2003년도 대한전자공학회 하계종합학술대회 논문집 제26권 제1호

LTS로 명세화 된 철도 신호제어용 프로토콜 검증

서미선*, 정창현*, 이재호**, 황종규**, 김성운*
*부경대 정보통신공학과
**철도기술연구원

Verifying Correctness of Rail Signal Control Protocols Specified in LTS

Miseon Seo*, Changhyun Jeong*, Jaeho Lee**, Jongkyu Hwang**, Sungun Kim*
*PuKyung National University, Department of Telematics Engineering
**Korea Railroad Research Institute
E-mail : jljpms@mail1.pknu.ac.kr

Abstract

대규모 시스템 명세의 올바름을 검증하기 위한 유한상태 LTS(Labeled Transition System)에 기반을 둔 CTL(Computation Tree Logic) 논리 적용의 문제점은 시스템 내부의 병렬 프로세스간 상호 작용으로 인한 상태폭발이다. 그러나 modal mu-calculus 논리를 시스템 안전성 및 필연성 특성 명세에 사용하면, 행위에 의한 순환적 정의가 가능하므로 상태폭발 문제가 해결된다. 본 논문에서는 LTS 로 명세화 된 철도 신호제어용 프로토콜 모델의 안전성 및 필연성 특성을 모형 검사 기법에 의해 검증하기 위해 시제 논리로 사용된 modal mu-calculus 를 사용하여 해당 검정 알고리즘을 구현 및 적용하였다.

I. 서론

통신 소프트웨어에 대한 사용자의 요구사항이 복잡화, 다양화, 대형화 되어짐에 따라 개발에 따르는 어려움은 더욱 증대되었고, 과거에 사용된 비정형적 방법(informal method)에 의한 개발은 많은 오류와 비효율성을 내포하고 있다. 따라서 개발에 소요되는 비용 및 시간을 절약하고 심각한 오류를 형식적 방법에 의해 검출하는 기술인 정형기법(formal method)의 적용이 증가하는 추세이며, 특히 통신 소프트웨어 개발 과정에서 요구사항과 명세와의 일치성 및 완전성을 검사하는 검증단계에 많이 사용되고 있다[1]. 정형 검증 기법은 명세의 일반적인 정확성(correctness)을 분석하는 과정으로 잘못된 상태가 존재하지 않아야 하는 안전성 특성(safety property)과 정의된 행위가 반드시 발생해야 하는 필연성 특성(liveness property)이 만족하는지 수학적이고 논리적인 방법에 의해 검증하는 기법이다[2].

일반적으로 유한 상태 레이블 천이 시스템(LTS)으로 모델링 된 concurrent 시스템 명세의 올바름을 검증하기 위한 모델 검증(model checking)에 시제논리에 기반을 둔 CTL 이 많이 사용되어 왔으나[3] 해당 검증 알고리즘의 구현 및 적용 과정에서 시스템 내부 병렬 프로세스간의 상호작용으로 인한 상태폭발 문제가 제기되어 왔다. 그러나 modal mu-calculus 논리를 시스템 안전성 특성 명세에 사용하면 행위에 의한 순환적 정의가 가능하므로 상태폭발 문제가 해결될 수 있다[4].

본 논문에서는 LTS 로 명세화 된 철도 신호제어용 프로토콜 모델의 안전성과 필연성 특성을 정형화된 모형 검사 기법으로 검증하기 위해, 기존의 시제 논리에 기반을 둔 CTL 에 순환개념을 첨가하여 Kozen 이 제시한 modal mu-calculus[5]로 해당 검정 알고리즘을 구현 및 적용하는 방안을 제시한다.

II. 검증사항 및 대상

2.1 검증사항

통신 프로토콜이 적절한 기능을 하기 위해서는 프로토콜 각 해당 상태의 deadlock 과 livelock 및 비정상적인 도달과 같은 잠재적 설계 에러가 없어야 하며, 사용자 요구사항과 일치하고 다른 프로세서와의 원활한 통신이 이루어지는지 검사해야 한다.

프로토콜 검증은 프로토콜 명세의 정확성(correctness), 안전성(safety)과 필연성(liveness) 및 일관성(consistency) 등을 알아보는 것으로 model checking 에서 보다 구체적으로 검증해야 할 프로토콜의 특성은 아래와 같다.

- Deadlock : 한 상태에서 다른 어떤 상태로의 천이가 존재하지 않기 때문에 다음 행위를 할 수 없는 경우, 즉 그 상태에서 나가는 천이가 존재하지 않는다.
- Livelock : 프로토콜(명세) 상태들의 부분집합 내에서 그 상태들만을 무한히 반복적으로 천이하는 경우로서 그 부분집합 이외의 다른 상태로의 천이가 존재하지 않는다.
- Reachability : 초기상태로부터 프로토콜(명세)이 정의된 천이의 순서에 의해 정의된 다른 상태로 도달 가능하다면 그 천이와 상태에 대해 이 명세는 올바른 프로토콜(명세)이다.
- Liveness : 프로토콜의 어떤 정당한 특성(상태 또는 행위)이 결국 만족되어지는 것으로, 결국 도달되어야 하는 상태와 반드시 발생해야 하는 행위를 나타낸다.

2.2 검증대상 프로토콜

철도에 있어 신호제어시스템은 제한된 철도자원을 효율적이고 경제적으로 운용할 수 있도록 도와주며 열차의 안전 운행을 위한 필수적인 시스템으로 철도 신호제어용 프로토콜에 의해 제어되어진다.

본 논문에서는 국내 철도 신호제어용 프로토콜 중의 하나인 중앙집중제어장치와 전자연동장치 간의 정보전송을 담당하는 역정보전송장치(LDTS : Local Data Transmission System)와 현장신호설비를 제어하고 감시하는 전자연동장치(EIS : Electronic Interlocking System) 사이의 정보전송방식을 검증한다. 그림 1 은 이들간의 메시지 흐름을 I/O FSM (Input/Output Finite State Machine) 중간 모델로 생성한 것이며, 그림 2 는 철도 프로토콜을 LTS 로 모델링 한 것이다.

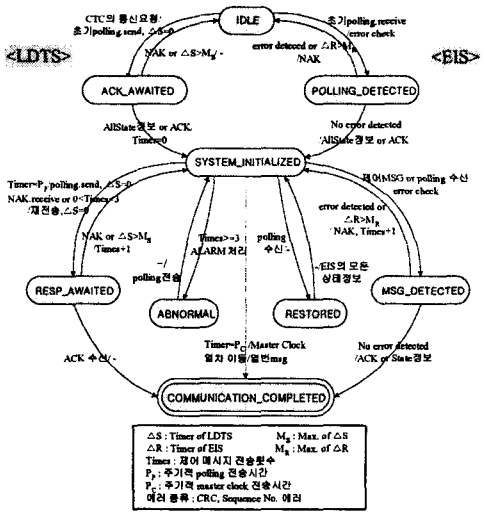

그림 1. 철도 프로토콜의 I/O FSM 생성

상태	상태 설명
IDLE	LDTS 와 EIS 사이의 통신이 시작되기 전 상태
ACK_AWAITED	LDTS 에서 polling 을 전송하고 ACK 의 수신을 기다리는 상태
POLLING_DETECTED	polling을 수신한 EIS가 메시지를 검사하는 상태
SYSTEM_INITIALIZED	LDTS가 ACK를 수신함으로 통신이 초기화된 상태
RESP_AWAITED	LDTS에서 제어 메시지 전송 후, ACK 수신을 기다리는 상태
MSG_DETECTED	제어 메시지를 수신한 EIS가 메시지를 검사하는 상태
ABNORMAL	재전송 3회 이후에도 NAK 또는 어떤 응답도 받지 못할 경우 발생하는 통신 이상 상태
RESTORED	통신 이상 상태에서 정상적인 통신으로 복구되는 상태
COMM_COMPLETED	정상적인 통신 완결 상태

표 1. I/O FSM 모델의 상태 설명

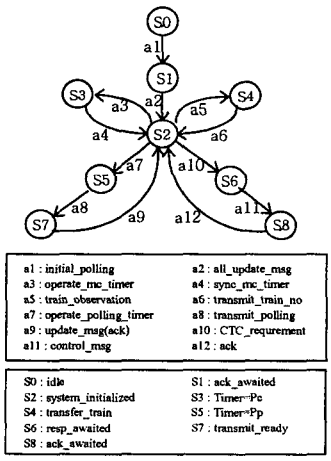

그림 2. 철도 프로토콜 LTS

III. Modal mu-calculus

Modal mu-calculus 는 시스템 상태의 부분 집합이 공

통적으로 만족하는 논리식인 고정점을 가진 modal 논리이며, 시스템의 시간적 변화 특성뿐만 아니라 시스템 명세 특성을 다양하게 표현하는 명제적 temporal 논리이다. 연산자는 원자명제(atomic proposition), ∧(논리곱: conjunction), ∨(논리합: disjunction), [ ](필연성: necessarily), < >(가능성: possibly), ν(최대고정점), μ(최소고정점)으로 구성되어 있으며, 일반화된 modal mu-calculus 의 논리식은 다음과 같다[6].

$\Phi ::= tt \mid ff \mid Z \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [k]\Phi \mid \langle k \rangle \Phi \mid \nu Z.\Phi \mid \mu Z.\Phi$

여기서 tt 는 모든 상태에 대해 참인 것을 나타내고 ff 는 거짓임을 나타내며 Z 는 명제적 변수, k 는 천이 집합의 원소, Φ 는 프로세스 특성을 나타낸 논리식이다. νZ 는 시스템의 상태 집합들이 일반적으로 만족하는 특성인 최대고정점연산자(*greatest fixed point operator*)이며, μZ 는 시스템의 상태 집합들이 공통적으로 만족하는 최소고정점연산자(least fixed point operator)이다[2].

3.1 안전성(Safety)

프로토콜의 부당한 상태 즉, deadlock 이나 livelock 과 같은 상태를 배제하는 특성이다.

만약 정당한 상태를 나타내는 식이 Φ 라면 상태에 대한 안전성은 νZ.Φ ∧ [-]Z 로 표현되고 여기서 [-]는 모든 행위를 나타낸다. 그림 3 에서 예를 들면, 위 식의 명제 값이 참이 된다는 것은 그림 3 의 LTS 가 상태에 대한 안전성 특성을 가지고 있음을 의미한다.

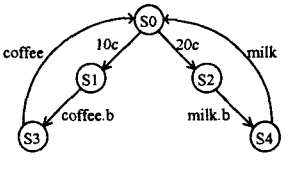

그림 3. 자판기 LTS

또한 행위에 대한 안전성 표현은 νZ.[k]ff ∧ [-]Z 이며, 식의 명제 값이 참이 된다는 것은 LTS 가 행위에 대한 안전성 특성을 가짐을 의미한다. 즉 행위 k 가 절대 발생하지 않음을 나타낸다.

3.2 필연성(Liveness)

프로토콜의 어떤 정당한 특성(상태 또는 행위)이 결국 만족되어지는 것으로, 결국에는 도달되어져야 하는 상태와 반드시 발생해야 하는 행위 즉, reachability 와 liveness 를 만족하는 특성이다.

LTS 모델로 표현된 프로세스의 상태에 대한 필연성은 μZ.Φ ∨ (<->tt ∧ [-]Z)로 표현되고, 이 식이 참이 된다는 것은 μZ.Φ 와 (<->tt ∧ [-]Z)이 동시에 거짓이 될 수 없고 식 Φ 에 해당되는 상태는 항상 도달되어져야 함을 나타낸다. 또한 LTS 모델로 표현된 프로세스에서 행위에 대한 필연성의 논리식은 μZ.<->tt ∧ [k]Z 로 표현하는데, 이 식이 참이 되기 위해 [k]Z 는 반드시 참이 된다. 즉, 결국 행위 k 는 발생할 수 밖에 없음을 나타낸다. 필연성에 대한 식이 참이 된다는 것은 검증 대상 LTS 가 필연성 특성을 가지고 있다는 것을 의미한다.

위에서 기술한 프로토콜의 특성(safety, liveness)을 만족하고, 원자명제인 상태 A 에 결국 도달되어져야 하며 deadlock 및 livelock 과 같은 치명적 오류가 존재하지 않음을 의미하는 modal mu-calculus 의 논리식은 νZ.(μY.A ∨(<-> tt ∧ [-]Y)) ∧ [-]Z 로 정의한다.

IV. Modal mu-calculus 를 적용한 철도 신호시스템 검증

본 소절에서는 LTS 에 대한 정확성 검증에 대해 앞 소절에서 기술한 modal mu-calculus 식에 model checking 알고리즘을 적용하여 검증 결과를 검출한다.

그림 4 는 그림 2 의 LTS 에서 임의로 상태 S5 에 deadlock, 상태 S8 에 livelock 을 발생시킨 LTS 모델이며 원자명제는 상태 S2 이다.

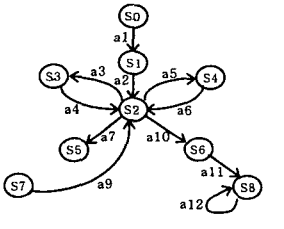

그림 4. Deadlock 및 Livelock LTS 모델

그림 4 의 모델을 검증하기 위한 modal mu-calculus 의 식은 다음과 같다.

νZ. (μY.A ∨(<-> tt ∧ [-]Y)) ∧ [-]Z 단, A={S2}

논리식은 LTS 의 행위 집합 S 에 포함된 어떠한 행위가 발생하더라도 그 LTS 를 만족해야 하고, 원자명제 A 를 만족함을 나타낸다.

위 식에 model checking 알고리즘인 solve 알고리즘을 적용하여 검증결과를 검출한다[6]. 그림 5 는 변수 Xi 들의

천이 관계를 나타낸 edge-labeled directed graph G 를 나타낸다.

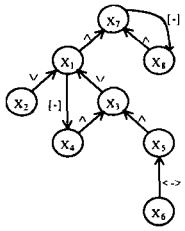

그림 5. Edge-labeled directed graph G

그림 6 은 최대고정점과 최소고정점을 사용하여 생성된 max block 과 min block 을 나타내며, 그림 7 은 초기화 규칙에 의해 초기화 된 bit-vector, counter, 그리고 배열 M[i]를 나타낸다.

$B_1 \equiv min\{X_1 = X_2 \vee X_3$	$B_2 \equiv max\{X_7 = X_1 \wedge X_8$
$X_2 = A$	$X_8 = [-]X_7\}$
$X_3 = X_4 \wedge X_5$	
$X_4 = [-]X_1$	
$X_5 = <>X_6$	
$X_6 = tt\}$	

그림 6. Max block, Min block

X	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
S0	0	0	0	0	0	1	1	1
S1	0	0	0	0	0	1	1	1
S2	0	1	0	0	0	1	1	1
S3	0	0	0	0	0	1	1	1
S4	0	0	0	0	0	1	1	1
S5	0	0	0	1	0	1	1	1
S6	0	0	0	0	0	1	1	1
S7	0	0	0	0	0	1	1	1
S8	0	0	0	0	0	1	1	1

C	X_3	X_4
S0	2	1
S1	2	1
S2	2	4
S3	2	1
S4	2	1
S5	1	0
S6	2	1
S7	2	1
S8	2	1

M[1]=<<S2, X2>, <S5, X4>, <S0, X6>, <S1, X6>, <S2, X6>, <S3, X6>, <S4, X6>, <S5, X6>, <S6, X6>, <S7, X6>, <S8, X6>>
M[2]=< >

그림 7. Bit-vector, Counter and Array M[i]

그림 8 은 배열 M[i]가 공집합(empty)이 될 때까지 solve 알고리즘을 적용하여, bit-vector 및 counter 를 갱신한 결과이다. Deadlock 은 bit-vector 요소에 의해 판단되는데 어떠한 행위에 대해서도 참임을 나타내는 변수 X5 의 등식 <->tt 를 만족하지 않는 상태 <S5, X5>가 0 으로 갱신됨으로써 검출된다. 또한 livelock 은 관련된 상태의 counter 요소에 의해 판단되며 그림 8 의 counter 에서 상태 S2, S6, S8 과 관련된 요소가 1 로 갱신된다. 이는 상태 S8 이 safety 를 표현한 부논리식 $\mu Y.A \vee (<-> tt \wedge [-]Y)$과 관련된 변수 X3 및 X4 를 만족하지 않으므로 livelock 이 발생했음을 나타낸다.

X	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
S0	1	0	1	1	1	1	0	0
S1	1	0	1	1	1	1	0	0
S2	1	1	0	0	1	1	0	0
S3	1	0	1	1	1	1	0	0
S4	1	0	1	1	1	1	0	0
S5	1	0	1	1	0	1	0	0
S6	0	0	0	0	1	1	0	0
S7	1	0	1	1	1	1	0	0
S8	0	0	0	0	1	1	0	0

C	X_3	X_4
S0	0	0
S1	0	0
S2	1	1
S3	0	0
S4	0	0
S5	0	0
S6	1	1
S7	0	0
S8	1	1

M[1]=< > M[2]=< >

그림 8. Bit-vector, Counter and Array M[i]

V. 결론

본 논문에서는 기존의 CTL 논리에 순환 개념을 첨가한 modal mu-calculus 를 적용함으로서 LTS 로 명세화된 철도 신호제어용 프로토콜 모델이 안전성 및 필연성 특성을 만족하는지를 형식적으로 검증하는 방법을 제시하였고, 그림 1 및 그림 2 에서 나타낸 프로토콜 모델에 적용한 결과 안전성과 필연성이 모두 만족됨을 검증하였다.

향후 연구사항은 본 논문에서 제시한 방법을 이용해서 사용자가 보다 쉽고 정확하게 검증할 수 있도록, LTS 행위 명세 검증도구를 컴퓨터 환경에서 구현하여 철도 프로토콜을 검증하게 하는 것이다.

참고문헌

[1] D.schwabe, "Formal Techniques for the Specification and Verification of Protocol", Ph.D Thesis, Univ. of California Los Angeles, Apr., 1981.

[2] Kenneth L.McMillan, "Symbolic model checking", Kluwer Acacenic Publishers, Model checking, 1996.

[3] E.M.Clarke and E.A.Emerson, "Synthesis of synchronization skeletons for branching time temporal logic", In Dexter Kozen, editor, Logic of Trograms: Workshop, volume 131 of Lecture Notes in Computer Science, Yorktown Heights, New York, May 1981. Springer-Verlag.

[4] 한국정보과학회, "모형검사를 위한 Modal Mu-Calculus 와 CTL 의 안전성 및 필연성 논리식 증명", 1999 년, 12 월.

[5] D.Kozen, "Results on the Propositional Mu-Calculus", Theoretical Computer Science 27:333-354, 1983.

[6] A.Arnold and B. Crubille. "A Linear Algorithm To Solve Fixed-Point Equations on Transition System." Information Processing Letters 29:57-66, 30 September 1988.