항공기 소프트웨어에서 경합치유도구를 검증하기 위한 합성프로그램 모음

(A Synthetic Program Suite to Validate Healing Race Tool in Airborne Software)

백 형 진†, 이 건 표‡, 최 으 뜸‡, 전 용 기†‡*
†국립 경상대학교 항공우주 및 소프트웨어학과, ‡국립 경상대학교 정보과학과
(Hyoung-Jin Baek, Keon-Pyo Lee, Eu-Teum Choi, Yong-Kee Jun)
(†Department of Aerospace and Software Engineering, Gyeongsang National Univ.,
‡Department of Informatics, Gyeongsang National Univ.)

Abstract : It is difficult to verify for healing race tool in multi-threaded airborne software. because, existing verification method for healing such as benchmark, real-world programs, and in-house synthetic programs are ineffective to use in the airborne software. This paper provides a synthetic race program suite made by classifying race patterns in real-world programs and benchmark programs. To evaluate this suite, we are tested with Simulated-IMA that is supported ARINC 653 specification.

Keywords : airborne software, race, healing, synthetic program

Ⅰ. 서론

멀티스레드를 사용하는 항공기 소프트웨어의 수행 중에 발생할 수 있는 경합은 프로그램의 신뢰성 보장을 어렵게 한다. 따라서 경합은 탐지하여 제거되거나 수행 중에 치유되어야 한다. 이러한 경합치유기법은 시스템의 신뢰성을 보장하기 위해 성능검증이 필요하다. 하지만 벤치마크와 실세계 프로그램은 항공기 소프트웨어에 탑재할 수 없고 자체 합성프로그램은 경합이 누락될 수 있기 때문에 검증결과의 신뢰성이 낮다.

본 논문은 벤치마크 프로그램과 실세계 프로그램을 분석하고 분류하여 항공기 소프트웨어에서 경합치유도구를 검증하기 위한 합성프로그램 모음을 제시한다. 합성프로그램 모음의 경합유형에 따라 순서위배, 원자성위배 및 양성위배를 발생시키는 프로그램으로 설계한다. 설계된 프로그램을 ARINC-653를 지원하는 Simulated-IMA에서 구현한다. 최종적으로 합성프로그램을 검증하기 위해 실행하여 결과 값과 실제 실행 값을 비교한다.

* 교신저자(Corresponding Author)
백형진, 이건표, 최으뜸 : 경상대학교
전용기: 경상대학교 정보과학과, 항공우주및소프트웨어학과, 항공기부품기술연구소, 공학연구원

※ 이 논문은 교육부 지방대학특성화(CK-I)사업의 재원으로 경상대학교 창의적항공IT기계융합인력양성사업의 지원을 받아 수행되었음.

Ⅱ. 연구배경

항공기 소프트웨어는 고신뢰성을 요구하는 임베디드 소프트웨어이다. 멀티스레드를 사용하는 항공기 소프트웨어에서 경합은 의도하지 않은 결과를 발생시켜 신뢰성 보장을 어렵게 한다. 따라서 경합은 탐지하여 제거되거나 수행 중에 치유되어야 한다.

동시성오류의 한 종류인 경합[1]은 발생 유형에 따라 원자성위배(atomicity violations)와 순서위배(order violations)로 분류된다. 원자성위배는 원자적으로 실행되어야 하는 공유자원을 가진 코드영역에서 개발자의 잘못된 추정으로 인해 위배가 발생하는 것이다. 순서위배는 서로 다른 스레드들이 공유자원에 접근하는 이벤트의 순서를 보장할 수 없는 것이다.

경합치유[2]는 프로그램 수행 중에 존재하는 경합에 의해 시스템이 실패(failure)할 가능성을 낮추어 정상적으로 운용될 수 있도록 하는 기법이다. 이러한 치유기법은 시스템의 신뢰성을 보장하기 위해 모든 경합 발생유형에 대한 성능검증이 필요하다.

기존의 치유기법의 성능검증 방법은 벤치마크 프로그램, 실세계 프로그램, 또는 자체 개발한 합성프로그램을 이용하는 방법이 일반적이다. 하지만 벤치마크와 실세계 프로그램은 항공기 소프트웨어에 탑재할 수 없고 자체 합성프로그램은 경합이 누락될 수 있기 때문에 검증결과의 신뢰성이 낮다.

표 1. 경합 합성프로그램 모음
Table 1. The Synthetic Program Suite for Race

경합유형	코드	의도된 실행	경합 패턴
순서 위배	OV01	$W_1 \rightarrow R_2$	$R_2 \rightarrow W_1$
	OV02	$R_1 \rightarrow W_2$	$W_2 \rightarrow R_1$
	OV03	$W_1 \rightarrow W_2$	$W_2 \rightarrow W_1$
원자성 위배	AV01	$(R_1 \rightarrow R_1) \rightarrow W_2$	$R_1 \rightarrow W_2 \rightarrow R_1$
	AV02	$(W_1 \rightarrow R_1) \rightarrow W_2$	$W_1 \rightarrow W_2 \rightarrow R_1$
	AV03	$(W_1 \rightarrow W_1) \rightarrow R_2$	$W_1 \rightarrow R_2 \rightarrow W_1$
	AV04	$(R_1 \rightarrow W_1) \rightarrow W_2$	$R_1 \rightarrow W_2 \rightarrow W_1$
	AV05	$(W_1 \rightarrow W_1) \rightarrow W_2$	$W_1 \rightarrow W_2 \rightarrow W_1$
양성 위배	BV01	$R_1 \rightarrow R_2$	$R_2 \rightarrow R_1$
	BV02	$(R_1 \rightarrow R_1) \rightarrow R_2$	$R_1 \rightarrow R_2 \rightarrow R_1$
	BV03	$(R_1 \rightarrow R_1) \rightarrow W_2$	$R_1 \rightarrow R_2 \rightarrow W_1$
	BV04	$(W_1 \rightarrow R_1) \rightarrow R_2$	$W_1 \rightarrow R_2 \rightarrow R_1$

Ⅲ. 경합 합성프로그램 모음

본 논문은 경합치유도구의 검증을 돕기 위해 Pasec benchmark, MySQL, Mozila 등과 같은 벤치마크 및 실세계 프로그램에 경합 탐지도구와 버그 리포트를 이용하여 경합패턴[3]을 분석하고 분류하여 경합 합성프로그램 모음을 만들었다. 경합의 종류는 순서위배, 원자성위배, 및 양성위배(benign violations)로 구분하였다.

순서위배는 서로 다른 스레드들이 공유자원에 접근하는 이벤트 순서를 보장할 수 없는 것이다. 따라서 순서위배는 $W_1 \rightarrow R_2$로 의도한 실행이 $R_2 \rightarrow W_1$과 같이 실행되는 것으로 세 가지 종류가 있음을 확인하였다.

원자성위배는 원자적으로 실행되어야 하는 공유자원을 가진 코드영역이 개발자의 잘못된 추정으로 위배가 발생하는 것이다. 따라서 원자성위배는 $(R_1 \rightarrow R_1) \rightarrow W_2$에서 $(R_1 \rightarrow R_1)$이 원자적 실행영역으로 의도하였지만 실제 실행은 $R_1 \rightarrow W_2 \rightarrow R_1$와 같이 원자성이 위배되는 것으로 다섯 종류가 있음을 확인하였다.

양성위배는 순서위배 또는 원자성위배가 발생하더라도 결과에 영향을 미치지 않는 것이다. 분석결과 순서위배 한 종류와 원자성위배 세 종류로 총 네 종류가 있음을 확인하였다.

표1은 완성된 경합 합성프로그램 모음을 보여준다. 합성프로그램의 식별코드는 경합유형 및 종류에 따라 부여하였다. R과 W는 읽기와 쓰기접근을 의미하며 오른쪽 하단 숫자는 쓰레드 번호를 의미한다. 접근 순서는 오른쪽 화살표를 이용하여 표현하였다.

Ⅳ. 실 험

본 논문에서 제안하는 경합 합성프로그램 모음의 실험을 위한 환경으로 ARINC-653을 지원하는 Simulated-IMA 운영체제에 GCC 4.1.2 버전의 컴파일러를 사용하여 구현한 환경에서 실험하였다.

표 2. 실험 결과
Table 2. The Result of Experimentation

경합유형	코드	의도 결과	실행결과			
			#1	#2	#3	#4
순서 위배	OV01	105	105	5	105	105
	OV02	1	0	0	0	1
	OV03	100	100	200	100	200
원자성 위배	AV01	0	0	50	0	0
	AV02	100	100	100	100	150
	AV03	100	100	20	20	20
	AV04	9995	9995	9995	7215	6858
	AV05	150	200	150	200	150
양성 위배	BV01	100	100	100	100	100
	BV02	100	100	100	100	100
	BV03	100	100	100	100	100
	BV04	100	100	100	100	100

실험 결과는 표 2와 같다. 의도결과 열은 개발자가 의도한 결과이며 실행결과는 합성프로그램을 실행하여 얻은 결과이다. 실험 결과를 통해 순서위배 및 원자성위배 합성프로그램들은 실행결과가 의도 실행결과와 일치하지 않는 경우가 발생함을 알 수 있으며, 양성위배는 매 실행마다 의도결과와 일치함을 알 수 있었다. 따라서 경합치유도구의 검증을 위한 합성프로그램 모음이 정상적으로 경합을 발생시킴을 확인하였다.

Ⅴ. 결 론

본 논문에서는 항공기 소프트웨어를 위한 경합치유도구를 검증하기 위한 합성프로그램 모음을 구현하고 실험하였다. 실험 결과를 통해 순서위배 및 원자성위배 합성프로그램에서 비결정적 실행이 발생함을 알 수 있었다. 향후 연구는 동기화 명령어를 고려한 경합패턴을 분류 할 연구할 계획이다.

참 고 문 헌

[1] S. Lu, S. Park, E. Seo, and Y. Zhou, “Learning from mistakes: a comprehensive study on real world concurrency bug characteristics”. Sigplan Notices Vol. 43, No. 3, pp. 329-339, 2008.

[2] G. M. Tchamgoue, O. K. Ha, K. H. Kim, and Y. K. Jun, “A framework for on-the-fly race healing in ARINC-653 applications”. International Journal of Hybrid Information Technology, 2011.

[3] S. Park, R. W. Vuduc, and M. J. Harrold, “Falcon: fault localization in concurrent programs”. In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, Vol 1, pp. 245-254, May 2010.