실시간 시스템의 실행 공간상에서 구문 및 의미 패턴에 기반한 상태 최소화를 위한 추상화 방법

(An Abstraction Method for State Minimization based on Syntactic and Semantic Patterns in the Execution Space of Real-Time Systems)

박 지 연 † 조 기 환 †† 이 문 근 ††

(Ji-yeon Park) (Gi-hwan Cho) (Moon-kun Lee)

요 약 정형기법을 사용하여 실시간 시스템을 명세할 때, 상태 기반 정형 기법이 가지는 큰 문제 중의 하나는 시간 값, 자료 값, 위치 값을 통한 상태 표현으로 발생하는 상태 폭발이다. 본 논문에서는 상태 폭발 문제를 접근하기 위해, 시스템의 명세에 적용하는 추상화와, 명세된 시스템의 실행에 적용하는 추상화 기법을 정의하였다. 명세 구문에 정의한 추상화를 구문 추상화라 정의하고 명세 구문이 가진 패턴(연산 정보, 구조)을 정의하여 추상화한다. 실행에 적용되는 추상화는 의미 추상화라 정의하고 실행 시 생성되는 시간, 자료, 위치 상태 값이 가진 실행 의미의 패턴을 추상화한다. 추상화를 통하여 명세 모델과 실행 모델에 계층을 생성하여 상위 계층에서는 복잡도가 낮은 단계에서 시스템의 개략적인 정보를 분석할 수 있다. 하위 계층에서는 정확도가 높은 분석을 수행할 수 있는 반면에 많은 상태를 살펴야되기 때문에 높은 복잡도를 가진다. 본 논문에서는 추상화의 정의와 더불어, 적용 사례를 통하여 상태 감소와 계층성 생성, 복잡도 감소를 보인다.

키워드 : 정형 기법, 상태 감소, 추상화

Abstract *States explosion* due to composition of spaces of data, temporal, and locational values is one of the well-known critical problems which cause difficulty in understanding and analysing real-time systems specified with state-based formal methods. In order to overcome this problem, this paper presents an abstraction method for state minimization based on an abstraction in system specification and an abstraction in system execution. The first is named the syntactic abstraction, through which the patterns of the unconditionally internalized computation and the repetition and selection structures are abstracted. The latter is named the semantic abstraction, through which the patterns of the execution space represented with data, temporal and locational value in the forms of the patterns in the syntactic abstraction are abstracted. Through the abstractions, the components of a system in specification and execution model is hierarchically organized. The system can be analyzed briefly in the upper level in an skeleton manner with low complexity. The system, however, can be analyzed in detail in the lower level with high and dense complexity. This report presents the abstraction method for the state minimization and the decrease in analysis complexity through the abstraction with examples.

Key words : formal method, state minimization, abstraction

· 본 연구는 한국과학재단 특정기초연구(1999-2-303-003-3) 지원으로 수행되었음

† 비 회 원 : 전북대학교 컴퓨터통계정보학과
jypark@cs.chonbuk.ac.kr
†† 종신회원 : 전북대학교 전자정보공학부 교수
ghcho@cs.chonbuk.ac.kr
mklee@cs.chonbuk.ac.kr

논문접수 : 2002년 5월 2일
심사완료 : 2002년 11월 5일

1. 서 론

상태 기반 정형 기법에서 상태 폭발 문제[1,2]는 상태 기계로 표현된 시스템의 행위에 대한 이해와 분석을 어렵게 한다. 이는 각 상태 기계의 수와 각 기계가 가진 상태의 수가 증가할수록 전체 시스템의 상태의 수는 기하급수적으로 증가하기 때문이다. 특히, 실시간 시스템에서

는 실시간 속성으로 인하여 일반 시스템에 비해 상태 공간의 복잡도가 높아지며 따라서 시스템의 이해와 분석도 더욱 어렵게 된다.

상태 폭발 문제를 해결하기 위해 많은 연구[1, 2, 3, 4, 5, 6, 7, 8, 9]가 진행되고 있으며 이 연구들은 시스템의 실행 시 발생하는 상태 또는 상태 공간의 크기를 줄이는 것을 주요 목적으로 하고 있다. 이러한 연구들은 프로세스의 조합으로 발생하는 상태 폭발을 다루기 위해 유한 상태 시스템의 조합적 분석(*compositional analysis*) 방법[2]과 무한한 시간 공간을 유한한 시간 공간으로 감소시키는 방법[3], 무한한 데이타 공간을 유한한 데이타 공간으로 감소시키는 방법[4], 영역 그래프(*region graph*)로 표현되는 상태 공간을 시간에 대해 동치 관계에 있는 상태들을 분할을 통해 상태 감소를 유도하는 방법[5], 프로그램에 의해 생성되는 상태 집합에 대해 modulo와 같은 특정 관계성을 제시하여 추상화하는 방법[7], 디지털 하드웨어에 적용되는 상태 감소 방법으로 입력과 출력에 대한 *bisimulation*을 정의하여 상태를 분할하여 상태 감소를 유도하는 방법[9] 등을 포함하고 있다.

그러나 이러한 관련 연구들은 2절에서 기술하듯이 각각의 단점들을 가지고 있다. 단점과 제약점들을 효과적으로 해결하기 위하여, 즉 가능한 모든 경우의 상태감소를 보장하고 각 상태감소의 근거와 내용을 표현할 수 있는 새로운 상태감소 방법을 제안한다.

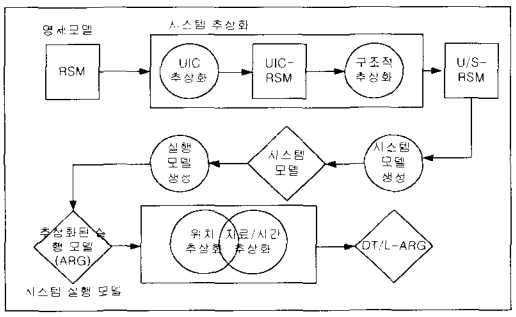


그림 1 추상화 흐름도

상태감소 방법은 그림 1에서 보여주는 바와 같이 다음 2개 유형 각 2단계(총 4단계)의 추상화로 이루어 졌다:

i) 명세 모델 기반 구문 추상화: 명세 모델은 각 시스템의 동작을 상태와 전이로 기술한다. 명세 모델에 적용되는 추상화는 명세된 구문이 특정한 조건을 만족할 경우 전이와 상태를 추상화하여 상태가 감소한 명세 모델을 생성한다.

① 비조건 내부 연산(UIC: *Unconditionally Internalized Computational*) 추상화: UIC 추상화에서는 한 상태 기계 내에 명세된 전이 중 상태 분기를 나타내는 조건이 없으며 다른 상태 기계와의 상호 작용을 가지지 않고 내부 연산만을 표현한 전이는 다른 상태 기계와의 상호 작용과 조건적으로 상태 분기에 영향을 끼치지 않음으로 이와 같은 상태와 전이를 추상화한다.

② 구조(*Structural*) 추상화: 구조 추상화는 특정 구조 조건에 만족하는 상태와 전이를 추상화한다. 논문에서는 정규적 반복 구조, 정규적 선택 구조, 혼합 구조를 정의하여 이 구조를 가진 상태와 전이를 추상화한다. 구조적 추상화를 통하여 세부 상태 변화를 실행 상태로 생성하지 않음으로 분석의 복잡도를 감소시킬 수 있다.

명세 모델은 조합을 통하여 명세 모델에 의해 생성될 수 있는 모든 발생 가능한 상태를 각 상태 기계의 위치 값을 기준으로 생성한다. 조합된 결과를 통하여 실행 시 발생하는 위치 값의 전이 관계와 시간, 자료 값을 상태로 가진 실행 모델을 생성한다. 실행 모델의 추상화를 수행할 때, 조합된 결과가 가진 구문 형태를 추상화된 정보 생성에 적용한다.

ii) 실행 모델 기반 의미 추상화: 실행 모델은 명세된 시스템의 실제 실행을 모델링한 것이다. 생성된 실행 모델이 가진 위치, 자료, 시간 정보가 의미적으로 특정한 규칙, 또는 반복적 형태 등으로 파악될 때 이를 추상화하는 방법이다.

① 위치(*Locational*) 공간 추상화: 시스템의 실행이 동일 위치 경로가 반복적으로 수행될 때 반복되는 동일 경로를 추상화한다.

② 자료/시간(*Data/Temporal*) 공간 추상화: 시스템의 실행이 추상화될 경우 추상화된 정보는 정확한 자료 값이나 시간 값을 가지지 못한다. 본 논문에서는 이러한 자료나 시간 값의 정보를 특정 함수나 값의 범위로 추상화하여 추상화한 후에도 정보의 손실을 최소화한다.

본 논문에서 정의한 추상화의 장점은 상태 감소를 명세 모델에 적용하는 구문적 추상화와 실행 모델에 수행하는 의미적 추상화를 정의하고, 실행 모델을 추상화할 때 명세 모델의 추상화된 정보를 이용한다는 점이다. 또한, 추상화된 상태 정보는 모두 계층적으로 표현하여 계층간의 상태 변화도를 이해할 수 있다.

구문 추상화와 의미 추상화의 적용은 각 모델이 가진 상태를 현저히 감소시켜 계층 구조에서 상위 계층에서는 시스템의 이해를 낮은 복잡도를 가지고 개략적 분석을 수행할 수 있다. 계층적 구조를 가진 명세 모델과 실

행 모델을 top-down 또는 bottom-up 방식을 통하여 원하는 계층 단계에서의 시스템을 이해할 수 있게 한다.

논문의 구성은 다음과 같다. 2절에서는 관련연구를 기술하고, 3절에서는 명세 모델 기반 분법 추상화 방법을 정의하며, 4절에서는 실행 모델 기반 의미 추상화 방법을 정의한다. 5절에서는 정의된 추상화의 적용에 대한 예를 보이며, 6절에서 결론과 향후연구에 대하여 기술한다.

2. 관련 연구

실시간 시스템의 상태 폭발 문제를 해결하기 위해 다양한 연구가 보고되고 있다[2, 3, 4, 5, 6, 7, 8, 9].

[2]의 연구는 다수의 시스템이 조합되어 병렬적으로 동작하는 시스템의 도달성 분석을 위해 동기화 구조 분석 방법을 제공한다. 시스템의 조합에 의해 발생 가능한 상태 폭발은 구조적 계층성을 이용하여 서브시스템의 세부 사항을 숨김으로써 방지한다. 이와 같은 방법은 유용하지만, 실제 시스템에 대한 도달성 분석과 상태 감소에는 적용하기 어렵다. 이는 시스템의 상태 분기에 영향을 미치는 많은 조건이 변수 값을 기준으로 하는 경우가 많으나 [2]의 연구에서는 상태 변화에 대한 제어 결정이 데이타 값과 독립적으로 수행된다는 가정에 의해 연구에서 고려되지 않았기 때문이다.

CSM(*Communicating State Machine*)[3]에서는 데이타와 통신에 적용되는 이벤트의 변화를 분석하여 데이타 도메인을 다수의 동치 클래스로 클러스터링하는 상태 최소화 방법을 제안한다. 그러나 이 방법은 실시간 시스템과 같이 많은 변수가 사용되는 시스템에는 적용이 어렵다. 변수가 많아지면 많아질수록 모든 변수가 동치 클래스로 클러스터링되더라도 복잡도는 기하급수적으로 증가하기 때문이다. 또한, CSM은 시스템의 시간 속성 명세를 위한 정의를 제공하지 않아 실시간 시스템에는 적용될 수 없으며 시스템의 모든 변수를 고려하기 때문에 불필요한 데이타 도메인을 생성할 수 있는 가능성이 존재한다.

[4]는 CRSM(*Communicating Real-time State Machine*)을 사용하여 명세한 실시간 시스템의 안전성을 검증하기 위해 도달성에 대한 분석 방법을 제공한다. [4]의 접근 방법은 시간 상태 공간의 처리를 통하여 무한한 도달성 그래프를 유한한 도달성 그래프로 변환한다. 그러나, 변환 영역은 시간 공간에만 적용하며 최소화를 위해 데이타와 제어 공간은 고려하지 않는다.

[5]은 TA(*Timed Automata*)로 모델링된 시스템의 상태 공간을 *Region Graph*를 이용하여 표현한다. 생성되는 region graph의 상태 공간을 감소하기 위하여 TA가 가진 시간에 대한 동치 관계를 적용하여 시간 공간을 여러 파티션으로 분할한다. 이 연구의 시간에 대한 분할 방법과 [3]의 변수의 상태 공간을 전이에 따른 도달 가능한 영역으로 분할하면서 시스템의 상태를 중복되지 않게 표현한 방법은 유사한 방법론을 사용하여 상태 감소를 하고 있다. [5]의 연구는 상태를 시간과 위치 정보로 표현하고 있어 변수의 값이 실행과 검증에 영향을 주는 소프트웨어의 정형적 모델링에는 적용하기 어렵다.

[6]은 TA에 대한 상태 최소화 방법은 발생 상태를 최소화하기 위해 *history equivalence*와 *transition bisimulation* 개념을 사용한다. 두 개념의 적용 결과는 시간 개념이 존재하는 상태를 시간 개념이 없는 오토마타에 의해 생성된 상태로 만든다. 즉, 데이타 공간이나 이벤트에 대한 도달성 분석은 가능하나 시간 공간에 대한 분석은 고려할 수 없다.

[7]은 원래 프로그램이 가진 속성을 검증할 수 있도록 추상화 모델을 제안한다. 추상화는 상태 집합에 대해 특정한 함수 관계 h를 정의하고 적용하여 사상된 상태의 집합에 의해 이루어진다. 또한 원래의 상태와 추상화된 상태의 안정성(*soundness*)을 제공하기 위하여 *homomorphism*을 정의하고 있다.

[8]의 연구는 전체 시스템에 대한 도달성 그래프를 한번에 생성하지 않고 상태 감소를 수행하는 방법으로 전체 시스템은 무한할 수 있으나 상태 감소가 이루어진 시스템은 유한하다. 이 연구에서는 *'largest equivalence relation'*개념을 사용하여 도달성 그래프 생성과 상태 감소를 동시에 수행하고자 하였다.

[9]는 디지털 하드웨어에 대한 상태 감소 방법을 제시한다. 이 연구는 기계의 집합을 입력과 출력의 값에 대해 *bisimulation* 관계를 통하여 동치 클래스를 생성하는 상태 감소 방법을 사용한다. 이 연구에서 제시하는 입력과 출력에 대한 동치 클래스 생성 방법은 입력과 출력이 다양하고 그 영역이 광범위한 소프트웨어의 상태 감소 방법으로 적용하기는 어렵다.

3. 명세 모델 기반 추상화

명세 모델의 추상화는 명세 모델이 가진 특정 구문을 정의하여 이 구문을 만족하는 노드와 전이를 추상화한다. 명세 모델의 추상화는 시스템 아키텍처가 가진 본래의 계층성뿐만 아니라 구문적 계층성을 생성함으로써 명세 모델을 단순화하고, 단순화된 명세 모델들을 조합함으로써 병렬 실행되는 시스템의 분석 복잡도를 낮출 수 있다.

3.1 명세 모델

시스템은 실시간 상태 기계인 RSM을 사용하여 명세한다[10]. 하나의 RSM은 다른 RSM과 병렬적으로 수행될 수 있으며 계층적으로 위치할 수 있다. RSM의 정의는 다음과 같다.

정의 3.1: RSM M은 $<\Sigma, N, n_0, F, T, V, C>$의 튜플로 정의된다. Σ는 유한한 포트의 집합을, N은 노드의 유한 집합을, $n_0 \subseteq N$은 시작 노드를 나타낸다. $F \subseteq N$는 최종 노드의 유한 집합이며, T는 전이 조건과 시간 제약을 가진 노드 간 전이의 유한 집합, V는 데이타 변수의 유한 집합을, C는 지역 시계(*local clock*)를 나타낸다.

RSM의 노드는 기계의 특정 순간에서의 상태(동작 위치)를 명세한다. 명세 모델 기반 추상화는 상태를 나타내는 노드와 전이가 정의된 특정 구문을 만족하는 경우 노드와 전이를 추상화한다. 명세 모델에 적용되는 추상화는 비조건 내부연산 추상화와 구조적 추상화이다.

그림 2와 3은 RSM 명세 예제로, 이중선으로된 사각형은 하나의 RSM 기계를 나타낸다. M_1은 *E_ch* 포트를 통하여 M_2와 통신을 수행함을 명세하며 기계의 동작은 화살표로 표현된 전이와 원, 사각형 등으로 표현된 노드를 통해 명세한다. RSM의 표기법에 대한 정의는 RSM에 대한 정의는 [10, 11]에 기술되어 있다.

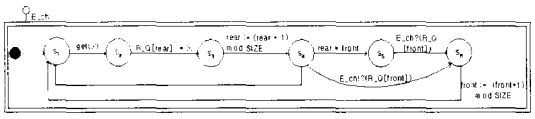


그림 2 RSM M_1

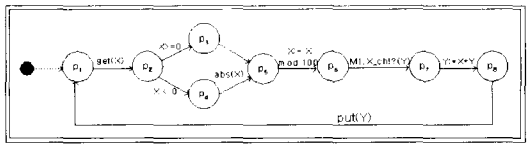


그림 3 RSM M_2

3.2 상태 공간

실시간 시스템이 가지는 상태는 자료, 시간, 위치 값으로 정의된다. 자료 공간은 시스템을 명세하는 과정에서 사용되는 변수의 도메인이며 특정 상태에서의 자료 값은 변수의 도메인을 벗어나지 않는다. 시간 공간은 RSM이 실행될 때 경과되는 지역 시계 값이다. 위치 공간은 각각의 RSM이 가지는 노드의 값을 나타낸다.

실시간 시스템을 명세하는 과정에서는 각 RSM의 동작은 위치 공간 즉, 노드를 중심으로 명세된다. 명세된 모델로부터 시스템의 실행 상태를 자료 값과 시간 값, 각 RSM의 위치 값을 사용하여 모델링한다.

3.3 비조건 내부연산 추상화

기계 내부에 표현된 노드와 전이 명세 중 다른 기계와의 상호 작용을 나타내는 경우나, 실행 시 발생되는 상태의 분기에 관여하는 조건 문을 갖지 않은 노드와 전이는 실행 상태 분석 과정에 많은 영향을 미치지 않는다. 이와 같이 통신과 조건의 속성을 갖지 않는 전이와 노드의 관계를 UIC-관계(UIC-relation)라 정의하고 UIC-관계에 있는 전이와 노드를 추상화하는 것을 UIC(*Unconditionally Internalized Computational*) 추상화라 한다. UIC 추상화는 추상화 과정 중 잘못 생성될 수 있는 전이 관계를 방지하기 위해 추상화 과정에 제약을 둔다.

정의 3.2: (UIC relation) RSM의 노드 s_i, $s_{i-1} \in N$와 전이 $t_i \in T$에 대해서 t_i의 원시 노드가 s_i이고 목적 노드가 s_{i+1}일 때 s_i를 원시 노드로 하는 전이가 t_i(1개)뿐이고, t_i의 전이의 내용이 다른 RSM과의 상호 작용과 논리 연산을 포함하지 않을 때 s_i와 s_{i-1}의 관계를 t_i에 의한 UIC 관계(UIC relation)라 정의한다. □

그림 4의 (a)에서 d_1과 d_2, d_2와 d_3가, (b)에서는 d_1과 d_2, d_2와 d_3, d_3와 d_4, d_4와 d_5가 각각 할당하는 연산만을 가진 전이이므로 UIC-관계에 있다. UIC-관계를 가진 노드와 전이를 추상화하여 생성된 노드를 UIC-모드라하고, UIC 추상화를 수행한 결과 생성되는 RSM을 UIC-RSM이라 한다.

정의 3.3: (UIC 추상화) UIC 추상화는 노드와 전이 사이의 UIC 관계를 기반으로 하여 다음과 같은 두 가지 경우에 따라 이루어진다:

경우 1) 노드 s_i, s_{i+1}, s_{i-2}에 대해 s_i, s_{i-1}가 전이 t_i에 의해 UIC 관계에 있고 s_{i+1}과 s_{i-2}가 UIC 관계가 아닐 때 s_{i+1}이 s_i로부터가 아닌 다른 노드로부터의 전이를 가진 경우:

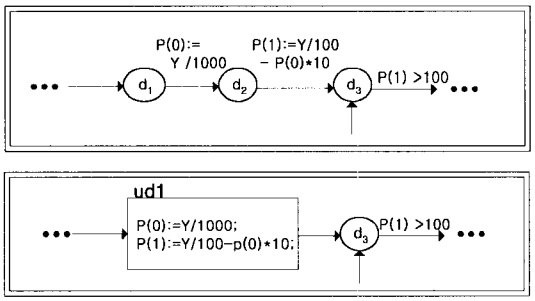
UIC 모드는 t_i까지 만을 추상화하고 추상화된 UIC 모드로부터 s_{i+1}로의 레이블이 없는 전이를 생성하며 추상화를 수행한다.

경우 2) 경우 1이 아닌 경우:

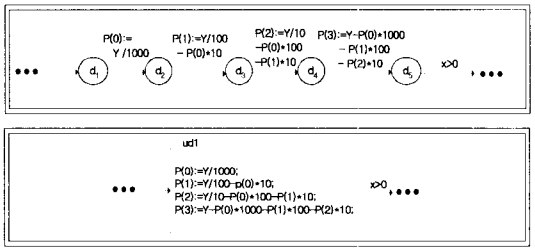
UIC 관계에 있는 노드와 전이를 UIC 모드로 추상화한다. □

그림 4는 UIC 경우에 대한 추상화 예를 보여준다. 두 경우 모두 UIC 관계를 가진 노드와 전이가 ud1의 UIC 모드로 추상화된다. 경우 1에서는 노드 d_2가 UIC 관계

에 있으나 다른 노드로부터의 전이를 가졌으므로 d_2는 추상화하지 않는다.



(a) 경우 1



(b) 경우 2

그림 4 비조건 내부 연산 추상화 예

UIC 추상화는 명세 모델이 가진 비조건 내부 연산 관계에 있는 노드와 전이를 추상화함으로써 노드 전이 관계가 논리적 조건과 상호 작용, 시간 제약을 가진 전이에만 초점을 둔 명세 모델이 생성된다. 그림 5, 6은 그림 2, 3의 RSM에 UIC추상화를 적용한 결과이다.

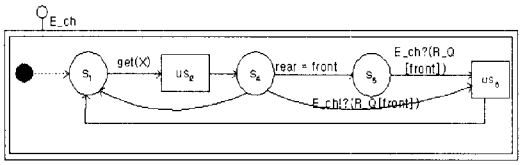


그림 5 UIC-RSM M_1

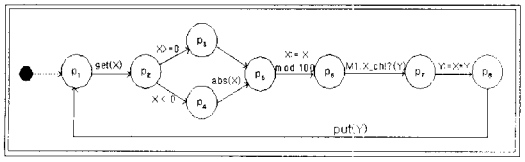


그림 6 RSM M_1

3.4 구조적 추상화

명세 모델이 가진 구조를 추상화하기 위해 본 논문에서는 정규적 선택 구조(*Regular Selection Structure*, RS Structure)와 정규적 반복 구조(*Regular Repetition Structure*, RR Structure), 혼합 구조(*Merged Structure*, MG Structure)를 정의한다. RS, RR, MG 구조를 정의함으로써 시스템이 가지는 구조를 추상화하여 시스템에 구문에 의한 계층성을 제공한다. 구조에 의한 계층성을 통하여 실행을 구조가 생성하는 계층적 관계를 통해 복잡도가 낮은 단계에서 분석할 수 있다.

RS 구조는 그림 7과 같이 특정 노드(s_i)로부터의 선택된 결과가 단일 노드(s_k)로 전이되는 구조이다. 단, 구조 내부의 전이는 실행 모델을 생성하기 위한 상대 RSM과의 동기를 나타내는 전이를 포함하지 않는다.

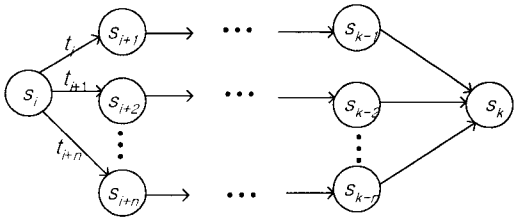


그림 7 정규적 선택 구조

RR 구조는 그림 9과 같이 s_i에서 s_k로의 전이가 존재하고 s_k의에서 발생하는 전이 중 한 개의 목적 노드가 s_i인 반복 형태를 갖는 구조를 나타낸다. 단, 구조 내부의 전이는 실행 모델을 생성하기 위한 상대 RSM과의 동기를 나타내는 전이를 포함하지 않는다.

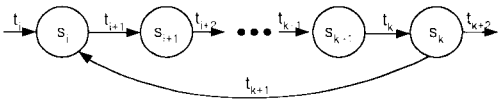


그림 8 정규적 반복 구조

두 개 이상의 RR, RS 구조가 일부 노드들을 공유하여 겹쳐지는 경우가 발생하는데 이와 같이 일부 노드들을 공유하여 발생하는 구조는 혼합 구조라 정의한다.

정규적인 형태를 가진 구조는 명세 모델이나 실행 모델 상에서 다중으로 포함되어 발생할 수 있다. 본 논문에서는 정규 구조들 간의 포함 관계를 단층 구조(*Flat Structure*)와 중첩 구조(*Nested Structure*)로 정의한

다. 정규적 구조가 명세 모델의 계층성을 제공할 때 중첩 구조는 상·하위 포함 관계에 따라 명세 모델과 실행 모델에 다중 계층을 제공한다.

정의 3.4: (구조적 추상화) RSM 또는 UIC-RSM이 RR, RS, MG 구조를 가졌을 경우 이 구조를 추상화하는 것을 구조적 추상화라 한다. 구조적 추상화를 추상화하였을 때 발생되는 새로운 노드를 S 모드(*S-mode*)라 하고 이 RSM을 S-RSM이라고 정의한다. □

그림 9, 10은 그림 5, 6의 UIC RSM에 구조적 추상화를 적용한 결과이다. 그림 5에서 s_1, us_2, s_4가 가진 RR 구조를 추상화한 결과가 그림 9의 ss_1노드이다. 그림 10의 sp_2는 그림 6이 가진 RS구조(p_2, p_3, p_4, p_5)를 추상화한 노드이다.

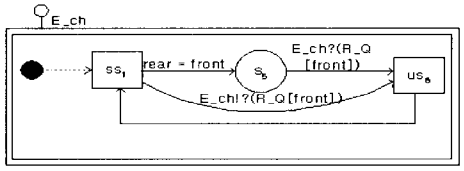


그림 9 S-RSM M_1

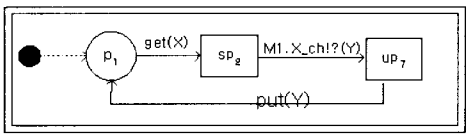


그림 10 S-RSM M_2

3.5 시스템 모델

각 명세 모델의 노드를 조합함으로써 병렬 실행되는 시스템이 발생시킬 수 있는 모든 상태를 노드의 위치 값을 중심으로 생성한다. 명세 모델의 조합은 각 명세 모델이 가진 노드와 전이의 조건을 통해 조합된 노드와 전이를 가지며 이를 시스템 모델이라 정의한다. 시스템 모델을 정의하면 다음과 같다.

정의 3.5: (시스템 모델) 조합되는 명세 모델의 집합을 $S = \{M_1, M_2, ..., M_n\}$라 할 때, 각 명세 모델의 노드의 집합이 $N_1, N_2, ..., N_n$이고 전이의 집합이 $T_1, T_2, ..., T_n$이라면 시스템 모델의 노드 집합은 $N_1 \times N_2 \times ... \times N_n$이고, 전이의 집합은 $\bigcup_{i=1}^{n} \times T^j$, $(1 \le i \le n,\ 1 \le j \le n)$의 부분집합이다. 단, 명세 모델의 전이의 집합 중 동기적인 전이에 대한 조합의 전이는 동시에 만족할 때만 생성된다. □

그림 11은 그림 9, 10의 시스템 모델이다. 상호 작용하는 두 개의 RSM이 존재할 때, 시스템 모델은 두 RSM이 가진 모든 노드와 전이를 조합하여 발생가능한 노드를 생성한다. 시스템 모델의 노드는 RSM의 위치 값을 조합한 것이며 전이는 두 개의 RSM이 병렬 수행될 때 발생할 수 있는 각 RSM의 위치 값을 나타낸다.

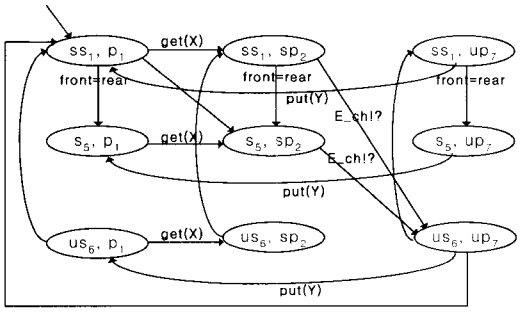


그림 11 S-RSM M_1, M_2의 시스템 모델

4. 실행 모델 기반 추상화

하나의 시스템은 여러 개의 서브시스템이 독립적으로 실행되거나 통신을 통한 상호작용을 한다. 실행 모델은 병렬 시스템의 동작을 모델링하여 구현되어질 시스템의 속성 검증 및 실행 예측을 가능하게 한다. 그러나, 실행 모델을 생성하는데 관여하는 시스템이 많아질수록 생성되는 실행의 상태는 큰 폭으로 증가하게 된다. 명세 모델과 마찬가지로 실행 모델에 추상화를 적용하여 계층을 생성함으로써 실행 정보를 복잡도가 낮은 상위 계층에서 시스템의 실행 정보나 시스템간의 상호 작용 등을 분석할 수 있다.

4.1 실행 모델

실행 모델은 명세된 시스템의 실행 정보를 모델링 한 것으로 도달성 그래프의 형태로 표현된다. 즉, 실행 과정에서 도달 가능한 상태를 시스템 모델로부터 생성한다. 실행 모델의 노드는 자료와 시간, 위치 값으로 정의되며 노드의 위치 값은 시스템 모델의 노드에 따른다. 에지는 실행 시 발생되는 전이에 의해 결정되며 시스템 모델이 가진 전이 집합의 부분 집합이다.

정의 4.1: 주어진 일련의 RSM의 집합 $S=\{M_1, M_2, ..., M_n\}$, $n \ge 1$,의 실행 모델은 그래프 $G<N, n_0, E>$의 3-튜플로 정의한다.

- N은 도달성 그래프의 노드의 집합으로 RSM들의 노드와 자료, 시간으로 정의된다.
- $n_0 \in N$은 그래프의 시작 노드이며,

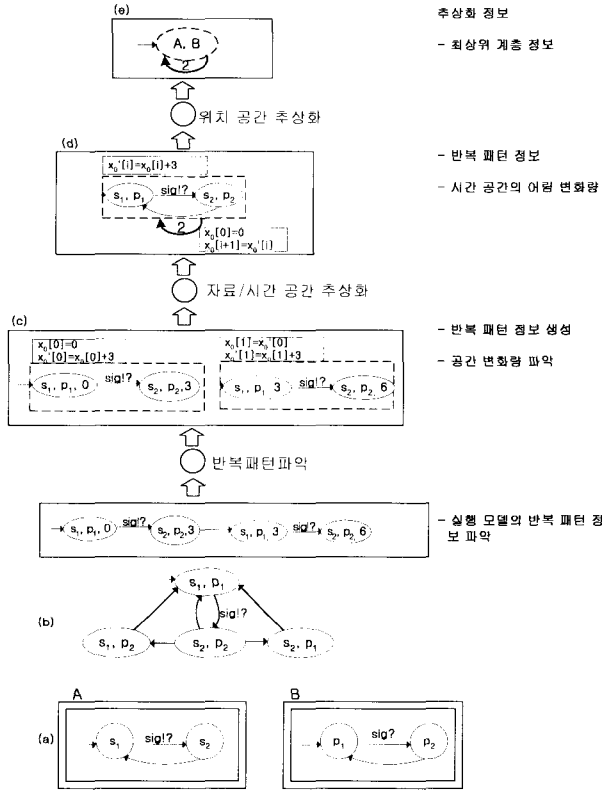


그림 12 위치, 가료 공간 추상화와 예제의 의미 추상화 과정

· $E = N \times T \times N$는 에지의 집합으로 T는 N에서 발생 가능한 전이들의 집합이다. □

4.2 의미 추상화

의미 추상화는 실행 모델에 적용되는 추상화를 대표하는 것으로 실행 모델의 노드가 가진 위치, 자료, 시간의 변화 관계나 특정 범위 관계와 같은 의미를 파악하여 추상화하는 방법이다. 실행 모델에 의미 추상화를 적용하여 구체적이지 않은 개략적인 실행 정보를 복잡도가 낮은 상위 계층에서 제공할 수 있다. 의미 추상화의 종류로는 노드가 가진 위치 값에 따라 적용되는 위치 공간 추상화와 자료와 시간 공간에 적용하는 자료, 시간 공간 추상화가 있다.

정의 4.2: (위치 공간 추상화) 위치 추상화는 실행 모델의 노드가 RSM의 전이 관계에 의해 반복적으로 생성될 때, 즉, 일정한 패턴이 반복될 때 실행 모델의 반복되는 패턴을 추상화한다. 유/무한하게 생성될 수 있는 실행 모델의 노드와 에지를 위치에 의해 추상화한다. 위치 추상화에서는 추상화된 실행 노드와 에지가 몇 번 반복되는지 혹은 무한하게 반복되는지에 대한 정보를 예측 값 또는 실행 결과로서 추상화하여 표현한다. □

정의 4.3: (자료와 시간 공간 추상화) 노드를 정의하는 것 중 자료와 시간에 적용되는 추상화이다. 이는 위치 공간 추상화와 동시에, 또는 독립적으로 적용될 수도 있다. 자료 공간 추상화는 노드를 정의하는 값 중 자료 값이 일정한 비율 또는 규칙성을 가지고 변하거나 특정 범위에 존재하는 경우와 같이 자료 값이 특정 의미로 표현될 수 있을 경우 그와 관련한 실행 모델의 노드와 전이를 추상화한다. 시간 공간 추상화는 노드를 정의하는 값 중 시간 값과 관련한 것으로 적용 방법은 자료 공간 추상화와 같다. □

그림 12는 위치 공간 추상화와 자료, 시간 공간 추상화에 대한 예와 추상화 과정을 나타낸다. 그림 중 (a)는 병렬 실행하는 RSM으로 두 RSM은 동기적 통신을 수행한다. (b)는 (a) RSM과의 조합 결과인 시스템 모델로 두 RSM의 노드의 조합을 노드의 위치 값으로 가지며, 두 RSM이 가진 전이의 집합을 조합의 전이로 가진다.

(c)는 실행 모델로부터 유도된 실행 결과로 특정 경로가 2회 반복되어 수행되었다. 자료 변수 x가 1회 반복 때

마다 3씩 증가하는 실행 모델이다. 추상화 과정을 설명하기 위해 (c)에서의 상태는 위치 공간과 자료 공간만으로 정의하였다. (c)의 노드를 감싼 점선으로 된 사각형에 표현된 정보는 동일 위치 값의 반복을 추상화하기 위해 각 실행에 대한 정보를 추상화한 것이다. 즉, 반복되는 노드들을 추상화할 때 변화하는 값의 패턴을 표현한 것이다. x_0는 실행 모델이 가진 첫 번째 위치 공간 패턴을 의미한다. $x_0[0]$은 패턴이 반복될 때, 반복되는 패턴 중 첫 번째 반복에서의 변수 x의 초기 값을 나타낸다. $x_0'[0]$은 $x_0[0]$으로 표현된 값의 실행에 의한 결과 값을 의미한다. 동일하게 $x_0[1]$은 x_0 패턴이 두 번째 반복되는 경우의 초기 값을, $x_0'[1]$는 결과 값을 의미한다.

(d)는 (c)에 대한 위치 추상화를 적용하기 위해 동일 위치 값을 가진 노드로 에지를 연결하였다. 반복되는 위치공간 패턴 내에서의 변수 x 값의 변화는 1회 반복마다 3씩 증가함을 $x_0'[i] = x_0[i] + 3$으로 표현하였다. 또한, 각 반복 패턴이 다시 시작될 때 새로운 패턴의 시작이 갖는 변수의 값이 이전 실행의 변수 값을 변화 없이 실행하기 때문에, 반복 사이의 x 값의 변화는 $x_0[i+1] = x_0'[i]$로 표현하였다. 즉, 그림 9는 패턴이 두 번 반복되는데, 두 번째 반복 패턴에서의 x의 시작 값($x_0[1]$)은 패턴의 첫 번째 실행의 결과 값인 $x_0'[0]$을 가진다는 것을 의미한다. 패턴의 첫 실행의 초기 값은 $x_0[0] = 0$으로 표현하였다.

(e)는 위치, 자료 공간 추상화가 적용된 최종 실행 모델이다. 점선으로 된 노드는 (b)에서 표현된 시스템 상태의 일부를 추상화한 했다는 것, 즉, (d)의 위치 공간 패턴의 추상화를 의미한다.

4.2 의미 추상화와 구문 추상화의 결합

의미 추상화를 수행할 때 실행 모델이 가진 위치나 자료, 시간 값의 변화를 추상화뿐만 아니라, 구문 추상화에 의해 생성된 각 구문 모드의 정보를 의미 추상화 시 고려한다. 즉, 실행 모델의 특정 실행 경로가 구문 추상화에 의해 추상화된 모드와 같을 때, 구문 추상화를 근거로 실행 정보를 추상화한다. 이 결과 구문 추상화의 구문 정보가 실행 모델에 투영되어 분석이 보다 용이해진다. 예를 들어 구문 추상화과정에서 RR 구조를 가진 노드들에 대한 실행 모델을 추상화할 경우 명세 모델의 RR 구조 정보를 이용한다.

4.3 추상화 정보의 표현 방법

명세 모델에서의 추상화 정보는 UIC 모드 또는 S 모드와 같은 추상화된 노드와 전이를 대표하는 노드를 정의하였다. 실행 모델 또한 명세 모델과 같이 실행 노드와 전이가 추상화되었음을 노드를 통하여 표현한다. 그러나, 명세 모델과 달리 실행 모델은 자료와, 시간, 노드의 위치 값에 대한 추상화 정보를 표현할 수 있어야 하며, 이를 위해 부가적인 표기법을 정의하였다. 표 1은 표기법의 정의를 보인다.

표 1 추상화 정보를 표기하기 위한 기호

표기	내용	예
+a, -a (단, a는 상수)	· 자료, 시간 값의 증/감 폭을 나타냄. · 일정한 값으로 증/감.	$t' = t-3$, $t' = t+3$ · 변화한 t의 값(t')은 이전 값(t)에 ±3의 변화 값을 가진다.
+[a, b], -[a, b] (단, a, b는 상수)	· 자료, 시간 값의 증/감 폭을 나타냄. · a와 b사이의 값만큼 증/감.	$t' = t-[4, 10]$, $t' = t+[4, 10]$ · 변화한 t의 값(t')은 이전 값(t)에 4와 10사이의 한 값만큼 증가, 감소한 값이다.
<, <=, >, >=	· 자료, 시간 값이 특정한 범위에 있음을 나타내는 기호.	$10<t<=20$ · t의 값은 10 초과 20미만의 범위에 존재한다.
~	· 자료, 시간 값의 추상화된 정보를 알 수 없는 경우를 나타내는 기호.	$t = \sim$ · t의 값은 알 수 없음.
repetition #	· 위치 값의 반복을 나타내는 기호, 반복 회수를 레이블로 가짐.	· 그림 5의 (d), (e).

4.4 추상화 단계 정리

3절과 4절에서 기술한 명세 모델에 구문을 바탕으로 적용하는 추상화와, 실행 모델에 자료, 시간, 공간의 의미를 바탕으로 추상화하는 단계를 도표로 나타내면 그림 13과 같다. 실행 모델은 더 이상 추상화가 되지 않을 때까지 반복적으로 의미 추상화를 수행한다.

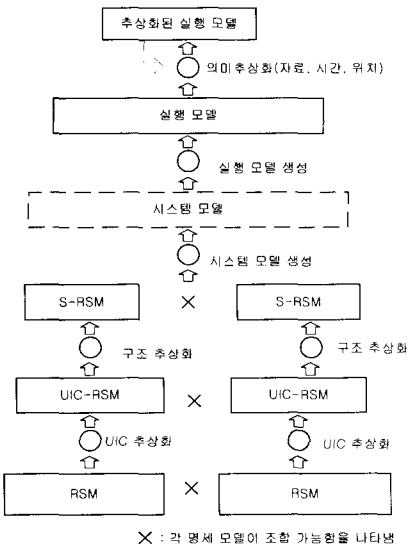


그림 13 구문/의미 추상화 단계

그림 14는 추상화 단계를 나타낸 알고리즘이다.

```
Abstraction()
begin
      //각각의 모델에 대한 구문 추상화 수행
      abstract_spec_model[] = syntax_abstraction(model[]);

      //시스템 모델 생성
      system_model =
      generate_system_model(abstract_spec_model[]);

      //실행 모델 생성
      execution_model =
      generate_execution_model(system_model);

      //의미 추상화 수행
      abstract_exection_model = execution_model;

      while(exist abstractable node) //추상화될 수 있는
      노드가 없을
      begin                          //때 까지 수행
            abstarct_execution_model =
                  semantic_abstraction
                  (abstract_exectuion_model);
      end
end
```

그림 14 추상화 과정 알고리즘

5. 예제

5절은 논문에서 정의한 추상화를 적용한 예제를 기술한다. 예제 RSM은 그림 2, 3에서 보여준 RSM으로 RSM M_1은 사용자로부터 수를 입력받아 원형 큐에 보관하고 RSM M_2로부터 요청이 있을 때 입력받은 순서대로 수를 전달해 주는 작업을 반복적으로 수행하는 동작을 명세한 것이다. M_1과 M_2에 적용되는 추상화 결과는 각 절에서 보였다. 5절에서는 각 단계별 시스템 모델 예와 실행 모델 예를 보인다.

시스템 모델 생성과 계층 관계

RSM을 조합한 시스템 모델의 복잡도는 노드의 수가 가장 적은 S-RSM을 조합한 시스템 모델이 가장 낮다. 그림 15은 그림 9, 10의 S-RSM M_1, M_2이 각각 3개의 노드를 가졌기 때문에 최대 9개의 노드가 생성되었으며 이 레벨에서는 노드들의 전이 관계에 영향을 주는 조건 전이, 동기화 전이, IO등에 대한 분석이 가능하다.

그림 16은 UIC-RSM M_1, M_2의 조합 결과인 시스템 모델이다. S-RSM의 시스템 모델 보다 복잡도가 높은 것을 볼 수 있다. 이 단계에서는 구조가 포함하던 정규적 반복 구조와 선택 구조에 대한 상세 노드와 전이 관계를 볼 수 있다.

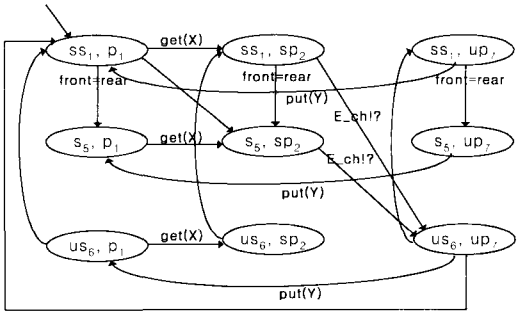


그림 15 S-RSM M_1, M_2의 시스템 모델

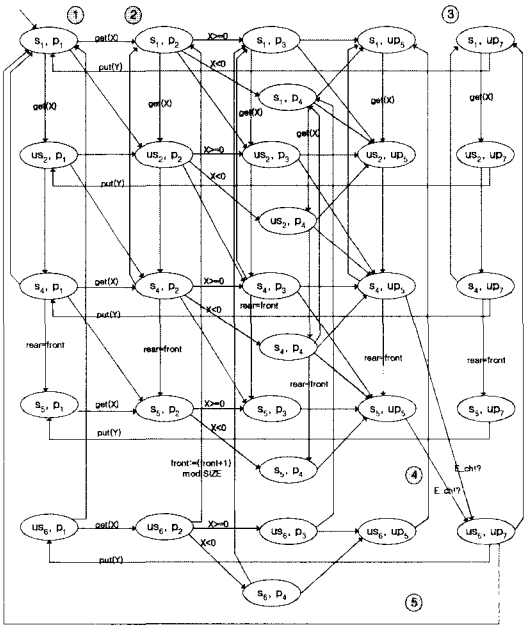


그림 16 UIC RSM M_1, M_2의 시스템 모델

각 추상화 수준에 대한 시스템 모델은 계층적 관계를 가진다. 그림 16에서 ①은 ss_1과 p_1의 조합, ②는 ss_1과 sp_2의 조합, ③은 ss_1과 up_7의 조합에 해당한다. ④는 s_5와 sp_2의 조합, ⑤는 us_5와 sp_2의 조합에 해당한다.

UIC 추상화를 수행하지 않은 RSM의 시스템 모델은 그림 17과 같이 생성된다. 그림 17은UIC RSM을 통해 생성된 시스템 모델보다 많은 노드를 가진다. 이는 RSM M_1, M_2의 모든 노드와 전이의 조합 관계를 표현하기 때문이다.

실행 모델 생성과 의미 추상화 적용

본 논문에서 각 RSM의 실행 모델은 시스템 모델로

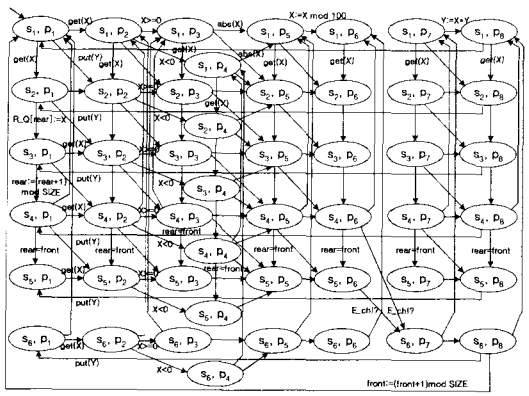


그림 17 RSM M_1, M_2의 시스템 모델

부터 생성한다. 즉, 위치 중심으로 조합되었던 노드와 전이를 실행했을 때 발생하는 시간과 자료 값을 포함하여 실행 모델로 나타낸다. RSM의 실행 모델은 무한하거나 유한한 노드와 에지를 생성할 수 있으나, 생성 과정에서 실행 모델을 위치, 자료, 시간 공간 추상화를 통하여 실행 모델의 노드와 에지의 수를 감소시킴으로써 계층적 상위 단계에서 실행에 대한 대략적 정보를 분석할 수 있도록 한다.

실행 모델의 노드의 위치 값은 시스템 모델의 위치 값과 같으며 실행 시 생성되는 자료 값과 시간 값을 갖는다. 실행 모델의 에지는 시스템 모델의 에지와 같으며 시간 제약을 가진 전이일 경우 발생 시간 값을 갖는다. 노드와 에지의 생성 규칙은 [10]을 따른다.

예를 들어, UIC-RSM M_1과 M_2의 실행 모델이 그림 16로부터 그림 18와 같이 생성되었다고 하자. RSM의 실행 모델 생성은 복잡도 관계로 생략한다. 여기에서는 추상화의 적용 과정을 쉽게 기술하기 위해 실행 모델의 자료 값은 고려하지 않고, 시간 만을 고려하였다. '*get*

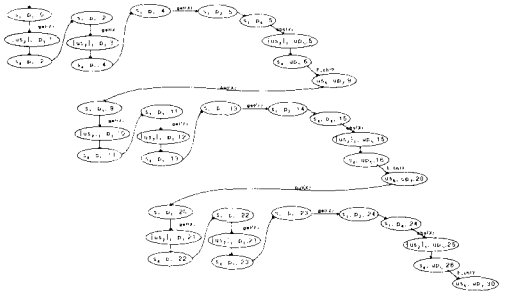


그림 18 UIC-RMS 시스템 모델의 실행 모델

(X)'는 입력문으로 1 단위 시간을 소모하고, 동기화 전이에 필요한 시간은 3~4 단위 시간이 걸린다고 했을 때, 각 노드에 표기된 수는 이에 따른 시간 값을 의미한다.

그림 18의 반복되는 위치 패턴에 따라 분석하면 그림 19에서처럼 특정 경로가 3회 반복되는 패턴을 가지며, 반복 패턴 내에 또 다른 위치 반복 패턴과 구조에 관한 실행 경로가 분석된다. 패턴 내의 반복 정보는 시스템 모델로부터 알 수 있는 정보이다. 그림 18의 UIC-RSM의 실행 모델을 위치, 시간, 자료 공간 추상화와 S RSM의 조합을 통해 파악된 구조를 적용하여 위치 공간 패턴을 수행하면 그림 21과 같이 추상화한 정보로 표현된다.

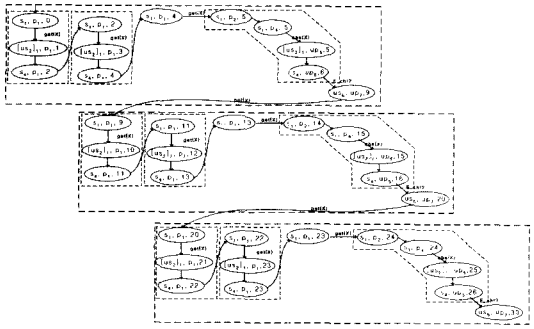


그림 19 UIC-RSM 실행 모델의 반복 패턴 및 구조 분석

그림 19를 살펴보면 노드 중 실행 경로의 일부(p_2→p_1→up_5를 포함한 노드)가 S-RSM의 구문 구조를 가졌으며 이에 따라 "ss_1, sp_2" 노드로 추상화하였다. ss_1과 sp_2는 구조적 추상화에서 추상화한 s-모드이다. 이 때, ss_1과 sp_2의 조합에 의해 생성된 노드는 sp_2의 모든 노드가 생성한 경로가 아닌 여러 가지 선택 경로 중 한 가지 경로가 선택되어 실행되었다는 정보를 표현하기 위해 점선으로 표기하였다.

추상화된 노드는 추상화하기 전 실행 모델이 가졌던 노드와 전이에 따른 시간 변화에 대한 추상화 정보를 가진다. 또한, 내부 반복 패턴을 추상화하기 위해 반복되는 패턴이 가진 노드의 위치와 시간 값을 추상화된 형태로 표현하였다. 그림 20에서 노드 "ss_1, p_1"는 반복되는 패턴으로 2회씩 총 6번 반복된다. 2회의 반복은 연속적으로 수행되며, 이에 대한 관계를 시간 값의 변화에 대한 표현으로 나타내었다. 즉, 첫 번째 노드에서 "$t_0[0]$ =0"는 첫 번째 패턴의 첫 번째 수행에 대한 초기 값이 0임을 나타내고, 패턴 1회의 수행에 대한 결과 값, t_0' [0],은 초기 값에 대한 증가치 $t_0[0]$+2를 가진다. 두 번째 반복은 첫 번째 반복의 결과 값을 그대로 가지므로

$t_0[1]=t_0'[0]$로 표현된다. 두 번째 반복의 시간의 변화치도 첫 번째 반복과 같은 형태를 가진다. 마찬가지로, "ss_1, sp_1"노드에 표현된 시간에 대한 추상화 정보도 초기 값과 실행에 의한 변화치를 가진다.

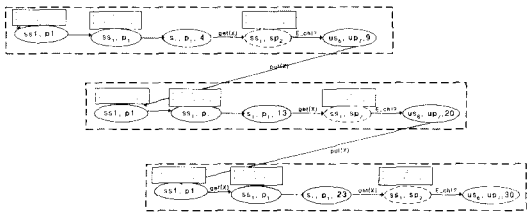


그림 20 UIC-RSM의 실행 모델을 추상화한 결과

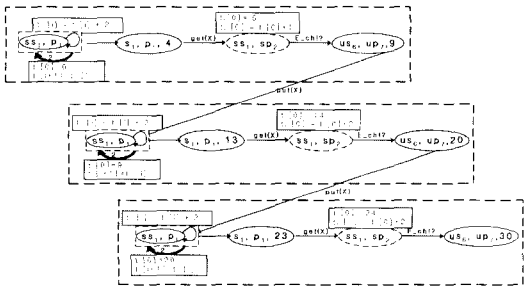


그림 21 1차 위치 공간 추상화를 적용한 실행 모델

그림 21는 그림 20의 실행 모델이 가진 노드가 반복되는 패턴을 가진 점을 추상화한, 즉 위치공간 추상화를 적용한 실행 모델이다. 반복되는 노드는 굵은 화살표를 가지며 화살표 안쪽에 있는 수로 반복 회수를 나타낸다. 이때, 노드에 표현된 정보는 1회 실행 시 변화되는 시간을 표현한 것이다. 굵은 화살표가 가지는 추상화 정보는 패턴이 처음 실행되는 초기 값과 반복이 실행될 때 이전 결과 값이 실행에 시작에 미치는 값의 변화, 즉, 반복과 반복 사이에 존재하는 시간의 변화에 대한 정보를 가진다. 그림 21의 반복에 대한 시간 정보는 재실행될 때의 시간 값은 이전 실행의 마지막 시간 값과 같다는 것을 의미한다.

그림 21이 가진 3회 반복 패턴(점선 사각형)을 추상화하면 그림 22와 같이 추상화할 수 있다. 추상화된 각각의 노드는 실행에 따른 시간 정보를 초기 값과, 결과 값의 변화 정보를 추상화한 형태로 가진다. 여기에 반복되는 패턴에 따른, 즉, 위치 공간 추상화를 수행하면 그림 23의 최종 추상화된 실행 모델을 생성할 수 있다. 그림 22를 분석하면 3번 반복되며 처음 실행으로부터 1회 반복될 때마다 시간 소비가 9~10 단위 시간이며, 재실행될 때 시간 값의 변화는 없다는 정보를 알 수 있으며 이러한 정보는 그림 23에 표현된다.

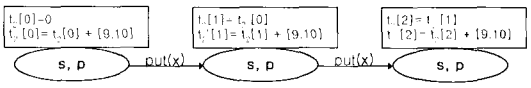


그림 22 반복되는 실행 경로를 추상화한 모델

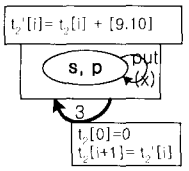


그림 23 2차 위치 공간 추상화 결과

예제 분석

표 2, 3, 4는 예제의 명세 모델과 실행 모델에 추상화를 적용했을 때 변화하는 노드와 전이 수를 분석한 표이다. 명세 모델은 구문 추상화 결과 상태와 전이 수가 감소했으며 그 결과로 시스템 모델의 노드와 전이의 수가 급격히 줄었음을 알 수 있다.

표 2 명세 모델의 추상화에 따른 노드와 전이 분석

명세 모델	M_1		M_2	
	상태 수	전이 수	상태 수	전이 수
RSM	6	8	8	9
UIC RSM	5	7	6	6
S-RSM	3	4	3	3

표 3 각 추상화 단계별 시스템 모델의 노드와 전이 수 분석

	상태 수	전이 수
RSM 시스템 모델	6	8
UIC-RSM 시스템 모델	5	7
S-RSM 시스템 모델	3	4

표 4 실행 모델의 추상화에 따른 노드와 전이 수 분석

추상화단계		상태 수	추상화 정보
1단계	UIC 기반 실행모델	36	
2단계	구문 정보에 의한 추상화	15	구문 구조에 포함된 노드와 에지 정보 추상화
3단계	1차 위치 추상화	12	반복되는 실행 패턴 추상화
4단계	2차 위치 추상화를 위한 공간 추상화	3	반복되는 패턴의 정보와 시간/자료 공간 추상화
5단계	2차 위치 추상화	1	반복 패턴 정보를 추상화

표 5 비교 연구 분석

비교 항목		[2]	[3]	[4]	[5]	[6]	[7]	[9]
상태 감소 대상	syntax	×	×	×	×	×	×	×
	semantic	○	○	○	○	○	○	○
계층성		○	×	×	×	×	×	×
원시 정보 상실		×	△	○	△	○	○	×
상태 감소 영역	시간			○	○	○ (정보제거)		
	자료		○	○		○	○	
	위치	○		○				○
실행 패턴 파악	시간			○		○		
	자료		○				○	○
	위치							
상태 감소 방법		동기화 구조 분석	상태들의 부분집합들을 동치 클래스로 클러스터링	의미 없는 상태 값을 특정 표기(*)로 대신 표현	불가산 상태공간을 유한한 다수의 region으로 파티션	history equivalence와 transition equivalence를 사용하여 상태들을 동치 클래스로 분류	abstraction	input output equivalence를 이용한 상태 분류

6. 분 석

본 논문의 추상화 방법과 비교해 볼 수 있는 연구로는 [2, 3, 4, 5, 6, 7, 9]가 있다. 2절에서 간략하게 기술한 관련 연구들을 본 논문의 상태 감소 방법과 비교 항목을 두어 분석하면 표 6과 같은 결과를 볼 수 있다.

표에서 볼 수 있듯이 각각의 연구들은 다수의 시스템들이 동작할 때 발생되는 상태들의 감소, 즉, 의미(*semantic*)에 초점을 맞추고 있다. 또한 상태 감소 영역을 시간, 자료, 위치 중 한 상태 공간을 대상으로 하며, 평면(flat)적 관점에서 수행하므로 계층성을 제공하지 않는다. 계층성의 결여는 추상화되기 이전의 원시 정보를 쉽게 살펴 볼 수 없음을 의미한다고 할 수 있다. 이와 같은 면에서 본 논문의 상태 감소 방법이 가진 장점을 보면, 구문이나 실행 의미에 추상화를 적용함으써 명세가 가진 구조적 정보를 실행 의미에 적용하여 추상화할 수 있으며, 계층관계를 통해 추상화된 내용의 세부 내용을 탐색할 수 있으므로 원시 정보를 상실하지 않는다는 점이다.

상태 감소 분석

RSM M_1과 M_2에 구문과 의미 추상화를 적용할 때, 만약 M_1과 M_2의 노드의 수가 각각 n, m개라고 하자. 이에 대한 시스템 모델은 $n \times m$의 복잡도로 생성된다. 그러나, UIC 추상화 결과, 각각 $\frac{1}{k_1} \times n$, $\frac{1}{k'_1} \times m$으로 줄었다면, UIC-RSM의 시스템 모델은 $\frac{1}{k_1 \cdot k'_1}(n \times m)$의 복잡도를 가지며 생성된다. 다시, 구조적 추상화를 적용한 S-RSM의 노드의 수가 각각 $\frac{1}{k_2}\left(\frac{1}{k_1} \times n\right)$, $\frac{1}{k'_2}\left(\frac{1}{k'_1} \times m\right)$으로 줄었다면, 이에 대한 시스템 모델은 $\frac{1}{k_1 \cdot k_2 \cdot k'_1 \cdot k'_2}(n \times m)$만큼의 노드를 가진다. 이는 초기 RSM에 의한 복잡도보다 구문 추상화를 적용한 결과 $\frac{1}{k_1 \cdot k_2 \cdot k'_1 \cdot k'_2}$배 감소된 복잡도를 가진 시스템 모델을 생성한 것이며, 낮은 모델로부터 생성하는 실행 모델의 복잡도 또한 감소시킨다. 명세 모델이 가진 구조가 중첩되었다면 복잡도는 중첩된 구조에 관련된 노드의 수만큼 감소한다(여기에서, k_i, k'_i는 상수).

실행 모델이 가진 복잡도는 시스템 모델의 복잡도를 바탕으로 한다. 시스템 모델로부터 생성된 실행 모델의 노드 수가 f개라고 하자. 이때, 노드의 특정 패턴이 l회 반복된다면 노드의 수는 $f = a + b \times l$(단, b는 반복 패턴의 노드 수)이라고 할 수 있다. 이를 의미 추상화하면, 감소되는 상태의 수는 $a+1$ 개로 현저히 줄어들어, 낮은 분석 복잡도를 제공하게 된다.

표 6은 n과 m개의 노드 수를 가진 명세 모델의 복잡도와 노드의 수가 f개인 실행 모델의 분석 복잡도를 나타낸 표이다. 실행모델은 두 개의 반복 패턴을 가지고 있다.

표 6 상태 감소를 통한 복잡도 분석

추상화단계	명세 모델		시스템모델 노드 수
	M1	M2	
RSM	n	m	n×m
UIC 추상화	$\frac{1}{k_1}\times n$	$\frac{1}{k'_1}\times m$	$\frac{1}{k_1\cdot k'_1}(n\times m)$
구조 추상화	$\frac{1}{k_2}\left(\frac{1}{k_1}\times n\right)$	$\frac{1}{k'_2}\left(\frac{1}{k'_1}\times m\right)$	$\frac{1}{k_1\cdot k_2\cdot k'_1\cdot k'_2}(n\times m)$
	실행 모델		실행 모델 노드 수
	노드 수	패턴	
	f		$a+(b\times p+c)\times q$
1단계 의미 추상화	$f-(b\times p+c)\times q$ $+(1+c)\times q$	b의 p회 반복	$a+(1+c)\times q$
2단계 의미 추상화	$f-(b\times p+c)\times q+1$	$(1+c)$의 q회 반복	$a+1$

7. 결론 및 향후 연구

본 논문에서는 명세 모델과 실행 모델을 정의하여 각 모델에 적용되는 구문 추상화와 의미 추상화를 정의하였다. 구문 추상화가 적용된 명세 모델은 구문이 가진 구조에 따라 계층성을 생성하며 복잡도가 낮은 추상화된 명세 모델이 된다. 추상화된 명세 모델을 조합한 후 조합에 따른 실행 모델을 생성하여 실행 모델이 가진 시간, 위치, 자료 공간의 의미에 따라 추상화를 수행하는 방법을 기술하였다. 명세 모델과 실행 모델에 추상화를 적용함으로써 모델 즉, 시스템이 가진 명세 정보, 실행 정보를 최상위 계층에서는 개략적이긴 하지만 복잡도가 낮은 형태로, 최하위 계층에서는 정확한 정보를 높은 복잡도의 형태로 제공하여 시스템의 이해를 다양한 각도에서 제공할 수 있다.

향후 연구로는 추상화된 명세 모델을 통하여 추상화된 계층에서 적용할 수 있는 모형 검사(*Model Checking*) 기법을 정의, 적용하는 것이다. 추상화를 통한 모형 검사 연구[6, 7]는 계층성이 없는 모델로부터 생성된 실행 모델이 특정 속성들을 만족하는지 여부를 결정하는 형태이다. 본 연구에서는 그림 24와 같이 계층성에 따라 검증할 수 있는 속성을 분류하여, 시스템 간의 상호작용 여부와 같이 복잡도가 낮은 상위 계층에서 검증할 수 있는 속성은 높은 복잡도를 가진 하위 계층의 정보를 사용하지 않고서도 검증할 수 있도록 하여 시스템의 이해도뿐만 아니라 검증의 복잡도 또한 감소시킬 수 있는 연구를 지속적으로 수행할 것이다. 이를 위해서는 CTL[1]과 같이 추상화된 정보와 속성을 표현, 검증할 수 있는 논리의 정의가 필요하다.

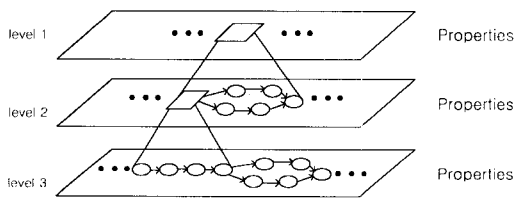


그림 24 모델의 계층성과 속성

명세 모델과 실행 모델에 대한 추상화를 적용한 시스템의 효율성을 보여줄 수 있는 구현과 사용 사례 적용이 향후 연구로 지속될 것이다.

참 고 문 헌

[1] 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*, pp. 244-263, April 1986, 8.

[2] W. J. Yeh and M. Young. Compositional Reachability Analysis using Process Algebra. In *Proc. of Conf. on Testing, Analysis and Verification*, pp. 49-59, August 1992.

[3] I. Kang and I. Lee. State Minimization for Concurrent System Analysis Based on State Space Exploration. In *Proc. of Conf. on Computer Assurance*, pp. 123-134, 1994.

[4] S. Raju. An Automatic Verification Technique for Communicating Real-Time State Machines. Technical Report 93-04-08, Dept. of Computer Science and Engineering, Univ. of Washington, April, 1993.

[5] R. Alur, C. Courcoubetis, N. Halbwachs, D.Dill, H. Wong-Toi, Minimization of Timed Transition Systems, In W. R. Cleaveland, editor, *CONCUR'92: 3rd Intl. Conf. on Concurrency Theory*, Lecture

Notes in Computer Science vol. 630, Springer, pp. 340-354, 1992.
[6] I. Kang, I. Lee and Y. Kim. An Efficient State Space Generation for the Analysis of Real-Time Systems. *IEEE Transaction on Software Engineering*, Vol. 26, No. 5, pp. 453-477, May 2000.
[7] Edmund M. Clarke, Orna Grumberg, David E. Long, Model Checking and Abstraction, In *Proc. of the 19th Annual ACM Symposium on Principles of Programming Laguages*, pp. 343-354, January. 1992.
[8] David Lee, Mihalis Yannakakis, Online Minimization of Transition Systems(Extended Abstract), In *Proc. 24th ACM Symposium on Theory of Computing*, pp. 264-274, May 1992.
[9] Adan Aziz, Vigyan Singhal, Gitanjali M. Swamy, Minimizing Interacting Finite State Machines: A Compositional Approach to Language Containment, In *Proc. of Intl. Conf. on Computer Design*, pp.255-263, October. 1994.
[10] 박지연, 이문근, 추상 시간 기계를 이용한 실시간 시스템의 도달성에 대한 검증 방법, 정보과학회논문지, 제28권 제3호, pp. 224-238, 2001년 3월.
[11] 노경주, 박지연, 이문근. 추상시간기계를 기반으로한 실시간 시스템의 시간명세와 분석. 한국정보과학회 소프트웨어공학회지, 제13권 제3호, pp. 45-54, 2000, 9.
[12] Kathi Fisler and Moshe Y. Vardi, Bisimulation Minimization in an Automata-Theoretic Verification Framework, In *Proceedings of International Conference on Formal Methods in Computer Aided Design*, Lecture Notes in Computer Science vol. 1552, Springer-Verlag, pp. 115-132, 1998.
[13] E. M. Clarke, D. E. Long, K. L. McMillan, Compositional Model Checking, In *Proceedings of 4th Annual Symposium on Logic in Computer Science*, pp 353-362, 1989.
[14] Douglas L. Long, Lori A. Clarke, Task Interaction Graphs for Concurrency Analysis, In *Proceedings of the International Conference on Software Engineering*, pp. 44-52, 1989.
[15] Jeffrey Fischer, Richard Gerber, Compositional Model Checking of Ada Tasking Programs, In *Proceedings of COMPASS'94*, pp. 135-147, 1994.
[16] Fabrice Derepas, Paul Gastin, and David Plainfossé, *Avoiding Sate Explosion for Distributed Systems with Timestamps*, Lecture Notes in Computer Science vol. 2021, pp. 119-134, 2001.
[17] Matthew B. Dwyer, David A. Schmidt, Limiting State Explosion with Filter-Based Refinement, In *Proceedings of the International Workshop on Verification, Model Checking and Abstract Interpretation*, 1997.

박 지 연
1999년 2월 전북대학교 컴퓨터과학과 졸업(이학사). 2001년 2월 전북대학교 전산통계학과 졸업(이학석사). 2001년 3월 ~ 현재 전북대학교 컴퓨터통계정보학과(박사과정). 관심분야는 정형기법, 실시간 시스템, 검증, 소프트웨어 재·역공학, 운영체제 등

조 기 환
1985년 전남대학교 계산통계학과 졸업(학사). 1987년 서울대학교 계산통계학과 졸업(석사). 1996년 영국 Newcastle 대학교 전산학과 졸업(박사). 1987년 ~ 1997년 한국전자통신연구원 선임연구원 1997년-1999년 목포대학교 컴퓨터과학과 전임강사. 1999년 ~ 현재 전북대학교 전자정보공학부 조교수. 관심분야는 이동컴퓨팅, 무선인터넷, 분산처리시스템, 컴퓨터통신

이 문 근
1989년 The Pennsylvania State University, Computer Science 학과 졸업(이학사). 1992년, The University of Pennsylvania, Computer and Information Science 학과 졸업(이공학석사). 1995년, The University of Pennsylvania, Computer and Information Science 학과 졸업(이공학박사). 1992년 5월 ~ 1996년 1월 미국, Computer Command and Control Company, Computer Scientist로 근무. 1996년 4월 ~ 1998년 3월 전북대학교 컴퓨터과학과 전임강사. 1998년 4월 ~ 1999년 2월 전북대학교 컴퓨터과학과 조교수. 1999년 3월 ~ 2002년 3월 전북대학교 전자정보 공학부 조교수. 2002년 4월 ~ 현재 전북대학교 전자정보 공학부 부교수. 관심분야는 소프트웨어 재·역공학, 실시간 시스템, 운영체제, 형식언어, 병렬함수언어, 컴파일러 등