ISSN 2287-4348

제2권 제1호

Vol.2 No.1

한국스마트미디어학회
2013년도 춘계학술대회 학술발표 논문집

Proceedings of KISM Spring Conference 2013
2013 순천정원엑스포 ICT 합동학술대회

일시 : 2013년 5월 31(금) ~ 6월 1일(토)
장소 : 순천대학교 70주년 기념관
주최 : (사)한국스마트미디어학회
(사)한국인터넷정보학회
주관 : 순천대학교, 2013순천만국제정원박람회조직위원회

http://www.kism.or.kr

KOREAN INSTITUTE OF SMART MEDIA
한국스마트미디어학회

10:22-10:35 제목 : **부분 암호화를 통한 안드로이드 애플리케이션 암호화 기법의 성능 개선 /040**
저자 : 김동진(단국대), 정윤식(단국대), 조성제(단국대), 최종무(단국대)

10:35-10:48 제목 : **선형시간시상논리를 이용한 스마트미디어의 자동 검증에 관한 연구 /043**
저자 : 조성진(홍익대), 김보연(홍익대), 양효석(홍익대), 손현승(홍익대), 박보경(홍익대) 서채연(홍익대), 김영철(홍익대)

10:48-11:00 제목 : **태스크 의존성 그래프 추출 도구의 개발 /046**
저자 : 백한별(숭실대), 이준우(숭실대), 김강희(숭실대)

S1-C. Image Processing I (영상처리I)

(3강의실 / 09:30~11:00)
좌 장 : 이귀상(전남대)

09:30-09:43 제목 : **숫자에지의 형태학적 특징을 이용한 차량 번호판 검출 /050**
저자 : 김광복(전남대), 김수형(전남대), 나인섭(전남대)

09:43-09:56 제목 : **Haar-like Features and Adaboost를 기반으로 한 번호판 검출 /054**
저자 : 엠디 무스타파 카말 사커(전북대), 장대석(전북대), 윤숙(목포대), 박동선(전북대)

09:56-10:09 제목 : **모폴로지 연산과 신경망 시스템을 이용한 차량 번호판 검출 및 숫자 인식 /058**
저자 : 최인수(전북대), 진문용(전북대), 박동선(전북대)

10:09-10:22 제목 : **컬러 및 구조적 특징을 이용한 차량 번호판 영역 추출 시스템 /062**
저자 : 김선윤(전북대), 홍창표(전북대), 김형석(전북대)

10:22-10:35 제목 : **Drop Fall 알고리즘기반의 개선된 악보기호분할 /066**
저자 : 반티미융(전남대), 김수형(전남대), 나인섭(전남대), 오아란(전남대)

선형시간시상논리를 이용한 스마트미디어의 자동 검증에 관한 연구

조성진, 김보연, 양효석, 손현승, 박보경, 서채연, 김영철
홍익대학교 소프트웨어공학연구실
e-mail : sjcho@selab.hongik.ac.kr

A Study on the Automatic Verification of Smart Media using Linear Time Temporal Logic

Seong Jin Cho, B. Kim, H. Yang, H. Son, B. Park, C. Chae, R. Kim
SE Lab., Hongik University

요 약

본 논문에서는 선형시간시상논리(Linear Time Temporal Logic, LTTL)을 이용한 스마트미디어(Smart Media)의 자동검증시스템(Automatic Verification System) 구현을 제안한다. 스마트미디어의 선형시간시상논리를 이용한 자동검증시스템은 유한상태그래프의 생성(Generation of Finite State Graph)하는 부분과 모델검사(Verification of Model)를 위한 부분으로 구성된다. 본 논문에서 설계한 확장된 Tempura(Extended Tempura, E-Tempura)는 인터리빙(interleaving) 모델에 대한 비결정적 기능(Nondeterministic Ability)을 제공하며, 유한상태그래프를 생성한다. 본 논문에서 E-Tempura는 인터리빙을 위한 기능과 동기화된 메카니즘(Synchronized Mechanism)을 포함하며, E-Tempura 인터프리터(E-Tempura Interpreter)는 유한상태그래프를 생성할 수 있도록 구현되었다.

키워드 : 스마트미디어, 선형시간시상논리, 분기시간시상논리, 유한상태그래프, E-Tempura

1. 서 론

스마트미디어(Smart Media)들은 집, 사무실, 공장 등의 공간에 다양한 형태로 존재하며, 삶의 편리함을 제공하여 준다. 이와 같은 스마트미디어들은 공간 내의 상황을 인지(Context Awareness)하고, 다양한 센서(Sensor)들로부터 수집된 정보를 기반으로 어떠한 서비스를 제공할 수 있는지를 결정하게 되며, 이를 유비쿼터스 환경(Ubiquitous Environments)라고 한다.

유비쿼터스 환경에서의 스마트미디어들과 다양한 형태의 센서들은 미들웨어(Middleware)를 통하여, 입력되는 다양한 정보들을 처리하며, 이들 장치들로의 입력과 출력, 소프트웨어와 하드웨어의 올바른 동작여부의 검증은 매우 중요한 요소로 작용되고 있다. 본 논문에서는 이러한 검증을 위하여 시상논리(Temporal Logic)를 이용한 검증방법을 제안한다.

시상논리(Temporal Logic)는 기존의 논리(Logic)에 시간 개념을 추가한 것으로서, 소프트웨어(Software)와 하드웨어 행동(Hardware Behaviour)을 기술하고 증명하는 유용한 도구로서 이용되어지고 있으며, 유한상태그래프(Finite State Graph)를 구성하여 검증한다.

유한상태그래프를 효율적으로 간편하게 검증하기 위한 방법은 시스템을 상태전이그래프(State Transition Graph)로 표현하여, 이 구조가 시스템의 여러 가지 특성을 만족하는 지를 결정하기 위해 효율적인 검증 알고리즘을 사용하는 것이다. 시상논리가 프로그램의 검증에 처음 사용된 것은 Bustall에 의해서이며, 이후 Pnuelli, Lamport, Owicki 등의 여러 학자에 의해 발전되어 왔다[1,4,7].

시상논리는 크게 선형시간시상논리(Linear Time Temporal Logic, LTTL)과 분기시간시상논리(Branching Time Temporal Logic, BTTL)로 나누어진다. 이들 논리를 이용하여, 시상구조식(Temporal Formula)에 적합한 상태를 생성할 수 있는 프로그래밍 언어로 Tempura, Tokio 등이 개발되었으며, 이 중에 Tempura는 선형시간시상논리를 기초로 하여, 명령형 프로그래밍 언어의 기능을 추가한 프로그래밍 언어이다[2].

Moszkowski는 1983년에 프로그래밍 언어 Prolog, 1984년 프로그래밍 언어 LISP를 이용하여, Tempura 인터프리터(Interpreter)를 개발하였고, 1984년, 이중단계기억장치(Two Level Memory)와 다중통로스캐닝(Multi Pass Scanning)을 이용하여 수정하였다. 이후, 1985년, Roger Hale은 C 프로그래밍 언어로 Tempura 인터프리터를 개발하였다[3].

시상논리를 이용한 증명방법들로서는 추론방법, 규칙 및 공리를 이용하는 방법, 시상논리로 주어진 구조식을 만족할 수 있는 모델(Model)을 생성할 수 있는 방법, 그리고

* 본 연구는 지식경제부 및 한국산업기술평가관리원의 산업원천기술개발사업의 일환으로 수행하였음. [10035708, 고신뢰 자율제어 SW를 위한 CPS(Cyber-Physical Systems) 핵심 기술 개발]

유한상태그래프로 변환하고, 모델검사프로그램(Model Checking Program)에 의해 주어진 유한상태그래프가 증명하려는 구조식에 만족하는 가를 검사하는 방법 등이 있다. 이 중에서 유한상태그래프를 이용하는 방법은 증명하려는 구조식에 관계없이 동일한 방법으로 증명할 수 있을 뿐만 아니라, 증명과정을 자동화할 수 있는 장점이 있다[5]. 그러나 Tempura에서의 프로그램 수행과정은 결정적(Deterministic)하고, 미래의 상태가 항상 단일한 형태를 취하므로, 인터리빙 모델을 기술하기 위한 비결정성(Non-Deterministic)을 제공할 수 없다.

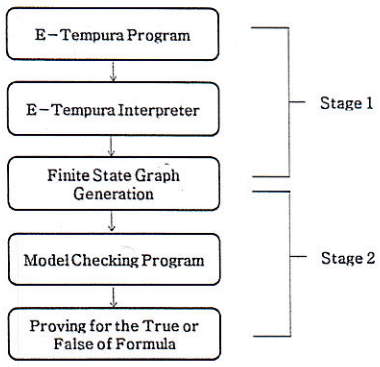
본 논문에서는 선형시간시상논리 프로그래밍 언어인 Tempura에 <cobegin> - <coend>, <await> 등의 구문과 기능을 추가하여, 비결정성을 제공하는 E-Tempura (Extended Tempura)를 설계하였으며, E-Tempura는 변수의 변환에 따른 하드웨어 행동을 증명할 수 있는 선형시간시상논리 프로그래밍 언어이다.

2. 관련 연구

2.1 자동검증시스템의 개요

자동검증시스템(Automatic Verification System)은 그림 1과 같이 구분되며, 각 단계(Stage)의 내용은 다음과 같다.

- 단계 1 : 실행할 프로그램을 번역하여 가능한 모든 상태(State)를 생성한다. 인터프리터에 의해 번역된 유한상태그래프의 각 상태는 지역변수의 값(Value of Local Variable), 다음 상태(Next State)에 관한 정보 및 다른 상태로의 전이(Transition)에 관한 정보를 포함할 수 있어야 한다.
- 단계 2 : 증명하려는 구조식과 유한상태그래프를 입력으로 하여 필요한 모델을 생성하고, 증명하려는 구조식의 진위여부(True or False)를 판단한다. 즉, 모델검사프로그램을 통하여 단계 1에서 생성된 유한상태그래프에서 안전성(Safety), 유효화(Liveness) 등의 성질을 만족하는 가의 여부를 증명한다.



(그림 1) 선형시간시상논리 검증시스템

2.2 시상논리와 Tempura

시상논리는 크게 분기시간시상논리(BTTL)와 선형시간시상논리(BTTL)로 구분된다.

분기시간시상논리의 유한상태시스템의 자동검증은 매우 효율적이며, 시간복잡도(Time Complexity)가 모델의 크기와 증명하고자하는 구조식에 선형적으로 비례한다. 그러나 분기시간시상논리는 상태 간의 모든 전이가 비조건적(Unconditional)이므로, 병렬시스템(Parallel System)의 중요한 성질인 공정원칙(Fairness) 등을 표현하기 어려우며, 따라서 공정원칙을 표현하기 위해서는 분기시간시상논리의 논리적 의미를 변경해야한다(예 : CTL^*, CTL^F). 분기시간시상논리는 Clarke, Emerson 등에 의해 자동검증시스템으로 개발되었으며, 간단한 하드웨어 검증의 예가 제시되었다[5].

선형시간시상논리는 프로그램이 유한상태그래프로 표현되는 경우, 분기시간시상논리와 같이 모델검증프로그램을 이용하여 자동검증할 수 있으며, 시간복잡도는 상태의 수에 선형적으로 비례하나, 구조식의 길이에는 지수적으로 비례한다[8]. 그러나 실제 응용의 경우에서는 증명하고자 하는 안전성, 유효화 등은 구조식의 길이가 비교적으로 짧으므로 분기시간시상논리와 같이 모델의 크기에 선형적으로 비례한다. 또한 선형시간시상논리는 공정원칙의 표현력이 우수하므로, 병렬시스템의 모델링 및 검증에 유용하게 사용할 수 있다[11,12].

2.3 Tempura의 특성 및 실행방법

Tempura는 일반 수학적인 논리에 의한 프로그래밍 언어에 □("always"), ○("next") 등의 연산자가 추가되었다.

□ (x=0) ∧ ○(¬○true)
State 0: (x=0) ∧ ○(□ (x=0) ∧ (¬○true))
State 1: (x=0) ∧ ¬○true ∧ ○□(x=0)≡(x=0)∧○(false∧□(x=0))

(그림 2) Tempura 구조식 수행과정의 예

Tempura는 그림 2에서와 같이 주어진 프로그램의 시간에 따른 행동을 현재와 나머지 상태로 나누어 프로그램에 맞는 상태들을 생성한다. 그러나 Tempura는 한 프로그램 내의 실행방법에 있어 단일한 형태의 미래만이 존재하므로, 유비쿼터스 환경의 스마트미디어와 센서들의 동시성 프로그램(Concurrent Program)과 하드웨어 행동을 기술하기에 적합하지 않다.

2.3 E-Tempura의 유도된 연산자

Tempura는 스마트미디어와 센서들에 대해 주어진 프로그램에 맞는 상태를 자동적으로 생성할 수 있으며, 아울러 소프트웨어 및 하드웨어의 검증을 위한 모의실험(Simulation)에 매우 편리하며, 특히 병렬처리(Parallel Processing) 등을 기술하는 데에 유용하게 사용되어질 수 있다[9,10]. 스마트미디어와 각종 센서들에 대한 병렬처리 과정의 모의실험을 위해 Tempura의 기본 연산자를 이용하여 E-Tempura에서 유도한 연산자는 그림 3과 같다.

empty	$\equiv_{def}$	¬○true
skip	$\equiv_{def}$	○empty
len n	$\equiv_{def}$	$○^n$ empty
keep w	$\equiv_{def}$	always(¬empty → w)
A gets B	$\equiv_{def}$	keep((○A) = B)
stable A	$\equiv_{def}$	A gets A
halt C	$\equiv_{def}$	always(C ≡ empty)
fin w	$\equiv_{def}$	always(empty → w)

(그림 3) E-Tempura의 유도된 연산자

2.4 E-Tempura에 추가된 구문들

유비쿼터스 환경에서의 스마트미디어와 다양한 센서들은 모의실험을 위하여 동시성 프로그램과 병렬프로그램으로 모델링해야하는 경우가 일반적이며, 이를 위한 인터리빙 모델구성을 위해 기존의 Tempura에 추가된 구문들은 그림 4와 같다.

```
<concurrent_component>
    ::= cobegin <statement_sequence>
          {;; <statement_sequence>}
        coend
<statement_sequence>
    ::= <statement>
      | <statement> ; <statement_sequence>
<statement>
    ::= <label> : <unlabelled statement>
      | <unlabelled statement>
<loop_statement>
    ::= loop <statement> endloop
      ≡ while more do <statement>
<await_statement>
    ::= await (<condition>)
```

(그림 4) E-Tempura에 추가된 구문들

3. 결 론

본 논문에서는 유비쿼터스 환경에서의 스마트미디어와 각종 센서들의 검증을 위하여, 선형시간시상논리 프로그래밍 언어인 Tempura를 확장한 E-Tempura를 제안하고 구현하였으며, 새로이 <cobegin> - <coend>, <await> 등의 구문을 추가하여, 인터리빙 모델에 대한 비결정성 제공과, 유한상태그래프의 구성을 가능하도록 하였다.

E-Tempura에 의해 생성된 유한상태그래프는 모델검사 프로그램을 이용하여, 병렬프로그램과 소프트웨어 및 하드웨어 행동에 대한 자동검증을 위해 이용되어질 수 있다. 이에 대한 문제점으로서는 유한상태그래프의 생성과정에서 상태정보를 저장하기 위해 많은 기억용량이 필요하며, 향후 이에 따른 기억용량의 효율성을 높이기 위한 연구가 이루어져야한다.

참 고 문 헌

[1] Susan Owicki, Leslie Lamport, "Proving Liveness Properties of Concurrent Programs", ACM Transactions, on Programming Languages and Systems, Vol. 4, No. 3, July, 1982.

[2] M. Fujita, S. Kono, H. Tanaka, T. MotooKa, "Tokio : Logic Programming Language based on Temporal Logic and it's Complication to Prolog".

[3] B. C. Moszkowski, "Executing Temporal Logic Programs", Cambridge University Press, 1986.

[4] Leslie Lamport, "Specifying Concurrent Program Modules" ACM Transactions on Programming Languages and Systems, Vol. 5, No. 2, April, 1983.

[5] E. M. Clarke, E. A. Emerson, A. P. Sistla, "Automatic Verification of Finite State Concurrent Systems using Temporal Logic Specifications", ACM Transactions on Programming Languages and Systems, Vol. 8, No. 2, April, 1986.

[6] Michael C. Browne, "An improved Algorithm for the Automatic Verification of Finite State Systems using Temporal Logic", IEEE, 1986.

[7] B. Hailpern and Owicki, "Verifying Network Protocols using Temporal Logic", NBS Trends and Applications Symposium, May 29, 1980, pp. 18-28.

[8] A. P. Sistla, and E. M. Clarke, "The complexity of Propositional Linear Temporal Logic", Journal of the Association for Computing Machinery, Vol. 32, No. 3, July, 1985, pp. 733-749.

[9] Roger Hale, "Temporal Logic Programming", In Galton, ed., "Temporal Logic and it's Applications, Academic Press, London, 1987.

[10] Roger Hale and Ben Moszkowski, "Parallel Programming in Temporal Logic", In Proceedings of PARLE 87, Eindhoven, Netherlands, June, 1987, Published as Lecture Notes in Computer Science 259, Springer Verlag.

[11] Orna Linchtenstein and Amir Pnueli, "Checking that Finite State Concurrent Programs Satisfy their Linear Specification", ACM, 1987.

[12] E. Allen Emerson and Chin Laung Lei, "Modalities for Model Checking: Branching Time Strikes Back", ACM, 1984.

[13] Zohar Manna, Amir Pnueli, "The Temporal Logic of Reactive and Concurrent Systems", Springer, 1992, pp. 179-273.

[14] E. M. Clark, E. A. Emerson, A. P. Sistla, "Automatic Verification of finite-state concurrent systems using Temporal logic specifications", ACM, 1986, pp. 244-263