대규모 유한 상태 기계의 근사 도달성 분석

정회원 홍 유 표*

Approximate Reachability Analysis of Large Finite State Machines

Youpyo Hong* *Regular Member*

요 약

유한 상태 기계(finite state machine)의 도달성 분석(reachability analysis)은 통신 프로토콜이나 마이크로 프로세서 설계 등의 다양한 컴퓨터 원용설계응용(computer-aided design applications)에 매우 유용하다. 도달성 분석은 정확한 도달가능상태를 계산하는 정해분석(exact analysis)과 도달 불가능상태의 일부만을 계산하는 근사분석(approximate analysis)으로 나뉘는데, 본 논문은 기존의 방법보다 크게 정확도를 향상시킨 근사 도달분석 기법을 소개하며, 그 기본적인 원리는 근사분석 알고리즘을 반복 적용하되 이전 근사분석 결과를 이후의 근사분석에 활용하는 반복적 근사 도달성 분석 (iterative reachability analysis)을 통해 근사분석의 정확도를 향상시킬 수 있도록 하는 것이다. 반복적 근사 도달성 방식을 이용하여 기존의 근사분석보다 크게 향상된 근사도달상태를 계산할 수 있음을 실험적으로 증명하였다.

ABSTRACT

Reachability analysis of finite state machines is very useful for many computer-aided design applications such as communication protocol or microprocessor design. We present new techniques to improve approximate reachability analysis. The key idea is to use an iterative approximate reachability analysis technique in which don't care sets derived from previous iterations are used to improve the approximation in subsequent iterations. Experimental results show that the new techniques can improve reachability analysis significantly compared to existing analysis techniques.

I. 서 론

유한 상태 기계(finite state machine)의 도달성 분석(reachability analysis)은 통신 프로토콜이나 마이크로 프로세서 설계 등의 다양한 컴퓨터 원용설계응용(computer-aided design applications)에 널리 사용되고 있다. 다양한 도달성 분석 방법 중 대표적으로 많이 사용되고 있는 binary decision diagram (BDD)[1]을 이용한 도달성 분석 기법은 Coudert[2]에 의해 처음 제안되었으며, 그 이래로 대규모의 유한 상태 기계를 분석하는데 널리 사용되어오고 있다. 그러나, 복잡한 통신 프로토콜이나 집적회로 등 대규모의 회로를 분석하려면 막대한 용량의 기억장치나 매우 오랜 계산시간이 소요된다는 한계가 있다. 이 때문에 최근까지도 도달상태 계산툴의 용량과 성능을 개선하기 위한 다양한 방법이 연구되고 있다. 이중 대표적인 방법중의 하나는 BDD내에서 변수의 순서를 바꾸는 것인데, 이는 BDD의 크기가 BDD 변수 순서에 매우 민감하게 의존하기 때문이다[3,4]. 이 이외에도 대규모의 유한 상태 기계 분석에 유용한 다양한 방식들이 제안되었다. 가령, Ravi 등의 연구[5]에서는 특정 상태 집합을 표현하기 위한 BDD가 지나치게 큰 경우 넓이 우선(breadth-first)과 깊이 우선(depth-first)방식을 혼용하여 그 부분집합을 작은 BDD로 표현함으로써 도달상태 분석툴의 용량을 크게 확장하였다. 그러나, 이 방식의 단점은

* 동국대학교 전자공학과
논문번호: 010096-0516, 접수일자: 2001년 5월 16일
※ 본 연구는 한국과학재단 산학협력연구(2000- 30200-005-1) 지원으로 수행되었음.

동일한 최종 결과를 계산하기 위하여 필요한 연산의 횟수가 지나치게 증가하는 경우가 많다는 점이다. Cabodi 등[6] 과 Narayan 등[7] 은 연산 횟수를 줄이기 위하여 상태 집합을 분리(partition) 하는 방식을 제안하였다. 또 다른 연구 경향중의 하나는 BDD의 크기를 줄이기 위하여 돈케어(don't care, DC) 정보를 이용하는 것이다. 예를 들면 Coudert 등[2]은 각 연산 단계마다 상태 영역 탐색 전위(a state space search frontier)로 사용되는 frontier set BDD 의 크기를 줄이는 방식을 제안하였다. 또한, Brayton 등[8]은 transition relation(TR) BDD의 크기를 줄이는 방식을 제안하였다. 그들은 비전위 상태들로부터의 천이와 계산 시점에서 이미 도달한 상태로부터의 천이를 돈케어로 사용할 것을 제안하였다. 이 방식의 단점은 전위상태와 도달상태가 매 단계마다 바뀌게 되므로 TR의 압축이 매 계산 단계마다 반복되어야 한다는 점이다.

한편, Ranjan 등[9]은 과도근사된 도달가능상태(overapproximated reachable states)로부터 추출된 돈케어를 이용하여 TR의 크기를 줄이는 방법을 제안하였다. 이 방식의 장점은 도달가능상태 계산 전 과정을 통하여 BDD압축이 한번만 수행되면 된다는 점이다. 그러나, 놀랍게도 그들의 실험결과에 의하면 BDD 압축이 더욱 큰 BDD를 생성하는 모순된 경우가 많았으며, 전체적인 성능 개선에 대한 실험 결과도 보고된 바가 없다.

본 논문에서는, 과도근사된 도달가능상태를 이용하여 근사 도달가능상태 계산결과의 정확성을 높이는 방법을 소개한다. 제안된 방식의 특징은 크게 다음과 같이 요약할 수 있다.

- 돈케어 BDD의 크기를 적절히 유지하며 고압축률을 유지하기 위한 클러스터드 돈케어 BDD 개념의 제안.
- 이전 단계에서 계산한 결과를 다음 단계의 돈케어로 활용하는 반복적(iterative) 근사 도달가능상태 계산.

II장에서는 배경이론을 설명한 뒤, III장에서는 새로이 제안되는 이론을, 그리고 IV장에서는 실험 결과를, 그리고 V장에서 결론을 제시한다.

Ⅱ. 배 경

1. 유한 상태 기계의 도달상태 계산

N개의 상태변수를 갖는 유한 상태 기계를 위한 차기상태함수는 천이함수벡터(transition function vector)라 불리는 TF_i ($1 \le i \le n$)로 구성되어 있으며, TR로 표시되는 transition relation[2]은 천이함수벡터들을 이용하여 모든 가능한 현재/다음 상태의 천이 관계를 다음과 같이 표현할 수 있다.

$$TR(x,y) = \prod_{i=1}^{n} TF_i(x,y)$$

여기서 x는 현재 상태와 입력 변수를, y는 다음 상태 변수를 나타낸다. TR을 표현하는 여러 가지 방법 중 대표적인 방법이 BDD를 사용하는 상징적 기법이다. 이 경우 TR을 단일 BDD를 사용하여 표현하면 그 BDD의 크기가 지나치게 커질 수가 있기 때문에, partitioned transition relation, 즉 TF_i를 몇 개의 상호 배타적인 그룹으로 나누어 표현하는 기법[10]이 제안되었으며, 이렇게 나누어져 논리곱(conjunction)을 통하여 결합된 TF_i 집합을 클러스터라고 부르며 기호로는 TR_i로 표현한다. 이 경우 TR은 클러스터를 사용하여 다음과 같이 표현된다.

$$TR = \prod_{i=1}^{n} TR_i(x,y)$$

여기서 n은 클러스터의 개수이다. 일단의 상태집합, From으로부터 한번의 상태 천이를 통해 도달가능한 상태집합 To는 TR을 나타내는 BDD에 existential quantification 연산을 적용함으로서 다음과 같이 계산될 수 있다.

$$To(y) = From(y) \cup \exists_{x_i \in x} [From(x) \cdot TR(x,y)]$$

여기서 논리곱연산은 많은 중간결과들을 생성하며 매 계산단계마다 반복적으로 수행된다. 만약 어떠한 클러스터들이 의존하지 않는 일단의 변수들이 존재하면, existential quantification은 부분 결과들에 걸쳐 분산될 수 있다. 특히, 전체 곱셈이 끝마쳐지기 전에 중간결과들로부터 그러한 변수들을 제거하면 중간 결과 BDD의 크기를 크게 감소시킬 수 있으며 이러한 기법을 조기 quantification이라고 부른다.

2. 돈케어를 이용한 BDD 압축

단일 출력을 갖는 부울함수 ff는 ff_{on}(on-set), ff_{off}(off-set), 그리고 ff_{dc}(DC set)의 세 가지의 값을 갖는다. 이들 중 어느 두 가지만으로 부분적(incompletely)으로 표현된 함수를 정확히 나타낼 수 있다. 불완전하게 표현된 함수 ff는 $f_{on} \supseteq ff_{on}$, f_{off}

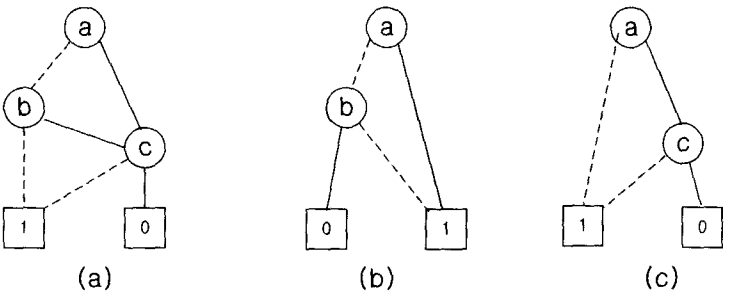

그림 1. (a) F BDD (b) C BDD (c) C BDD를 이용해 F BDD를 최소화한 결과

$\supseteq ff_{off}$, 그리고 $c = ff_{on} \cup f_{off}$를 만족하는 완전히 표현된 함수 [f, c]의 쌍으로 표현될 수 있다[10]. [f, c]는 일반적으로 [F, C]의 BDD 쌍의 형식으로 주어진다. 주어진 [F, C]에서 cover F'을 찾는 작업을 BDD 최소화 작업이라고 하고 이 때, F는 초기 BDD, F'은 최소화된 BDD라고 한다. 그림 1은 F BDD와 케어를 나타내는 C BDD 가 주어졌을 때 C BDD를 이용하여 F BDD를 최소화한 예를 보여주고 있다.

그러나 이러한 BDD 최소화는 그 복잡도가 NP-Complete인 것으로 밝혀졌으며, 이는 대부분의 큰 BDD를 최소화하는 것은 실질적으로 불가능함을 의미하며, 결과적으로 실제로는 BDD 압축 휴리스틱들[2,11]을 대신 사용하고 있다.

Ⅲ. 돈케어를 이용한 상징적 계산(Symbolic Traversal)

도달 불가능한 상태로부터의 상태천이는 TR에서 돈케어에 해당하는데 그 이유는 회로가 정상적으로 동작하는 한 도달 불가능한 상태에 도달할 수 없으며, 결과적으로 도달 불가능한 상태로부터의 상태천이는 실제로 도달 상태 계산 시 사용되지 않으므로 돈케어에 해당된다.

도달 불가능한 상태가 유용한 이유는 그 크기가 일반적으로 매우 크기 때문이다. 여기서 돈케어의 크기는 도달 불가능한 상태의 개수로 나타낼 수 있으나, 그 상대적 크기를 직관적으로 비교하기 위하여 존재하는 모든 상태의 개수 중 도달불가능한 상태의 개수를 백분율을 이용하여 표시한 값을 주로 이용한다. 많은 대용량 유한 상태 기계는 도달 불가능한 상태가 도달가능한 상태의 수천배 내지 수만배 이상이 되는 것이 일반적인데, 이는 대부분의 대용량 회로들은 계층적으로 결합하여 설계하며, 상태 부호화도 모듈별로 독립적으로 이루어지기 때문에 전체회로의 일부 상태만 도달 가능하기 때문이다.

도달 불가능한 상태가 효과적으로 사용되기 위한 중요한 전제 조건중의 하나는 그 계산이 쉽고 빠르게 수행될 수 있어야 한다는 점이다. 만약, 도달 불가능 상태의 계산 자체가 매우 어려운 일이라면 이를 돈케어로 활용한다는 것 자체가 무의미하기 때문이다. 여기서 주목할 점은 정확한 도달 불가능 상태 계산은 도달가능상태의 계산과 동등하며 그 계산은 매우 어려운 일임이 분명하다. 다행히도, 정확한 도달 불가능 상태 대신 도달 불가능 상태의 일부를 쉽게 구할 수 있고, 그 크기가 충분히 크면 TR을 압축하는데 효과적으로 이용할 수 있으며, 도달 불가능한 상태 중 일부는 근사 도달성 분석[12]을 이용하여 매우 쉽게 계산해 낼 수 있다는 연구 결과가 이미 발표된 바 있다. 본 연구에서는 이러한 돈케어들을 이용하여 근사 도달 상태 분석의 성능을 향상시키는 방법을 제안하고 있으며, 우선 데이터 구조의 크기는 작게 유지하면서 가급적 많은 돈케어를 표현할 수 있는 클러스터드 돈케어를 만들어 낸 뒤, 돈케어의 크기를 점진적으로 증가시키며 근사 분석을 반복적으로 수행하는 방법에 대해 설명한다.

1. 돈케어 천이와 클러스터드(clustered) 돈케어 BDD

현재까지 알려진 근사 분석 기법들은 대부분 상태공간 분해[11,12,13]를 기반으로 하고 있으며, 그 결과로 얻어지는 근사 도달상태는 분해된 상태공간마다의 도달가능상태들이 다. 예를 들어 어떠한 유한 상태 기계의 상태공간을 n개로 분리하여 근사 도달상태를 계산하면 n개의 BDD, $R^+_1, \ldots, R^+_n$가 각 상태공간별 도달상태를 나타내며, 이들의 논리곱이 총 도달가능상태의 근사 결과를 의미한다. 즉, 이러

한 R^+_i의 논리곱은 전체 상태 공간에서의 도달상태의 최소 superset를 표현하며, 그 역(complement)을 취하면 가장 큰 돈케어를 취할 수 있게 된다. 그러나, 실제로 이러한 논리곱의 결과 BDD는 그 크기가 매우 클 수 있다는 치명적인 문제점이 있다. 이러한 문제점을 해결하기 위한 가장 간단한 방법은 모든 TR_i에 대하여 R^+_i를 케어로 이용하여 개별적으로 압축하는 것이다. 그렇지만, 이러한 방법은 각각의 R^+_i로부터 얻은 돈케어의 크기가 비교적 크지 않기 때문에 TR_i 압축율이 상당히 저조할 가능성이 높다.

본 논문에서는 가급적 큰 돈케어 값을 유지하되 그 BDD 크기가 너무 크지 않은 BDD를 돈케어 BDD로 사용하기 위하여 클러스터드 돈케어(clustered DC)의 사용을 제안하는데, 클러스터드 돈케어는 각 TR_i에 대하여 그 TR_i의 압축에 기여할 확률이 높은 케어 BDD만을 추려 결합한 결과를 의미하며, 그 구체적인 계산과 활용 방법은 다음과 같다. 우선 이전 근사 도달상태 계산으로부터 R_0, R_1, . .R_i . , R_n을 구하였다고 가정한다. (여기서 R_i는 i 번째 sub-FSM, 즉 TR_i의 상태공간으로부터 구한 근사도달상태이다.) 만약 TR_i를 압축하고자 한다면 TR_i의 support set과 R_j의 support set을 관찰하여 최소 한 개 이상의 공통 변수가 있을 경우에만 그 R_j가 TR_i의 압축에 기여할 수 있다고 판단한다. 사용되는 BDD 압축 알고리즘에 따라, 이와 같은 support set조건을 만족하지 않는 R_j도 다른 R들과 결합된 뒤 TR_i의 압축에 기여할 수 있는 가능성을 배제할 수 없으나, 그 가능성은 일반적으로 매우 적기 때문에 이러한 기준을 택하였다. 이와 같은 기준에 의하여 채택된 R_j만을 추려 그들을 논리곱 연산을 이용하여 통합한 것을 클러스터드 돈케어라고 부르며 TR_i를 위한 클러스터드 돈케어 BDD를 기호로 표시할 경우 C_i라 표시한다. 그림 2는 C_i를 구하기 위한 알고리즘을 나타낸다.

```
Procedure Build-clustered-DC ({TRi}, {R+j})
  for i=1 to m /* m : TRi의 개수 */
    Ci = bdd_one
    for j = 1 to n
      if support(TRi) ∩ support(Rj) ≠ ∅ then
Ci = Ci · ( ∃ x∉TRi R+j )
    return ({C̄i})
```

그림 2. 클러스터드 돈케어 생성 알고리즘

2. 반복적 근사 도달가능상태 분석

TF는 TR의 일종이라고 볼 수 있으며, 따라서 앞 절에서 설명된 돈케어 BDD는 TR_i뿐만 아니라 TF의 압축을 위해서도 이용될 수 있다. 돈케어를 TR에 적용할 경우는 TR BDD의 크기를 줄여 TR이 관련된 연산속도를 단축시키는 데 크게 기여를 하는 반면, TF BDD의 압축은 단순한 크기 감소 이상의 다음과 같은 중요한 의미가 있다. 우선 근사 도달 상태 분석이 정해 분석에 비해 훨씬 큰 유한 상태 기계를 다룰 수 있는 이유는 전체 회로를 여러 개의 sub-FSM으로 분리하여 개별적으로 분석하기 때문이다. 그러나, 이러한 개별적 분석은 각 sub-FSM의 상호작용이 무시되기 때문에 상당한 정보유실을 의미하기도 한다. 따라서 너무 많은 sub-FSM으로 분리하여 분석할 경우는 이러한 정보유실이 더욱 증가하여 근사 결과의 정확성이 더욱 떨어지게 된다. 즉, 유한 상태 기계의 분리와 근사 도달상태계산 결과의 정확도간에는 trade-off가 존재한다. 따라서, 근사정확도를 결정하는 중요한 요소 중의 하나는 상태분할의 효율성이다. 기존 상태분할 기법들은 대개 서로 상관관계가 깊은 TF_i를 단일 클러스터로 묶는 것이며 두 TF_i간의 상관관계는 그들 support set의 공통 변수의 수효로 평가한다. BDD 최소화는 간접적으로 상태분할 향상에 기여할 수 있는데 그 이유는 BDD최소화가 대상 BDD의 support set의 크기를 줄여 보다 정확한 TF들간의 상관도 분석으로 이어질 수 있기 때문이다. 이와 같이 TF의 압축이 support set의 감소로 이어지기 때문에, 일단 TF들의 support set이 감소하고 나면 상관도가 높은 TF를 묶어 TR을 만들어 내는 과정을 반복하여 보다 양질의 TR을 만들어 낼 수 있다.

따라서, BDD최소화 알고리즘의 선택은 매우 중요하며, 우리는 restrict[2]를 사용키로 하였는데 그 이유는 restrict가 크기축소는 물론 support set 감소에 있어서도 매우 우수한 성능을 갖고 있기 때문이다. 반면, compact[13]는 크기 축소 능력은 우수하나 support set 감소의 성능은 상대적으로 약한 단점이 있다.

한편 근사도달상태 계산 알고리즘을 반복적으로 적용하는 데 있어서 중요한 결정 요소 중의 하나는 언제 반복 계산을 멈출 것이냐 이다. 가장 단순하면서도 효과적인 시점은 매 반복이 끝날 때마다 근사 결과의 향상도를 측정하여 더 이상의 향상이 없을 때 더 이상의 반복을 멈추는 것이나, 이를 위해서는

모든 $R^+_1, \ldots, R^+_n$ 들의 논리곱을 구해야 하는데 그 결과 BDD의 크기 매우 커질 수 있는 가능성이 높기 때문에, 본 연구에서는 그 대신 매번 TF에 BDD 압축 알고리즘을 적용하여 새로운 TR을 만들었을 때 그 결과를 이전의 TR들과 비교하여 그 TR의 개수가 차이가 없을 경우 반복을 멈추는 방법을 택하였다.

마지막으로 주목할 점은 sub-FSM의 감소된 support set은 근사 도달상태 계산의 수행 속도를 줄이는데 큰 도움을 준다는 점이다. 그 이유는 만약 어떤 특정 sub-FSM과의 support-set 일부가 공유되는 sub-FSM의 수가 적으면 결과적으로 그 특정 sub-FSM의 분석 후, 그 결과에 따라 분석되어야 하는 sub-FSM의 숫자가 감소하기 때문이다. 이는 곧 총 sub-FSM 분석시간의 감소로 이어진다.

3. 최근 연구 결과와의 비교

근사도달상태계산방식에 대한 연구는 최근까지 지속되고 있으며, 특히 주목할 만한 연구 결과로는 [13,14,15] 등을 들 수 있다. 이들을 크게 분류하면 유한상태기의 분할 방식, 즉 sub-FSM을 구성하는 방식을 개선하는 부류[13,15]와 기존의 sub-FSM구성을 그대로 채용하되 그들을 이용한 계산 순서나 중간 계산 결과의 크기를 감소시키는[14] 두 가지로 분류할 수 있다. 본 논문에서 제안한 방식의 장점은 이러한 기존의 도달상태계산 방식의 특징에 구애받지 않고 적용이 가능하다는 직교성(orthgonality)이다. 즉, sub-FSM 구성에 관해서는 non-supporting 변수를 sub-FSM 구성 이전에 제거시켜 줌으로서 보다 우수한 sub-FSM을 만드는데 기여하게 된다. 또한, sub-FSM구성보다는 그의 효율적인 활용에 치중하는 경우에도, 보다 정확한 sub-FSM간의 상관관계 정보를 제공함으로서 불필요한 계산을 생략시켜 프로그램의 수행속도 및 용량 증대에 기여할 수 있다.

이후 실험결과를 위해서 본 논문의 제안 방식은 VIS[9]에 이식되어 기존 연구 결과와의 상대적 비교를 꾀하였다. VIS는 현재 학술용 정형 검증 툴로서의 인지도가 가장 높을 뿐만 아니라 새로운 연구결과의 이식 및 기존 연구결과와의 상대적 비교가 가능하다는 이유에서 채택되었다.

Ⅳ. 실험 결과

본 논문에서 제안된 방법은 David Long의 BDD 패키지를 이용하여 VIS-1.2 툴에 구현되었다. 도달상태계산은 ISCAS89과 ISCAS-Addendum93 벤치마크 회로를 대상으로 SUN UltraSparc 128MHz를 이용하여 실시되었으며, 클러스터링 알고리즘은[8]에 기술된 것을 그대로 사용하였다. 동적 변수 재정렬 (dynamic variable ordering) 기능은 제안 알고리즘의 성능향상 기여도의 정확한 측정을 위하여 사용하지 않았다. 표 1에서는 기존 근사 방식 중 VIS의 근사 결과와 본 논문에서 제안한 방식을 적용한 결과를 비교하였으며, 여기서 R^+는 해당 상태기계의 존재 가능한 모든 상태 중 도달 가능서이 있는 상태를 백분율로 나타낸 것으로 이 값이 작을수록 정확도는 큰 것을 의미한다.

실험 결과에 의하면 반복적 근사방식을 적용하였을 경우 모든 경우 근사 결과의 질이 향상하였으며, 크게는 10배 이상 혹은 1000배 이상까지 근사 성능이 개선된 것을 볼 수 있다. 반복적 근사 분석방식을 적용하였을 경우 계산 소요시간은 다소 증가하였다.

Ⅴ. 결 론

본 논문은 돈케어를 이용하여 근사 도달상태의

표 1. 근사도달상태 계산 시간 및 근사 성능 비교

회로	기존 근사 분석 (VIS)		제안된 방식	
	R^+ (%)	계산시간 (CPU sec)	R^+ (%)	계산시간 (CPU sec)
s1423	0.61	170	0.42	201
s3271	5.32	98	3.89	122
s3330	7.15	109	1.43e-3	120
s4863	1.16e-3	410	2.04e-4	502
s5378	4.33e-8	241	2.83e-8	579
s13207	3.83e-6	9,900	2.00e-5	17,020
s15850	2.34e-5	7,201	1.23e-5	19,202

정확도를 향상시키는 기법을 제안하였으며, 그 원리는 이전 근사 도달상태 분석결과를 이용하여 분석하고자 하는 유한 상태 기계의 상태천이표현을 간략화 시키는 것이다.

기존 근사도달상태 계산 방식과의 실험결과 비교에 의하면 새로이 제안된 방식은 근사 정확도 향상에 크게 기여하였다. 제안된 알고리즘은 근사도달상태 계산 결과를 직접 사용하는 응용의 경우는 물론 정해 분석에 응용될 경우에도 그 실행 속도 및 분석 용량 증대에 크게 기여할 수 있다.

참 고 문 헌

[1] R. E. Bryant, Graph-Based Algorithms for Boolean Function Manipulation, IEEE Trans. Computers, vol. C-35, pp. 677-691, 1986.

[2] O. Coudert, C. Berthet, and J. C. Madre, Verification of Synchronous Sequential Machines Based on Symbolic Execution, Automatic Verification Methods for Finite State systems, Springer-Verlag, pp.365-373, 1989.

[3] A. Aziz, S. Tasiran, and R. Brayton, BDD Variable Ordering for Interacting Finite State Machines, Proc. DAC, pp. 283-288, 1994.

[4] R. Rudell, Dynamic Variable Ordering for Ordered Binary Decision Diagrams, Proc. ICCAD, pp. 42-47, 1993.

[5] K. Ravi and F. Somenzi, High-Density Reachability Analysis, Proc. ICCAD, pp. 154-158, 1995.

[6] G. Cabodi, P. Camurati, and S. Quer, Improved Reachability Analysis of Large Finite State Machines, Proc. ICCAD, pp. 354 - 360, 1996.

[7] A. Narayan, A. J. Isles, J. Jain, R. K. Brayton, and A. Sangiovanni-Vincentelli, Reachability Analysis Using Partitioned-ROBDDs, Proc. ICCAD, pp. 388-393, 1997.

[8] R. K. Brayton et. al., VIS: A System for Verification and Synthesis, Proc. Int'l Conference on CAV, pp. 428-432, July, 1996.

[9] R. K. Ranjan, A. Aziz, R. K. Brayton, B. Plessier and C. Pixely, Efficient Formal Design Verification: Data Structure + Algorithms, Technical Report UCB/ERL M94, University of California, Berkeley, Oct., 1994.

[10] J. R. Burch, E. M. Clarke, D. Long, K. L. McMillan, and D. L. Dill, Sequential Circuit Verification Using Symbolic Model Checking, Proc. DAC, pp. 46-51, 1990.

[11] Y. Hong, P. A. Beerel, J. R. Burch, and K. L. McMillan, Safe BDD Minimization Using Don't Cares, Proc. DAC, pp.208-213, 1997.

[12] H. Cho, G. D. Hachtel, E. Macii, B. Plessier, and F. Somenzi, Algorithms for Approximate FSM Traversal, Proc. DAC, pp. 25-30, 1993.

[13] S. G. Govindaraju, D. L. Dill, A. J. Hu, and M. A. Horowitz, Approximate Reachability with BDDs Using Overlapping Projections, Proc. DAC, pp. 451-456,1998.

[14] I. Moon, J. Kukula, T. Shiple, F. Somenzi, Least Fixpoint Approximations for Reachability Analysis, Proc. ICCAD, pp.41 - 44, 1999.

[15] S. G. Govindaraju, D. L. Dill, J. P. Bergmann, Improved Approximate Reachability using Auxilary State Variables, Proc. DAC, pp. 312 - 316, 1999.

홍 유 표(Youpyo Hong) 정회원

1991년 2월 : 연세대학교 전기공학과 학사

1993년 5월 : University of Southern California 전기공학과 석사

1998년 8월 : University of Southern California 컴퓨터공학과 박사

1998년 7월~1999년 2월 : Synopsys, Hillsboro, Senior Engineer

1999년 3월~현재 : 동국대학교 전자공학과 조교수

<주관심 분야> ASIC설계, VLSI CAD