신뢰성 확보를 위한 철도 신호제어용 프로토콜 검정기 개발
Development for Verification Tool Guaranteeing Reliability of Rail Signal Control Protocol

서미선* 황진호* 황종규** 이재호** 김성운***
Seo, Mi-Seon Hwang, Jin-Ho Hwang, Jong-Gyu Lee, Jae-Ho Kim, Sung-Un

ABSTRACT

In this paper, we develope a protocol verification tool that verifies the correctness of rail signal control protocol type 2 specified in LTS(Labeled Transition System) by using model checking method. This tool automatically checks several properties for deadlock, livelock and reachability of states and actions on LTS. and removes many errors and ambiguities of an informal method used in the past, so saves down expenditures and times required in the protocol development. Therefore it is expected that there will be an increase in safety, reliability and efficiency in terms of the maintenance of the signaling system by using the developed verification tool.

1. 서론

국내의 철도현장에서 사용되는 정보전송방식이 각 제조회사별, 각 노선별로 서로 상이하고, 또한 정형기법에 의해 검정되지 않은 기준을 사용함으로 인해 통신시스템의 안정성 저하는 물론이고 유지보수에도 많은 비용 소요 및 어려움을 겪어왔다. 따라서 철도 신호장치간 정보전송방식의 프로토콜 표준화 및 제품개발에 따른 정형기법에 기초한 검정기 개발은 철도 신호 전체 제어시스템의 안정성 및 신뢰성 보장뿐만 아니라 열차의 안전운행, 그리고 철도 신호보안설비 유지보수 최적화를 위해 필수적으로 요구되는 기술이다.

프로토콜 검정이란 사용자 요구사항과 명세와의 일치성을 구현 전에 확인하는 단계로, 모든 프로토콜에 필수적인 정확성을 만족하는지를 모형검사(model checking)방법을 사용하여 자동적, 형식적으로 검정하는 기술이다[1]. 검정을 위해 과거에 사용된 비정형적 방법은 명세의 모호함과 불완전성 등의 많은 오류와 비효율성을 내포하고 있다.

본 논문에서는 형식적 방법에 의한 모델 검정을 위해 프로토콜의 행위 특성을 명세화하는 LTS(Labeled Transition System)로 변환한 모델에 기반을 두고 대수적 명세 기법인 modal mu-calculus 논리를 적용하여 프로토콜을 검정하는 '프로토콜 검정기'를 개발하였다. 개발된 도구를 이용하여 국내 철도 신호제어 프로토콜 (type 2 : 이더넷 기반의 프로토콜표준)의 안전성 및 필연성 특성을 검정함으로써 안정적이고 효율적인 검정을 수행함을 보인다. 이를 위해 2장에서는 모형검사를 이용하는 프로토콜 검정방법을 설명하고, 3장에서는 검정하고자 하는 대상인 철도 신호제어 프로토콜 type 2와 이를 LTS로 변환한 모델에 대해 기술한다. 그리고 4장에서는 개발된 프로토콜 검정기의 구성 및 동작과 결과를 분석하는 방법을 서술하고 5장에서 검정 예시와 검정기 실행 결과를 보인 다음, 마지막으로 6장에서 본 논문의 결론과 향후 추진 사항에 대해 기술한다.

* 부경대학교 정보통신공학과, 학생회원
** 철도연구원 선임연구원, 정회원
*** 부경대학교 정보통신공학과, 정회원

2. 모형검사를 이용한 프로토콜 검정 방법

통신 프로토콜이 적절한 기능을 하기 위해서는 프로토콜 각 해당 상태의 deadlock과 livelock 및 비정상적인 도달(reachability)과 같은 잠재적 설계 에러가 없어야 하며, 사용자 요구사항과 일치하고 다른 프로세서와의 원활한 통신이 이루어져야 한다. 모형검사방법은 프로토콜의 안전성과 필연성 특성을 보다 구체적으로 검정하기 위한 방법으로, 이를 위해서는 프로토콜의 행위 특성을 나타내는 레이블 천이 시스템(Labeled Transition System)을 이용해 시스템을 명세화한다[2]. LTS로 명세화된 프로토콜에 기반을 두고 Modal mu-calculus를 이용하여 검정하고자 하는 대상에 따라 시스템의 특성을 (그림 1)과 같이 명세한다. 논리식에 따라 max block과 min block에 대한 변수를 생성하여 상태와 변수의 천이관계를 나타낸 edge-labeled directed graph를 생성한 다음, modal mu-calculus의 재귀적 특성을 이용한 Solve 알고리즘의 초기화 및 갱신알고리즘을 적용하여 bit-vector, counter, 그리고 배열 M의 결과를 검정한다. (그림 2)는 이러한 절차를 나타낸다.

프로토콜 검정을 위한 Modal Mu-Calculus 표현식

1. Safety Check 식 (Deadlock 및 Livelock) : $\nu Z.\ (\mu Y.A \vee (\langle -\rangle tt \wedge [-]Y)) \wedge [-]Z$ 단, $A=\{S_0\}$
2. State Reachability Check 식 : $\nu Z.\ (\mu Y.A \vee (\langle action_2 \rangle tt \wedge [action_1]Y)) \wedge [-]Z$ 단, $A=\{state\}$
3. Action Reachability Check 식 : $\nu Z.\ [action_1](\mu Y.\ \langle action_3 \rangle tt \wedge [action_2]Y) \wedge [-]Z$

그림 1. Modal Mu-Calculus 논리식

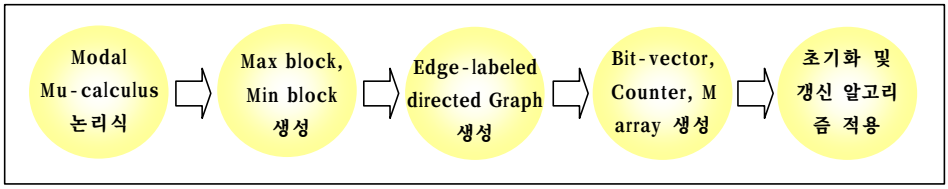

그림 2. 모형검사방법의 절차

3. 검정 대상

본 논문에서 검정할 대상인 철도 신호제어 프로토콜 type 2는 열차집중제어장치(CTC : Centralized Traffic Control)와 전철전력원격감시제어장치(SCADA : Supervisory Control And Data Acquisition) 간의 열차정보와 SCADA 상태정보 전송을 위한 프로토콜이다. CTC와 SCADA 사이의 인터페이스 링크 구성은 라우터를 통한 LAN 접속방식을 사용하고, CTC는 Server, SCADA는 Client의 역할을 각각 담당한다. CTC는 SCADA로 해당 구간에 운행 중인 열차의 정보를 나타내는 메시지를 전송하고, SCADA는 CTC로 각 구간별 전차선의 가압 상태를 나타내는 SCADA 상태정보를 전송한다. 철도 신호 제어 프로토콜 type 2의 행위 특성을 레이블 천이 시스템으로 나타내면 (그림 4)와 같다. 이것은 6개의 상태와 8개의 행위로 구성되어 있으며, 각 상태명과 행위명은 (그림 4)의 table에 기술되어 있다.

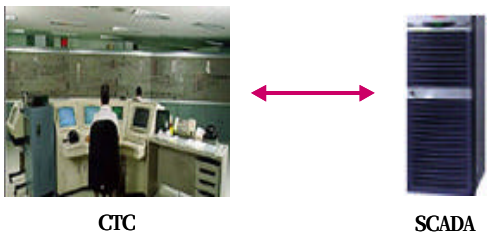

그림 3. 철도 신호 제어 프로토콜 type 2

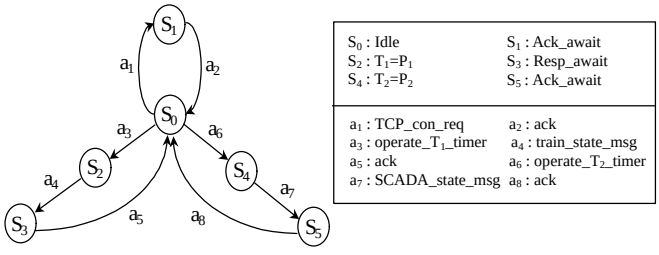

그림 4. 철도 신호제어 프로토콜 type 2에 대한 LTS 모델링

4. 프로토콜 검정기의 구성 및 동작

4.1 검정기의 동작

기술된 검정 방법을 기반으로 개발된 '프로토콜 검정기'는 사용자가 쉽게 다룰 수 있도록 윈도우 NT 환경 하의 GUI(Graphic User Interface) 기능에 의해 visual C++을 이용하여 구현되었고, deadlock, livelock, reachability, liveness 그리고 determinist 와 같은 프로토콜의 특성을 보다 구체적으로 검정해 준다[4]. 구현된 프로토콜 검정기의 구성은 (그림 5)와 같다.

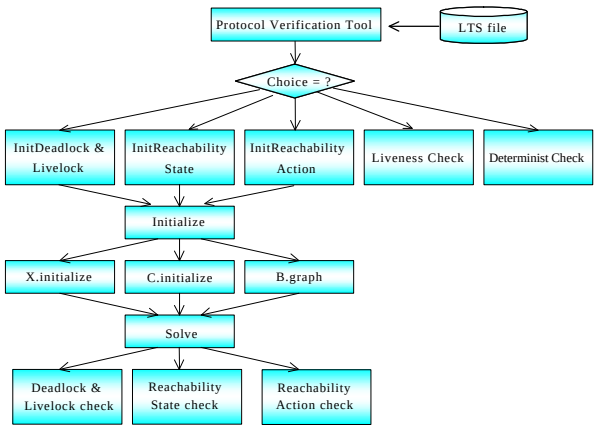

그림 5. 프로토콜 검정기의 동작절차

본 프로토콜 검정기는 LTS 모델의 각 시퀀스를 현재 상태(S_i), 행위(Action), 다음 상태(S_o)로 구성한 LTS 파일을 입력으로 한다. 선택항목 중 사용자가 'Deadlock & Livelock', 'Reachability State', 'Reachability Action' 중에서 한 항목을 선택하면 'InitDeadlock & Livelock', 'InitReachability State', 'InitReachability Action' 부모듈을 호출하여, 정의된 modal mu-calculus 논리식에 따라 각각 max block과 min block을 생성하고, max block과 min block에서 생성된 변수와 입력 LTS의 state를 이용해 bit-vector와 counter를 만든다(Initialize). Initialize 모듈은 bit-vector를 초기화하는 X.initialize, counter를 초기화하는 C.initialize, 그리고 edged- labeled directed graph를 생성하는 B.graph 부모듈을 호출하게 되고, 다음으로 X.initialize와 C.initialize, B.graph에 배열 M이 공집합(empty)이 될 때까지 Solve의 갱신 알고리즘을 적용시켜 Deadlock & Livelock, Reachability State, 그리고 Reachability Action의 검정 결과를 판단한다.

그러나 만약 사용자가 'Liveness' 또는 'Determinist' 를 선택할 경우, 모형검사 알고리즘과는 상관없이 'Liveness check'는 초기상태에서 도달가능한 모든 상태와 행위를 출력하고, 'Determinist check'는 어떤 특정한 상태에서 동일한 행위에 의해 다음 상태가 두 가지 이상 존재하는지를 검사한다.

4.2 검정기 결과 분석

본 소절에서는 각 검정항목에 대해 검정 결과를 판단하는 절차에 대해 간략히 기술한다.

(1) Deadlock & Livelock Check

Solve 알고리즘을 통해서 배열 M이 공집합이 될 때까지 BitArray X와 CounterArray C를 갱신한 결과에서 검정 결과를 판단하는데, deadlock 판정 procedure는 (그림 6)과 같고, BitArray의 요소 X[i][X_5]가 0인 경우의 상태를 검출하면 deadlock 발생여부를 판단할 수 있다. 또한 Livelock Check는 검출 논리식에서 안정성(safety)을 표현한 부 논리식과 관련된 변수 X_3와 X_4가 CounterArray에서 1로 set 되어 있는 경우 검출된다.

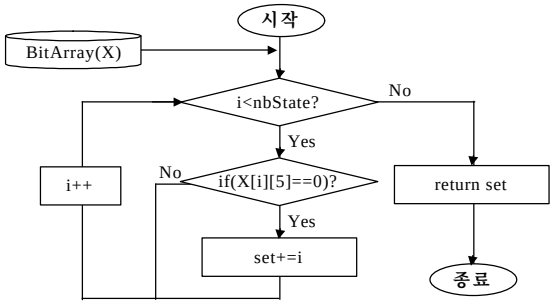

그림 6. Deadlock Check 부모듈의 procedure

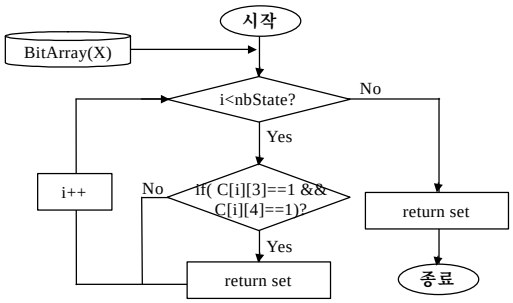

그림 7. Livelock Check 부모듈의 procedure

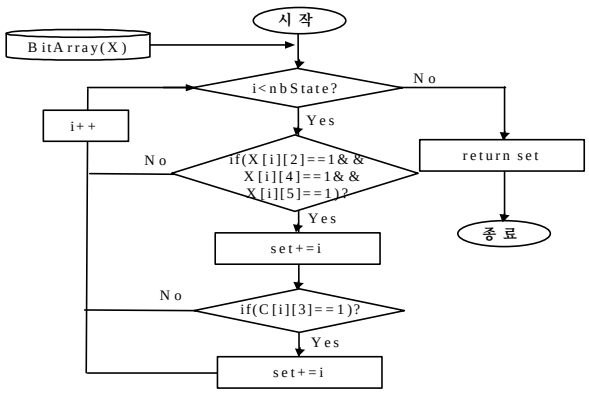

그림 8. Reachability State Check 부모듈의 procedure

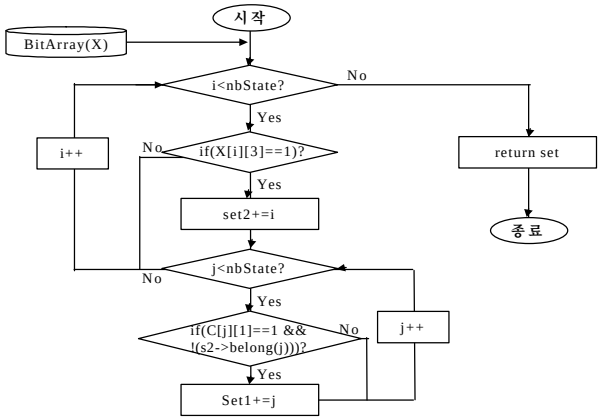

그림 9. Reachability Action Check부모듈의 procedure

(2) Reachability State Check

Reachability State는 먼저 검사 상태인 s와 관련된 BitArray의 요소 X[i][X_2]가 1인 경우와 이 상태에서 act1과 act2와 각각 관련된 BitArray의 요소 X[i][X_4]와 X[i][X_5]가 모두 1인지를 검사하고, 또한 CounterArray C의 요소 C[i][3]의 요소가 1이 되는 값을 검출함으로써 행위 발생여부를 검사한다. 이에 대한 프로시져는 (그림 8)에 나타나 있다.

(3) Reachability Action Check

Reachability Action은 우선 $act_2 \rightarrow act_3$가 발생하는 상태와 관련된 BitArray의 요소 X[i][X_3]가 1인 상태 's2'를 검출하여 set2에 저장하고, 이러한 조건을 만족하는 상태가 존재하면 $act_1 \rightarrow act_2$가 발생하는 상태와 관련된 CounterArray의 요소 C[i][1]가 1이고 act2를 거쳐 's2'를 발생하는 상태인 's1'을 검출하여 set1에 저장한다. Action reachability를 만족하는 이 상태를 return하였을 때, main 함수에서는 set1과 set2안에 값이 있는지 없는지를 판단하여, 둘 중 하나라도 값이 없거나 둘 다 없다면 입력된 3개의 행위에 대해 reachability action을 만족한 것으로 판단한다.

(4) Liveness Check

Liveness Check는 초기 상태와 거기에서 파생되는 행위를 출력한다. 그리고 만약 다음 상태가 초기 상태와 같지 않을 경우, 시작 상태를 다음 상태로 설정하여 그 상태와 거기서 파생되는 행위를 출력한다. 이 과정을 계속 반복하다가 다음상태가 초기상태와 같아지면 부모듈을 종료하며, 이에 대한 절차는 (그림 10)과 같다.

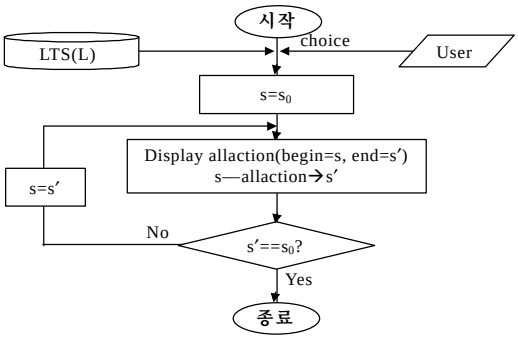

그림 10. Liveness Check 부모듈의 procedure

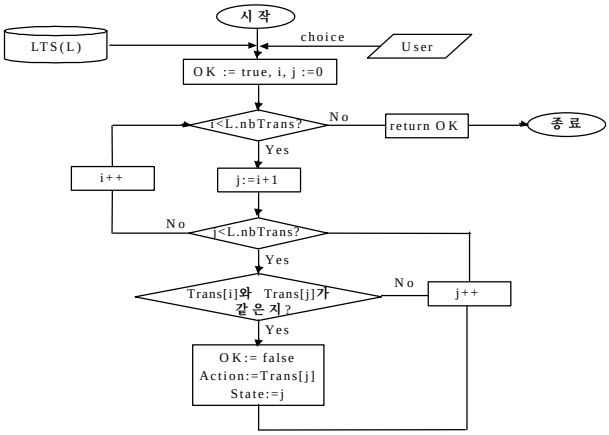

그림 11. Determinist Check 부모듈의 procedure

(5) Determinist Check

Determinist Check는 각 상태에서 동일한 행위에 대해 서로 다른 다음 상태가 있는 경우에는 0을 리턴하고, 그렇지 않은 경우에는 1을 리턴한다. 이에 대한 절차는 (그림 11)과 같다.

5. 검정기 알고리즘을 이용한 검정 예시

4장의 검정 동작의 예를 들어, 프로토콜의 안전성 특성을 표현하는 modal mu-calculus 식으로부터 철도 신호제어용 프로토콜 type 2의 완전성을 검정한다. 결국에는 원자 명제인 상태 A에 도달되어야 하며 deadlock과 livelock이 없음을 나타내는 modal mu-calculus의 논리식은 식(1)과 같고, 본 식은 LTS의 행위 집합 S에 포함된 어떠한 행위가 발생하더라도 그 LTS를 만족함을 나타낸다.

$$\nu Z.(\mu Y.A \vee (<->tt \wedge [-]Y)) \wedge [-]Z,\ A=\{S_0\} \quad \cdots\cdots\cdots\cdots\cdots\cdots \quad (1)$$

(그림 12)는 식(1)에서 최대고정점과 최소고정점을 사용하여 생성된 max block과 min block을 나타내고, (그림 13)은 변수 X_i들의 천이 관계를 나타낸 edge-labeled directed graph G이다.

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

그림 12. Max block, min block

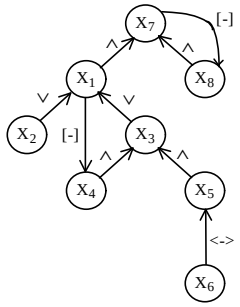

그림 13. Edge-labeled directed graph G

Solve 알고리즘의 초기화 알고리즘을 따라, X_1 - X_6으로 이루어진 min block은 0로, X_7과 X_8로 이루어진 max block은 1로 bit-vector의 값이 초기화되어지고, 초기화 규칙에 대해 값이 변한 요소들의 상태와 변수쌍은 배열 M에 더해진다. 이와 같은 초기화의 결과를 (그림 14)에서 보인다.

배열 M이 공집합(empty)이 될 때까지 Solve의 갱신 알고리즘을 적용시켜 검정 결과를 판단한다. 4.2 소절에 기술하였듯이 bit-vector의 결과에 의해서는 deadlock을, counter의 결과에 의해서는 livelock 여부를 판단하는데, 본 논문에서 임의로 든 예시의 결과 (그림 15)에서는 bit-vector의 X_5의 모든 요소가 1로, counter의 X_3와 X_4의 모든 요소가 0로 갱신되었음을 알 수 있다. 즉, 모든

요소가 max block과 min block을 만족하므로 (그림 4)의 철도 신호 제어 프로토콜 type 2의 LTS 모델은 deadlock과 livelock이 발생하지 않고 논리식 νZ.(μY.A∨(<->tt∧[-]Y))∧[-]Z, A={S_0}를 만족하는 완전한 프로토콜 모델임을 판단한다.

X	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
S_0	1	1	1	1	1	1	1	1
S1	1	0	1	1	1	1	1	1
S2	1	0	1	1	1	1	1	1
S3	1	0	1	1	1	1	1	1
S4	1	0	1	1	1	1	1	1
S5	1	0	1	1	1	1	1	1

C	X_3	X_4
S_0	0	0
S_1	0	0
S_2	0	0
S_3	0	0
S_4	0	0
S_5	0	0

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

그림 14. Bit-vector, counter, 배열 M의 갱신결과

X	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
S_0	0	1	0	0	0	1	1	1
S_1	0	0	0	0	0	1	1	1
S_2	0	0	0	0	0	1	1	1
S_3	0	0	0	0	0	1	1	1
S_3	0	0	0	0	0	1	1	1
S_5	0	0	0	0	0	1	1	1

C	X_3	X_4
S_0	2	3
S_1	2	1
S_2	2	1
S_3	2	1
S_4	2	1
S_5	2	1

M[1]=<<S_0,X_2>, <S_0,X_6>, <S_1,X_6>, <S_2,X_6>, <S_3,X_6>, <S_4,X_6>, <S_5,X_6>>
M[2]=< >

그림 15. Bit-vector, counter, 배열 M의 초기화 결과

(그림 16)은 이상에서 예로든 철도 신호 제어 프로토콜 type 2 모델의 완전성을 검정하는 프로토콜 검정기의 실행 결과이다.

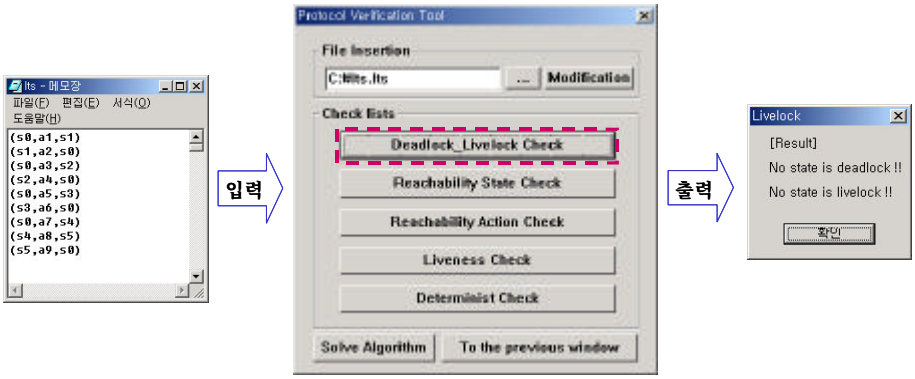

그림 16. 프로토콜 검정기의 실행결과

6. 결론 및 향후 계획

본 논문에서는 철도 신호 제어 프로토콜 type 2를 대상으로 LTS로 모델링된 프로토콜을 기반으로 시스템 특성 언어인 Modal Mu-Calculus를 이용한 검정기를 컴퓨터로 구현함을 보였다. 개발에 사용된 알고리즘은 검정하고자 하는 사항별로 모형 검사 방법을 사용함으로써, 시스템 모델을 매우 정확하고 명료하게 검사하도록 하여 비형식적 방법에서 야기될 수 있는 오류와 모호함을 제거하였다.

해당 프로토콜의 정확성을 검정한 결과에 따르면, 열차집중제어장치(CTC)와 전력 SCADA 장치 사이의 상태정보 전송방식 프로토콜인 type 2 프로토콜은 안전성과 필연성을 모두 만족하는 규격으로 그 내용이 검정되었다.

본 연구의 개발 도구는 철도 신호장치간 인터페이스 정보전송방식 표준을 정형기법에 의해 검정함으로써 관련 프로토콜의 유지보수를 위한 추가적인 인건비와 계속되는 오류상황에 소요되는 운용비용 절감 효과가 기대되고 개발된 검정기법을 철도 신호 제어시스템 표준 개발에 응용하여 각종 철도 통신장비의 안전성과 신뢰성 보장에 기여한다.

향후 추진 사항은 어떤 시간 단위의 정수배로 증가하는 이산시간(discrete time) 영역뿐만 아니라 실시간에서도 검정이 가능한 새로운 정형 기법의 개발을 연구하는 것이다.

참고 문헌

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 Academic Publishers, Model Checking, 1996.
3. R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus, Formal Methods in System Design 2(2) : pp.121-147, 1993.
4. R. Cleaveland, Tableau - Based Model-Checking in the Propositional Mu-Calculus, Acta Informatica 27 : 725-727, 1990.

후 기

본 연구는 철도청 철도기술연구개발사업으로 지원된 “신뢰성 확보를 위한 프로토콜 검정기 및 적합성시험 생성도구 연구” 과제의 연구결과의 일부입니다.