1989.

[16] W. J. Yeh, Controlling State Explosion in Reachability Analysis, PhD. Thesis, Purdue University, Dec. 1993.

[17] W. J. Lee, S. D. Cha, Y. R. Kwon, Integration and Analysis of Use Cases Using Modular Petri Nets in Requirements Engineering, IEEE Transaction on Software Engineering, Vol. 24, No. 12, 1998.

[18] Y. Cai, T. Sekiguchi, H. Tanaka, M. Hikichi and Y. Maruyama, A method for analyzing Petri Net structure and adding counter-place to its incidence matrix, The Transactions of the SICE of Japan, Vol.29, No. 12, pp. 1458-1464, 1993.

저자소개

이종근(E-Mail : kjlee@sarim.changwon.ac.kr)

1974 숭실대학교 전산과 졸업, 동 대학원 수료

1977 고려대학교 경영대학원 경영학과 수료

1990 Univ. de Montpellier II 전산과 박사 수료

1983 ~ 현재 창원대학교 컴퓨터공학과 교수, 창원대학교 전산소장, 자연대부학장 역임

관심분야: 패트리 넷, 성능분석, 정보보호, 스케쥴링 연구

송유진(E-Mail : syj@sarim.changwon.ac.kr)

1992 창원대학교 컴퓨터공학과 졸업(학사)

1995 창원대학교 대학원 컴퓨터공학과 졸업(석사)

1999 ~ 현재 창원대학교 대학원 컴퓨터공학과 박사과정

1995 ~ 현재 창원대학교 강사

관심분야: 패트리 넷, 성능분석, 정보보호, 스케쥴링 연구