

# 다중 측면 기반의 분산 데이터 할당을 위한 MAPN 모델링

박 성 진<sup>†</sup>

## 요 약

분산 데이터베이스 설계에서 데이터의 할당은 중요한 설계 이슈이다. 그러나, DAP(Data Allocation Problem)에 관한 대부분의 기존 연구들은 오직 비용 측면만을 고려함으로써 성능과 가용성의 향상을 가져올 수 없고 높은 가용성을 요구하거나 실시간 처리를 필요로 하는 시스템들의 경우 적합하지 않다. 따라서, 보다 정형화되고 다중 측면을 동시에 표현할 수 있는 데이터 할당 모델이 요구된다.

본 논문에서는 분산 트랜잭션 모델링을 위한 MAPN 모델링 방법을 제안하였다. 확장된 페트리 넷 모델인 MAPN(Multiple Aspects Petri Net) 모델은 다중 측면(비용, 성능 및 가용성)을 동시에 고려하는 정형화된 모델링을 위해 제안되었다. 제안된 MAPN 구조와 그래프를 사용함으로써 비용뿐만 아니라 성능, 가용성 측면을 동시에 고려하는 타당한 DAP 평가 모델을 구성할 수 있다.

## The MAPN Modeling for the Distributed Data Allocation based on Multiple Aspects

Seong-Jin Park<sup>†</sup>

### ABSTRACT

In designing distributed databases, DAP (Data Allocation Problem) is one of the key design issues. Because, however, most of previous researches on DAP have considered only cost aspect, they cannot increase the performance and availability and they are not proper to the systems requiring high-availability or real-time processing. Therefore, we need a more formal data allocation model considering multiple aspects.

In this paper, we propose the MAPN (Multiple Aspects Petri Net) modeling method for the distributed transaction modeling. The MAPN model, an extended classical petri net, is proposed for the formal modeling considering multiple aspects (cost, performance and availability) concurrently. We demonstrate that we can compose the valid DAP evaluation model considering not only cost but also performance and availability concurrently by using the MAPN structure and MAPN graph.

### 1. 서 론

분산 데이터베이스 시스템으로의 변환은 비용(cost)

감소, 성능(performance) 향상, 데이터의 가용성(availability) 및 신뢰성(reliability) 증가 등의 이점을 갖는다. 이러한 이점들은 효율적인 데이터의 할당(allocation)을 통해서 가능하다[1, 2]. 한편, 기존의 데이터 할당 연구 대부분은 비용 또는 성능 중의 한 측면만을 고려하고

<sup>†</sup> 정 회 원 : 한신대학교 정보처리학과 조교수  
논문접수 : 1999년 2월 18일, 심사완료 : 2000년 1월 12일

있다[3-6]. 이것은 둘 이상의 측면들을 고려할 경우, 모델링이 복잡하며 모델링이 가능하더라도 그에 따른 최적화가 어렵기 때문이다. 그러나, 데이터의 할당 결과가 분산 데이터베이스의 다양한 이점들을 충족시킬 수 있는 최적의 형태로 결정되어야한다는 점을 고려한다면 문제점으로 지적될 수 있다[1].

따라서, 분산 데이터베이스의 이점을 충분히 살릴 수 있도록 데이터를 할당하기 위해서는 기존의 비용 측면과 성능 측면을 동시에 고려해야한다. 또한, 적정 수준의 데이터 중복을 결정하기위한 가용성 측면 역시 함께 고려될 필요성이 있다.

본 논문에서는 비용, 성능 그리고 가용성 등의 다양한 측면을 동시에 고려할 수 있는 새로운 데이터 할당을 위해 MAPN(Multiple Aspects Petri Net) 모델링 방법을 제안함으로써 다양한 환경의 분산 데이터베이스 할당을 위한 형식론적 모델링 방법을 제안하였다. 제안한 MAPN 모델링을 통해서 비용뿐만 아니라 성능, 가용성 측면이 중시되는 실시간 시스템 또는 고장-감내(fault-tolerant) 시스템 환경을 위한 다중측면을 고려하는 데이터 할당 모델의 구성이 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 데이터 할당 연구들에 관한 문제점을 고찰함으로써 다중측면을 고려하는 데이터 할당 모델의 필요성을 지적하였다. 3장에서는 고전적인 페트리 네트를 확장한 MAPN 모델을 정의하고 MAPN이 필요로하는 다중측면 표현 능력을 갖음을 보였다. 4장에서는 제안한 MAPN 모델링 방법을 데이터 할당 문제에 적용, 분석한 예를 보였으며 마지막으로 결론을 5장에서 기술하였다.

## 2. 데이터 할당 연구

분산 데이터베이스에서의 데이터 할당 문제(Data Allocation Problem)는 시스템 전체의 비용, 성능 그리고 가용성에 크게 영향을 미치는 매우 중요한 연구 분야이다. 이 문제에 대한 연구는 분산 파일 시스템에서의 파일 할당 문제(FAP : File Allocation Problem)에 대한 연구 결과들을 토대로 분산 데이터베이스 환경에 맞게 확장하려는 방향으로 진행되어왔다[3-7]. 기존의 데이터 할당 방법들을 평가한 결과 다음과 같은 문제점들을 분석할 수 있다. 첫째, 구성된 데이터 할당 모델에 대한 타당성 검증이 어렵다는 점이다[12, 13]. 둘

째, 둘 이상의 측면들을 동시에 최적화할 수 있는 통합 모델과 그에 따른 최적화 기법이 제시되지 않았다는 점이다.

일반적으로 시스템에 대한 분석 평가 방법의 선택은 시스템의 행위 예측, 시스템 동작의 이해 및 시스템 성능 평가 등의 분석 목적, 시스템의 특성 및 분석가의 개인 능력에 따라 다르다. 비용, 시간 그리고 가용성 측면에서 분산 트랜잭션의 처리 과정을 분석, 평가하는 데 있어 가장 정확한 방법은 시뮬레이션을 통해 분석하는 방법이다[11]. 시뮬레이션 방법은 동적인 시스템 분석을 위한 효과적인 접근 방법으로 다양한 상황에 대한 할당 결과를 분석할 수 있는 이점을 갖는다[10].

기존 연구에서는 DAP 평가 모델을 수학적 모델로 제시하고 있다. 수학적 모델은 분석 시간이 적고 제한된 영역 안에서 정확한 해답을 구할 수 있는 장점을 갖는다. 그러나, 모델 구성 과정에서 많은 가정들을 설정해야하므로 DAP 평가 모델을 구성하는데 있어 단순한 통신 비용이 아닌 동시성 제어(concurrency control) 기법과 같은 복잡한 분산 트랜잭션 처리 과정을 고려하는 경우, 수학적 모델은 불완전한 모델이 구축될 수 있는 문제점을 갖고 있다[12]. 특히, 단일 측면이 아닌 다중 측면을 동시에 고려하는 경우 일관적이지 못한 모델링이 될 가능성이 높다[13].

한편, 상태/사건(state/event) 모델로 간주되는 페트리 네트(petri net)는 병행적 또는 병렬적 프로세스들의 특성을 연구하기 위해 고안된 도식적, 수학적 모델링 도구이다[9, 10]. 도식적 도구로서의 페트리 네트는 순서도(flowchart)나 블록 다이어그램(block diagram)과 같이 시각적으로 의사 소통을 가능하게 해주며 모델의 해석적인 분석을 가능하게 해준다. 본 연구에서는 형식적인 모델링 방법으로 고전적인 페트리 네트(classical Petri Net)를 다중 측면을 표현할 수 있도록 확장한 MAPN(Multiple Aspects Petri Net)을 제안하였다. 그 이유는 페트리 네트가 도식적 모델이어서 이해가 용이하며 동시에 명확한 형식론에 기초한 수학적 모델이어서 모호성이 배제된 트랜잭션 명세에 적합하기 때문이다[10, 12]. 특히, 트랜잭션 명세에는 트랜잭션의 병행적(concurrent), 비결정적(nondeterministic), 병렬적(parallel), 분산적(distributed) 특성을 기술할 수 있어야 하는데 페트리 네트는 이러한 특성을 잘 표현할 수 있는 형식 모델이다.

그러나, 분산 트랜잭션 처리 과정을 모델링하기 위

하여 페트리 넷을 사용할 경우 다음과 같은 문제점을 갖는다. 첫째, 트랜잭션 처리에 따른 비용, 시간, 가용성 등과 같은 측면값들을 표현할 수 없다. 둘째, 사건이 발생하기 위해서 모든 사전조건이 만족해야만 하는 AND 논리만으로는 읽기/쓰기 트랜잭션 모델링이 어렵다. 이러한 문제점을 해결하기 위해 제안된 MAPN은 고전적인 페트리 넷로부터 크게 다음 3가지 즉, 1) 측면 토큰 정의, 2) 사건 정의, 3) 측면 마킹에 의한 상태 전이 표현을 확장하였다.

제안된 MAPN 모델은 데이터 할당 모델의 모델링 과정에서의 모호성, 불완전성을 배제하고 분산 트랜잭션 처리에 대한 다양한 측면에서의 체계적인 분석을 위한 평가 모델을 구성할 수 있도록 해준다.

### 3. MAPN 모델링

고전적인 페트리 넷 모델은 제어 흐름만을 표현하기 때문에 분석하고자 하는 측면에 대한 표현이 어렵다. 기존의 고전적인 페트리 넷 모델은 다중 측면값을 다루지 않으며 측면값의 변화를 표현하는 기법이 없기 때문에 측면 표현능력이 없는 모델이다[10]. 따라서, 분산 트랜잭션처리 과정을 명세하고 다양한 측면에서 분석하기 위해서는 페트리 넷 모델에 다중측면 표현능력을 부여할 필요가 있다. 즉, 비용, 시간 및 가용성 등이 각각 하나의 측면으로 정의될 수 있어야 한다. 먼저, 다중 측면 표현 능력을 갖는 모델을 정의하면 다음과 같다.

#### 【정의 1】 모델의 다중측면 표현능력

모델 M이 다음 특성을 만족할 때 다중측면 표현능력(multiple aspects expressability)을 갖는다고 한다.

- ① 다중 측면의 표현 : 임의의 시점에서 시스템의 상태를 다양한 측면 관점에서 그 값을 표현할 수 있다.
- ② 측면 전이함수의 존재 : 시스템의 상태 전이를 다양한 측면에서 서술할 수 있는 방법이 모델 M의 형식론에 포함되어 있다.

기존의 페트리 넷 모델에 측면 표현 능력을 부여하기 위해 측면 토큰(aspect token)과 사건(event) 개념을 추가하여 확장한 MAPN은 다음과 같이 정의된다.

#### 【정의 2】 MAPN

다중측면 페트리 넷의 구조 MAPN은 다음과 같이 위치(place)의 유한 집합 P, 전이(transition)의 유한

집합 T, 입력함수 I, 출력함수 O, 사건(event)의 유한 집합 E, 측면(aspect) 튜플들의 도메인 A, 초기 측면 마킹  $m_0$ 로 구성된다.

MAPN = ( P, T, I, O, E, A,  $m_0$  )  
 여기서,  
 $P = \{p_1, p_2, \dots, p_k\}, k \geq 1$   
 $T = \{t_1, t_2, \dots, t_i\}, i \geq 1, P \cap T = \emptyset$   
 $I : T \rightarrow P^\infty, I(t_i)$ 는  $t_i$ 의 입력 위치들의 집합  
 $O : T \rightarrow P^\infty, O(t_i)$ 는  $t_i$ 의 출력 위치들의 집합  
 $E = \{e_{11}, \dots, e_{ij}\}, e_{ij}$ 는  $t_i$ 의 j번째 이벤트  
 $A = A_1 \times A_2 \times \dots \times A_n$   
 $(n \geq 2, A_1$ 는 위치정보,  
 $A_2, A_3, \dots, A_n$ 는 각 측면값들의 유한 집합)  
 $m_0$  : 초기 측면 마킹  
 단,  $P^\infty$ 은 위치의 백(bag)[10]

위에서  $m_0$ 와 E, A는 다중 측면을 고려하기 위하여 고전적인 페트리 넷에 새로 추가된 요소들이다. MAPN 형식론에서는 분석하고자 하는 다양한 측면들을 각각 측면  $A_2, A_3, \dots, A_n$ 으로 정의하고 다음과 같은 측면 토큰을 사용하여 표현한다.

#### ● 측면 토큰(Aspect token)

측면 토큰은 정의된 각 측면에 대한 특정 값을 가질 수 있는 MAPN에서 새롭게 제시되는 토큰으로 기존 페트리 넷의 토큰을 확장한 것이다. 각 측면은 고려하고자 하는 척도값의 변화를 분석할 목적으로 측면값을 위해 정의되는 것 과 각 토큰과 관련된 데이터를 위해 정의되는 것 두 형태가 있다. 이러한 측면 값들은 전이가 점화될 때 사건 규칙에 의해서 변경된다.

먼저, 측면 토큰  $k(k \in A)$ 는 측면들의 튜플  $\langle a_1, a_2, \dots, a_n \rangle$ 으로 표현하며 이때,  $a_1(a_1 \in P)$ 은 항상 토큰이 위치한 위치를 나타내고 나머지  $a_2, \dots, a_n(a_2 \in A_2, \dots, a_n \in A_n)$ 은 정의된 측면들에 따른 토큰의 측면값들을 나타낸다.

측면 토큰은 특정 위치에 존재하면서 제어 토큰과는 달리 측면 토큰의 갯수와 측면값이 전이의 발생 조건이 된다. 따라서, 적정 수의 측면 토큰이 조건에 부합되면 사건이 발생되어 전이를 점화시키고 측면 토큰은 사건 규칙에 의해 다른 위치로 이동된다.

측면 토큰은 전체적인 처리 과정에 따른 각 측면값들의 변화를 표현하기 위해 필요한 토큰이다. 고전적인 페트리 넷에서 토큰은 주로 제어 마킹을 위해 사용됨으로써 개별적으로 식별되기보다는 오직 그 갯수

만이 의미가 있었다. 그러나, 동일한 상태 전이에 대한 다양한 관점에서의 분석을 위해서 그리고, 보다 복잡한 처리 과정을 모델링하기 위해서는 시스템에 대한 정보들과 분석 측면들을 토큰을 통해 표현해야 한다. 예를 들면, 트랜잭션이 발생한 노드 및 타입이나 각 트랜잭션에 대한 수행 비용, 성능, 가용성 등의 측면값 등이 측면 토큰을 통해 표현된다.

● 제어 토큰(control token)

제어 토큰은 고전적인 페트리 넷에서의 토큰과 같은 특성을 갖는다. 즉, 측면값을 갖지 않으며 오직 토큰의 위치만으로 표현된다. 제어 토큰은 중앙처리장치(CPU)나 디스크와 같은 자원의 가용성을 표현하는 등의 전체적인 처리 흐름을 제어하기 위해 사용된다. 따라서, 제어 토큰은 토큰의 위치와 토큰의 갯수만이 중요한 의미를 갖는다.

MAPN에서 동적 행위에 대한 명세는 초기 측면 마킹으로부터 시작해서 연속되는 측면 마킹에 의해 상태 변화를 표현함으로써 이루어진다. MAPN에서 특정한 위치의 토큰 갯수와 토큰 값들의 상태를 측면 마킹(aspect marking)이라고 부른다. 형식적으로 서술하면  $C = (P, T, I, O, E, A, m_0)$ 에서 측면 마킹  $m$ 은 다음과 같이 정의되는 함수이다.

【정의 3】 측면 마킹(Aspect marking)

MAPN의 위치들에 대한 측면 마킹  $m$ 은 다음의 함수로 정의한다.

$$m : P \rightarrow A^\infty$$

단,  $A^\infty$ 은 측면 도메인의 백(bag)

임의의 위치  $p_i$ 의 측면 마킹  $m(p_i)$ 는 위치  $p_i$ 에 존재하는 토큰의 각 측면값들이다. MAPN에서의 측면 마킹은 특정한 위치의 토큰들의 갯수와 각 토큰들의 측면 값들을 표시한다. 예를 들면, 위치 'busy'의 측면 마킹은  $m(\text{busy}) = 2\langle \text{busy}, 2, 0.5 \rangle + \langle \text{busy}, 4, 1.5 \rangle$ 로 표시할 수 있다. 이것은 위치 'busy'에 3개의 토큰이 존재함을 나타낸다. 또한, 3개의 토큰중 2개의 토큰이 동일한 측면값(두번째, 세번째 측면 값으로 각각 2와 0.5)을 가지며 세번째 토큰은 두번째, 세번째 측면 값으로 각각 4와 1.5를 갖음을 의미한다.

백(bag)은 다중집합(multiset)이라고도 부르며 일반 집합과는 달리 임의의 집합에서 동일한 원소(element)의 반복 발생을 허용하는 부분집합을 말한다[10]. 측면

도메인 집합  $A$ 에 대한 백은 다음과 같이 정의한다.

【정의 4】 측면 도메인의 백(bag)( $A^\infty$ )

먼저, 측면 도메인 집합  $A$ 로부터  $N$ 으로의 다음과 같은 함수  $b$ 를 가정한다.

$$b : A \rightarrow N$$

이때,  $N$ 은 자연수 집합을, 함수  $b$ 는 측면 튜플들중에서 동일한 튜플들의 갯수를 말한다.

따라서, 측면 도메인의 집합  $A$ 에 대한 백  $A^\infty$ 은 집합  $A$ 의 원소  $a$ (측면 튜플)와 동일한 원소  $a$ 의 갯수의 곱들에 대한 합으로 다음과 같이 정의한다.

$$A^\infty = \sum_{a \in A} b(a)a$$

한편, MAPN이 실행되는 동안 각 측면 토큰의 위치와 갯수, 측면값들이 바뀌게 되는데 이는 곧 MAPN 모델에서의 상태(state) 변화로 간주된다. 한편,  $A^\infty$ 은  $n$ 개의 위치를 가진 상태 공간(state space)으로 모든 측면 마킹의 집합을 의미할 수 있다. 따라서, 측면 마킹  $m$ 은 임의의 시점에서의 상태를 나타내며, 집합  $P$ 의 원소 갯수  $|P| = n$ 이고 각각의  $m_i \in A^\infty$ 일때  $m = (m_1 + m_2 + \dots + m_n)$ 으로 표현된다.

예를 들면, 'free', 'lock', 'busy' 3개의 위치만이 존재한다고 할때 임의의 상태는 측면 마킹  $m$ 에 의해 다음과 같이 표현한다.

$$m = ( 2\langle \text{free}, 0, 0 \rangle + \langle \text{lock}, 1, 0.2 \rangle + \langle \text{busy}, 4, 1.3 \rangle )$$

'free'                      'lock'                      'busy'

위의 경우 위치 'free'에는 2개의 토큰이 존재하며 각 측면값이 동일하다는 것을 알 수 있다. 위치 'lock'와 'busy'에는 각각 1개씩의 토큰이 존재하며 각기 다른 측면값을 가지고 있다.

위와 같이 MAPN은 측면 마킹  $m$ 에 의해서 모든 위치들의 측면값들을 나타냄으로써 임의의 상태를 표현한다. 따라서, 임의의 시스템 상태가 측면 마킹 벡터로 표현되는 MAPN 모델은 다중측면 표현능력을 갖기 위한 【정의 1】의 조건 ①을 만족하게 된다. ....(1)

한편, 고전적인 페트리 넷에서는 전이를 점화시키는 입력 상태와 출력 상태가 하나로 고정되어 있다. 즉, 오직 하나의 입력 상태에 대해서만 전이가 점화되고 그 결과 오직 하나의 상태로만 전이된다. 그러나, 분산 트랜잭션을 모델링하기 위한 MAPN에서는 하나

의 전이에 대해서 다양한 입출력 관계를 표현할 수 있어야 한다. 이를 위해서, MAPN에서는 사건 개념을 도입하였다. 사건은 임의의 전이 점화를 발생시키는 특정한 입출력 관계로 정의한다. 따라서, 하나의 전이를 점화시키는 복수개의 사건이 존재할 수 있다. 이것은 고전적인 페트리 네트에서의 전이에 대한 모델링 능력을 향상시킨 것이다.

**【정의 5】 사건(event)**

특정한 전이  $t$ 에 대한 모든 가능한 점화를 나타내는 사건  $e_{ij}$ 는 다음과 같이 점화시키는 전이  $t_i$ , 사전 조건  $e_{in}$ , 사후 조건  $e_{out}$ 으로 정의한다.

$$e_{ij} = \langle t_i, e_{in}, e_{out} \rangle$$

여기서,

- $t_i$  : 사건  $e_{ij}$ 가 점화시키는 전이(transition),  $t_i \in T$
- $e_{in}$  : 사전 조건(precondition),  $e_{in} \subset A^\infty$
- $e_{out}$  : 사후 조건(postcondition),  $e_{out} \subset A^\infty$

단, 사전 조건  $e_{ij}$ 는 특정 측면값으로 '\*'를 가질 수 있으며 이때 '\*'는 해당 측면값이 모든 사전 조건을 만족함을 의미한다.

따라서, 사건  $e_{ij}$ 는 튜플  $\langle t_i, e_{in}, e_{out} \rangle$ 으로 표현하며  $i$ 번째 전이에 관한  $j$ 번째 사건을 의미한다.  $e_{in}$ 은 사건  $e_{ij}$ 가 발생하기 위해 만족되어야할 사전 조건 집합을 표시하며  $e_{out}$ 은 사건  $e_{ij}$ 발생후의 사후 조건 집합을 나타낸다. 따라서,  $e_{in}$ 은  $t_i$ 가 점화되기 위한 토큰들에 대한 조건을 나타내며 점화후에 상태 집합에서 소멸된다. 반면에,  $e_{out}$ 은  $t_i$ 가 점화된 후에 새로 생성되는 토큰들의 상태를 나타내며 상태 집합에 새로 추가된다. 이때,  $t_i$ 를 점화시키는  $e_{in}$ 에 기술된 모든 위치들은 【정의 2】에서 정의된  $I(t_i)$  집합에 속해야하고  $e_{out}$ 에 기술된 모든 위치들은  $O(t_i)$  집합에 속한다.

● 사전 조건  $e_{in}$

사전 조건  $e_{in}$ 은  $t_i$ 의 점화를 위하여  $I(t_i)$ 안의 위치들에 대해 어떠한 측면 토큰이 필요한지를 명세한다. 특히, 사건에 대한 사전 조건은  $I(t_i)$ 안의 일부 위치만을 명세할 수 있다. 모든 입력 위치들의 충족을 의미하는 AND 조건뿐만 아니라 동일한 전이에 대해서 일부 위치들의 충족을 뜻하는 여러 사건들의 사전 조건을 명세함으로써 OR 조건도 나타낼 수 있다.  $e_{in}$ 에서 '\*' 측면값은 결국 특정 측면값이 사전 조건에 포함되지 않

음을 의미한다.

● 사후 조건  $e_{out}$

사후 조건  $e_{out}$ 은  $t_i$ 의 점화시에  $O(t_i)$ 안의 위치들에 대한 측면 토큰의 변화를 명세한다. 측면 토큰들을  $O(t_i)$ 안의 일부 위치들에만 발생할 수 있도록 확장함으로써 사건 발생시 측면 토큰들의 다양한 라우팅이 가능하다. 사후 조건  $e_{out}$ 은  $O(t_i)$ 안에 속하는 위치들에 새롭게 생성되는 토큰들에 대한 측면값들을 정의함으로써 전이  $t_i$ 의 점화에 따른 측면 토큰 값의 변화를 명세한다. 이때, 측면 토큰의 값은 상수값을 정의할 수도 있고  $e_{in}$ 에 명세된 특정 위치의 측면값들에 대한 상대적인 값으로 정의할 수도 있다.

한편, 측면 마킹  $m$ 은 MAPN에서의 위치들의 상태를 나타내고 상태 변화는 위치들에 대한 측면 마킹으로부터 다른 측면 마킹으로의 이동에 의해서 표현된다. 이러한 토큰의 이동과 측면 값의 변화는 사건의 활성화에 따른 특정 전이의 점화(firing)에 의해서 제어된다.

**【정의 6】 활성화(enable)**

사건  $e_{ij}(=\langle t_i, e_{in}, e_{out} \rangle)$ 는 측면 마킹  $m$ 이 다음과 같은 조건을 만족하는 상태일때 활성화(enabled) 된다.

$$e_{in} \subseteq m$$

사건  $e_{ij}$ 는 사전 조건에 명세된 측면 토큰들의 위치와 측면값들이 측면 마킹  $m$ 에 의한 상태에 포함될때 발생한다.

한편, MAPN에서의 상태 변화는 모든 위치들의 토큰 갯수의 변화와 각 토큰들의 측면값의 변화를 의미한다. 즉, 사건에 의한 전이의 점화는 임의 위치들의 측면 마킹을 바꾸게 하며 따라서, 페트리 네트 모델의 상태를 변경시킨다. MAPN에서의 토큰들의 흐름과 각 토큰값들의 변화는 명시적으로 나타나지는 않지만 사건에 의한 측면 마킹에 의해서 표현될 수 있다. 따라서, 사건  $e_{ij}(=\langle t_i, e_{in}, e_{out} \rangle)$ 가  $e_{in}$  사전 조건을 만족함에 따라 활성화됨으로써 전이  $t_i$ 를 점화하여  $e_{out}$ 과 같은 측면 마킹의 변화를 가져온다면 전체적인 상태 전이는 다음과 같이 정의한다.

**【정의 7】 상태 전이**

사건  $e_{ij}(=\langle t_i, e_{in}, e_{out} \rangle)$ 가 마킹  $m_1$ 에서 활성화되어  $t_i$ 를 점화시키고 마킹  $m_2$ 로 전이된다고 할때  $e_{ij}$ 에 의한

상태 전이는

$$m_2 = (m_1 - e_{in}) + e_{out}$$

로 정의한다. 즉,  $e_{in}$ 에 명세된 측면 토큰은 제거하고  $e_{out}$ 에 명세된 측면 토큰은 새로 추가한다.

이때, 마킹  $m_2$ 는 사건  $e_{ij} = \langle t_i, e_{in}, e_{out} \rangle$ 에 의하여 마킹  $m_1$ 으로부터 '도달가능(reachable)하다'고 하며 다음과 같이 표시한다 :

$$m_1 \rightarrow m_2$$

각 사건은 사건 정의에 의해 명시된 것처럼 사전 조건을 만족할 경우 활성화되어 사후 조건에 따라 토큰들의 분포와 토큰들의 측면값을 변화시킨다. 즉, 사전 조건에 명시된 입력 위치들에 대해서는 토큰 수가 감소되고, 사후 조건을 만족한 출력 위치들에 대해서는 토큰 수가 증가된다. 또한, 이때 사건 정의에 의해서 토큰들의 측면값 역시 변경된다. 그리고, 이외의 모든 다른 위치들의 측면 마킹은 변화하지 않는다.

예를 들어, 사건  $e_{ij} = \langle t_i, \langle lock, *, * \rangle + \langle free, *, * \rangle, \langle busy, A_1+A_2, A_1*A_2 \rangle \rangle$ 는 위치 'lock'과 'free'에 토큰이 하나 이상씩 존재할 경우 발생하며  $t_i$ 점화후에는 위치 'busy'에 새로운 토큰 하나를 생성시킨다. 생성된 토큰은 위치 'busy'에 생성되며 토큰의 두번째 측면 값은 위치 'lock'에 있던 토큰과 위치 'free'에 있던 토큰의 두번째 측면 값들의 합이고 세번째 측면 값은 위치 'lock'에 있던 토큰과 위치 'free'에 있던 토큰의 세번째 측면 값들의 곱으로 설정된다.  $\langle lock, *, * \rangle + \langle free, *, * \rangle$ 은 위치 'lock'과 'free'에 측면값에 상관없이 토큰이 존재하기만 한다면  $e_{ij}$ 가  $t_i$ 를 점화시킴을 의미한다.

【정의 7】에서 정의한바와 같이 MAPN에서 상태 전이는 사건의 사전 조건  $e_{in}$ 과 사후 조건  $e_{out}$ 에 의해서 표현된다. 이때,  $e_{in}$ 과  $e_{out}$ 은 전이에 대한 입력 위치들의 측면값과 출력 위치들의 출력값을 나타낸다. 즉, MAPN에서의 임의의 상태간의 전이는 다양한 측면값들의 변화에 의해서 표현된다. 따라서, MAPN은 전이의 점화를 위한 사건  $e_{ij}$ 의 정의를 통해서 시스템의 상태 변화를 다양한 측면에서 표현할 수 있으므로 【정의 1】의 조건 ②를 만족한다. .... (2)

한편, 위에서 '도달가능하다'라는 관계는 모든 상태 집합상에서의 전이성(transitive) 관계가 된다. 예를 들어 설명하면, 만일  $m_1 \rightarrow m_2$  이고  $m_2 \rightarrow m_3$  이면  $m_1$

$\rightarrow m_3$  이므로  $m_3$ 도  $m_1$ 으로부터 '도달가능'하다. 따라서, MAPN에서 초기 마킹  $m_0$ 로부터 '도달가능'한 모든 마킹들의 집합을 고려하면 다음과 같은 점화 순서를 정의할 수 있다.

**【정의 8】 점화 순서(firing sequence)**

점화 순서 F는 다음과 같이 마킹(markings)들과 사건(event)들의 순서로 표시한다.

$$F(m_0) : m_0 \xrightarrow{e_{11}} m_1 \xrightarrow{e_{22}} m_2 \xrightarrow{e_{33}} m_3 \xrightarrow{e_{44}} m_4 \rightarrow \dots$$

또한, 마킹  $m_1$ 에서 시작하여 마킹  $m_n$ 으로 끝나는 유한 길이의 점화 순서가 존재한다면 마킹  $m_n$ 은 마킹  $m_1$ 으로부터 '도달 가능(reachable)하다'고 한다.

$$m_1 \xrightarrow{e_{11}} m_2 \xrightarrow{e_{22}} m_3 \rightarrow \dots \xrightarrow{e_{n-1n-1}} m_n$$

상기 (1)과 (2)의 내용으로부터 다음의 정리를 밝힐 수 있다.

**【정리 1】 다중측면 표현능력을 갖는 페트리 넷**

모델 MAPN = ( P, T, I, O, E, A,  $m_0$  )에 대해

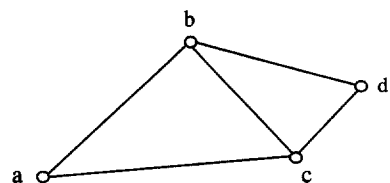
【정의 3】의 각 위치들의 측면값들로 표현되는 마킹  $m$ 과 【정의 7】의 상태 전이를 표현하는 사건  $e_{ij}$ 의  $e_{in}$ 과  $e_{out}$ 을 고려하면 MAPN은 다중측면 표현능력을 갖는 모델이다.

**【증명】**

모델 MAPN = ( P, T, I, O, E, A,  $m_0$  )에서 마킹값들의 벡터로 표현되는 측면 마킹  $m$ 과 마킹 전이를 표현하는  $e_{in}$ 과  $e_{out}$ 은 각각 모델의 다중측면 표현능력 조건 ①항과 ②항을 만족시키므로 MAPN은 다중측면 표현능력을 갖는다. ■

**4. MAPN 모델링 및 분석**

**4.1 DAP 모델링 예**



(그림 1) 분산 환경

MAPN 구조와 그래프를 사용하여 (그림 1)과 같은 분산 환경에서의 분산 트랜잭션 처리 과정을 모델링함으로써 DAP 평가 모델을 구성할 수 있다.

먼저, 읽기 트랜잭션의 경우 오직 한 데이터만을 판독하고, 쓰기 트랜잭션의 경우 모든 데이터를 갱신하는 ROWA(Read-One-Write-All) 규칙을 사용함을 가정한다. 따라서, 중복된 데이터에 대해 읽기 트랜잭션은 데이터 저장 사이트중 한 사이트만을, 쓰기 트랜잭션의 경우 모든 데이터 저장 사이트를 접근하여야 하므로 읽기 트랜잭션과 쓰기 트랜잭션 각각에 대해 독립된 DAP 평가 모델을 구성한다.

MAPN = ( P, T, I, O, E, A, m<sub>0</sub> )

P = {p<sub>1</sub>, p<sub>2</sub>, p<sub>3</sub>, p<sub>4</sub>, p<sub>5</sub>, p<sub>6</sub>, p<sub>7</sub>, p<sub>8</sub>, p<sub>9</sub>, p<sub>10</sub>, p<sub>11</sub>, p<sub>12</sub>, p<sub>13</sub>, p<sub>14</sub>, p<sub>15</sub>, p<sub>16</sub>}

T = {t<sub>1</sub>, t<sub>2</sub>, t<sub>3</sub>, t<sub>4</sub>, t<sub>5</sub>, t<sub>6</sub>, t<sub>7</sub>, t<sub>8</sub>}

I(t <sub>0</sub> ) = ∅	O(t <sub>0</sub> ) = {p <sub>1</sub> }
I(t <sub>1</sub> ) = {p <sub>1</sub> , p <sub>3</sub> }	O(t <sub>1</sub> ) = {p <sub>2</sub> }
I(t <sub>2</sub> ) = {p <sub>2</sub> , p <sub>4</sub> , p <sub>16</sub> }	O(t <sub>2</sub> ) = {p <sub>3</sub> , p <sub>4</sub> , p <sub>5</sub> , p <sub>10</sub> }
I(t <sub>3</sub> ) = {p <sub>4</sub> , p <sub>5</sub> , p <sub>6</sub> }	O(t <sub>3</sub> ) = {p <sub>4</sub> , p <sub>6</sub> , p <sub>7</sub> }
I(t <sub>4</sub> ) = {p <sub>7</sub> }	O(t <sub>4</sub> ) = {p <sub>5</sub> , p <sub>8</sub> }
I(t <sub>5</sub> ) = {p <sub>9</sub> , p <sub>10</sub> , p <sub>11</sub> }	O(t <sub>5</sub> ) = {p <sub>9</sub> , p <sub>11</sub> , p <sub>12</sub> }
I(t <sub>6</sub> ) = {p <sub>12</sub> }	O(t <sub>6</sub> ) = {p <sub>10</sub> , p <sub>13</sub> }
I(t <sub>7</sub> ) = {p <sub>4</sub> , p <sub>8</sub> , p <sub>13</sub> }	O(t <sub>7</sub> ) = {p <sub>4</sub> , p <sub>14</sub> , p <sub>15</sub> }
I(t <sub>8</sub> ) = {p <sub>14</sub> , p <sub>15</sub> }	O(t <sub>8</sub> ) = {p <sub>16</sub> }

E = {e<sub>11</sub>, e<sub>21</sub>, e<sub>31</sub>, e<sub>32</sub>, e<sub>33</sub>, e<sub>41</sub>, e<sub>42</sub>, e<sub>43</sub>, e<sub>51</sub>, e<sub>52</sub>, e<sub>53</sub>, e<sub>61</sub>, e<sub>62</sub>, e<sub>63</sub>, e<sub>71</sub>, e<sub>72</sub>, e<sub>81</sub>}

m<sub>0</sub> = < <p<sub>3</sub>>+<p<sub>4</sub>>+<p<sub>6</sub>>+<p<sub>9</sub>>+<p<sub>11</sub>> >

A = P × T × C × A × N × S × Q

P(place) = {p<sub>1</sub>, . . . , p<sub>16</sub>}

N(현재사이트) = {s<sub>0</sub>, s<sub>1</sub>}

T(응답시간) = R(실수)      C(비용) = R

A(가용성) = R

S(처리상태) = {pre-lock, post-lock, pre-op, post-op, pre-unlock, post-unlock}

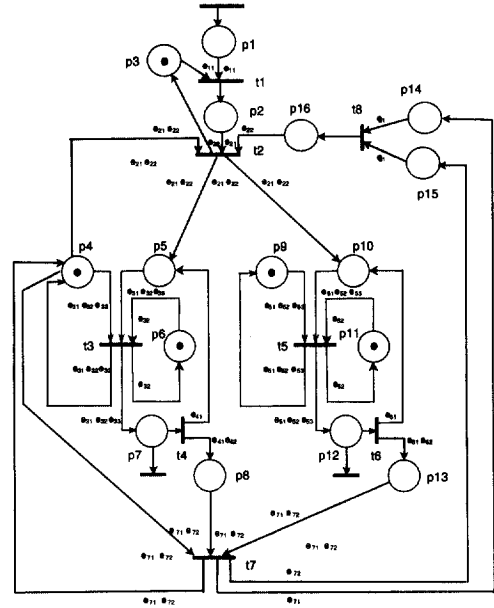
Q(타입) = {r}

MAPN 그래프는 고전적인 페트리 넷 그래프와 유사하다. 그러나, 측면 토큰의 정의와 이에 따른 측면 마킹, 그리고 사건의 사전 조건 및 사후 조건의 확장 등을 표현할 수 있도록 그래프 역시 확장되었다. 정적인 구조를 나타내는 MAPN 구조는 다음과 같다.

< 읽기 트랜잭션의 MAPN 구조 예 >

위와 같은 MAPN 구조는 전체적인 토큰의 분포는 나타낼 수 있지만 사건에 따른 입출력 위치들의 관계를 표현할 수 없다. 또한, 사건 발생시 각 측면 토큰의 측면값의 변화를 나타낼 수 없다. MAPN 그래프에서

는 사건과 관련된 동적인 모델링을 표현할 수 있도록 사건에 대한 표시와 명세가 필요하다. 따라서, 사건과 사건 발생에 따른 측면 토큰의 변화를 표현하기 위해 다음과 같이 MAPN 그래프와 사건 명세를 함께 나타낸다.



(그림 2) 읽기 트랜잭션을 모델링한 MAPN 그래프

읽기 트랜잭션에 대한 MAPN 구조를 다음 (그림 2)의 MAPN 그래프로 표현함으로써 분산된 사이트간의 분산 트랜잭션 처리 과정을 나타낼 수 있다. (그림 2)의 MAPN 그래프는 분산 2단계 로킹(distributed 2-phase locking) 기법과 2단계 완료(2-phase commit) 규약을 사용한다고 가정할 때, 읽기 트랜잭션이 임의의 사이트에서 발생하여 지역 사이트를 포함한 2개의 사이트중 읽고자하는 데이터가 저장되어 있는 사이트들에 대해 접근하여 데이터를 읽어오는 과정을 모델링한 것이다.

(그림 2)에서 place는 다음의 조건을 나타낸다.

- p1 : 발생된 읽기 트랜잭션의 읽기 연산들이 대기중임
- p2 : 임의의 읽기 연산이 발생됨
- p3 : 대기중인 읽기 연산에 대한 발생이 허용됨
- p4,p9 : CPU 자원이 가용함
- p5,p10 : 읽기 연산을 위한 read lock의 설정, 해제 요청이 도착하거나 읽기수행이 요청됨
- p6,p11 : DISK 자원이 가용함

- p7,p12: 읽기 연산을 위한 read lock 설정, read lock 해제, 읽기수행이 처리됨
- p8,p13: 읽기 연산을 위한 read lock 설정에 대한 확인 메시지 혹은 읽기수행 결과가 도착함
- p14: read lock 설정에 대한 확인 메시지가 도착함
- p15: 읽기 수행 결과가 도착함
- p16: 읽기 연산이 완료됨

또한, (그림 2)에서 event  $e_{ij}$ 는 다음의 사건을 수행한다.

- $e_{11}$ : 읽기 연산이 허용된 경우 임의의 읽기 연산을 발생시킨다.
- $e_{21}$ : 단편이 저장된 사이트들에 read lock 설정을 요청한다. (트랜잭션 발생 사이트의 CPU 자원이 가용한 경우)
- $e_{22}$ : 단편이 저장된 사이트들에 read lock 해제를 요청하고 또다른 읽기 연산의 발생을 허용한다. (트랜잭션 발생 사이트의 CPU 자원이 가용한 경우)
- $e_{31}, e_{31}$ : 요청된 단편에 대해 read lock을 설정한다. (CPU 자원이 가용한 경우)
- $e_{32}, e_{32}$ : 요청된 단편에 대해 읽기 연산을 수행한다. (CPU, DISK 자원이 가용한 경우)
- $e_{33}, e_{33}$ : 요청된 단편에 대해 read lock을 해제한다. (CPU 자원이 가용한 경우)
- $e_{41}, e_{41}$ : read lock 설정에 대한 확인 메시지를 전송한다.
- $e_{42}, e_{42}$ : 읽기 연산의 수행을 요청한다.
- $e_{43}, e_{43}$ : 읽기 연산의 수행 결과인 단편을 전송한다.
- $e_{71}$ : read lock 설정 확인 메시지의 도착을 확인한다.
- $e_{72}$ : 읽기 수행한 결과인 단편의 도착을 확인한다.
- $e_{81}$ : read lock 설정 확인 메시지와 읽기 수행한 결과인 단편이 도착할 경우 read lock의 해제를 요청하고 해당 읽기 연산의 수행종료를 지시한다.

그러나, (그림 2)와 같은 MAPN 그래프는 토큰들의 이동 가능한 경로들을 추측할 수만 있으며 상세한 측면값들의 변화를 알 수 없다. 따라서, MAPN 그래프에 다음과 같이 사건과 토큰에 관한 명세가 필요하다.

**< 읽기 트랜잭션의 MAPN 명세 예 >**

**[측면 토큰 정의]**

$$k_{R11} = \langle p_1, \text{init\_tm}, \text{init\_co}, \text{init\_av}, s_0, \text{pre\_lock}, r \rangle$$

**[측면 변수 정의]**

$$\begin{aligned} \text{init\_tm} &= 0, \text{init\_co} = 0, \text{init\_av} = 1 \\ \text{com\_tm}_0 &= 0, \text{com\_tm}_1 = 0.02, \\ \text{com\_co}_0 &= 0, \text{com\_co}_2 = 0.01, \\ \text{com\_av}_0 &= 1, \text{com\_av}_2 = 0.97, \end{aligned}$$

$$\begin{aligned} \text{cpu\_tm}_0 &= 0.002, \text{cpu\_tm}_1 = 0.002, \\ \text{cpu\_co}_0 &= 0.001, \text{cpu\_co}_1 = 0.001, \\ \text{cpu\_av}_0 &= 0.99, \text{cpu\_av}_1 = 0.99, \\ \text{io\_tm}_0 &= 0.0025, \text{io\_tm}_1 = 0.0025, \\ \text{io\_co}_0 &= 0.1, \text{io\_co}_1 = 0.1, \\ \text{io\_av}_0 &= 0.98, \text{io\_av}_1 = 0.98 \end{aligned}$$

**[event 정의]**

- $e_{11} = \langle t_1, \langle p_1, *, *, *, s_0, \text{pre\_lock}, r \rangle + \langle p_3 \rangle, \langle p_2, T_1, C_1, A_1, s_0, \text{pre\_lock}, r \rangle \rangle$
- $e_{21} = \langle t_2, \langle p_2, *, *, *, s_0, \text{pre\_lock}, r \rangle + \langle p_4 \rangle, \langle p_5, T_1 + \text{cpu\_tm}_0 + \text{com\_tm}_0, C_1 + \text{cpu\_co}_0 + \text{com\_co}_0, A_1 \times \text{cpu\_av}_0 \times \text{com\_av}_0, s_0, \text{pre\_lock}, r \rangle + \langle p_{10}, T_1 + \text{cpu\_tm}_0 + \text{com\_tm}_0, C_1 + \text{cpu\_co}_0 + \text{com\_co}_0, A_1 \times \text{cpu\_av}_0 \times \text{com\_av}_0, s_1, \text{pre\_lock}, r \rangle + \langle p_4 \rangle \rangle$
- $e_{22} = \langle t_2, \langle p_{26}, *, *, *, s_0, \text{pre\_unlock}, r \rangle + \langle p_{26}, *, *, *, s_1, \text{pre\_unlock}, r \rangle + \langle p_{26}, *, *, *, s_2, \text{pre\_unlock}, r \rangle + \langle p_{26}, *, *, *, s_3, \text{pre\_unlock}, r \rangle + \langle p_4 \rangle, \langle p_5, T_1 + \text{cpu\_tm}_0 + \text{com\_tm}_0, C_1 + \text{cpu\_co}_0 + \text{com\_co}_0, A_1 \times \text{cpu\_av}_0 \times \text{com\_av}_0, s_0, \text{pre\_unlock}, r \rangle + \langle p_{10}, T_1 + \text{cpu\_tm}_0 + \text{com\_tm}_0, C_1 + \text{cpu\_co}_0 + \text{com\_co}_0, A_1 \times \text{cpu\_av}_0 \times \text{com\_av}_0, s_1, \text{pre\_unlock}, r \rangle + \langle p_3 \rangle + \langle p_4 \rangle \rangle$
- $e_{31} = \langle t_3, \langle p_5, *, *, *, s_0, \text{pre\_lock}, r \rangle + \langle p_4 \rangle, \langle p_7, T_1 + \text{cpu\_tm}_0, C_1 + \text{cpu\_co}_0, A_1 \times \text{cpu\_av}_0, s_0, \text{post\_lock}, r \rangle + \langle p_4 \rangle \rangle$
- $e_{32} = \langle t_3, \langle p_5, *, *, *, s_0, \text{pre\_op}, r \rangle + \langle p_4 \rangle + \langle p_6 \rangle, \langle p_7, T_1 + \text{cpu\_tm}_0 + \text{io\_tm}_0, C_1 + \text{cpu\_co}_0 + \text{io\_co}_0, A_1 \times \text{cpu\_av}_0 \times \text{io\_av}_0, s_0, \text{post\_op}, r \rangle + \langle p_4 \rangle + \langle p_6 \rangle \rangle$
- $e_{33} = \langle t_3, \langle p_5, *, *, *, s_0, \text{pre\_unlock}, r \rangle + \langle p_4 \rangle, \langle p_7, T_1 + \text{cpu\_tm}_0, C_1 + \text{cpu\_co}_0, A_1 \times \text{cpu\_av}_0, s_0, \text{post\_unlock}, r \rangle + \langle p_4 \rangle \rangle$
- $e_{41} = \langle t_4, \langle p_7, *, *, *, s_0, \text{post\_lock}, r \rangle, \langle p_8, \text{com\_tm}_0, \text{com\_co}_0, \text{com\_av}_0, s_0, \text{post\_lock}, r \rangle \rangle$
- $e_{42} = \langle t_4, \langle p_7, *, *, *, s_0, \text{post\_lock}, r \rangle, \langle p_5, T_1, C_1, A_1, s_0, \text{pre\_op}, r \rangle \rangle$
- $e_{43} = \langle t_4, \langle p_7, *, *, *, s_0, \text{post\_op}, r \rangle, \langle p_8, T_1 + \text{com\_tm}_0, C_1 + \text{com\_co}_0, A_1 \times \text{com\_av}_0, s_0, \text{post\_op}, r \rangle \rangle$
- $e_{51} = \langle t_5, \langle p_{10}, *, *, *, s_1, \text{pre\_lock}, r \rangle + \langle p_9 \rangle, \langle p_{12}, T_1 + \text{cpu\_tm}_1, C_1 + \text{cpu\_co}_1, A_1 \times \text{cpu\_av}_1, s_1, \text{post\_lock}, r \rangle + \langle p_9 \rangle \rangle$
- $e_{52} = \langle t_5, \langle p_{10}, *, *, *, s_1, \text{pre\_op}, r \rangle + \langle p_9 \rangle + \langle p_{11} \rangle, \langle p_{12}, T_1 + \text{cpu\_tm}_1 + \text{io\_tm}_1, C_1 + \text{cpu\_co}_1 + \text{io\_co}_1, A_1 \times \text{cpu\_av}_1 \times \text{io\_av}_1, s_1, \text{post\_op}, r \rangle + \langle p_9 \rangle + \langle p_{10} \rangle \rangle$
- $e_{53} = \langle t_5, \langle p_{10}, *, *, *, s_1, \text{pre\_unlock}, r \rangle + \langle p_9 \rangle, \langle p_{12}, T_1 + \text{cpu\_tm}_1, C_1 + \text{cpu\_co}_1, A_1 \times \text{cpu\_av}_1, s_1, \text{post\_unlock}, r \rangle + \langle p_9 \rangle \rangle$



×cpu\_av<sub>1</sub>,s<sub>1</sub>,post\_unlock,r>+<p9> >  
 e<sub>61</sub> = < t6, <p12,\*,\*,s<sub>1</sub>,post\_lock,r>, <p13,com\_tm<sub>0</sub>,com\_co<sub>10</sub>,com\_av<sub>10</sub>,s<sub>0</sub>,post\_lock,r> >  
 e<sub>62</sub> = < t6, <p12,\*,\*,s<sub>1</sub>,post\_lock,r>, <p10,T<sub>1</sub>,C<sub>1</sub>,A<sub>1</sub>,s<sub>1</sub>,pre\_op,r> >  
 e<sub>63</sub> = < t6, <p12,\*,\*,s<sub>1</sub>,post\_op,r>, <p13,T<sub>1</sub>+com\_tm<sub>0</sub>,C<sub>1</sub>+com\_co<sub>10</sub>,A<sub>1</sub>×com\_av<sub>10</sub>,s<sub>0</sub>,post\_op,r> >  
 e<sub>71</sub> = < t7, <p8,\*,\*,s<sub>0</sub>,post\_lock,r> +<p13,\*,\*,s<sub>1</sub>,post\_lock,r>+<p4>, +<p14,T<sub>1</sub>+cpu\_tm<sub>0</sub>,C<sub>1</sub>+cpu\_co<sub>0</sub>,A<sub>1</sub>×cpu\_av<sub>0</sub>,s<sub>0</sub>,post\_lock,r> +<p14,T<sub>2</sub>+cpu\_tm<sub>0</sub>,C<sub>2</sub>+cpu\_co<sub>0</sub>,A<sub>2</sub>×cpu\_av<sub>0</sub>,s<sub>1</sub>,post\_lock,r>+<p4> >  
 e<sub>72</sub> = < t7, <p8,\*,\*,s<sub>0</sub>,post\_op,r> +<p13,\*,\*,s<sub>1</sub>,post\_op,r>+<p4>, +<p15,T<sub>1</sub>+cpu\_tm<sub>0</sub>,C<sub>1</sub>+cpu\_co<sub>0</sub>,A<sub>1</sub>×cpu\_av<sub>0</sub>,s<sub>0</sub>,post\_op,r> +<p15,T<sub>2</sub>+cpu\_tm<sub>0</sub>,C<sub>2</sub>+cpu\_co<sub>0</sub>,A<sub>2</sub>×cpu\_av<sub>0</sub>,s<sub>1</sub>,post\_op,r>+<p4> >  
 e<sub>81</sub> = < t8, <p14,\*,\*,s<sub>0</sub>,post\_lock,r>+<p15,\*,\*,s<sub>0</sub>,post\_op,r> +<p14,\*,\*,s<sub>1</sub>,post\_lock,r>+<p15,\*,\*,s<sub>1</sub>,post\_op,r>, <p26,T<sub>1</sub>+T<sub>2</sub>,C<sub>1</sub>+C<sub>2</sub>,A<sub>1</sub>×A<sub>2</sub>,s<sub>0</sub>,pre\_unlock,r> +<p26,T<sub>3</sub>+T<sub>4</sub>,C<sub>3</sub>+C<sub>4</sub>,A<sub>3</sub>×A<sub>4</sub>,s<sub>1</sub>,pre\_unlock,r> >

위와 같은 명세에 의해 (그림 2) MAPN 그래프에서의 다음과 같은 분산 트랜잭션의 처리 과정을 파악할 수 있다.

먼저, 측면 A가 place, 시간, 비용, 가용성, 현재사이트, 상태, 타입의 7가지로 정의됨에 따라서 모든 측면 토큰들은 그에 상응하는 7개의 측면값을 갖는다. (그림 2)에서 place p4, p9와 p6, p11 의 제어 토큰들은 각 사이트에서의 CPU와 DISK 자원이 가용함을 의미한다. 측면 토큰 k<sub>R11</sub>은 읽기 트랜잭션으로 발생가능한 여러 트랜잭션중의 하나이다. 트랜잭션의 갯수를 의미하는 토큰의 갯수는 정보분석 단계의 결과에 따라 달라진다.

한편, 각 토큰들의 측면 초기값은 임의로 설정될 수 있으며 [측면 변수 정의]에 나타난 것처럼 시간, 비용 및 가용성 측면값을 초기화할 수 있다. MAPN에서 측면값들을 변화시키는 각 사건들은 [event 정의]에 명세된다. 이러한 사건 명세는 각 사건이 발생되기 위해서 만족되어야할 조건과 사건 발생시 전이에 의한 측면값들의 변화를 표현하고 있다. 예를 들어, 사건 e<sub>11</sub>은

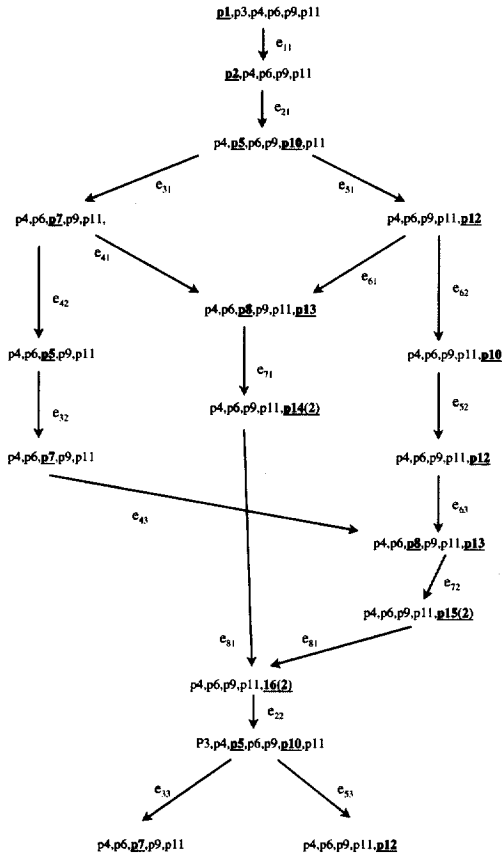
첫번째 측면값(place)이 p1이고, 다섯 번째(현재사이트), 여섯번째(상태), 일곱번째(타입) 측면값이 각각 s<sub>0</sub>, prelock과 r인 측면 토큰과 위치 p3에 제어 토큰이 존재할 경우에만 전이를 점화시킨다. 그리고, 사건 e<sub>11</sub>에 의한 전이가 점화되면 위치 p2에 새로운 측면 토큰이 생성된다. 이때, 생성되는 측면 토큰은 시간, 비용, 가용성들에 대한 초기 측면값을 할당받는다.

#### 4.2 DAP 모델링 분석

일반적으로 페트리 넷 모델은 도달할 수 있는 모든 상태들과 그들 사이의 점화되는 전이를 '토큰 기계'라 불리는 상태 기계를 사용하여 표현하고 이를 이용하여 도달성 분석을 한다[9, 10]. 그러나, 앞에서 MAPN을 이용하여 모델링한 DAP 모델은 전이가 아닌 사건을 통해 전이되기 때문에 각 위치에서 적용가능한 모든 사건을 고려한 (그림 3)과 같은 상태 전이 그래프를 이용해서 검증할 수 있다. (그림 3)의 상태전이 그래프는 (그림 2)의 MAPN 그래프가 정상적으로 동작한다는 것을 증명하는 것으로 각 위치에서의 발생하는 사건들에 의한 토큰의 순차적인 흐름을 표현한다. 사전 조건이 만족할 경우에 발생하는 사건과 그 사건의 사후조건들의 연속된 흐름을 따라 도달성 분석을 함으로써 제시한 MAPN이 2PC 규약을 정확히 모델로서 표현한 것인지를 검증할 수 있다. (그림 3)에서 p4, p6, p8, p9, p11, p13은 사건 e<sub>41</sub>과 e<sub>61</sub> 발생에 따른 사후조건에 의해 p4, p6, p8, p9, p11, p13 위치로 토큰이 이동했으며 p8과 p13에는 신규로 토큰이 생성되었음을 나타낸다. (그림 3)에서 보는 것처럼 위치 p14와 p15에 각각 2개의 토큰이 존재해야만 사건 e<sub>81</sub>이 발생하기 때문에 p14는 두 사이트에 대한 읽기 록(read lock) 설정 상태를, p15는 읽기 연산 결과가 도착한 상태를 표현함을 확인할 수 있다. 즉, 먼저 읽기 록을 설정한후 읽기 연산이 완료되어야만 사건 e<sub>22</sub>에 의한 읽기 록을 해제할 수 있음을 통해 분산 2-PL(Phase Locking), 2-PC(Phase Commit) 규약 등에 대한 모델링 결과의 타당성을 검사할 수 있다.

제안한 MAPN 모델링은 동시성 제어와 같은 복잡한 트랜잭션 처리 과정뿐만 아니라 무결성 유지 등과 같은 분산 데이터 운영환경과 관련한 다양한 처리 과정을 모두 표현할 수 있으며 본 논문에서는 2PC 처리 과정을 예로 제시하였다. 또한, MAPN 모델링 방법은 주어진 데이터 할당 문제에 있어 고려해야하는 모든

복잡한 처리 과정을 MAPN으로 표현할 수 있을뿐만 아니라 그러한 처리 과정상에서 분석하고자 하는 다양한 측면들을 일관되게 분석, 평가할 수 있다. 제시한 예에서는 비용, 시간, 가용성을 분석 측면으로 정의하였으며 구성된 분석 모델은 PCGA(Pareto Crowding Genetic Algorithms) 기법[14, 15]을 사용하여 최적의 할당해를 찾을 수 있다.



(그림 3) 읽기 트랜잭션을 위한 상태 전이 트리

### 5. 결 론

분산 데이터베이스 할당 문제는 분산 데이터베이스 시스템 전체의 비용, 성능 그리고 가용성에 영향을 미치는 매우 중요한 문제이다. 데이터 할당 모델에 관한 기존 수학적 모델들은 분산 트랜잭션 기법과 같은 복잡한 처리 과정에 대해서 모호하고 불완전한 모델이 구축될 가능성이 높으며 또한, 다양한 측면들을 동시

에 고려하지 못하는 문제점이 있다.

이 논문에서는 비용, 성능 및 가용성 등의 다중 측면을 동시에 고려하기 위해 다양한 측면을 표현할 수 있는 트랜잭션 모델링 방법으로 MAPN 모델링 기법을 제안하였다. 이 MAPN 모델은 시스템의 다양한 측면 상태를 모델링할 수 있도록 고전적인 페트리 넷트를 측면 토큰과 사건 개념을 가지고 확장한 것으로 MAPN 모델링을 통하여 분산 트랜잭션 처리 과정중의 각 측면값들을 명확하고 투명하게 분석할 수 있다.

제안된 MAPN 구조와 그래프를 사용함으로써 다양한 측면을 동시에 고려하면서 분산 트랜잭션 처리 과정을 정형화된 형태로 모델링할 수 있으며 따라서, 타당한 DAP 평가 모델을 구성할 수 있음을 보였다.

### 참 고 문 헌

- [1] M. Ozsu and P. Valduriez, Principles of Distributed Database Systems, Englewood Cliffs, N. J. : Prentice-Hall Inc., 1991.
- [2] P.M.G. Apers, "Data allocation in distributed database systems," ACM Trans. on Database Systems, Vol.13, No.3, pp.263-304, Sept. 1988.
- [3] D.W. Cornell and P.S. Yu, "On optimal site assignment for relations in the distributed database environment," IEEE Trans. on Software Engineering, Vol.15, No.8, pp.1004-1009, Aug. 1989.
- [4] S. Ram, and R.E. Marsten, "A model for database allocation incorporating a concurrency control mechanism," IEEE Trans. on Knowledge and Data Engineering, Vol.3, No.3, Sept. 1991.
- [5] S.T. March, and S. Rho, "Allocating data and operations to nodes in distributed database design," IEEE Trans. on Knowledge and Data Engineering, Vol.7, pp.305-317, 1995.
- [6] W.W. Chu, Optimal file allocation in a computer network, Computer Communications Systems, pp. 82-94, Prentice Hall, Englewood Cliffs, N.J., 1973.
- [7] H.K. Jain, "A comprehensive model for the design of distributed computer systems," IEEE Trans. Software Eng., Vol.SE-13, pp.1092-1104, Oct. 1987.

[8] C.M. Fonseca and P.J. Fleming, "Genetic algorithms for multiobjective optimization : formulation, discussion and generalization," Proceedings of the Fifth International Conference on Genetic Algorithms, Morgan Kauffman(1993), pp.416-423.

[9] T. Murata, Petri Nets : Properties, Analysis and Applications, Proceedings of the IEEE, 77(1989), pp.541-580.

[10] J.L. Peterson, Petri net theory and the modeling of systems, Prentice-Hall, Englewood Cliffs, 1981.

[11] 박성진, 노전필, 백두권, "분산 데이터베이스의 데이터 중복 기법 및 시뮬레이션", 추계학술대회발표논문집, 한국시뮬레이션학회, pp.5, 1996.

[12] 박성진, 백두권, "다중 측면 페트리 넷 모델링을 이용한 데이터 할당 방법론", 가을학술발표논문집, 한국정보처리학회, pp.416-421, 1997.

[13] 박성진, 손주찬, 이교상, 백두권, "분산 데이터의 할당을 위한 MAPN 모델링", 가을학술발표논문집, 한국정보처리학회, pp.179-182, 1998.

[14] 박성진, 박화규, 손주찬, 박상봉, 백두권, "유전자

알고리즘을 이용한 분산 데이터베이스 할당 방법론", 정보기술과 데이터베이스저널, 5권 1호, 한국 데이터베이스학회, pp.1-12, 1998.

[15] 박성진, 백종명, 백두권, "다중측면 최적화 문제를 위한 Pareto Crowding 유전자 알고리즘", 정보과학회논문지(B), 25권 5호, pp.820-830, 1998.



### 박성진

e-mail : sipark@hucc.hanshin.ac.kr

1991년 고려대학교 전산학과  
졸업(학사)

1993년 고려대학교 대학원 전산과  
학과(이학석사)

1998년 고려대학교 대학원 전산과  
학과(이학박사)

1998년~2000년 한국전자통신연구원 선임연구원

2000년~현재 한신대학교 정보처리학과 조교수

관심분야 : 분산데이터베이스, 리파지토리, 정보공학, 컴  
포넌트 공학 등