항공기 소프트웨어에서 자료경합치유를 위한 공간 효율적인 소프트웨어 트랜잭셔널 메모리

(Space Efficient Software Transactional Memory to Heal Data Races in Airborne Software.)

박 주 혁†, 옥 지 훈‡, 최 으 뜸†, 전 용 기†‡*
†경상대학교 정보과학과, ‡경상대학교 항공우주및소프트웨어학과
(Ju-Hyeok Park, Ji-Hoon Ok, Eu-Teum Choi, Yong-Kee Jun)
(†Department of Informatics, Gyeongsang National Univ.,
‡Department of Aerospace and Software Engineering, Gyeongsang National Univ.)

Abstract : Healing data races using software transactional memory record all accesses to shared variables in the transaction region. It is impractical to apply software transactional memory to airborne software with limited resources. In this paper, we propose a technique to reduce the space overhead by the log recorded for total of two in each transaction region.

Keywords : software transactional memory, data races, healing, airborne software

Ⅰ. 서 론

2000년도 초반 등장하는 다중 프로세서 기반의 통합 모듈형 항공전자(IMA)[1]를 위한 항공기 소프트웨어는 멀티스레드 프로그래밍을 통해 실용성과 효율성을 높일 수 있지만 자료경합과 같은 동시성 오류가 발생할 수도 있다. 자료경합은 비결정적인 수행으로 인해 프로그래머가 의도하지 않은 결과를 발생시킨다. 이러한 비결정적인 수행은 소프트웨어를 테스트 하는 동안에 자료경합을 제거하는 것을 어렵게 한다. 따라서 자료경합을 프로그램 수행 중에 용인하고 치유하는 것이 효과적이다.

본 논문은 항공기 소프트웨어에서 발생하는 자료경합을 치유하기 위해 자료경합이 발생할 수 있는 위치를 트랜잭션 영역으로 설정하고 트랜잭션 영역안의 접근을 필터링하여 기존의 기법에서 발생하는 공간오버헤드를 개선한 소프트웨어 트랜잭션 메모리(STM)를 제안한다. 그리고 실험을 통하여 기존의 STM과 필터링을 적용한 STM의 메모리 사용량, 실행시간을 비교하여 효율성을 검증한다.

* 교신저자(Corresponding Author)
박주혁, 옥지훈, 최으뜸: 경상대학교
전용기: 경상대학교 정보과학과, 항공우주및소프트웨어학과, 항공기부품기술연구소, 공학연구원
※ 이 논문은 교육부 지방대학특성화(CK-I)사업의 재원으로 경상대학교 창의적항공IT기계융합인력양성사업의 지원을 받아 수행되었음.

Ⅱ. 연구배경

멀티스레드 프로그램에서 자료경합[2]은 두 개 이상의 스레드에서 하나의 공유변수에 적절한 동기화 없이 최소 하나 이상의 쓰기 접근을 포함하면 발생할 수 있다. 자료경합은 비결정적인 수행으로 인해 프로그래머가 의도하지 않은 결과를 발생시켜 프로그램의 비정상적인 실행을 발생 시킬 수 있다. 이러한 비결정적인 수행은 소프트웨어를 테스트 하는 동안에 자료경합을 제거하는 것은 매우 어렵다. 따라서 자료경합을 프로그램 수행 중에 용인하고 치유하는 것이 효과적이다.

항공기 소프트웨어는 안전성과 신뢰성이 중요하다. 또한 항공기에서 소프트웨어의 비중과 기능에 대한 요구가 증가하면서 멀티스레드 프로그래밍의 필요성이 높아지고 있지만 멀티스레드 프로그래밍을 사용하는 항공기 소프트웨어는 자료경합 때문에 시스템의 신뢰성을 보장하기 어렵다. 따라서 항공기 소프트웨어에 존재하는 경합의 치유를 위한 효율적 도구가 필요하다.

트랜잭셔널 메모리(TM)[2-3]는 병행 프로그램에서 공유메모리의 접근을 제어하기 위한 동시성 제어 방법이다. 이러한 TM은 트랜잭션 영역을 설정하면 충돌을 감지하고 자료경합을 치유 할 수 있다. STM은 TM을 소프트웨어로 구현한 것으로 HTM 보다 구현하기 쉬워 치유 기법에 적용하기 효율적이다. 따라서 본 논문에서는 STM을 이용하여 항공기 소프트웨어에서 자료경합을 치유할 수 있는 기법을 제시한다.

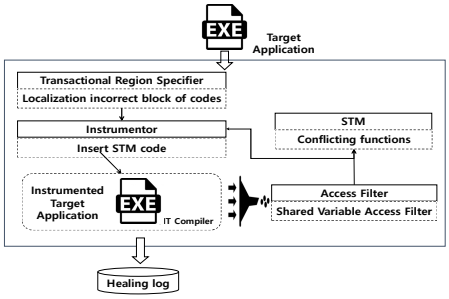

그림 1. 제안하는 아키텍처

Fig. 1. proposed architecture

Ⅲ. 공간 효율적인 STM

본 논문은 STM을 이용하여 공간 효율적인 자료경합 치유를 위해 실행로그 필터링 기법을 제시한다. STM은 동시성 제어 방법으로 기존의 동시성 제어 방법인 lock의 문제점인 우선순위 역전, 데드락과 같은 문제를 개선하여 힐링에 적용하기에 적합하다. 하지만 기존의 연구는 트랜잭션 영역안의 명령을 모두 기록하여 높은 공간 오버헤드가 발생하였다. 따라서 본 논문에서는 공간효율성의 향상을 위해 자료경합의 치유에 필요한 로그만을 기록한다. 그림 1은 공간 효율적인 STM의 전체 아키텍처를 보여주며 Transaction Region Specifier (TRS), STM, Access Filter(AF), 그리고 Instrumentor (INS)와 같이 4개의 모듈로 나타낼 수 있다.

TRS는 트랜잭션 영역의 식별을 위해 사용되는 모듈로 바이너리 코드를 분석하여 적절한 트랜잭션 영역 범위를 식별한다. 트랜잭션 영역 범위는 자료경합이 발생할 수 있는 부분을 반복문, 조건문을 고려하여 식별한 정보를 Instrumentor 와 Access Filter(AF)에 전달한다.

AF는 공유변수로 접근한 사건의 트랜잭션 ID를 식별하고 불필요한 로그 정보는 필터링하여 필요한 로그 정보만을 기록함으로써 공간 오버헤드를 감소시킨다.

STM은 탐지와 복구를 담당 하는 모듈로 트랜잭션 영역 수행 중 자료경합을 탐지하고 자료 경합이 발견되면 힐링하는 모듈이다.

INS는 STM을 삽입하는 모듈로 TRS 로부터 전달받은 트랜잭션 영역정보를 가지고 타켓 프로그램에서 자료경합이 발생할 수 있는 위치에 STM을 삽입하여준다.

표 1. 실험결과

Table 1. performance result table

	메모리사용량	실행시간	치료여부
Norec STM	75(Mb)	0.27(ms)	O
개선된 STM	1(Mb)	0.17(ms)	O

Ⅳ. 실 험

항공기 소프트웨어에서 공간 효율적인 STM의 효율성 실험을 위하여 ARINC-653 운영체제인 Simulated IMA ver. 1.0을 사용하였으며, GenuineIntel CPU, Intel PIN 2.14와 g++4.8.4을 이용하여 구현하였다.

자료경합이 발생하는 합성프로그램을 만들어 STM의 성능을 실험한 결과 표 1과 같다. 실험결과 기존의 Norec-STM의 실행로그 기록방식을 본 논문에서 제시하는 필터링을 적용한 결과 메모리의 효율성은 75배 증가하였고 실행시간은 0.1ms가 감소하였다.

Ⅳ. 결 론

본 논문에서는 STM의 트랜잭션 영역 내의 불필요한 공유변수 접근을 필터링하여 공간 효율적으로 자료경합을 치유 할 수 있는 기법을 제시 하였다. 공간 효율적인 STM의 효율성 검증을 위하여 합성프로그램에 적용하여 실험하였다. 그 결과 공간 오버헤드가 기존의 STM보다 감소한 것을 확인 하였다. 향후 연구는 실제 운용중인 항공기 소프트웨어에 적용하여 실험을 진행할 계획이다.

참 고 문 헌

[1] S. H. VanderLeest,, "ARINC 653 hypervisor", 29th Digital Avionics Systems Conference, pp. 5.E.2-1-5.E.2-2, 2010

[2] 최으뜸, 하옥균, 전용기, "소프트웨어 트랜잭셔널 메모리를 이용한 자료경합 치유 기술 설계", 한국컴퓨터정보학회 학술발표논문집, 제 24권, 제 2호, 3-4쪽, 2016.

[3] C. Calin, B. Colin, M. Maged, C. Harold W, W.Peng, C. Stefanie, C. Siddhartha, " Software Transactional Memory: Why Is It Only a Research Toy?", The Concurrency Problem Queue Homepage table of contents archive, Vol, 6, pp.4, 2008