동작 특성 기반의 시스템 신뢰도 분석

정회원 나 윤 지*, 고 일 석**, 조 용 환***

Reliability Analysis of A System Based on the Execution Characteristics of sub-Modules

Na Yun Ji*, Ko Il Seok**, Cho Yong Hwan** *Regular Members*

요 약

시스템의 대규모, 복잡화는 시스템의 결함 여부를 테스트하는 것을 어렵게 하고 있다. 시스템의 구성은 기능별 모듈로 나눌 수 있으며, 시스템의 각 모듈이 전체 시스템의 신뢰도에 미치는 영향의 효율적인 분석을 통해 시스템 전체에 대한 효율적인 테스트 전략을 수립할 수 있고 이는 전체 시스템 테스트의 효율을 증대시킬 수 있을 것이다. 또한 시스템 모델을 정형화된 모듈로 나타낼 경우 그 시스템의 동작상의 논리적인 정확성, 성능분석 및 신뢰성 예측 등의 여러 특성 등의 효율적인 분석이 가능하다. 본 논문에서는 시스템 모듈을 그래프 모델로 정형화하고, 이것의 시스템 동작 특성을 기반으로 한 분석을 통하여 각 모듈이 전체 시스템의 신뢰성에 미치는 영향을 분석하는 기법을 제안한다.

ABSTRACT

Complexity of the hardware system grows larger, the fault testing becomes more difficult. As we divide system into the functional sub-modules and analyze reliability of a sub-modules on the system, We improve the system test performance. And the analysing results help us to set up the effective strategies for the system test. Also describing the system to a formalized sub-modules, we can analyze the logical accuracy and the characteristics of system. So we can predict the reliability of the system based on execution characteristics. In this paper we propose a reliability analysis method of a system based on the execution characteristics of sub-module.

Ⅰ. 서 론

시스템의 대규모, 복잡화는 테스트의 문제를 더욱 어렵게 만들고 있다. 또한 집적도의 증가는 시스템 내부에 여러 가지 형태의 고장 요소가 포함될 확률을 더욱 높이고 있다[1,2,3]. 따라서 시스템의 테스트는 더욱 중요한 문제로 부각될 것이고 이러한 시스템의 신뢰도를 검증하기 위한 테스트는 효율적으로 수행되어야만 비용과 노력의 절감을 가져올 수 있다[3,4].

이에 따라 시스템의 설계 시 시스템 동작 특성을 고려한 분석을 통하여 각 모듈이 전체 시스템에 미치는 신뢰도를 분석할 수 있다면, 이것의 분석 결과를 이용하여 전체 시스템에 대해 효율적이고 적절한 비용을 가지는 테스트 전략을 세워 테스트의 효율을 증대시킬 수 있다. 또한 재사용성과 이식성, 확장성의 향상을 위해 컴포넌트와 같은 형태의 부모듈을 사용한 시스템의 개발이 이루어지고 있다[2,5].

본 논문에서는 시스템의 설계단계에서 각 시스템 모듈간의 동작 특성을 분석, 이에 따라 시스템의 각 모듈이 시스템의 신뢰성도에 미치는 영향을 규명할 수 있는 기법을 제안한다.

* 대전보건대학 컴퓨터정보처리과(yjna2967@kebi.com) ** 충북과학대학 전자상거래과(isko@ctech.ac.kr)
*** 충북대학교 전기전자 및 컴퓨터공학부
논문번호 : K01195-0830, 접수일자 : 2001년 8월 30일

Ⅱ. 패트리네트

패트리네트는 1962년 독일의 C. A. Petri 박사의 학위 논문에서 제시된 모델[6,7]로서, 비동기적이고 동시발생 시스템의 정보 흐름을 모델링하는 모델로 개발되었다[8,9].

1. 패트리네트 모델

패트리네트(Pn)은 플레이스(Place)의 유한집합 P, 트랜지션(Transition)의 유한 집합 T, 정방향 영향함수(Forward incidence function) F, 역방향 영향함수(Backward incidence function) B로 구성된다[1,2]. 또한 전형적인 패트리네트를 시스템의 상황을 표현하기 위한 동적 모형으로 표현하기 위해서는 프로세서들에 토큰이 할당된 모델을 필요로 한다. 정의 1과 같이 패트리네트의 프로세서는 조건(Condition), 사건(Event) 및 그들간의 관계를 서술하는 규칙(Rule)들로 구성된다[9].

[정의 1] 토큰을 갖는 패트리네트(Pnt)

Pnt = {P, T, F, B, t}

t = $\{t_1, t_2, \cdots, t_n\}$이고 $t_{i \in N}$, i=1,2,⋯,n이고

P_i의 토큰의 수는 t_i이다.

P = {p1, p2, ……, pn}:플레이스들의 유한집합

T = {t1, t2, ……, tn}:트랜지션들의 유한집합

F : P × T → N:트랜지션이 1개 이상의 플레이스로 매핑되는 입력 함수들의 집합

B : P × T → N:트랜지션이 1개 이상의 플레이스로 매핑되는 출력함수들의 집합

P ≠ ø, T ≠ ø, P ∩ T = ø

1. 하나의 트랜지션 t에 대하여, 입력 플레이스 p가 최소한 W(p, t)만큼의 토큰을 가지고 있으면, t는 장전되었다고 한다.
2. 장전된 트랜지션 t는 격발이 될 수도 있고, 아니 될 수도 있다.
3. t의 격발은 W(p, t) 만큼의 토큰을 각 입력 플레이스 p에서 제거하고, W(p, t) 만큼의 토큰을 각 출력 플레이스에 더해준다.

조건을 만족하는 것은 플레이스에 토큰을 표현하며 플레이스(place)와 트랜지션(transition)의 연결은 방향성 화살표로 나타낸다. 사건(transition)은 자신에게로 입력되는 조건(place)이 만족되어야 (즉, 토큰을 보유하고 있는 장전의 상태) 격발을 통해 점화(fire)된다. 트랜지션이 점화되면 자신의 각 입력 플레이스로부터 토큰을 하나씩 제거하고, 각 출력 플레이스에 토큰을 하나씩 첨가한다.

또한 패트리네트 모델의 도식적 표현인 패트리네트 그래프에서 이들 조건과 사건을 각각 원으로 표시한 플레이스와 선분으로 표시한 트랜지션으로 나타낸다.

2. 결함율 패트리네트(FPn)

일반적인 신뢰성 분석을 위한 신뢰도 패트리네트는 트랜지션이 성공적으로 점화할 확률로 정의된다. 기존의 신뢰도 패트리네트를 시스템의 고장율이나 시스템간의 신호전달 고정에서의 고장율을 모델링하기 의해 다음과 같은 결함율 패트리네트(FPn;Faulted Pn)을 정의한다.

[정의 2] 결함율 패트리네트는 다음과 같이 7가지로 구성된 튜플이다.

FPn = (P, T, I, O, M_0, F(i), F(o))

P={P_1, P_2, ⋯, P_m}:플레이스들의 유한집합

T = {t_1, t_2, ……, t_n}:트랜지션들의 유한 집합

I : P × T → N : 트랜지션이 1개 이상의 플레이스로 매핑되는 입력 함수들의 집합

O : P × T → N : 트랜지션이 1개 이상의 플레이스로 매핑되는 출력함수들의 집합

$M_0 \in M=\{M|M:P \rightarrow N\}$: M_0는 초기 토큰상태

F(i) : 입력함수들의 결함율의 집합

F(o) : 출력함수들의 결함율의 집합

P ≠ ø, T ≠ ø, P ∩ T = ø

정의 2에 의해 입력함수 I(T_i, P_j)는 P_j플레이스에서 T_i트랜지션 사이의 간선으로 표시하고 T_i트랜지션의 입력 플레이스가 P_j임을 나타내며, 출력함수 O(T_i, P_j)는 T_i트랜지션에서 P_j플레이스 사이의 간선으로 표시하고 T_i트랜지션의 출력 플레이스가 P_j임을 의미한다. 정의 4와 정의 5는 입력함수와 출력함수의 결함율을 나타낸다.

[정의 4] 입력함수의 결함율

입력함수의 결함율은 플레이스 P_i에서 트랜지션 t_j로 정확한 정보가 전달되지 않을 확률로서 입력함수 I(t_j, P_i)의 결함율 f(P_i, I(t_j))로 나타낸다.

[정의 5] 출력함수의 결함율

출력함수의 결함율은 ti트랜지션에서 Pj플레이스로 정확한 정보가 전달되지 않을 확률로서 출력함수 $O(t_i, P_j)$의 결함율 $f(O(t_i), P_j)$로 나타낸다. 동일한 트랜지션의 출력함수의 결함율은 모두 같다.

Ⅲ. 시스템 모델의 정형화

1. 블록 그래프

모듈별로 설계된 시스템은 방향성이 있는 다중그래프로 나타낼 수 있으며 따라서 다음의 정의와 같은 블록그래프로 정형화 할 수 있다.

[정의 5] 블록그래프 BDG = (M, S, R)
M : 모듈들의 유한 집합
S : 모듈간의 매핑의 유한 집합
R : 모듈과 접속의 관계 $M \times S \rightarrow M$

블록그래프에서 적어도 하나의 매핑관계가 존재하면 그 매핑관계의 시작점과 끝점인 모듈이 반드시 존재한다. $M_j = R(M_i, S_k)$가 존재하면 M_j모듈에서 M_i모듈로의 신호 S_k가 존재하는 것이다. 이와 같이 블록그래프에서 모든 매핑관계는 시작점 모듈과 이를 받는 끝점 모듈과의 관계로 표현한다. 또한 한 모듈이 특정 모듈사이에 여러 개의 매핑관계가 존재할 수 있다. 그림 1은 본 논문에서 사용할 블록그래프의 예이다. 여기에서 DIvIdend 모듈과 Control 모듈 사이의 매핑관계를 표현하면 $M_{DIvIdend} = R(M_{Control}, S_{EZERO})$, $M_{Control} = R(M_{DIVIdend}, S_{load})$, $M_{Control} = R(M_{DIvIdend}, S_{down})$로 표현 할 수 있다.

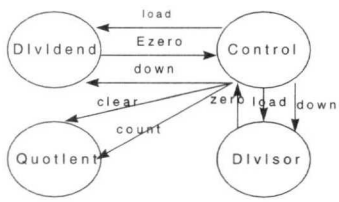


그림 1. 블록그래프

2. 블록그래프를 FPn으로 변환

FPn은 그림 1과 같이 한 노드에서 다른 노드로 여러 개의 간선을 가질 수 있으므로 다중 그래프이고, 각 트랜지션에 대해 입력함수와 출력함수의 방향성을 갖기 때문에 방향성 그래프이고, 결함율은 입력함수와 출력함수의 간선에 기록하기 때문에 가중치를 갖는 그래프이다. 즉, FPn은 방향성과 가중치를 갖는 다중 그래프이다. 알고리즘 1은 결함율 $f(P_i, I(t_j))$를 이용하여 블록 그래프를 FPn로 변환하는 알고리즘이다.

[알고리즘 1]
Finput : 1) 블록그래프 BDG = (M, S, R)
2) 결함율 $f(p_i, I(t_j))$
output : FPn 그래프 G=(V, A), 이때 $V=P \cup T$이고 $\{v_1, v_2, \cdots, v_n\}$이며 $P \cap T=\phi$이다. 또한 A는 간선의 집합 $A=\{a_1, a_2, \cdots, a_n\}$이며 모든 $p_i \in P$, $t_j \in T$에 대해 P_i에서 t_j로의 입력함수의 수가 $\#((p_i, t_j), A)$가 되고 t_i에서 P_j로의 출력함수의 수가 $\#((t_i, p_j), A)$가 된다. 또한 입력함수의 결함율 $f(p_i, I(t_j))$는 그 간선의 가중치가 되고 출력함수의 결함율 $f(O(t_i), P_j)$ 또한 그 간선의 가중치이다.

단계1 : 플레이스와 트랜지션을 결정한다.
이때 플레이스 $P = M \cup S \cup \{P_s, P_f\}$이고 P_s는 시작 플레이스, P_f는 종료 플레이스이다. 또한 트랜지션은 다음과 같은 과정을 거쳐 결정한다.
1) for i = 1 to k (k는 모듈간의 매핑의 수)에 대해 t_i를 생성한다
2) P_s와 P_f에 대해 t_s와 t_f를 결정한다

단계2 : 입력함수와 출력함수를 결정한다. 입력함수의 결정 과정은 다음과 같다.
1) for i = 1 to k (k는 모듈간의 매핑의 수)
$t_j \in s$에서 $I(t_i)$를 결정한다.
2) 블록그래프의 각 모듈에서 모듈로의 플레이스에서 입력 간선이 존재하는 것에 대해 단계1에서 만든 모듈의 플레이스에서 해당 매핑의 트랜지션으로 입력함수를 결정한다. 다음으로 블록그래프의 모듈에서 모듈로 출력 간선이 있는 것에 대해 해당 매핑의 트랜지션의 단계1에서 생성한 모듈의 플레이스로 출력함수를 결정한다.

단계3 : 입력으로 받은 모듈 및 신호선의 결함율 $f(P_i, I(t_j))$를 해당 트랜지션의 입력 결함율로 결정한다.

$F(I) = \{ \forall i, \forall j, f(p_i, I(t_j))\}$

단계4 : 입력으로 받은 모듈 및 신호선의 결함(O)를 결정한다.

$$F(O)=\{\forall i, \forall j, f(O(t_j), P_i) \mid f(O(t_j), P_i)=1-\prod_{k=1}^{n}(1-f(P_k, I(t_j)))\}$$

단계5 : 토큰의 초기치를 결정한다.
단계6 : 단계1부터 단계4에서 생성된 FPn의 구조를 이용하여 FPn 그래프 G=(V, A)로 변환한다.

그림 2는 그림 1이 위의 알고리즘에 의해 변환된 FPn그래프이다. 여기에서 P={ P_s, P_f, load, Ezero, down, zero, load, down, clear, count, control, dividend, divisor, Quotient } 이고 T={ t_s, t_f, t1, t2, ···, t8 }이다. 이때 그림 3의 control 모듈에서 Dividend 모듈로의 점화를 위해서는 시작 플레이스인 P_s에서의 토큰과 load 토큰이 모두 만족되어야만 하므로, 초기 토큰을 가진 플레이스는 P_s, load, Ezero, down, zero, load, down, clear, count이다.

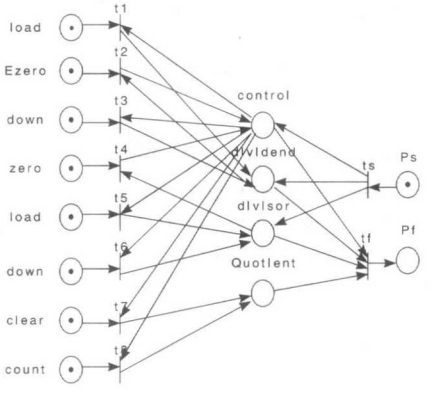


그림 2. FPn 그래프

Ⅳ. 시스템의 분석과 영향도 계산

FPn로 모델링된 시스템은 시스템의 동적인 분석을 위하여 도달성 분석 트리로 구성이 되며[9] 이를 이용하여 수행 경로를 생성하며, 각 모듈이 시스템에 미치는 신뢰도를 계산하게 된다.

1. 도달성 분석 트리의 생성

FPn의 트랜지션의 실행 경로를 분석하기 위하여 도달성 분석 트리를 이용한다. 도달성 분석 트리의 생성은 먼저 FPn에서 시작트랜지션을 도달성 분석 트리의 루트 노드로 설정한다. 다음으로 입력되는 노드가 플레이스이면 버리고, 트랜지션에 대해 다음의 과정을 계속해서 수행한다. 시작플레이스의 트랜지션의 출력함수의 플레이스를 입력함수로 갖는 트랜지션은 루터 노드의 자식 노드이며 이때 생성되는 자식 노드들은 루트 노드에서 같은 레벨에 있으므로 형제노드가 된다. 또한 완전 트랜지션이 아닌 노드에 대해 그 트랜지션의 출력함수의 플레이스를 입력함수로 갖는 트랜지션은 그 노드의 자식 노드가 된다. 이전 과정에서 생성된 노드가 종료 트랜지션인 경우는 그 노드의 자식 노드의 생성은 완료되고, 생성한 노드 중 완전 트랜지션이 아닌 노드가 존재할 경우에는 이전 과정에서 생성된 노드 중 완전 트랜지션이 아닌 노드가 존재할 경우 그 노드의 상위 노드들 중에서 그 노드와 같은 노드가 있으면 그 경로의 수행을 종료한다. 마지막으로 생성된 노드들이 모두 완전트랜지션이거나 그 상위 노드들 중에서 같은 노드가 있는 경우에는 수행을 종료하고 아니면 알고리즘을 계속해서 수행하게 된다. 이때 완전 플레이스란 어떤 플레이스에서 전이가 일어나는 모든 트랜지션들이 종료 트랜지션이거나 그 트랜지션의 모든 입력함수들이 점화된 트랜지션일 경우이며, 형제 플레이스란 시작 트랜지션에서 전이가 일어날 경우에 트랜지션의 깊이(depth)가 같은 플레이스를 말한다. 여기에서 깊이가 같다는 것은 루트 노드로부터 해당 노드까지의 간선의 수가 같다는 것이다.

알고리즘 2는 FPn을 도달성 분석 트리로 만드는 과정을 슈도우코드(pseudo-code)로 나타낸 것이다.

```
[알고리즘 2] 도달성 분석 트리의 생성
input=EPG
output=Rtree
node()={a set of places, a set of transitions}
algorithm make_Rtree()
{
  if(node(i)=place)
        discard(node(i));
  else if(node(i)=transition)
  {
    if(node(i)=start transition){
      Rtree(root_node)=start transition of EPG;
      }
      if(input_function of transition is child node
of
        Rtree(root_node)){
          node(i) is child node of root_node;
          s3:flag=1;
          while(flag){
            if(node(i) is not complete transition){
              if(input function of node(i) is
output
                function of parent node){
                  node(i) is child node of
parent node;
              } /* end of if */
              else if(node(i)=final node){
                complete make_node of this
path;
                  flag=0;
              } /* end of else if */
```

```
                else if(there is not_complete node)
                        flag=0;
        } /* end of while */
    if((generate_node is complete node)||(there is
civiling node(i)){
                exit();
        } /* end of if */
    else goto s3;
  } /* end of else if */
}  /* end of algorithm */
```

그림 3은 그림 2의 FPn을 위의 알고리즘에 따라 도달성 분석 트리로 구성한 것이다.

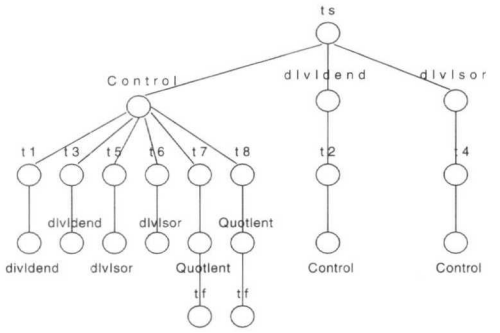


그림 3. 도달성 분석 트리

2. 수행 경로의 생성

알고리즘 3은 도달성 분석 트리를 이용하여 수행 경로를 생성하는 과정이다.

[정의 6] 도달성 분석 트리에서 싸이클의 존재
도달성 분석 트리의 서로 다른 두 플레이스 노드 pi, pj에서 $p_i \rightarrow t_i \rightarrow \cdots \rightarrow p_j$의 수행 경로와 $p_j \rightarrow t_j \rightarrow \cdots \rightarrow p_i$의 수행 경로가 있을 경우 싸이클이 존재한다.

[알고리즘 3] 도달성 분석 트리를 사용한 수행 경로의 생성
input : Rtree
output : a set of exe_path

단계1 : 종료노드가 종료트랜지션(t_f)인 경우는 루트노드에서 그 종료 노드까지가 수행 경로이다.
단계2 : 플레이스 노드가 종료 노드가 되는 경우 :

1. 자식의 자식 노드와 자신의 노드가 같은 경우 : 그 사이에 존재하는 트랜지션 노드를 자신의 노드가 다른 수행 경로를 구성할 때에 삽입하여 새로운 수행 경로를 만든다.
2. 자식의 자식 노드와 자신이 서로 다른 플레이스 노드지만 형제 플레이스인 경우 :
1) 싸이클이 없는 경우 : 서로 다른 두 플레이스 p_i, p_j에서 $p_i \rightarrow p_j$로의 관계가 있는 경우 p_j 밑의 모든 수행경로를 p_i밑의 수행 경로로 간주한다.
2) 싸이클이 있을 경우 : 서로 다른 두 플레이스 노드 p_i, p_j가 순환관계가 존재할 경우에는 p_i밑의 모든 수행 경로에 p_i플레이스의 자식인 트랜지션 노드t_i와 순환되는 플레이스 노드인 p_j플레이스 노드 대신에 이것의 자식노드인 t_j를 삽입하여 새로운 수행 경로를 만들고 p_j 밑의 모든 수행 경로에 대해서도 같은 방법으로 새로운 수행 경로를 만든다.

표 1은 그림 3의 도달성 분석 트리에서 루트 노드에서 단말 노드까지의 수행 경로이며 표 2는 알고리즘 3을 적용하여 수행경로와 트랜지션, 그리고 점화확률을 분석한 것이다.

표 1. 단말 노드까지의 수행경로

경로번호	수행경로
1	ts t7 tf
2	ts t8 tf
3	ts t1 dividend
4	ts t3 dividend
5	ts t5 divisor
6	ts t6 divisor
7	ts t2 control
8	ts t4 control

3. 시스템의 신뢰도에 미치는 영향 분석

신뢰도는 수행 경로상의 모든 트랜지션이 점화되어야 때문에 각 트랜지션의 결함율이 f_1, f_2,…, f_k일 때 다음과 같이 점화확률 T_i를 다음과 같이 나타낼 수 있다.

$$\text{점화확률} \quad T_i = \prod_{k=1}^{n} (1 - f_k)$$

표 2에서의 점화확률은 이를 이용하여 그림 3의 블록그래프에서 control모듈의 결함율이 2×10^{-3}이고 dividend모듈에서의 결함율이 3×10^{-3}, divisor모듈의 결함율이 1×10^{-3}, Quotient의 결함율이 4×10^{-3} 이라 가정하고 또한 각 신호선 간의 매핑관계의 결함율이 1×10^{-4}이라 가정하여 계산한 것이다.

표 3에서는 표 2에서 계산한 점화확률과 각 트랜지션의 빈도수를 고려해 모듈의 신뢰도를 계산한 것이다. 또한 표 3에서 평균 점화확률이란 표 2의 수행 경로 상에서 트랜지션 t_i가 생성된 경로의 점화확률의 평균값으로 다음과 같다.

평균 점화확률 $Pm(t_i) = \left(\sum_{j=1}^{m} T_j \right) / k$

(k는 트랜지션 t_i의 전체 빈도수)

모듈의 신뢰도는 수행경로의 점화확률이 낮을수록 시스템에 미치는 영향이 많으므로 점화확률과 신뢰도는 반비례한다. 따라서 발생 빈도수 k를 고려한 신뢰도 $Ef(t_i)$는 다음과 같다.

신뢰도 $Ef(t_i) = \prod_{j=1}^{k} (1 / Pm(t_j))$

표 2. 수행경로와 점화확률

경로번호	확장된 수행경로	점화확률
1	t_s t_7 t_f	0.9939
2	t_s t_8 t_f	0.9939
3	t_s t_1 t_2 t_f	0.9948
4	t_s t_3 t_2 t_f	0.9948
5	t_s t_5 t_4 t_f	0.9968
6	t_s t_6 t_4 t_f	0.9968
7.1	t_s t_2 t_1 t_f	0.9948
7.2	t_s t_2 t_3 t_f	0.9948
7.3	t_s t_2 t_5 t_4 t_f	0.9937
7.4	t_s t_2 t_6 t_4 t_f	0.9937
7.5	t_s t_2 t_7 t_f	0.9908
7.6	t_s t_2 t_8 t_f	0.9908
8.1	t_s t_4 t_1 t_f	0.9968
8.2	t_s t_4 t_3 t_f	0.9968
8.3	t_s t_4 t_5 t_f	0.9968
8.4	t_s t_4 t_6 t_f	0.9968
8.5	t_s t_4 t_7 t_f	0.9928
8.6	t_5 t_4 t_8 tf	0.9928

표 3. 신뢰도

트랜지션	평균점화확률	빈도수	신뢰도
t_1	0.9955	3	1.0136
t_2	0.9935	8	1.0536
t_3	0.9955	3	1.0136
t_4	0.9954	10	1.0472
t_5	0.9958	3	1.0127
t_6	0.9958	3	1.0127
t_7	0.9925	3	1.0228
t_8	0.9925	3	1.0228

표 3에서 시스템에 영향을 가장 많이 미치는 트랜지션은 t_2이며 이 트랜지션은 그림 3의 블록그래프에서 $M_{Dividend} = R(M_{Control}, S_{EZERO})$이다. 또한 이 관계는 그림 1에서 Dividend 모듈에서 Control 모듈로 매핑관계이다. 여기에서 Dividend 모듈이 Control 모듈에 영향을 미치게 되므로 Divisend 모듈이 전체 시스템에서 가장 영향을 많이 미치는 모듈이고 그때의 매핑관계는 $M_{Dividend} = R(M_{Control}, S_{EZERO})$이다.

V. 결 론

향후 시스템의 테스트는 더욱 중요한 문제로 부각될 것이고 이러한 시스템의 테스트는 효율적으로 수행되어야만 비용과 노력의 절감을 가져올 수 있다. 이에 따라 시스템의 설계 시 시스템 동작 특성을 고려한 분석을 통하여 각 모듈이 전체 시스템에 미치는 신뢰도를 분석할 수 있다면, 이것의 분석 결과를 이용하여 전체 시스템에 대해 효율적이고 적절한 비용을 가지는 테스트 전략을 세워 테스트의 효율을 증대시킬 수 있을 것이다.

본 논문에서는 모듈별로 구성된 시스템에 대해 각 모듈의 동작 특성을 분석하고, 시스템의 각 모듈이 전체 시스템의 신뢰성에 미치는 신뢰도를 규명할 수 있는 기법을 제안하였고 예제 모델을 통해 분석하였다. 본 논문에서 사용되는 정형화 도구는 블록 그래프와 패트리네트를 확장한 결함율 패트리네트이며 분석방법으로는 도달성 분석 트리를 이용하였다.

향후 실제 시스템의 테스트와 연계한 설계 및 분석, 이에 따른 적절한 테스트를 통해 테스트 커버리지(Coverage) 및 비용과의 Trade-off 관계가 규명되어야할 것이다. 또한 테스트의 효율을 높이기 위한 명확한 모듈의 분리에 대한 연구가 필요하며 이를 위해서 객체 기반의 모듈 구성이나 컴포넌트 기반의 모듈 구성에 대한 연구가 필요하다.

참 고 문 헌

[1] S. Cataldo, S. Chiusano, P. Prinetto, "Optimal Hardware Pattern Generation for Functional BIST," Design, Automation and Test in Europe, March 2000.

[2] S. Stoica, "Generating Functional Design Verification Tests," IEEE Design & Test of Computer, Vol.16, no.3, pp.53-63, July-September 1999.

[3] M. Jacomet, R. Walti, L. Winzenried, J. Perez, M. Gysel, "ProTset: A Low Cost Rapid Prototyping and Test System for ASICs and FPGAs," Proceeding of the 1997 International

Conference on Microelectronics Systems Education (MSE'97), 1997.

[4] S. Cataldo, S. Chiusano, P. Prinetto, "Optimal Hardware Pattern Generation for Functional BIST," Design & Automation and Test in Europe, 27-30 March 2000.

[5] A. Speck, "Component-Based Control System," Proceedings of the 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2000.

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

[7] T. Murata. Petri Nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541-580, April 1989.

[8] P. Senac, P. de Saqui-Sannes, R. Willrich, "Hierarchical Time Stream Petri Net: A Model for Hypermedia Systems," Application and Theory of Petri Nets 1995, pp.451-470, June 1995.

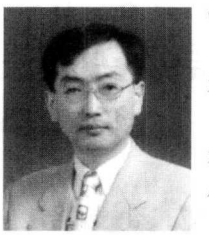
[9] M. Notomi and T. Murata, "Hierarchical Reachibility Graph of Bounded Petri Nets for Concurrent Software Analysis," *IEEE Transaction on Software Engineering*, Vol.20, No.5, pp.325-336, 1994.

나 윤 지(Yun Ji Na)

충북대학교컴퓨터공학(박사과정)
충북대학교컴퓨터공학(공학석사)
뉴욕공과대학교(NYIT)
Communication ART 전공
석사과정 2학기 수료
경북대학교졸업(이학사)

현)대전보건대학 컴퓨터정보처리과 초빙교수
<주관심 분야> 멀티미디어콘텐츠공학, 전자상거래시스템, 에이전트기반시스템, 원격교육, 컴퓨터그래픽스

고 일 석(Il Seok Ko)



연세대학교 컴퓨터산업시스템공학(박사수료)
경북대학교 컴퓨터공학
(공학석사)
경북대학교 전산전공(공학사)
성균관대학교경영대학원
앤소프전략스쿨 수료
미) USIU College of Business
Administration 졸업(MBA)
현) 충북과학대학 전자상거래과 교수
<주관심 분야> 전자상거래시스템, 에이전트기반시스템, 멀티미디어콘텐츠공학, 원격교육, 기술경영

조 용 환(Yong Hwan Cho)

고려대학교대학원 이학박사
(1989)
충북대학교 전기전자 및 컴퓨터공학부 교수(1982-현재)
현) 사단법인 한국콘텐츠학회
회장
<주관심 분야> 무선 인터넷, 멀티미디어통신, 트래픽공학, ATM, 정보통신정책