논•문

계층형 크립키 구조에서 HiCTL 해석

박 사 천, 권 기 현
경기대학교 정보과학부 소프트웨어공학 전공
{sachem, khkwon}@kyonggi.ac.kr

요 약

모형 검사기는 시스템을 기술한 모형과 시스템에서 만족되어야 할 속성을 받아들여, 모형이 속성을 만족하는지 검사한다. 시스템은 대부분 상태도와 같은 계층형 유한 상태 기계들로 기술되고, 속성을 기술하는 명세 언어로는 CTL 혹은 LTL과 같은 시제 논리를 많이 사용한다. 그런데 현재 사용되는 대부분의 시제 논리는 크립키 구조와 같은 평면 구조에서 정의된다. 따라서 계층 구조를 갖는 시스템을 모형 검사하기 위해서는 평탄화 과정이 필요하다. 평탄화를 하게 되면 상태수의 증가로 인한 상태 폭발이 가중되고 본래 기술된 계층성을 온전히 지킬 수 없다. 평탄화로 인해서 야기되는 문제점을 해결하기 위한 기본 연구로서, 본 논문에서는 계층형 크립키 구조를 정의하고 계층형 구조에서 해석되는 계층형 CTL인 HiCTL을 정의한다.

1. 서론

근래 20여년간 모형 검사에 관한 많은 연구가 진행되어왔다. 모형 검사는 논리를 이용한 의미적 접근법, 언어포함관계를 이용한 오토마타 이론적 접근법, 타블루 방법 등 크게 세 가지로 분류할 수 있다.[1] 모형 검사기는 주어진 모형과 시스템에서 만족되어야 할 속성을 받아들여 모형이 속성을 만족하는지 검사하는 것이다. 모형은 크립키(Kripke) 구조나 전이가 라벨된 시스템(Labeled Transition System)으로 표현되고, 속성은 선형 시제논리인 LTL(Linear Temporal Logic)과 분기 시제논리인 CTL (Computation Tree Logic)로 표현된다.[2]

그런데 실제 시스템은 상태도(state charts)와 같은 계층형 유한상태 기계로 명세 되는 반면 모형 검사기에서 받아들이는 모형은 크립키 구조로 표현된다.[3,4] 크립키

구조는 평면적이기 때문에 계층적 특성을 갖는 상태도로 기술된 시스템은 검증되기 위해서 평탄화 과정이 선행되어야 했다. 명세의 규모가 점점 커지게 되었고 복잡한 시스템의 설계에 계층성은 더욱 요구되었다.[4] 따라서 평탄화를 거치게 되면 시스템은 본래의 계층적인 의미를 잃게 되고 상태와 전이의 수가 계층의 깊이에 따라 지수적으로 증가하게 되어 상태 폭발이 가중된다. 최근 수년간 이러한 평탄화 과정을 생략하려는 연구가 활발히 진행되었다.[3] 그러나 아직까지 계층형 크립키 구조에서 해석되는 계층형 시제논리는 정의되지 않았다.

계층형 시제논리는 평면 구조상에서 해석되는 시제논리보다 직접적으로 시스템의 속성들을 명세 할 수 있다. 평면 구조는 모두 동등한 전이 관계로만 이루어지므로 명세에서 확보한 계층성의 이점을 해석에서 활용할 수 없다는 치명적인 단점이 있다. 계층형 CTL인 HiCTL(Hierarchical CTL)을 정의하기 위해서 먼저 계층형 크립키 구조를 정의한다. 정의된 계층형 크립키 구조에는 상위 구조에서 기술된 속성이 하위 구조에 그대로 상속되는 것으로 보았다. 상속성은 객체지향 방법론이 등장한 이후에 소프트웨어 개발에 적용되는 대표적인 특성이다. 계층적으로 설계된 시스템에서 상위 객체에서 정의된 속성이 있다면 그런 속성에 대한 검증은 상위 구조에 대한 검사만으로 충분하다고 보았다.

2장에서는 크립키 구조와 CTL을 정의하고 3장에서는 계층형 크립키 구조, HiCTL의 구문과 의미를 정의한다. 4장에서는 평탄화된 구조에서 CTL식 해석과 계층형 크립키 구조에서 HiCTL식 해석을 비교하고 5장에서는 결론과 향후 연구방향을 제시한다.

2 기본개념

2.1 크립키 구조

크립키 구조는 양상논리의 의미를 정의하기위해 처음으로 사용되었다가 CTL을 해석하는데 광범위하게 사용된다. 크립키 구조는 상태 전이도이다. 이것은 시스템을 이산적인 시간으로 구분되는 상태와 상태 간의 전이관계로 기술한 것이다.

정의 2.1: 크립키 구조 M = (S, R, *in*, L)은 기본명제(atomic proposition) 집합 AP상에서 다음과 같이 정의된다.

-S: M에 속하는 모든 상태들의 집합

-$R \subseteq S \times S$: 전이관계 R은 전체관계(total relation)로서 $\forall s \in S. \exists t \in S.(s, t) \in R$이어야 한다.

-$in \in S$: 시작 상태이다.

-$L: S \rightarrow 2^{AP}$ 각 상태에 기본명제의 집합을 사상시키는 함수이다.

2.2 CTL구문과 의미

대표적인 분기시제 논리인 CTL은 크립키 구조를 무한 트리의 관점에서 해석한 것이다. CTL은 명제논리를 포함하며 경로 한정자(path quantifier)와 시제 연산자(temporal operator)로 구성된다. 두개의 경로 한정자 **A**, **E**는 각각 “모든 경로”와 “어떤 경로”로 해석된다. 네 개의 시제 연산자(**X**, **F**, **G**, **U**)는 모두 현재 상태를 기준으로 해석된다. **X**p는 “다음 상태에서 p가 만족된다.”를 의미하고 **F**p는 “언젠가는 p를 만족한다.”, **G**p는 “모든 상태에서 p가 만족된다.”는

의미로 해석된다. pUq는 "q가 만족 될 때까지 p가 만족된다."는 의미이다. **EG**p는 "현재 상태로부터 모든 상태에서 p를 만족하는 경로가 적어도 하나 존재한다."는 의미이다. CTL구문을 BNF형식으로 나타내면 다음과 같다.

$$\phi := p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \mathbf{EX}\phi \mid \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$$

CTL식의 다른 구문은 동치 관계로 모두 표현될 수 있다: $\phi_1 \vee \phi_2 \equiv \neg\phi_1 \wedge \neg\phi_2$, $\phi_1 \Rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$, $\phi_1 \Leftrightarrow \phi_2 \equiv \phi_1 \Rightarrow \phi_2 \wedge \phi_2 \Rightarrow \phi_1$, $\mathbf{A}\pi \equiv \neg\mathbf{E}\neg\pi$, $\mathbf{F}\phi \equiv true\mathbf{U}\phi$, $\mathbf{G}\phi \equiv \neg\mathbf{F}\neg\phi$.[5]

정의 2.2: CTL의 의미는 크립키 구조 M에서 아래와 같이 귀납적으로 정의된다.

$M,s_0 \models p$ iff $p \in L(s_0)$

$M,s_0 \models \neg\phi$ iff $M,s_0 \not\models \phi$

$M,s_0 \models \phi_1 \vee \phi_2$ iff $M,s_0 \models \phi_1$ or $M,s_0 \models \phi_2$

$M,s_0 \models \mathbf{EX}\phi$ iff $\exists t.(s_0,t) \in R$ and $M,t \models \phi$

$M,s_0 \models \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$ iff $\exists x = s_0,s_1,s_2,\cdots$, $\exists k \geq 0$, $M,s_k \models \phi_2 \wedge \forall i.0 \leq i < k$, $M,s_i \models \phi_1$

p는 기본명제 이고 x는 M에서 상태들의 무한 연속인 경로이다: $x = s_0, s_1,\cdots$, $i \geq 0$, $(s_i, s_{i+1}) \in R$. 정의하지 않은 CTL식 **AX**ϕ, **EF**ϕ, **EG**ϕ, **AF**ϕ, **AG**ϕ, **A**$[\phi_1 \mathbf{U} \phi_2]$등에 대한 의미는 위의 정의를 이용하여 유도될 수 있다.[5]

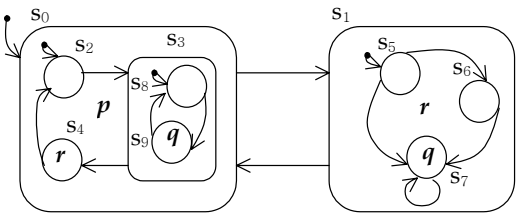
3 계층형 크립키 구조와 HiCTL 정의

3.1 계층형 크립키 구조

아래의 계층형 크립키 구조의 정의는 크립키 구조에는 없는 Υ함수가 사용된다. 이것은 정제 함수(refine function) 혹은 계층함수(hierarchy function)라고 하는데 상태를 입력받아 결과로 자신이 포함하는 상태집합을 돌려준다.

정의 3.1: 계층형 크립키 구조 M^H = (S, I, R, L, Υ)는 다섯 개의 항목으로 이루어지며 기본명제의 집합 AP상에서 다음과 같이 정의된다.

- S: M^H에 속하는 모든 상태들의 집합
- $I \subseteq S$: M^H안의 모든 시작상태들의 집합
- $R \subseteq S \times S$: 전이관계 $\forall s \in S.\exists t \in S.(s, t) \in R$은 $\exists s',t' \in S.\ s \in \Upsilon(s') \wedge t \in \Upsilon(t')$이면 $s'=t'$를 만족한다.
- L: $S \rightarrow 2^{AP}$
- Υ: $S \rightarrow 2^S$ 계층을 정의하는 함수는 상태 s를 상태집합 $S' \subset S$로 사상시킨다. S'은 s의 자식상태들의 집합이다. $\Upsilon^+(s)$는 s자신을 포함하지 않는 s의 모든 자손상태들의 집합이고, $\Upsilon^*(s) = \Upsilon^+(s) \cup \{s\}$이다.



(그림1) 계층형 크립키 구조 M^H

상태 집합 S는 암시적으로 최상위 상태인 root를 포함한다. 내부에 어떤 상태를 포함하는 상태로 전이되었을 경우 그 하위 상태들 중 시작 상태들로부터 출발한다. 전이관계 R은 같은 계층의 상태 안에서 정의된다. 라벨함수 L은 모든 상태에 라벨할 수 있음을

보여준다. 라벨함수에 의해서 상태 s에 기본 명제 p가 라벨 되었다면 상태s의 모든 자손 상태에도 동일하게 라벨 된다고 간주한다. 그리고 계층함수 Υ는 계층관계를 표현하는데 상태 s를 인자로 받아서 s의 자식 상태들의 집합을 돌려준다.

[그림 1]은 계층형 크립키 구조의 간단한 예이다. 계층형 크립키 구조 M^H의 상태 집합 $S=\{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9\}$이고 시작 상태 집합 $I=\{s_0, s_2, s_5, s_8\}$이다. 전이 관계 $R=\{(s_0,s_1), (s_1,s_0), (s_2,s_3), (s_3,s_4), (s_4,s_2), (s_5,s_6), (s_5,s_7), (s_6,s_7), (s_7,s_7), (s_8,s_9), (s_9,s_8)\}$ 이고, $AP=\{p, q, r\}$, $L(p) = \{s_0, s_2, s_3, s_4, s_8, s_9\}$, $L(q) = \{s_7, s_9\}$, $L(r) = \{s_1, s_4, s_5, s_6, s_7\}$이다. 계층관계는 함수에 의해서 정의된다: $\Upsilon(s_0)=\{s_2, s_3, s_4\}$, $\Upsilon(s_1)=\{s_5, s_6, s_7\}$, $\Upsilon(s_3)=\{s_8, s_9\}$, $\Upsilon(s_2)=\varnothing$, $\Upsilon(s_4)=\varnothing$, $\Upsilon(s_5)=\varnothing,\cdots,$ $\Upsilon(s_9)=\varnothing$. $\Upsilon^+(s_0)=\{s_2, s_3, s_4, s_8, s_9\}$, $\Upsilon^*(s_1)=\{s_1, s_5, s_6, s_7\}$.

3.2 HiCTL 구문(syntax) 정의

정의 3.2 HiCTL의 구문은 다음과 같이 상태식과 계층식, 경로식으로 정의된다.

-상태식 $\Phi := p \mid \neg\Phi \mid \phi_1 \vee \phi_2 \mid \mathbf{H}\chi \mid \mathbf{L}\chi$

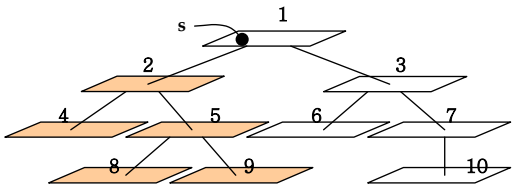
-계층식 $\chi := \neg\chi \mid \mathbf{E}\pi \mid \mathbf{A}\pi$

-경로식 $\pi := \neg\pi \mid \mathbf{X}\Phi \mid \mathbf{F}\Phi \mid \mathbf{G}\Phi \mid \phi_1\mathbf{U}\phi_2$

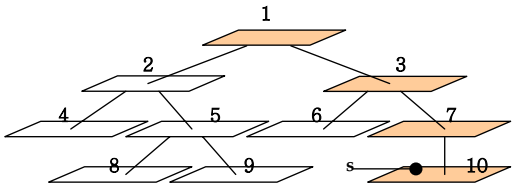
위 정의에서 p는 기본 명제이다. HiCTL구문은 CTL의 구문에 계층구조를 정해주는 계층 한정자(hierarchical quantifier)가 더해져서 확장된 것이다.

두개의 계층 한정자 **H**와 **L**은 각각 “현재 상태를 포함하고 있는 모든 상태” 그리고 “현재의 상태에 포함된 모든 상태”를 의미한다. [그림 2]는 계층 한정자의 의미를 설명하기 위해 계층형 크립키 구조를 트리 형태로 보인 것이다. [그림 2]에서 (a)는 1번 계층에 속한 상태 s에 포함된 모든 하위 상태들에서 속성 x를 만족하는 것이고, (b)는 계층10의 상태s를 포함하는 모든 상태에서 x가 만족한다는 의미이다.

[그림 2]에서 각 계층별로 번호를 부여한 것은 이해를 돕기 위함이고 실제로는 현재 상태 s로부터 계층 한정자가 적용된다. **H**한정자가 적용되면 (M^H,s$\models$**H**x) 범위가 직계조상으로 한정되고, **L**한정자가 적용되면(M^H,s$\models$**L**x) 자손 계층이 속성 검사의 범위가 된다. **H**한정자는 현재 상태를 포함하고 **L**한정자는 현재 상태를 포함하지 않는다. 계층 한정자는 속성을 계층별로 적용하기 위한 직접적인 구문으로 제공된다. 정의 3.3은 HiCTL식의 의미를 정형하게 정의한다.



(a) 2번 계층의 하위구조: $M^1,s \models L\chi$



(b) 10번 계층의 상위구조: $M^{10},s \models H\chi$

(그림2) 계층구조에서 계층한정자의 의미

3.3 HiCTL 의미 정의

HiCTL식은 CTL식에 **H**, **L**한정자가 추가된 형태이므로 여기에서는 정의2.2에 **HEX**Φ, **HE**$[\phi_1\mathbf{U}\phi_2]$, **LEX**Φ, **LE**$[\phi_1\mathbf{U}\phi_2]$, **HAX**Φ, **HA**$[\phi_1\mathbf{U}\phi_2]$,

LAXϕ, **LA**$[\phi_1 \mathbf{U} \phi_2]$의 의미를 추가하여 정의한다. 나머지 식의 의미는 아래의 정의로부터 유도될 수 있다.

정의 3.3 HiCTL은 계층형 크립키 구조 M^H에서 다음과 같이 귀납적으로 해석된다.

$M^H, s_0 \models p$ iff $p \in L(s_0)$

$M^H, s_0 \models \neg\phi$ iff $M^H, s_0 \not\models \phi$

$M^H, s_0 \models \phi_1 \vee \phi_2$ iff $M^H, s_0 \models \phi_1 \vee M^H, s_0 \models \phi_2$

$M^H, s_0 \models \mathbf{HEX}\phi$ iff $\exists s_0'.s_0 \in \Upsilon^*(s_0')$, $M^H, s_0' \models \mathbf{EX}\phi$

$M^H, s_0 \models \mathbf{HE}[\phi_1 \mathbf{U} \phi_2]$ iff $\exists s_0'.s_0 \in \Upsilon^*(s_0')$, $M^H, s_0' \models \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$

$M^H, s_0 \models \mathbf{HAX}\phi$ iff $\forall s_0'.s_0 \in \Upsilon^*(s_0')$, $M^H, s_0' \models \mathbf{AX}\phi$

$M^H, s_0 \models \mathbf{HE}[\phi_1 \mathbf{U} \phi_2]$ iff $\forall s_0'.s_0 \in \Upsilon^*(s_0')$, $M^H, s_0' \models \mathbf{A}[\phi_1 \mathbf{U} \phi_2]$

$M^H, s_0 \models \mathbf{LEX}\phi$ iff $\exists s_0'.s_0' \in \Upsilon^+(s_0) \wedge s_0' \in I$, $M^H, s_0' \models \mathbf{EX}\phi$

$M^H, s_0 \models \mathbf{LE}[\phi_1 \mathbf{U} \phi_2]$ iff $\exists s_0'.s_0' \in \Upsilon^+(s_0) \wedge s_0' \in I$, $M^H, s_0' \models \mathbf{E}[\phi_1 \mathbf{U} \phi_2]$

$M^H, s_0 \models \mathbf{LAX}\phi$ iff $\forall s_0'.s_0' \in \Upsilon^+(s_0) \wedge s_0' \in I$, $M^H, s_0' \models \mathbf{AX}\phi$

$M^H, s_0 \models \mathbf{LA}[\phi_1 \mathbf{U} \phi_2]$ iff $\forall s_0'.s_0' \in \Upsilon^+(s_0) \wedge s_0' \in I$, $M^H, s_0' \models \mathbf{A}[\phi_1 \mathbf{U} \phi_2]$

정의된 계층구조에서 자식 상태는 여럿이 있을 수 있더라도 부모상태는 오직 하나이므로 **H**한정자는 각 계층에 하나의 상태가 고려되고, **L**한정자는 현재 상태에 포함된 모든 시작 상태가 고려된다.

3.4 HiCTL 해석의 예

[그림 1]에서 M^H의 상태 s_9에서 HiCTL식 **HEX***r*이 만족하는지 다음과 같은 과정으로 해석한다.

$M^H, s_9 \models \mathbf{HAX}r$

$M^H, s_9 \models \mathbf{AX}r \wedge M^H, s_3 \models \mathbf{AX}r \wedge M^H, s_0 \models \mathbf{AX}r$

false $\wedge$ *true* $\wedge$ *true*

*false*이다.

root상태에서 **LAG***p*를 해석해 보면

M^H,root $\models \mathbf{LAG}p$

$s_0 \models \mathbf{AG}p \wedge s_1 \models \mathbf{AG}p \wedge s_5 \models \mathbf{AG}p \wedge s_8 \models \mathbf{AG}p$

그런데 $s_1 \not\models p$이므로 $s_0 \not\models \mathbf{AG}p$이다, 따라서 *false*이다.

위에서 **LAG***p*식을 해석할 때 루트로부터 모든 계층, 즉 각 계층의 모든 시작 상태에서 해석하는 것이지만 실제로는 최상위 계층에서만 검사하는 것과 같은 의미가 된다. 왜냐하면 공통 속성은 최상위 계층의 상태에 라벨 된다고 하였으므로 그 하부 계층의 상태를 직접 검사하지 않고도 알 수 있는 것이다. 이제 s_0에서 식 **LEF***q*를 해석해 보면

$M^H, s_0 \models \mathbf{LEF}q$

$M^H, s_2 \models \mathbf{EF}q \vee M^H, s_8 \models \mathbf{EF}q$

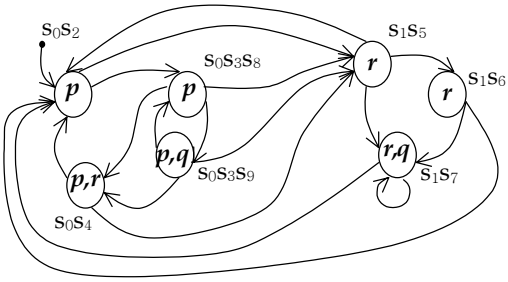
false $\vee$ *true*

true 이다.

계층 한정자를 적용할 때 정의한 것과 같이 경로 한정자가 **E**이면 어떠한 계층이던지 속성 ϕ를 만족하는 하나의 경로만 존재하면 되고, **A**이면 정의된 각 계층의 모든 상태에서 ϕ를 만족해야 한다. 따라서 경로 한정자가 **A**이면 $\wedge$로 **E**이면 $\vee$로 연결한다.

4. CTL과 HiCTL의 의미 비교

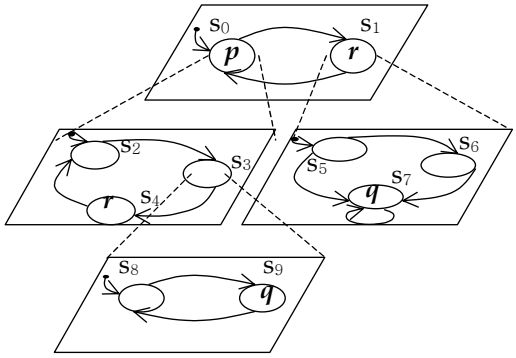
앞서 정의한 바와 같이 CTL식은 평면적인 크립키 구조에서 해석된다. 반면에 계층형 크립키 구조에서 계층은 [그림 4]와 같이 유한 트리로 표현되고 계층 속의 각각의 크립키 구조는 무한 트리로 표현된다. 이 장에서는 [그림 1]의 계층형 크립키 구조의 예를 가지고 평탄화 하여 크립키 구조로 만든 후에 CTL식을 적용하는 것과 계층형 크립키 구조에서 HiCTL식을 적용하는 예와 그 의미를 비교한다.



(그림3) 평탄화된 크립키 구조 M

4.1 평탄화

평탄화된 구조는 계층구조에서 계층성을 제거함으로써 얻어진다.[3] 가장 안쪽에 중첩된 구조부터 연속적으로 평탄화가 진행된다. 먼저 가장 안쪽의 상태는 $\Upsilon(S_3) = \{S_8, S_9\}$에 의해서 S_3S_8과 S_3S_9의 두개의 상태가 만들어진다. 새로 얻어진 상태의 이름은 부모와 자식상태를 연속해서 쓴다. 상태가 얻어진 다음 전이 관계를 결정해야 한다. S_2에서 (S_2, S_3S_8)의 전이관계가 있고, S_4로는 (S_3S_8, S_4)와 (S_3S_9, S_4)의 전이관계가 형성된다. 하위계층으로 들어오는 전이는 시작 상태로만 연결하고 하위에서 상위 계층으로의 전이는 하위 계층의 모든 상태에서 상위 상태로 전이를 만들어 준다. 상·하위 계층 사이에 한번의 평탄화가 끝나고 상태와 전이가 만들어지면 각 상태는 상위의 상태와 하위의 상태들에 라벨 되었던 기본 명제의 합집합으로 라벨된다. 이런 과정은 최상위 상태에 이르기까지 반복적으로 수행된다. [그림 3]은 [그림 1]의 계층형 크립키 구조를 평탄화한 크립키 구조이다. 평탄화한 결과를 보면 상태의 수가 감소했다. 크립키 구조는 상태도 관점[4]에서 보면 OR-상태로만 되어있기 때문에 상태도와는 달리 오히려 상태수가 감소한다. 실제로 계층형 크립키 구조에서 기본 상태의 수가 평탄화된 상태의 수가 된다. 계층형 크립키 구조에서 해석은 대상이 한번에 하나의 계층만을 고려하기 때문에 해석에 모든 상태가 고려되는 크립키 구조보다 적은 상태공간을 필요로 한다. 비록 상태수는 줄었지만 그렇다고 시제논리를 해석하는 상태공간을 줄인 것은 아니다.



(그림4) 계층형 크립키 구조 M^H의 계층도

4.2 CTL과 HiCTL의 의미

먼저 [그림 3]의 크립키 구조M에서 CTL식 **AG**ϕ를 해석하는 것과 계층도로 표현된 [그림 4]의 M^H 에서 HiCTL식 **LAG**ϕ의 해석을 비교해보자. 예를 들어 $s_0s_3s_8$에서는 **AG***p*가 만족되지 않는다. 그러나 $M^H,_{s3}$에서 해석해 보면 **LAG***p*가 만족된다. **LAG**ϕ는 계층 구조의 아래만을 고려하기 때문에 특정 계층 혹은 모듈에서 만족되어야 할 속성을 검사하려고 한다면 유용할 것이다. 반면 평탄화 된 구조에서는 직접적으로 특정 하위 구조를 검사하는데 자유롭지 못 할 것이다.

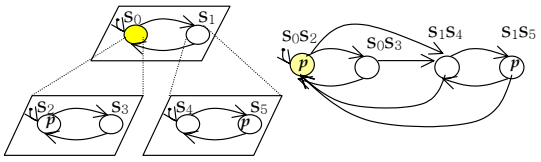
4.2.1 CTL과 HiCTL 해석의 차이

[표 1]은 CTL식이 의미하는 것을 HiCTL식으로 표현한 것이다. 평탄화 과정에서 상위로부터 하위 상태로 들어오는 전이는 시작 상태에만 연결되므로 **AX**ϕ와 **EX**ϕ는 **HAX**ϕ, **HEX**ϕ와 해석에 있어서 동일한 결과는 보인다. **HAG**ϕ는 계층 구조에서 최상위 상태들 간에 우선적으로 만족이 되어야 하기 때문에 계층구조에서 **HAG**ϕ가 만족되면 평탄화 된 구조에서 **AG**ϕ도 만족된다.

경로 / 시제	A	E
X	HAXϕ	HEXϕ
F	HAF(ϕ∨LAFϕ)	HEF(ϕ∨LEFϕ)
G	HAGϕ	HEG(ϕ∨LEGϕ)
$\phi_1 U \phi_2$	HA[ϕ_1U(ϕ_2∨LA[ϕ_1Uϕ_2])]	HE[ϕ_1U(ϕ_2∨LE[ϕ_1Uϕ_2])]

(표1) HiCTL 사상표

[그림 5]를 보면 평탄화 된 구조에서 **AF***p*를 만족하는 것에 반하여 계층형 구조에서는 **LAF***p*는 만족하지만 **HAF***p*는 만족하지 못하는 예를 볼 수 있다. 즉 CTL식과 동일하게 평탄화 된 경로 그대로를 표현하는 HiCTL 식을 간단히 표현하기가 어렵다. 평탄화 과정에서 계층 구조가 붕괴되고 암시적으로 표현되었던 전이관계를 명시적으로 나타냈기 때문에 단 방향의(상위, 하위) 식으로는 계층을 모두 추적하기 어렵다. 따라서 [표 1]에서와 같이 CTL식의 의미를 표현하기 위한 HiCTL식은 복잡해진다. 이 논문에서는 두 식간의 동치를 증명하지는 않는다.



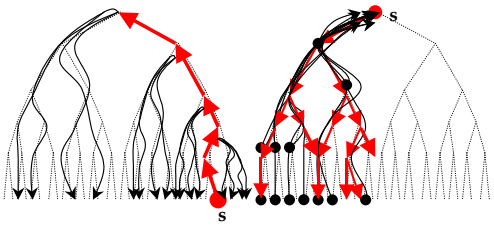
(그림5) **AF***p*해석($s_2 \not\models$ **HAF***p*, $s_0 \models$ **LAF***p*, $s_0s_2 \models$ **AF***p*)

4.2.2 계층한정자가 겹쳐진 식의 의미

정의된 HiCTL식은 **H**x**H**x, **H**x**L**x, **L**x**H**x, **L**x**L**x의 형태를 보일 수 있다. 현재 상태가 어디에 위치해 있는지에 따라 다르게 해석될 수 있지만 최상위의 상태와 최하위의 상태에서 적용시켜 보겠다. 먼저 가장 상위 계층의 상태 s를 생각해 볼 수 있다. s에서 **L**x**L**x는 하강하는 계층식을 중복했기 때문에 특별한 의미를 발견할 수 없다. s에서 **H**x**L**x은 **E**π(**A**π)**L**x와 유사하다. 앞의 계층식 **H**x는 계층 해석에 영향을 주지 못한다. **H**x**H**x 도 계층식의 의미를 충분히 표현하지 못한다. **L**x**H**x는 약간 복잡해지는데 [그림 6]의 (b)와 같이s에 포함된 계층에서 다시 s까지 거슬러 오르기 때문에 최상위 상태 s에 포함되는 모든 하위 상태들

의 계층적으로 연속된 집합을 이룬다. 상태도 같은 계층형 명세 언어로 작성된 시스템의 모형이라고 가정할 때 최상위 상태에서 수행 가능한 구성(configuration)으로 표현된다. 예를 들어서 **LEF(HAXϕ)**는 s의 어떤 하위 상태로부터 s까지 모두 **AXϕ**를 만족한다는 의미이다. 물론 상태도와 비교하면 계층형 크립키 구조에는 이벤트가 없고 전이의 갈등도 발생하지 않으며 AND-상태가 정의되지 않은 것 등 많은 차이점을 보이지만 계층 구조적인 관점에서 보면 유사하다고 할 수 있다.

이제 자손이 없는 상태인 기본 상태 t에서 위에서와 같이 각각의 식을 적용시키면 **LxHx**와 **LxLx**는 부적절한 표현이다. **HxHx**는 처음의 **Hx**에 의해서 형성된 계층 범위 내에서 적용된다. 즉 속성이 적용되는 계층구조에는 변화가 없다. 마지막으로 **HxLx** 은 3.4절에서 언급했던 것처럼 다른 가지에서 속성을 만족하는지 알아볼 수 있다. 그런데 [그림 6]의 (a)와 같이 최하위 상태에서 최상위 상태로 이어지는 부분은 하나의 특정한 구성이라고 볼 수 있다. 최하위 계층의 어떤 상태 s에 대해서 HiCTL식 **HAF(LAFϕ)**는 "앞으로 만들어질 구성 중에 언젠가는 ϕ를 만족하는 것이 있다." 라는 의미로 해석할 수 있다.



(그림6) (a) HxLx (b) LxHx

5 결론 및 향후 연구 방향

계층형 크립키 구조를 정의하고 계층형 구조에서 직접 적용할 수 있는 계층형 시제논리인 HiCTL을 정의했다. 계층형 크립키 구조에서는 속성이 상속된다고 보았고 이것은 전역적인 속성해석을 간단하게 할 수 있었다. CTL식과 HiCTL식을 비교 해본 결과 계층구조로 인한 해석의 차이를 비교 할 수 있었다. 그리고 CTL식으로는 표현할 수 없는 계층간의 속성도 표현 가능했다. 그러나 기존의 CTL식과 일치하는 의미를 표현함에 있어 식이 복잡함으로 직관적으로 이해하는데 어려움이 있었다.

앞으로의 연구는 HiCTL 계층 한정자가 중첩될 때의 의미와 CTL과의 의미 비교를 정형하게 정의하고 실제에서 적용할 모형검증 알고리즘을 만드는 것이 될 것이다.

참고문헌

[1] M. Muller-Olm, D. Schmidt, B. Steffen, "Model checking: A tutorial introduction," Proceedings of SAS'99, Lecture Notes in Computer Science, Springer, 1999

[2] E. A. Emerson, Model Checking and the Mu-Calculus, in Descriptive Complexity and Finite Models, N. Immerman and P. Kolaitis, eds., American Mathematical Society, 1997

[3] R. Alur, M. Yannakakis, Model Checking of Hierarchical State Machines, Proceedings of FSE, 1998.

[4] D. Harel, A. Naamad, The STATEMATE semantics of Statecharts, ACM Transactions on Software Engineering and Methodology, Vol.5, No.4, pp.293-333, 1996

[5] J. P. Katoen, Concepts, Algorithms, and Tools for Model Checking, Friedrich-Alexander-University, 1999