페트리 네트를 이용한 시스템 속성의 명세 및 분석

이 우 진†

요 약

소프트웨어 시스템 모델링에서는 정형적 기법으로 소프트웨어를 모델링하고 분석하여 소프트웨어 시스템이 가지는 문제점들을 구현에 앞서 미리 찾아 해결하고자 한다. 페트리 네트는 그래픽 정형 명세 언어로 병행적 시스템, 실시간 시스템, 통신 프로토콜 등의 소프트웨어 시스템 모델링 및 분석에 널리 이용되고 있다. 페트리 네트 분석에서, 교착상태(deadlock), 수행가능성(liveness) 등의 일반적인 시스템 특성 분석은 주로 도달성 분석을 통해 이루어지며 시스템 요구사항에 관한 고유 특성 분석은 모델 검사(model checking) 방법을 통해 이루어진다. 하지만 도달성 분석과 모델 검사 방법에서는 기본적으로 시스템의 모든 가능한 상태들을 나열하여 분석하므로 모델의 규모가 커짐에 따라 상태가 기하급수적으로 증가하는 상태 폭발(state explosion) 문제가 발생한다. 이 논문에서는 상태 폭발을 회피하면서 시스템의 요구사항을 체계적으로 분석할 수 있는 새로운 방법을 제안하고자 한다. 먼저 분석하고자 하는 요구사항을 속성 네트로 나타낸 후, 시스템 모델과 속성 네트를 합성하여 분석한다. 이러한 합성 분석에서는 분석 대상 속성과 연관되지 않는 모델의 부분들을 축약 규칙에 따라 축약함으로써 분석 도메인을 점진적으로 줄어나갈 수 있으며 요구사항 만족 여부를 간단히 검사할 수 있는 장점이 있다.

Specification and Analysis of System Properties by using Petri nets

Woo Jin Lee†

ABSTRACT

Software system modeling has a goal for finding and solving system's problems by describing and analyzing system model in formal notations. Petri nets, as graphical formalism, have been used in describing and analyzing the software systems such as parallel systems, real-time system, and protocols. In the analysis of Petri nets, general system properties such as deadlock and liveness are analyzed by the reachability analysis. On the other side, specific properties such as functional requirements and constraints are checked by model-checking. However, since these analysis methods are based on enumeration of all possible states, there may be state explosion problem, which means that system states exponentially increase as the size of system is larger. In this paper, we propose a new method for mechanically checking system properties with avoiding state explosion problem. At first, system properties are described in property nets then the system model and the property net are composed and analyzed. In the compositional analysis, system parts irrelevant to the specific property are reduced to minimize the analysis domain of the system. And it is possible to mechanically check whether a specific property is satisfied or not.

키워드 : 페트리 네트(Petri nets), 속성 네트(Property net), 시스템 속성(System Property), 속성 명세(Property Specification), 속성 분석(Property Analysis)

1. 서 론

페트리 네트(Petri nets)는 병행성 기술이 용이하고 도달성 분석, 상태불변(invariant) 분석 등 다양한 분석 방법이 제공되므로 프로토콜 검증[1], 실시간 시스템[2, 3]과 병행적 시스템[4] 등의 모델링 및 분석에 널리 이용되고 있다. 페트리 네트의 대표적인 분석 방법은 도달성 분석으로 시스템 모델의 가능한 모든 상태들을 도달성 그래프로 구성하여 교착상태(deadlock), 라이브락(livelock), 트랜지션의 수행가능성(liveness) 등의 일반적인 특성들을 분석한다. 그리고 시스템의 요구사항들을 검사하기 위해서는 모델 검사(model checking)를 이용한다[5]. 모델 검사에서는 시스템의 요구사항을 CTL로 표현하고 시스템의 모든 상태를 검사하여 CTL 논리식이 만족되는지 여부를 증명한다.

도달성 분석과 모델 검사 등의 페트리 네트 분석에서는 모든 시스템의 상태들을 나열하여야 함으로 시스템 규모가 커지면 상태의 수가 급증하는 상태 폭발(state explosion) 문제가 발생한다. 이러한 상태 폭발을 회피하는 방법으로는 동치 클래스(equivalence class)에 속하는 상태의 그룹화를 통한 상태 축소[6], 모듈화 개념을 적용하여 분할정복 방식

※ 이 논문은 2002년도 경북대학교의 연구비에 의하여 연구되었음.
† 정 회 원 : 경북대학교 컴퓨터과학과 교수
논문접수 : 2003년 1월 17일, 심사완료 : 2003년 9월 22일

으로 도달성 그래프를 생성하는 합성적 도달성 분석[4], 축약 규칙(reduction rules)을 적용한 모델 축소[7] 등이 제안되었다. 이러한 방법들은 교착상태, 트랜지션의 수행가능성 등의 일반적인 시스템 특성을 분석하는데 많이 이용되지만 구체적인 시스템의 요구사항을 검사하는 모델 검사에는 제대로 활용되지 못하고 있다. 모델 검사에서는 분석하고자 하는 특성에 따라 각각의 시스템 상태들이 나름의 의미를 가지므로 일반적인 관점에서 시스템 상태들을 추상화할 수 없다. 그러므로 모델 검사에서는 기본적으로 모든 시스템 상태들을 나열하는 것을 가정한다.

이 연구에서는 페트리 네트 분석에서 상태 폭발 문제를 회피할 수 있는 새로운 요구사항 검사 방법을 제안하고자 한다. 시스템에서 반드시 수행되어야 하는 행위를 나타내는 진행 속성(progressive property)과 시스템의 제약 사항들을 나타내는 제약 속성(constraining property)을 페트리 네트로 나타낸다. 이러한 요구사항을 나타내는 페트리 네트 모델을 속성 네트라고 한다. 속성 네트는 시스템의 상황을 모니터링하는 역할을 담당하며 시스템 모델과 병행적으로 합성되어 수행된다. 즉, 진행적 속성의 경우 일련의 요구되는 행위들이 일어나는지 검사하며 제약적 속성의 경우는 금하는 상황이 발생하는지를 검사한다. 일반적으로, 하나의 시스템 속성은 국부적인 혹은 추상적인 특성을 가지므로 속성과 연관되지 않는 시스템 요소들이 많이 존재할 수 있다. 이러한 시스템 요소들을 축약 규칙(reduction rules)을 통해 추상화하여 분석 도메인을 최소화함으로써 효율적인 합성 분석을 수행할 수 있다.

모델 검사와 논리 명세의 Theorem Proving 기법은 대표적인 시스템 속성 분석 방법이며 특히 모델 검사는 유한상태머신, Statecharts, 페트리 네트 등과 같이 시스템 상태를 중심으로 분석을 수행하는 정형 언어에 널리 사용된다. 이 연구는 모델 검사의 상태 폭발 문제를 극복할 수 있는 새로운 속성 분석 방법으로 속성과 연관된 상태공간만을 검색하므로 전체 상태공간을 모두 검사하여야 하는 모델 검사에 비해 효율적인 분석을 수행할 수 있다. 이 연구는 현재 페트리 네트 모델을 대상으로 연구되어 있지만 상태 중심 분석을 수행하는 다른 정형 언어로 쉽게 확장될 수 있다.

이 논문의 구성은 다음과 같다. 먼저, 제 2장에서는 페트리 네트를 이용한 시스템 모델링과 도달성 분석 및 모델 검사 등의 분석 방법에 대해 살펴본다. 제 3장에서는 진행 속성과 제약 속성을 나타내는 속성 네트를 정의하고 속성 네트의 사용 패턴을 설명한다. 제 4장에서는 시스템 모델과 속성 네트를 합성하여 분석하는 방법에 대해 기술한다. 그리고 제 5장에서는 사례연구로 속성 네트를 이용하여 통신 분야의 서비스 상호작용(Service Feature Interaction) 문제[11]를 해결하는 방법에 대해 설명한다. 마지막으로 제 6장에서는 결론과 향후 연구방향을 제시한다.

2. 페트리 네트 모델링 및 분석

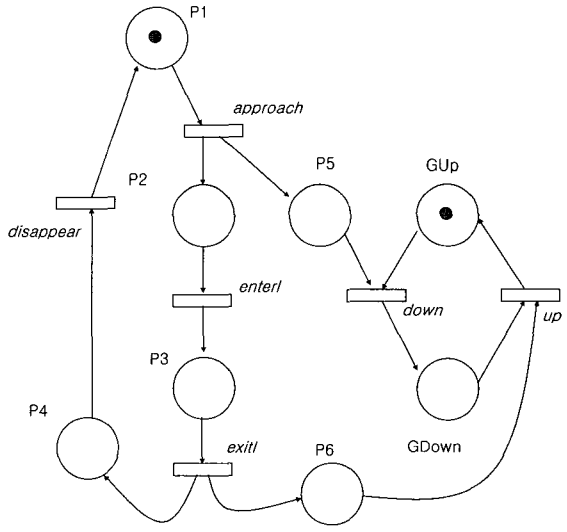
페트리 네트는 그래픽 모델링 언어로 유한 상태 머신(Finite State Machine)과 함께 시스템의 모델링 및 분석에 널리 이용되고 있다. 페트리 네트는 병행성을 쉽게 표현할 수 있다는 장점과 구조적 분석, 도달성 분석 등의 다양한 분석 방법을 제공하는 장점이 있다. 그러나, 기본 페트리 네트 개념에는 모듈화 특성이 반영되어 있지 않아 이러한 특성을 보완한 CMPNs[9], 수학적 논리식 등을 이용하여 표현력을 향상시킨 Colored Petri Nets[13], Predicate/Transition nets[14] 등 다양한 확장된 페트리 네트 개념들이 존재한다. 일반적인 페트리 네트로는 Place/Transition nets(P/T nets)을 널리 이용한다. P/T net은 아래와 같이 정의된다.

[정의 1] Place/Transition nets [8]

아래 조건을 만족하는 5-tuple N = (P, T, F, W, M)은 P/T nets이다.

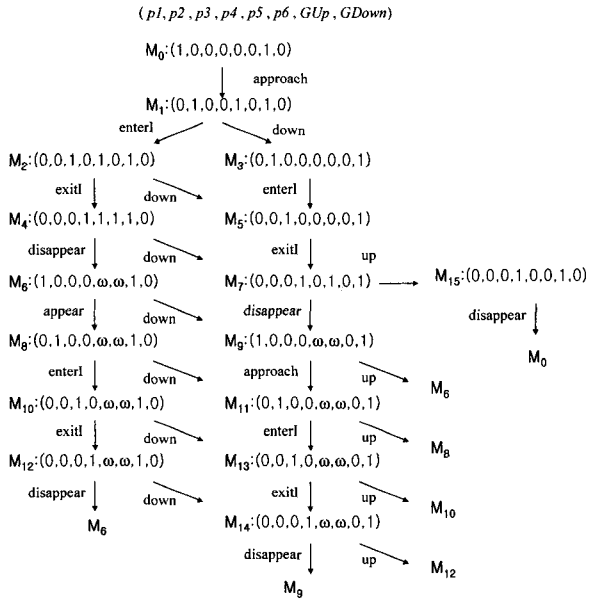
- (P, T, F)는 유한 네트이다. 여기에서 P는 플레이스의 집합, T는 트랜지션의 집합, F는 아크 집합을 나타낸다.
- $W : P \rightarrow N \setminus \{0\}$, 아크의 가중치를 나타낸다. 아크의 가중치는 일반적으로 1인 경우가 대부분이며 이 경우는 모델에 나타내지 않는다.
- $M : P \rightarrow N$, 각 플레이스의 초기 토큰을 나타낸다.

(그림 1)은 기차 건널목에 설치된 차단기를 P/T nets으로 나타내고 있다. 건널목 차단기 모델에서는 기차의 상태와 차단기의 상태를 병행적으로 나타내고 있다. 기차는 approach-enterI-exitI-disappear 과정을 반복하며 차단기는 down-up 과정을 반복한다. 기차가 접근하면 제어기가 차단기를 내리고 기차가 건널목을 통과하면 차단기를 올리도록 모델링되어 있다.



(그림 1) 기차 건널목 차단기의 페트리 네트 모델링

(그림 2)는 (그림 1)의 건널목 차단기 모델의 도달성 그래프를 나타내고 있다. 도달성 그래프에서 노드는 시스템 상태를 나타내는 것으로 각 플레이스의 토큰 수를 나타내는 벡터로 표현한다. 건널목 차단기 모델의 초기 시스템 상태는 $M_0(P_1, P_2, P_3, P_4, P_5, P_6, GUp, GDown) = (1, 0, 0, 0, 0, 0, 1, 0)$으로 표시된다. 도달성 그래프에서는 초기 시스템 상태에서 도달 가능한 모든 시스템 상태들을 나타내고 시스템 상태들간의 상태전이를 나타낸다. 페트리 네트에서 반복적인 행위를 하면서 대부분의 플레이스의 토큰은 그대로 유지한 채, 특정 플레이스의 토큰 수만 무한히 증가하는 경우가 발생하는데 이러한 경우는 항상 새로운 시스템의 상태가 만들어지므로 도달가능한 모든 시스템의 상태를 나열할 수 없다. 도달성 그래프에서는 이러한 경우 시스템 상태에 대한 포함관계(coverability)를 적용하여 토큰이 증가하는 플레이스를 무한(ω)으로 표시하여 추상화한다[8]. (그림 2)에서 $M_5 = (0, 0, 0, 1, 1, 1, 1, 0)$ 상태에서 disappear 트랜지션이 수행되면 $M_6 = (1, 0, 0, 0, 1, 1, 1, 0)$ 상태로 변환되는데 이 상태는 초기 상태 $M_0 = (1, 0, 0, 0, 0, 0, 1, 0)$를 포함하며 추가적으로 P_5, P_6에 토큰을 더 가진 경우이므로 approach-enterI-exitI-disappear-approach 트랜지션들이 무한히 반복될 수 있으며 P_5, P_6는 토큰을 무한히 누적할 수 있다. (그림 2)의 도달성 그래프에서는 P_5, P_6를 무한(ω)으로 표시하여 단순화시켜 나타낸다.



(그림 2) 건널목 차단기 모델의 도달성 그래프

모델 검사[5]는 도달성 그래프와 같은 유한상태 모델(M)에서 CTL로 명시된 명제 시제 논리식이 만족되는지 여부를 검사하는 것이다. 예를 들어, 건널목 차단기 모델에서 "기차가 지나갈 때는 반드시 차단기가 내려져 있어야 한다"라는 요구사항이 만족되는지 검사하려면, P_3와 GUp 플레이스 동시에 토큰을 가지지 않아야 하므로 $AG(\neg(P_3 \wedge GUp))$와 같이 검사하고자 하는 요구사항을 CTL로 나타내고 도달성 그래프의 모든 시스템 상태를 검사하여 $M \models AG(\neg(P_3 \wedge GUp))$의 만족여부를 검사한다. 모델 검사는 도달성 그래프 상에서 이루어져야 하므로 무한(ω)이 포함된 추상화된 도달성 그래프에서는 경우에 따라 검사할 수 없는 속성들이 있을 수 있다. 그리고 도달성 그래프 생성시에 발생할 수 있는 상태 폭발 문제가 모델 검사에 그대로 남아 있으므로 규모가 크고 복잡한 시스템 모델에는 모델 검사를 적용하기 어렵다.

3. 속성 네트를 이용한 시스템 요구사항 표현

속성 네트는 시스템 모델에서 만족되어야 하는 요구사항을 페트리 네트로 표현한 것이다. 유사한 연구로는 시스템의 안전성(safety) 속성을 오토마타로 표현하여 유한상태머신으로 표현된 시스템 모델과 합성적 분석을 통해 만족 여부를 검사하는 연구가 있다[10]. 시스템의 요구사항은 시스템의 행위에 반드시 포함되어 있어야 하는 진행 속성(progressive property)과 시스템의 행위를 제한하는 제약 속성(constraining property)으로 크게 나뉠 수 있다. 진행 속성과 제약 속성에 대한 정의는 아래와 같다.

[정의 2] 진행 속성

진행 속성은 시스템 행위에 반드시 포함되어야 하는 긍정적인 요구 사항으로 순차적인 이벤트 혹은 시스템 상태들로 표현된다.

[정의 3] 제약 속성

제약 속성은 시스템 안전성과 같이 시스템의 행위를 제한하는 제약사항을 나타낸다. 제약 속성은 시스템 모델에서 발생하지 않아야 하는 순차적인 이벤트와 시스템 상태들로 표현된다.

시스템의 진행 속성은 시스템의 부분 행위 혹은 추상적인 행위라 볼 수 있다. 전체 시스템의 행위 보다는 부분적인 관점에서 시스템을 보는 시각으로, 관심있는 이벤트 또는 시스템의 상태에 대해서만 기술한다. 예를들어, $Ea \rightarrow Eb \rightarrow Ec$, $상태(A, B) \rightarrow Ea \rightarrow Eb$와 같이 이벤트 혹은 토큰을 가진 플레이스로 표현된 상태의 연속(sequence)으로 기술된다.

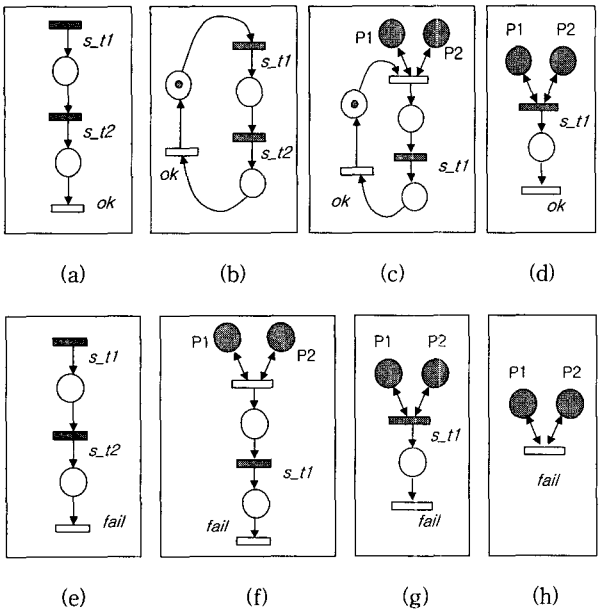
속성 네트는 진행 속성과 제약 속성을 표현하는 페트리 네트 모델이다. 속성 네트는 시스템이 특정 요구사항을 만족하는지 여부를 모니터링하고 검사하는 방편이기 때문에 시스템 모델에 새로운 기능 혹은 제약사항을 추가하거나 기존 행위를 변경시켜서는 안된다. 속성 네트는 시스템 모델에 나타난 트랜지션 또는 플레이스의 토큰 상태를 참조

하여 표현된다. 속성 네트는 [정의 4]와 같이 정의된다. 속성 네트는 일반 페트리 네트와 동일하게 표현되며 단지 시스템 모델의 트랜지션과 플레이스 참조를 통해 시스템 모델과의 연계성을 나타낸다. 그리고 분석의 편의를 위해 'ok', 'fail' 트랜지션을 가지고 있다. 'ok' 트랜지션은 진행 속성의 만족 여부를 간단하게 확인하기 위해 추가되며 'fail' 트랜지션은 제약 속성이 어겨지는지 여부를 확인하기 위해 추가된다.

[정의 4] 속성 네트(Property Nets)

시스템의 행위를 모델링한 페트리 네트 모델 M = (P, T, F, W, M_0)이 있을 때, 속성 네트 $PROP_N$ = (P_N, T_N, F_N, W_N, M_N)은 다음과 같이 정의된다.

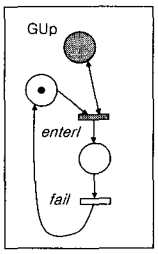
- P_N = subset(P) ∪ LP. 여기에서 LP는 모델 M에 없는 지역적 플레이스들을 나타낸다.
- T_N = subset(T) ∪ {ok, fail}. 여기에서 'ok'와 'fail'트랜지션은 속성을 나타내는 순차적인 이벤트 및 상태가 만족되는지 여부를 나타낸다.
- F_N는 플레이스와 트랜지션을 연결하는 아크를 나타낸다. 아크에는 양방향 아크를 포함하는데 양방향 아크는 방향이 반대인 두개의 단방향 아크와 동일한 의미를 가지며 이를 단순화시켜 나타낸 것이다.
- W_N는 속성 네트의 아크 가중치 함수이다.
- M_N는 속성 네트의 초기 토큰을 나타낸다.



(그림 3) 진행 속성과 제약 속성의 속성 네트의 예

(그림 3)은 진행 속성과 제약 속성의 예를 보여준다. 그림에서 색칠된 플레이스와 트랜지션은 시스템 모델에 나타난 플레이스와 트랜지션의 참조를 나타낸다. (그림 3)(a)와 (그림 3)(b)는 일회성 및 반복적으로 일어나는 연속적인 트랜지션(e.g $s\_t_1 \rightarrow s\_t_2$)을 가지는 진행 속성의 예를 보여준다. (그림 3)(c)는 항상 특정 시스템 상태가 만족된 다음에 특정 트랜지션이 발생하여야 함을 나타내는 진행 속성을 나타낸다. (그림 3)(d)는 특정 시스템 상태가 만족된 상황에서 특정 트랜지션이 반드시 수행되어야 함을 나타내는 진행 속성을 나타낸다. (그림 3)(c)와 (그림 3)(d)의 차이는 P_1과 P_2 플레이스가 동시에 토큰을 경우, (그림 3)(c)는 "향후 언제든지" s_t_1이 수행되면 된다는 것을 나타내고 (그림 3)(d)는 "이 상태에서" s_t_1이 수행되어야 한다는 것을 나타낸다. 즉, (그림 3)(d)의 경우는 플레이스 P_1 또는 P_2가 토큰을 동시에 가졌더라도 다른 트랜지션에 의해 그 중 하나의 플레이스가 토큰을 잃어버리면 s_t_1이 수행될 수 없다. (그림 3)(e), (그림 3)(f), (그림 3)(g)는 진행 속성인 (그림 3)(a), (그림 3)(c), (그림 3)(d)에 대응하는 제약 속성을 보여준다. 여기에서는 'ok' 트랜지션 대신에 'fail' 트랜지션이 이용된다. (그림 3)(h)는 P_1 과 P_2 플레이스가 동시에 토큰을 가지는 시스템의 상태가 발생하면 안된다는 것을 나타낸다.

(그림 4)는 기차 건널목 차단기 예에서 "기차가 지나갈 때는 차단기가 내려가 있어야 한다"는 안전 요구사항을 나타내는 제약 속성을 보여준다. 차단기가 GUp 플레이스에 토큰이 있을 때 enterI 트랜지션이 수행되면 'fail' 트랜지션이 수행된다는 것을 나타낸다.

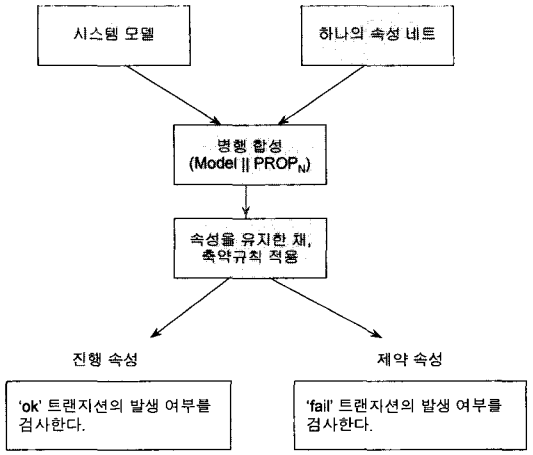


(그림 4) 기차 건널목 차단기의 안전성에 관한 제약 속성의 예

4. 속성 네트를 이용한 요구사항 분석

속성 네트의 만족 여부는 시스템 모델과 속성 네트의 합성을 통해 이루어진다. 시스템 모델과 속성 네트의 합성은 서로 공유하는 플레이스와 트랜지션들을 하나로 묶어 하나의 페트리 네트 모델로 만든다. 속성 네트의 만족 여부는 'ok', 'fail' 트랜지션의 발생 여부로 쉽게 검사할 수 있다. 진행 속성일 때, 합성 모델의 분석에서 'ok' 트랜지션이 일회적으로 또는 반복적으로 나타나는 경우는 진행 속성이 만족됨을 나타내고 'ok' 트랜지션이 나타나지 않는 것은 만족되지 않음을 나타낸다. 제약 속성일 때는 'fail' 트랜지션이 발생하면 제약 속성이 만족하지 않음을 나타내고 'fail' 트랜지션이 나타나지 않음은 제약 속성이 만족됨을 나타낸다.

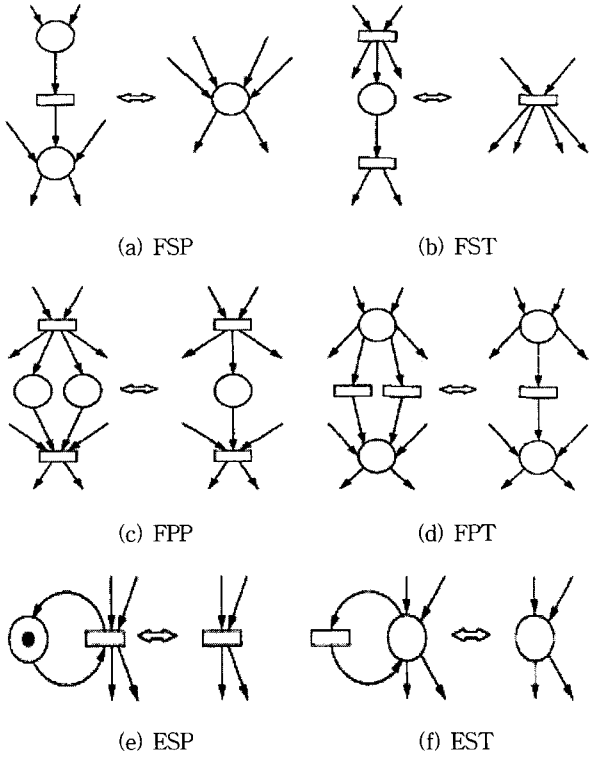
(그림 5)는 속성 네트를 이용한 요구사항 분석 과정을 보여준다.



(그림 5) 속성 네트를 이용한 요구사항 분석의 개략도

(그림 5)에서 합성 모델을 축약하는 과정이 있는데, 이는 합성 모델의 도달성 분석의 노력을 최소화하기 위해 기존의 축약규칙을 적용하여 합성 모델을 축약한다. 축약 규칙을 적용할 때는 속성과 연관된 부분은 유지하고 속성과 연관되지 않는 부분들을 추상화한다. 페트리 네트 모델의 기존 축약 규칙은 생존성(liveness), 안전성(safeness), 유한성(boundedness)을 유지시킨다[7]. (그림 6)은 6가지 축약규칙을 보여주고 있으며 각 축약규칙에 대한 구체적인 설명은 아래와 같다.

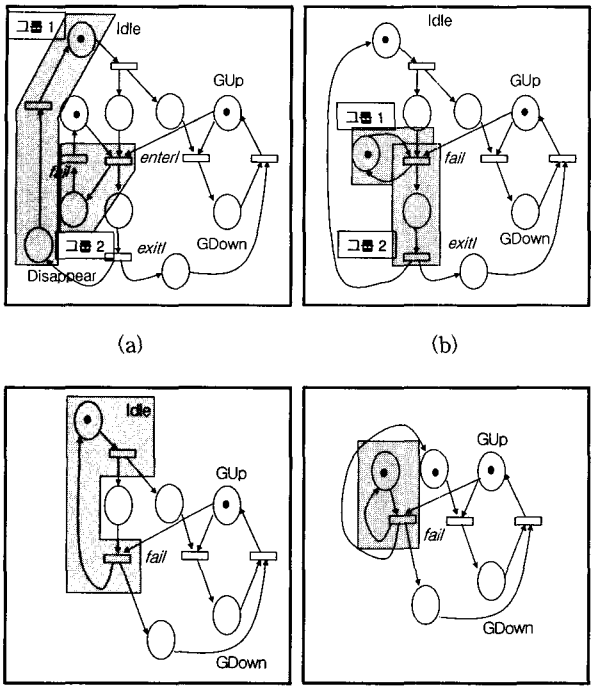
- (그림 6)(a) : 연속된 플레이스의 합성(Fusion of Series Places : FSP)은 두 개의 연속된 플레이스가 하나의 트랜지션에 의해서만 연결될 때 이 두 플레이스를 축약한다.
- (그림 6)(b) : 연속된 트랜지션의 합성(Fusion of Series Transitions : FST)은 두 개의 연속된 트랜지션이 하나의 플레이스에 의해서만 연결될 때 이 두 트랜지션을 하나로 축약한다.
- (그림 6)(c) : 병렬 플레이스의 합성(Fusion of Parallel Places : FPP)은 두 플레이스가 한 트랜지션에서 분기하여 바로 아래 트랜지션으로만 합병할 때 두 플레이스를 하나로 축약한다.
- (그림 6)(d) : 병렬 트랜지션의 합성(Fusion of Parallel Transitions : FPT)은 두 트랜지션이 한 플레이스에서 분기하여 바로 다음 플레이스로만 합병할 때 두 트랜지션을 하나로 축약한다.
- (그림 6)(e) : 셀프-루프 플레이스의 삭제(Elimination of Self-loop Places : ESP)는 트랜지션 수행에 전혀 영향을 주지 않는 토큰을 가진 셀프-루프 플레이스를 삭제한다.
- (그림 6)(f) : 셀프-루프 트랜지션의 삭제(Elimination of Self-loop Transitions : EST)는 항상 수행될 수 있는 셀프-루프 트랜지션을 삭제한다.



(그림 6) 페트리 네트의 축약 규칙

속성 네트를 이용하여 시스템의 요구사항을 검사하는 과정을 설명하기 위해 (그림 1)에 나타난 건널목 차단기 모델과 (그림 4)에 기술된 안전성 제약 속성을 합성하여 분석하는 과정을 설명한다. (그림 7)(a)는 건널목 차단기 모델과 안전성을 나타내는 속성 네트가 합성된 모델을 보여준다. 그림에서 시스템 모델과 속성 네트에서 공유하고 있는 GUp 플레이스와 enterI 트랜지션을 하나로 묶어 합성 모델에 나타내고 있다. 합성 모델에서 'fail' 트랜지션이 발생하는지 여부를 분석하기에 앞서 합성 모델에 축약 규칙을 적용하여 합성 모델을 최소화하는 과정을 거친다. (그림 6)(a)에서 그룹 1과 그룹 2는 각각 FSP, FST 축약 규칙이 적용되어 (그림 6)(b)와 같이 축약된다. 축약 규칙 적용 시에는 만족 여부 최종판단을 위해 'ok' 혹은 'fail' 트랜지션은 보존되어야 한다. (그림 6)(b)에서 그룹 1과 그룹 2는 각각 ESP와 FST 축약 규칙이 적용된다. (그림 6)(c)와 (그림 6)(d)에서는 각각 FST와 ESP 축약규칙이 적용된다. (그림 6)(d)에서 축약 부분에 ESP를 적용한 추상화된 모델에서는 굳이 도달성 분석을 수행하지 않더라고 쉽게 'fail' 트랜지션이 발생함을 알 수 있다. 즉, 건널목 차단기 모델에서는 차단기가 올려져 있는 상태에서 기차가 지나갈 수 있으므로 건널목

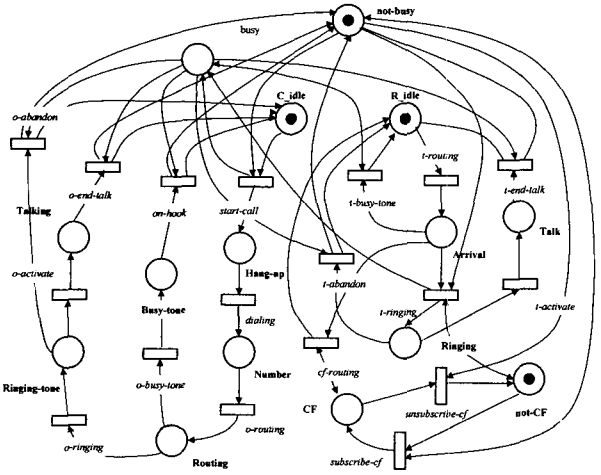
차단기의 안전성이 만족되지 못한다.



(그림 7) 합성 모델에서 축약규칙의 단계적 적용 과정

5. 사례 연구

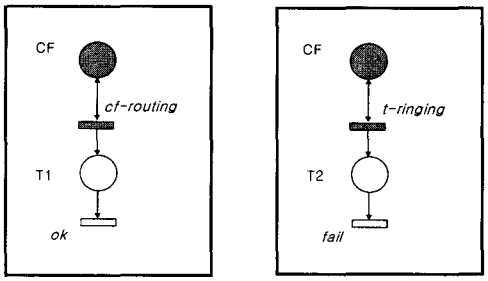
서비스 상호작용(Service Feature Interaction) 문제는 유선 통신에서 새로운 통신 서비스를 추가하려 할 때, 새로운 서비스와 기존 통신 서비스간의 충돌이 발생하는지 여부를 미리 검사하고자 하는 것이다. 서비스 상호작용 문제를 해결하기 위해 다양한 정형적 기법과 분석 방법들이 적용되어 왔다[11]. (그림 8)은 기본 호 처리 과정을 추상화하여 나타낸



(그림 8) 기본 호 처리 모델과 호전달 서비스에 대한 페트리 네트

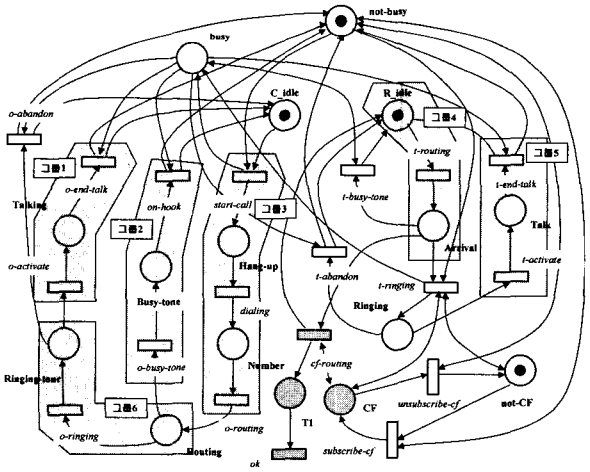
모델과 착신호를 다른 번호로 전달하는 호전달(call forwarding : CF) 서비스 [12]에 대한 페트리 네트 모델을 보여주고 있다. 기본 호 처리 과정은 전화를 거는 호 송신 기능과 전화를 받는 호 수신 기능으로 크게 나뉘어진다. (그림 8)에서 C_idle 플레이스와 연관된 모델의 왼쪽 부분이 호 송신 기능을 나타내고 있는 R_idle 플레이스와 연관된 오른쪽 부분이 호 수신 기능을 나타낸다. 그리고 오른쪽 아래에 CF, not-CF 플레이스와 연관된 부분이 호전달 서비스를 나타낸다.

(그림 9)는 호전달 서비스와 연관된 요구사항을 속성 네트로 표현한 것이다. (그림 9)(a)는 호전달 서비스의 기본 기능을 나타내는 진행 속성으로 CF 기능이 설정되면 cf-routing 트랜지션이 반드시 수행되어 호 전달 기능이 정상적으로 이루어져야 함을 나타낸다. (그림 9)(b)는 호전달 서비스에 관한 제약 속성으로 CF 기능이 설정되어 있으면 정상 수신 호 처리 트랜지션인 t-ringing이 수행되면 안된다는 것을 나타낸다.



(a) 진행속성 (b) 제약속성

(그림 9) CF 기능에 대한 속성 네트



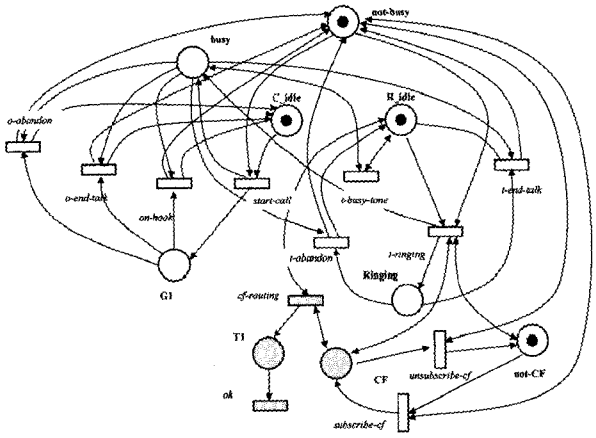
(그림 10) 시스템 모델과 진행 속성 네트의 합성 모델 및 축약규칙 적용예

(그림 10)은 (그림 8)의 기본호 처리 및 호전달 서비스 모델과 (그림 9)(a)의 진행 속성 네트를 합성한 모델을 보여준다. (그림 10)에서 색칠된 부분이 속성 네트를 나타낸다. 'ok' 트랜지션의 수행 여부를 분석하기에 앞서 합성 모델에서 추상화할 수 있는 부분들을 축약한다. 축약 규칙은 속성

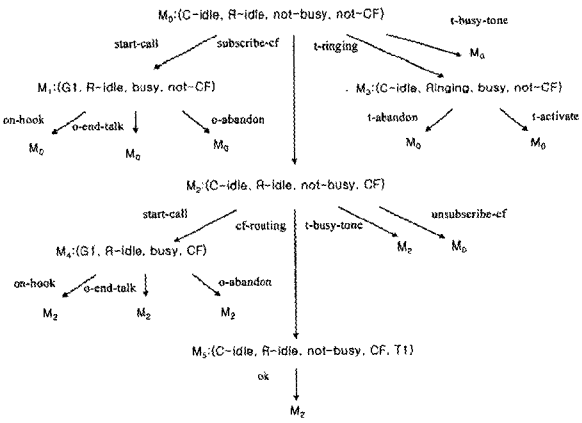
과 직접적으로 연관되지 않는 부분에만 적용된다. (그림 10)의 합성 모델에 축약 규칙을 적용하는 과정은 아래와 같다.

- 그룹 1 : FST 규칙 적용
- 그룹 2 : FST 규칙 적용
- 그룹 3 : FST 규칙을 연속적으로 두 차례 적용
- 그룹 4 : FSP 규칙 적용
- 그룹 5 : FST 규칙 적용
- 그룹 6 : FSP 규칙 적용

(그림 11)은 위의 축약 규칙들을 적용한 최종 축약 모델을 보여준다. 그리고 (그림 12)는 최종 축약 모델의 도달성 그래프를 나타내고 있다. 축약된 모델의 도달성 분석을 통해 쉽게 'ok' 트랜지션이 수행됨을 알 수 있다. 즉, 호전달 서비스 가입시에 기본 기능인 cf-routing 트랜지션이 항상 수행됨을 알 수 있다. (그림 9)(b)의 호전달 서비스의 제약사항을 나타내는 제약 속성도 위와 유사한 과정을 통해 분석이 가능하다. 제약 속성은 'fail' 트랜지션이 발생하면 제약사항이 만족되지 않음을 나타낸다. (그림 9)(b)의 제약 속성과 (그림 8)의 모델의 합성 모델의 도달성 분석에서는 'fail' 트랜지션이 발생하지 않음을 확인할 수 있다.



(그림 11) 축약된 합성 모델



(그림 12) 최종 축약 모델의 도달성 그래프

6. 결 론

본 논문에서는 효율적이고 체계적으로 시스템의 요구사항을 분석할 수 있는 새로운 분석 방법을 제안하였다. 시스템 요구사항을 속성 네트로 나타내고 속성 네트와 시스템 모델을 합성하여 축약한 후, 축약된 모델의 도달성 분석을 통해 요구사항의 만족 여부를 검사한다. 이러한 방법은 기존의 도달성 분석과 모델 검사에 내포된 상태 폭발 문제를 회피할 수 있으며 분석과정이 단순하여 쉽게 도구화할 수 있는 장점이 있다.

현재는 P/T nets에 근거하고 있지만 이를 확장하여 Colored Petri nets, Predicate/Transition nets 등의 고급 페트리 네트에 적용하는 연구가 필요하며 속성 네트를 이용한 분석 방법을 체계적으로 지원할 수 있는 분석 도구의 개발이 요구된다. 그리고 속성 네트의 표현력을 확장하여 시스템의 다양한 요구사항들을 나타낼 수 있도록 확장하여야 한다.

참 고 문 헌

[1] T. Suzuki, S. M. Shatz, "A Protocol Modeling and Verification Approach Based on a Specification Language and Petri Nets," IEEE Trans. on Software Engineering, Vol.16, No.5, pp.523-536, May, 1990.

[2] C. Ghezzi, D. Mandrioli, S. Modasca and M. Pezze, "A Unified High-Level Petri Net Formalism for Timed-Critical Systems," IEEE Trans. on Software Engineering, Vol.17, No.2, pp.160-172, 1991.

[3] G. Bucci and E. Vicario, "Compositional Validation of Time-Critical Systems Using Communicating Time Petri Nets," IEEE Trans. on Software Engineering, Vol.21, No.12, pp. 969-992, Dec., 1995.

[4] Wei Jen Yeh, "Controlling State Explosion in Reachability Analysis," Ph.D. Thesis, Purdue University, Dec., 1993.

[5] E. M. Clarke, E. A. Emerson and A. P. Sistla, "Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications," ACM Transactions on Programming Languages and Systems, Vol.8, No.2, pp.244-263, 1896.

[6] J. B. Jorgensen and L. M. Kristensen, "Verification of Coloured Petri Nets Using State Spaces with Equivalence Classes," Proc. of the Workshop on Petri Nets in Sys. Eng. (PNSE '97), Sep., 1997.

[7] T. Murata, "Petri Nets : Properties, Analysis and Application," Proceedings of The IEEE, Vol.77, No.4, Apr., 1989.

[8] Wolfgang Reisig, "Petri Nets : An Introduction," Springer-Verlag, 1985.

[9] W. J. Lee, S. D. Cha and Y. R. Kwon, "Integration and Analysis of Use Cases Using Modular Petri Nets in Requirements Engineering," IEEE Trans. On Software Engineering, Vol.24, No.12, pp.1115-1130, Dec., 1998.

[10] S. C. Cheung and J. Kramer, "Checking Subsystem Safety Properties in Compositional Reachability Analysis," Proceeding of ICSE, pp.144-154, 1996.

[11] Dirk O. Keck and Paul J. Kuehn, "The Feature and Service Interaction Problem in Telecommunications Systems : A Survey," IEEE Transactions on Software Engineering, Vol.24, No.10, Oct., 1998.

[12] ITU-T, "ITU-T Intelligent Network CS-2 Recommendations Q.1220-Q.1225," Geneva, Jan., 1997.

[13] K. Jensen, "Coloured Petri Nets : Basic Concepts, Analysis methods and Practical Use," Springer-Verlag, Vol.1, 1992.

[14] H. J. Genrich, "Predicate/Transition Nets," Petri Nets : Applications and Relationships to other Models of Concurrency (ed. W. Brauer, W. Reisig, and G. Rozenberg), Springer-Verlag, pp.207-247, 1987.

이 우 진

e-mail : woojin@knu.ac.kr

1992년 경북대학교 컴퓨터과학과(학사)

1994년 한국과학기술원 전산학과(공학석사)

1999년 한국과학기술원 전산학과(공학박사)

1999년~2002년 한국전자통신연구원 S/W 공학연구부 선임연구원

2002년~현재 경북대학교 컴퓨터과학과 전임강사

관심분야 : CBD, Requirements Engineering, 웹서비스 기술, Petri nets, 실시간 시스템 모델링