고장수목을 이용한 테스트 케이스의 안전성 측정

윤상현 ○, 조재연, 유준범
건국대학교 정보통신대학
pctkdgus@konkuk.ac.kr, tm77@konkuk.ac.kr, jbyoo@konkuk.ac.kr

A Safety Measurement of Test Case using Fault Tree Analysis

Sanghyun Yoon○, Jaeyeon Jo, Junbeom Yoo
College of Information and Communication, Konkuk University

요 약

테스팅은 테스트 요구사항을 기반으로 수행하기 때문에 테스트 요구사항의 품질은 테스팅 전체 과정의 품질과 직결된다고 할 수 있다. 그러나 테스트 요구사항에서 고려해야 할 점들은 시스템의 도메인과 목적에 따라 다르기 때문에 양질의 테스트 요구사항이라고 판별할 수 있는 기준을 정하는 것은 어려운 일이다. 본 연구에서는 테스트 케이스와 고장 수목의 최소 절단집합을 각각 정형모델로 변환하여 모델체킹을 함으로써, 테스트 요구사항의 안전성을 측정하는 방법을 제시한다. 테스트 요구사항이 반영된 테스트 케이스는 모델체킹의 대상이 되는 정형모델로 변환하였으며, 고장수목의 최소 절단집합은 CTL 검증 속성으로 변환하여 테스트 케이스에서 생성된 정형모델이 안전성을 만족하는지 만족하는지 모델체킹을 적용하여 확인하였다

1. 서 론

테스팅은 테스트 요구사항을 기반으로 수행하기 때문에 테스트 요구사항의 품질은 테스팅 전체 과정의 품질과 직결된다고 할 수 있다. 그러나 테스트 요구사항에서 고려해야 할 점들은 시스템의 도메인과 목적에 따라 다르기 때문에 양질의 테스트 요구사항이라고 판별할 수 있는 기준을 정하는 것은 어려운 일이다. 본 연구에서는 테스트 요구사항의 품질을 측정하기 위한 하나의 기준으로 소프트웨어 고장수목 분석(software fault tree analysis)[1]의 최소 절단집합(minimal cut set)[4]을 이용하였다. 테스트 요구사항과 고장수목의 최소 절단집합을 각각 정형모델로 변환하여 모델체킹 기법을 사용함으로써 테스트 요구사항이 안전성 분석의 관점에서 고려해야 할 점들을 감안하였는지 판단하는 접근을 해보았다.

일반적으로 테스트 요구사항은 주로 자연어로 명세되어 정형기법을 적용하기가 어려우므로 테스트 케이스에 테스트 요구사항이 충분히 반영되어 있다는 가정하에 테스트 케이스를 모델체킹의 대상인 정형모델로 변환하였다. 고장수목 분석의 대상은 주로 시스템에서 가장 중요한 실패 (failure)이기 때문에 테스팅에서 필수적으로 고려해야 할 부분에 대한 분석을 한다고 할 수 있으며, 고장수목의 최소 절단집합은 논리적인 조합으로 표현되기 때문에 분석된 결과를 다른 형태로 변환하기에도 유리하기에 연구의 목적에 적합하다는 판단을 하였다. 이를 CTL(Computational Tree Logic) 검증 속성으로 변환하여 테스트 케이스에서 생성된 정형 모델이 검증 속성을 만족하는지 모델체킹을 적용하였다.

본 논문의 구성은 다음과 같다. 2장에서는 연구에서 사용한 예제에 대한 가정과 전체 내용을 대략적으로 설명한다. 3장과 4장에서는 각각 고장 수목 분석의 최소 절단집합과 테스트 케이스를 정형 모델로 변환하기 위해 고려해야 할 점들과 과정을 소개하며, 5장에서 결론을 기술하였다.

2. 테스트 케이스와 소프트웨어 고장수목의 최소 절단집합의 비교

소프트웨어 요구사항과 테스트 요구사항을 비교하기 위해서 본 연구에서는 각각을 정형 모델로 변환하고 이들을 cadence SMV[2]를 사용하여 모델체킹을 하는 접근을 해보았다. 테스트케이스에서 변환된 SMV 입력 프로그램이 소프트웨어 고장수목에서 변환된 CTL property를 만족하면 테스트 케이스가 안전성을 고려하였다고 판단할 수 있다. 그림 1은 전체 과정을 보여주고 있다.

요구사항이 자연어로 기술되어 있으면 이를 정형 모델로 변환하거나 테스트 케이스, 디자인 모델 등에 반영

"본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과와 (NIPA-2011-(C1090-1131-0008)) 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원(2010-0002566) 그리고 지식경제부 및 한국산업기술평가관리원의 산업원천기술개발사업의 일환으로 수행하였음. [10035708, 고신뢰자율제어 SW를 위한 CPS(Cyber-Physical Systems) 핵심 기술 개발]"

되었는지 판별하기가 어렵다. 이를 해결하기 위해 소프트웨어 요구사항은 UML의 state diagram을 사용하여 요구사항 모델로 정의하였고, 테스트 요구사항은 테스트 케이스에 충분히 반영되어 있다는 가정을 하였다. 그리고 정형모델로의 변환을 단순화 하기 위해 테스트 케이스는 소프트웨어 요구사항 모델만을 대상으로 생성하였다. 3 장과 4 장에서 고장 수목 분석의 최소 절단집합과 테스트 케이스를 정형 모델로 변환하기 위해 고려해야 할 점을 살펴보겠다

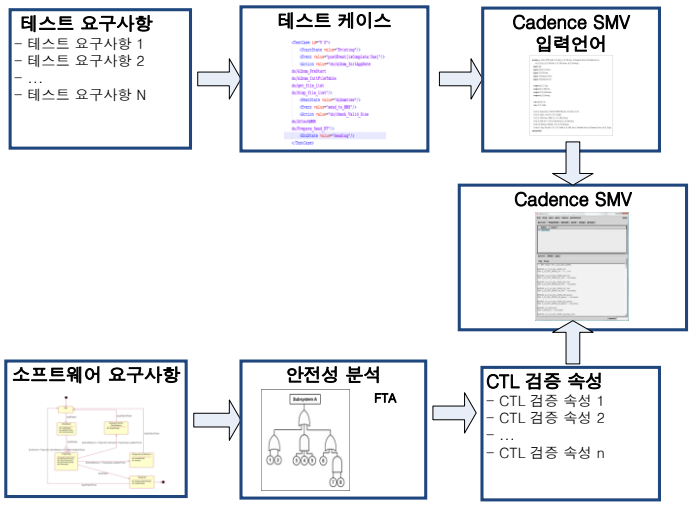

그림 1 소프트웨어 요구사항과 테스트 요구사항의 비교

3. 고장 수목 분석의 최소 절단 집합

고장 수목 분석은 전문가가 시스템에 대한 이해와 해당 도메인에 대한 지식을 가지고 시스템에 영향을 줄 수 있는 실패(failure)에 대한 원인을 나무형태로 그려나가면서 귀납적으로 분석하는 방법이다. 테스트 케이스와 최소 절단집합을 모델체커를 이용하여 비교하기 위해서는 각각에서 사용되는 시스템의 표현을 일치시킬 필요가 있다. 테스트 케이스는 소프트웨어 요구사항 모델에서만 생성한다는 가정을 하였으므로, 최소 절단 집합에서 사용되는 표현도 소프트웨어 요구사항 모델에 명세 되어있는 표현으로 나타낼 필요가 있다. 여기에서 연구의 목표를 위해 고려해야 할 점은 실패에 대한 원인, 즉 최소 절단집합이 자연어로 나올 수 있다는 점과 시스템을 명세한 사람과 고장 수목 분석을 하는 전문가들의 도메인에 대한 지식이 서로 다를 수가 있다는 것이다. 최소 절단집합이 자연어로 되어 있으면 정형 명세 모델의 표현으로의 체계적인 변환은 어려울뿐더러, 도메인에 대한 지식이 일치하지 않으면 자연어를 제외하고는 실패에 대한 원인을 표현할 방법이 없게 된다. 이런 문제점을 해결하기 위해서는 고장 수목 분석을 하기 전에 소프트웨어 요구사항 명세 고장 수목 분석의 도메인을 일치시켜 [3]와 같이 duration calculus등의 시멘틱을 사용하여 표현을 일치시키거나, 그림 2와 같이 고장 수목 분석의 결과에 따라 표현이 불가능한 부분을 채워 넣기 위해 소프트웨어 요구사항을 수정하고 이를 대상으로 다시 고장 수목 분석을 하는 방법을 생각해 볼 수 있다.

본 연구에서는 후자의 방법을 사용해 사람이 직접 주어진 소프트웨어 요구사항 모델을 수정하고 이를 기반으로 최소 절단집합의 표현을 일치시키는 시도를 해보았다. 그러나 사람이 직접 수행하기 때문에 자연어를 해석하기에 따라 정형적 표현이 달라질 수 있는 문제점이 있었고 이를 다시 CTL 검증 속성으로 다른 사람이 변환하면 원래의 자연어로 되어있는 최소 절단집합의 내용과는 동떨어질 우려가 있었다. 따라서 실제 모델체킹에서는 자연어를 포함하는 최소 절단집합을 보고 전문가가 직접 CTL 검증 속성을 생성하는 방법을 적용하였다.

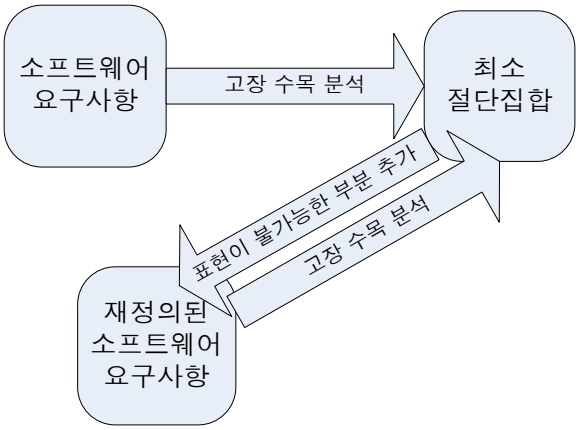

그림 2 고장 수목 분석의 결과를 소프트웨어 요구사항에 반영

4. 테스트 케이스의 SMV 입력 프로그램으로 변환

테스트 요구사항을 테스팅에 반영하기 위해서는 테스트 요구사항이 반영된 테스트 케이스, 그리고 이를 기반으로 한 실제 테스팅을 하기 위한 테스트 데이터를 생성하는 것이 중요하다고 할 수 있다. 일반적으로 자연어로 표현되는 테스트 요구사항을 테스트 케이스에 체계적으로 반영하기 위해서는 테스트 요구사항 자체가 정형 명세로 되어 있어야 하며 테스트 케이스와 데이터에 반영되어있는지 요구사항, 디자인, 소스코드간의 추적성을 살피는 방법이 제시되어야 할 것이다. 본 논문에서는 이런 과정을 간략하게 하기 위해 테스트 케이스를 UML의 state chart diagram만을 대상으로 생성하여 연구를 진행하였다.

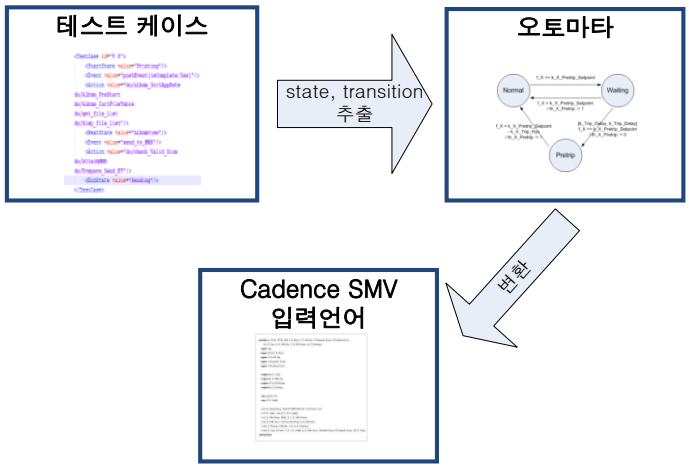

그림 3 테스트 케이스에서 Cadence SMV 입력언어로의 변환

그림 3은 본 연구에서 제시하는 테스트 케이스를 모델체킹의 대상이 되는 정형 모델로 변형하는 과정을 나타내고 있다. 테스트 케이스는 state diagram으로 생성한 시스템의 제어 흐름 그래프를 대상 생성하였기 때문에 state diagram의 상태와 상태들간의 천이를 포함하고 있다. 테스트 케이스를 cadence SMV의 입력 프로그램으로 변형하기 위해 오토마타 형태로 변경하는 과정에서 테스트 케이스에 있는 상태와 천이를 오토마타의 상태와 천이로 정함으로써 알고리즘을 사용한 변환이 가능하였다.

5. 결 론

본 연구에서는 고장 수목의 최소 절단집합을 이용하여 테스트 요구사항의 품질을 측정할 수 있는 기법을 제시하였다. 본 연구에서 제시한 방법은 아직 연구 초기단계이기 때문에 많은 가정을 가지고 있고 테스트 케이스의 변환 과정을 단순화 하기 위해 소프트웨어 요구사항만을 대상 생성한 테스트 케이스를 이용하였다. 이를 실제 예제를 가지고 적용하기 위해서는 시스템 디자인과 소스코드 단계까지 대상으로 연구를 확장해야 할 것이며 각 단계의 추적성에 대한 연구 및 각 변환과정을 체계적으로 하는 기준을 만드는 연구도 진행 되어야 할 것이다.

참고문헌

[1] N. G. Leveson and P. R. Harvey, “Software fault tree analysis”, *Journal of Systems and Software*, Vol. 3, pp. 173-181, 1983.

[2] K. L. McMillan, *Getting started with SMV*, Cadence Berkeley Laboratories, 1998

[3] Hansen, K. M., Ravn, A. P., and Stavridou, V., “From safety analysis to software requirements”, *IEEE Transactions on Software Engineering*, 24(7):574-584, July 1998

[4] Kececioglu, D., *Reliability Engineering Handbook*, Volume 2, Prentice Hall, Inc., New Jersey, 1991.