가치기반 접근법을 통한 테스트 케이스 우선순위기법

(A Test Case Prioritization Technique via Value-Based Approach)

박 현 철 [†] 류 호 연 [††] 백 종 문 [†††]

(Hyuncheol Park) (Hoyeon Ryu) (Jongmoon Baik)

요 약 한번 개발된 소프트웨어는 긴 수명을 가지며, 결함이나 사용자 요구사항의 변경, 환경의 변화 등의 원인으로 인한 수많은 모듈의 수정을 통해 진화한다. 소프트웨어의 진화와 더불어, 이러한 많은 모듈의 추가와 수정에 의한 소프트웨어의 다양한 버전의 생성은 소프트웨어 품질 향상을 가로막는 주요 요인 중의 하나이다. 한편 회귀테스트는 이러한 소프트웨어의 변경에 따른 초점을 둔 테스트이지만, 소프트웨어가 여러 버전을 거쳐 진화함에 따라 많은 수의 테스트 케이스의 추가 및 수정과 더불어 테스트 케이스의 전체 숫자가 급격히 증가함에 따라 비용이 많이 들기에 쉽게 수행하지 못하는 테스트이다. 이에 테스트 케이스 우선순위화 기법이 등장하여 회귀테스트의 수행을 지원하였다. 그러나, 테스트 케이스 우선순위화 기법에서 사용되는 우선순위의 기준이 가치중립적이거나 가치를 고려하더라도 특정한 단일 요인만을 고려하여 그 활용이 제한적이었다. 논문에서, 우리는 비용과 결함 심각도에 기반한 히스토리컬 가치 기반의 접근법을 제시하며, 이는 기존의 비용 인식 테스트 케이스 우선순위화 기법에서의 현재의 비용 및 결함 심각도를 예측하기 위하여 히스토리컬 정보를 사용하는 접근법이다. 본 논문의 공헌으로서, 제안된 접근법은 테스트의 이해관계자들이 어떻게 히스토리컬 가치가 가치의 관점에서 테스트 효과성의 향상을 위해 사용되고 있는가를 알 수 있도록 돕는다는데 있다. 결과적으로, 회귀 테스트를 수행하는 소프트웨어 테스터들은 그들의 테스트 케이스를 보다 더 효과적으로 우선순위화할 수 있기에 그들의 테스트를 통한 테스트 효과성은 향상될 수 있다.

키워드 : 테스트 케이스 우선순위화, 회귀테스트, 소프트웨어 테스팅

Abstract Software, once developed, has a long life and evolves through numerous additions and modifications because of the faults, the changes in user requirements, the changes in environments, and so forth. With the evolution of the software, assuring the quality of the software is getting more difficult because of numerous versions of the software. Meanwhile, regression testing has been used to support the software testing activities and assure acquiring appropriate quality through several versions of software. Regression testing, however, is too expensive because it requires lots of test cases executions and the number of test cases increases sharply as the software evolves. For this reason, several techniques are suggested to help conducting regression testing then test case prioritization technique is understood the most effective and efficient technique to support regression testing. In this paper, we propose an approach, Historical Value-Based Approach, which is based on the use of historical information to estimate the current cost and fault severity for cost-cognizant test

본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2009-(C1090-0902-0032)

이 논문은 2008 한국 소프트웨어공학 학술대회에서 '가치기반 접근법을 통한 효과적인 테스트 케이스 우선순위기법'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 삼성탈레스 기술연구소 선임연구원
hyuncheoll.park@samsung.com

†† 정 회 원 : (주)포스데이타 SW테스트팀 차장
hoyeon@posdata.co.kr

††† 종신회원 : 한국과학기술원 정보과학기술대학 교수
jbaik@kaist.ac.kr

논문접수 : 2008년 4월 14일
심사완료 : 2009년 3월 3일

정보과학회논문지: 소프트웨어 및 응용 제36권 제5호(2009.5)

case prioritization. As a result of the proposed approach, software testers who perform regression testing prioritize their test cases more effectively so that the test effectiveness of them can be improved in terms of APFDc.

Key words : Test Case Prioritization, Regression Testing, Software Testing

1. 서 론

한번 개발된 소프트웨어는 긴 수명을 가지며, 결함이나 사용자 요구사항의 변경, 환경의 변화 등의 원인으로 인한 수많은 모듈의 추가나 수정을 통해 진화한다. 소프트웨어의 진화와 더불어, 이러한 많은 모듈의 추가와 수정에 의한 소프트웨어의 다양한 버전의 생성은 소프트웨어 품질 향상을 가로막는 주요 요인 중의 하나였다. 회귀테스트는 이러한 소프트웨어의 다양한 버전과 모듈의 추가, 수정사항에 대한 고려를 바탕으로 새롭게 추가되거나 수정된 모듈이 기존의 모듈에 영향을 끼치는가를 테스트하여 소프트웨어 개발이나 유지보수 중의 어려움을 해소하기 위하여 등장하였다[1,2].

그러나, 소프트웨어가 여러 버전을 거쳐 진화함에 따라 많은 수의 테스트 케이스의 추가, 수정과 더불어 테스트 케이스의 전체 숫자가 급격히 증가함에 따라, 비용이 많이 들게 되기에 회귀테스트는 쉽게 수행하지 못하는 테스트이다[3]. 이러한 이유로 여러 기법들이 등장하여 회귀테스트의 수행을 돕게 되었고, 그 중에서 테스트 케이스 우선순위화 기법(Test Case Prioritization)이 가장 효과적이고 효율적인 기법으로 받아들여지고 있다[2]. 테스트 케이스 우선순위화 기법은 정해진 기준에 의해 테스트 케이스의 수행순서를 조절하는 우선순위화를 통해, 회귀테스트 수행에 있어서의 효율성 증대를 추구하는 기법이다.

그러나 소프트웨어 테스트의 특성을 반영한 우선순위의 기준이 너무나 다양함에 따라 다양한 테스트 케이스 우선순위화 기법이 등장하게 되었으나, 이러한 기법들은 기본적으로 하나의 큰 단점을 지니고 있었다. 즉, 이들 기법들은 기본적으로 가치 중립적인 생각(Value-neutral)에 기반하고 있기에 모든 테스트 케이스나 결함 등과 같은 요소들을 모두 동일하게 가치를 매기고 취급하고 있는 것이다. 이로 인해 비용인식에 근거한 테스트 케이스 우선순위화 기법(Cost-cognizant Test Case Prioritization)이 등장하였다[4]. 그러나 이 기법에서는 비용의 요소로 본 테스트 수행 비용(Test Cost)과 결함 심각도(Fault Severity)의 정량화를 위한 방안에 대한 방법의 제시가 없거나 미흡하였다.

이에 본 논문에서는 이전의 테스트 수행으로부터의 비용과 결함 심각도에 근거한 히스토리컬 가치 기반의 접근법을 제시하였다. 본 논문의 공헌으로서, 제안된 접근법은 테스트의 이해관계자들이 어떻게 히스토리컬 가치가 가치의 관점에서 테스트 효과성의 향상을 위해 사용되고 있는가를 알 수 있도록 돕는다는데 있다. 결과적으로, 회귀 테스트를 수행하는 소프트웨어 테스터들은 그들의 테스트 케이스를 보다 더 효과적으로 우선순위화할 수 있기에 그들의 테스트를 통한 테스트 효과성은 향상될 수 있다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 본 논문에서 제시하는 기법에 대한 제반 배경 및 관련연구에 대하여 설명하고, 3장에서는 제시한 기법에 대한 세부적인 사항을 소개한다. 마지막으로 4장에서 결론을 도출한다.

2. 배경 및 관련연구

본 절에서는 히스토리컬 가치 기반 접근법에 관련된 배경지식 및 관련연구로서 회귀테스트, 비용인식에 근거한 테스트 케이스 우선순위화 기법과 히스토리 기반의 테스트 우선순위화 기법, 테스트 우선순위화 기법의 효과성을 측정하기 위한 메트릭에 대하여 간략히 소개한다.

2.1 회귀테스트

회귀테스트는 소프트웨어 테스트의 한 종류로서, 소프트웨어 시스템의 다양한 버전에서의 선택적인 재 테스트(retesting)에 초점을 둔 테스트이다[2]. 회귀테스트에 대하여, IEEE에서는 다음과 같이 정의하고 있다.

"*Selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements*" [5]

즉, 회귀테스트는 소프트웨어 모듈의 수정이나 추가가 기존의 소프트웨어에 영향을 미쳐 결함이나 오류를 초래하였는지를 재검증하기 위한 목적으로 수행되는 소프트웨어 테스트이다. 다음의 그림은 소프트웨어 개발 및 유지보수의 수명주기에서의 회귀테스트를 보여주고 있다.

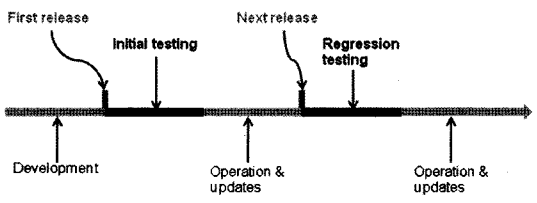

그림 1 소프트웨어 개발 및 유지보수 중의 회귀테스트

2.2 비용인식 기반의 테스트 케이스 우선순위화 기법

테스트 케이스 우선순위화 기법이 등장한 이래, 소개된 각 테스트 케이스 우선순위화 기법은 약점을 갖고 있었다. 즉, 테스트 수행 비용과 결함의 심각도에 대한 고려가 결여되었다는 점이다. 그러나, 실무에서는 각 테스트 케이스의 수행 비용과 각 테스트 케이스의 수행을 통해 발견된 결함이 동일하지 않기에 기존의 테스트 케이스 우선순위화 기법이 부적절한 테스트 케이스 순서를 생성하곤 하였다[3].

비용인식 기반의 테스트 케이스 우선순위화 기법은 테스트 수행 비용과 결함 심각도에 대한 고려를 추가한 기법으로서, 테스트 수행 비용과 결함 심각도를 테스트 케이스의 우선순위화를 위한 중요 요인으로 인식하고 이를 기존의 테스트 케이스 우선순위화 알고리즘에 추가하였다[3,4].

테스트 케이스의 수행과 검증에는 다양한 리소스가 필요하며 이러한 리소스로는 머신 타임, 휴먼 타임, 테스트 케이스의 수행과 검증에 관련된 비용을 화폐가치로 환산한 것, 하드웨어 비용, 임금, 릴리즈 일정의 지연으로 인한 손실 등 다양한 것들이 있다[6-8].

유사하게, 결함 심각도를 반영하는 비용이나 리소스 역시 마찬가지로 다양하다. 이러한 비용이나 리소스의 예로는 결함의 디버깅 비용에 대한 영향, 결함으로 인해 초래된 금액의 총액으로 나타낸 것 등과 같이 다양한 것들이 있다[2,3,9].

2.3 히스토리 기반의 테스트 우선순위화 기법

히스토리 기반의 테스트 우선순위화 기법은 히스토리컬 테스트 수행 데이터와 회귀테스트 선택 기법에 기반한 기법으로서, 다음의 두 가지 요인들을 그 핵심으로 하고 있다[10].

- 테스트 케이스 우선순위화를 위한 프로시져
- 테스트 케이스 선택을 위한 확률의 판단 및 설정

또한 테스트 케이스 선택을 위한 확률은 다음의 수식을 통하여 계산된다[10]. H_{tc}는 테스트 케이스 tc의 이전 수행으로부터 도출된 시간 t의 시간 순서의 테스트 케이스 실행에 관련된 히스토리의 집합, 즉 $\{h_1, h_2, h_3, \cdots h_t\}$ 이고, α가 개별 히스토리에 대한 가중 요인(Weighting factor)으로서 사용되는 상수이면, 시간 t에서의 각 테스트 케이스 tc의 선택 확률은 $P_{tc,t}(H_{tc}, \alpha)$로 표현된다. 아래의 수식은 이의 일반적인 형태이다.

$P_0 = h_1$

$P_k = \alpha h_k + (1 - \alpha)P_{k-1}$ $(0 =< \alpha =< 1, k => 1)$

α가 크다면 이는 최근의 테스트 케이스 수행에서의 값에 더 높은 가중요인을 주는 것이고, 반대로 이 값이 작다면 이는 이전의 테스트 케이스 수행에서의 값에 더 높은 가중요인을 주는 것이다. 즉, α의 값을 변경하는 것에 의해, 최근의 테스트 케이스 선택을 위한 확률과 그 이전의 테스트 케이스 선택을 위한 확률은 현재의 테스트 케이스 선택을 위한 확률에 영향을 주게 된다. 또한 이를 통해 테스트 케이스의 수행에 관한 히스토리컬한 정보는 현재의 테스팅 세션을 위한 선택 확률의 증가나 감소에 사용되게 된다.

2.4 테스트 케이스 우선순위화 기법의 효과성을 측정하기 위한 메트릭

테스트 케이스 우선순위화 기법의 효과성을 테스트 스위트의 테스트 케이스의 우선순위화된 순서와 이 순서에 의해 발견되는 결함의 비율로 나타내어 측정하는 발견된 결함의 평균 백분율(APFD: Average Percentage of Faults Detected)이란 메트릭이 있다[1,11]. 이 메트릭은 테스트 스위트에서의 결함 발견 비율의 증가에 초점을 맞춘 것으로서, 테스트 프로세스 중에 얼마나 빨리 결함이 발견되는지를 주어진 테스트 케이스의 수행 순서에 의한 누적적인 결함 발견 백분율의 평균으로 나타낸 값이다. 다음의 공식은 APFD를 계산하기 위한 공식이다[1,11].

$$APFD = 1 - \frac{\sum_{i=1}^{m} TF_i}{n \times m} + \frac{1}{2n}$$

T는 n개의 테스트 케이스로 이루어진 테스트 스위트이고, F는 m개의 결함을 포함한 결함의 집합이다. TF_i는 T에서의 테스트 케이스의 순서 중 결함 i를 노출시키는 첫 번째 테스트 케이스를 의미한다.

또한 APFD는 테스트 케이스 순서의 효과성과 효율성을 정량화하여 한 개의 테스트 스위트의 결함 발견 비율을 수행된 테스트 스위트의 백분율(Percentage of Test Suite Executed)과 발견된 결함의 백분율(Percentage of Fault Detected)의 두 가지 기준으로 측정하는 메트릭이다. 다음의 그림은 APFD의 예를 보여주고 있다.

기본적으로 APFD는 모든 테스트 케이스가 동일한 테스트 수행 비용을 갖고 모든 결함은 동일한 심각도를 가진다는 가정에 기반하고 있다. 그러나 실제로는 각 테

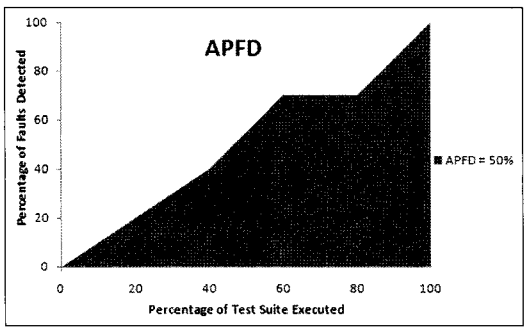

그림 2 APFD의 예

스트 케이스의 테스트 수행 비용과 각 결함의 결함 심각도는 같지 않고 서로 다르다. 그러므로 이러한 가정이 충족되지 못할 경우APFD는 부적절한 값을 표현하며, 이러한 이유로 APFD에 비용이란 개념을 추가한 비용당 발견된 결함의 평균 백분율(APFDc: Average Percentage of Faults Detected per Cost)이란 메트릭이 새로이 등장하였다[3,4]. 다음은 APFDc를 계산하기 위한 공식이다[3,4].

$$APFDc = \frac{\sum_{i=1}^{m}(f_i \times (\sum_{j=TF_i}^{n} t_j - \frac{1}{2}t_{TF_i}))}{\sum_{i=1}^{n} t_i \times \sum_{i=1}^{m} f_i}$$

APFDc는 테스트 수행 비용과 결함 심각도 측면에서의 테스트 케이스의 순서의 효과성을 정량화하여 측정하는 메트릭으로써, 소요된 전체 테스트 케이스의 비용의 백분율(Percentage of Total Test Case Incurred)과 발견된 결함들의 전체 결함 심각도의 백분율(Percentage of Total Fault Severities Detected)의 두 가지 기준에 의해 테스트 스위트의 결함 발견 비율을 측정한다. 다음의 그림은 APFDc의 예를 보여준다.

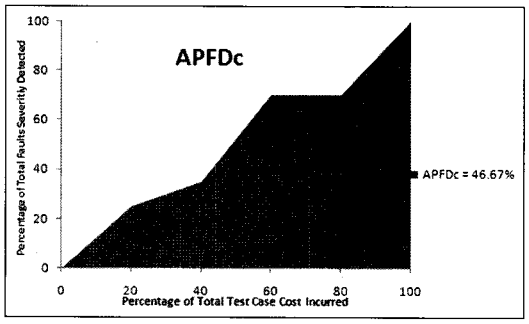

그림 3 APFDc의 예

정리하자면, APFD는 테스트 수행 비용과 결함 심각도가 가치중립적인(Value-neutral)할 때의 테스트 케이스 우선순위화의 효과성에 대한 메트릭이다. 이에 반해 APFDc는 테스트 케이스 우선순위화의 효과를 가치, 즉 테스트 수행 비용과 결함 심각도에 의거하여 보여주는 메트릭이다

3. 히스토리컬 가치 기반의 접근법

본 절에서는 비용인식 테스트 케이스 우선순위화를 위한 히스토리컬 가치 기반의 접근법을 설명하기 위하여 제안된 접근법의 개요 및 상세한 사항을 제공한다.

3.1 개요

제안된 히스토리컬 가치기반의 접근법은 주어진 테스트 케이스의 우선순위를 판단하기 위하여 히스토리컬 정보를 사용한다는 것에 초점을 맞추고 있다. 테스트 케이스의 수행 비용과 발견된 결함의 결함 심각도에 대한 히스토리컬 정보를 사용함으로써, 주어진 테스트 케이스의 히스토리컬 가치가 계산될 수 있으며, 이는 차후 테스트 케이스의 우선순위화를 위한 근거로서 사용된다. 추가적으로, 히스토리컬 가치는 비용인식 기반의 테스트 케이스 우선순위화 기법 이외의 기 존재하는 테스트 케이스 우선순위화 기법과도 같이 사용될 수 있으며 이는 기존의 테스트 케이스 우선순위화 기법을 보충하여 가치기반의 테스트 케이스 우선순위화 기법으로 사용되도록 할 수 있을 것이다.

히스토리컬 가치는 해당 테스트 케이스의 이전의 테스트 케이스 수행 비용과 이전에 해당 테스트 케이스에 의해 발견된 결함들의 심각도에 대한 정보를 통해서 계산되며, 이는 주어진 테스트 스위트에서의 테스트 케이스들을 우선순위화하는 요인으로서 사용된다. 다음의 그림은 제안된 접근법의 전체적인 개요를 보여주고 있다.

다음의 사항들은 위의 그림 4에 대한 설명이다.

- P는 소프트웨어 시스템이고 P' 은 P의 수정된 버전이다.
- P'에 대한 회귀테스트를 수행하기 위하여 테스트 케이스 저장소로부터 테스트 케이스들이 추출되어 하나의 테스트 케이스 스위트를 구성한다.
- 테스트 케이스의 수행을 통해 알게 된 테스트 케이스의 수행 비용과 발견된 결함의 결함 심각도는 히스토리컬 정보 저장소에 저장된다.
- 테스트 케이스의 우선순위화가 필요할 때, 히스토리컬 가치 모델은 저장된 히스토리컬 정보, 즉 각 테스트 케이스의 이전 수행에서의 테스트 케이스 수행 비용과 이 때 발견된 결함의 결함 심각도에 대한 정보를 이용하여 히스토리컬 가치를 산출한다.
- 계산된 히스토리컬 가치는 테스트 케이스 우선순위화 기법에서 테스트 케이스의 우선순위를 정하기 위한 기준으로 사용된다.

히스토리컬 가치기반의 접근법에 대한 보다 자세한 설명은 다음의 세부 절에서 계속된다.

3.2 히스토리컬 가치기반의 접근법의 상세

비용인식 기반의 테스트 케이스 우선순위화 기법을 위한 히스토리컬 가치기반의 접근법은 히스토리컬 가치 관점에서의 테스트 케이스 우선순위의 결정을 그 키포인트로 하고 있으며, 이는 이전의 테스트 케이스의 수행 비용과 테스트 케이스에 의해 발견된 결함의 결함 심각도가 테스트 케이스 우선순위화의 기준 및 근거로 사용된다는 뜻이다. 그러나 히스토리컬 가치기반의 접근법은 한가지 가정, 즉 소프트웨어의 한 릴리즈로부터 다음 릴리즈까지의 변경이 크지 않다는 것에 근거하고 있다. 그

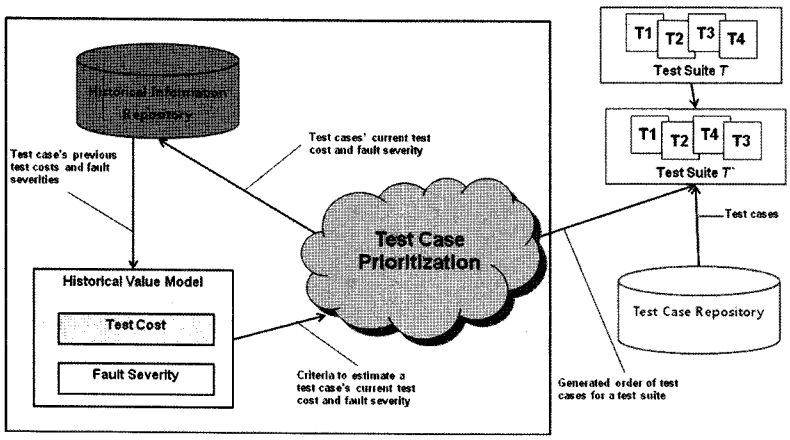

그림 4 히스토리컬 가치기반 접근법의 개요

러나 일반적으로 소프트웨어가 개발된 초기에는 소프트웨어 릴리즈 간의 변경이 클 수 있으나, 소프트웨어가 안정화된 이후의 유지보수 중에는 큰 변경이 없기에 이러한 가정이 유효하므로 큰 문제가 되지는 않는다[10].

3.2.1 테스트 케이스의 수행 비용과 발견된 결함의 결함 심각도

소프트웨어 테스트에서의 테스트 수행 비용은 상당히 다양한 요인으로 이루어지며 기준에 따라 머신 타임, 휴먼 타임, 테스트 케이스 수행 시간, 테스트 수행에 대한 화폐가치 기준의 비용과 같이 정의될 수 있다. 이와 유사하게 결함 심각도 역시 그 기준에 따라 테스트 중요도-결함을 탐지하는 테스트 케이스 자체의 중요도-, 함수 심각도- 테스트 케이스에 의해 커버되는 코드에서의 함수의 중요도-와 같이 다양하게 정의될 수 있다.

제안된 접근법에서는 테스트 수행 비용을 테스트 케이스의 수행에 따른 시간으로 한정하였다. 이는 이것이 테스트 수행 비용에 대해 폭넓게 받아들여지는 것이기 때문이며, 비용인식 기반의 테스트 케이스 우선순위 기법 등의 이전 연구에서도 마찬가지로 이를 테스트 수행 비용으로 보고 있다[3,4]. 또한 결함 심각도는 소프트웨어 테스터에 의해 각 테스트 케이스에 할당된 테스트 케이스 중요도로 한정하였다.

3.2.2 히스토리컬 가치기반의 접근법에서 사용되는 정의들

우리의 접근법에서는 여러 정의가 사용되며, 본 단락에서는 이러한 정의들을 소개한다.

- P는 소프트웨어 시스템이다.
- P'는 소프트웨어 시스템 P의 다음 버전이다.
- n은 하나의 테스트 스위트를 구성하는 테스트 케이스의 개수이다.
- T는 P에 대한 테스트를 위해 t_1부터 t_n까지의 테스트 케이스로 구성된 하나의 테스트 스위트이다.
- T'는 P'에 대한 테스트를 위해 t_1부터 t_n까지의 테스트 케이스로 구성된 하나의 테스트 스위트이다.
- t_i는 테스트 스위트 T에 포함된 테스트 케이스 중 해당 테스트 케이스의 순서에서i번째의 테스트 케이스이다.
- m은 테스트 케이스 T에 의해 발견되는 결함의 개수이다.
- $HV_{(t,\ i)}$는 테스트 케이스 t의 i번째 수행을 위해 i-1번째 수행으로부터 참조한 히스토리컬 가치에 대한 값이다. ($i > 1$)
- $C_{(t,\ i)}$는 테스트 케이스 t의 i번째 수행에서의 테스트 케이스 수행 비용을 해당 테스트 스위트에서의 테스트 케이스 중 최대의 테스트 케이스 수행 비용과 비교한 상대적인 값이다. (cf. C_0 = 테스트 스위트에서의 전체 테스트 케이스의 수행 비용의 평균값)
- $FS_{(t,\ i)}$는 테스트 케이스 t의 i번째 수행에서의 전체 결함 심각도를 해당 테스트 스위트에서의 테스트 케이스 중 최대의 전체 결함 심각도의 값과 비교한 상대적인 값이다. (cf. FS_0 = 테스트 스위트에서의 모든 테스트 케이스들의 전체 결함 심각도의 평균값)
- wC_i는 해당 테스트 케이스의 i번째 수행에서의 테스트 수행 비용Ci에 대한 가중치이다.
- wFS_i는 해당 테스트 케이스의 i번째 수행에서의 결함 심각도 FS_i에 대한 가중치이다.
- $min(X_1,\ X_n)$은 X_1으로부터 X_n의 값 중 최소값을 구하기 위한 함수이다.

3.2.3 히스토리컬 가치 모델

히스토리컬 가치 모델은 하나의 테스트 케이스의 히스토리컬 가치의 값 $HV_{(t,\ i)}$를 해당 테스트 케이스의 이전의 테스트 수행 비용 $C_{(t,\ i-1)}$과 해당 테스트 케이스를 통해 발견된 모든 결함들의 심각도인 $FS_{(t,\ i-1)}$의 관점에

서 정량화하기 위한 모델이며, 이전의 테스트 수행 비용 $C_{(t,\,i-1)}$과 이전의 결함 심각도 $FS_{(t,\,i-1)}$의 정량화를 위해 이들 중의 최대값에 대해 비교한 상대적인 값이 사용된다. 다음의 표는 상대적인 값에 대한 이해를 돕기 위한 예제이다. 소프트웨어 시스템 P가 있으며 이 소프트웨어를 테스트 하기 위한 5개의 테스트 케이스가 있다고 가정하자. 이들 각각은 다음의 표에서와 같이 각각의 테스트 케이스 수행 비용을 갖는다.

표 1 상대적인 테스트 케이스 수행 비용을 설명하기 위한 예제

Test cases	Cost	Relative Cost
A	2	50
B	1	25
C	4	100
D	2	50
E	1	25

예를 들어 다섯 개의 테스트 케이스 A, B, C, D, E에 대하여, 이들 각각은 1부터 4까지의 비용을 갖는다. 이 중 테스트 케이스 C의 비용 4가 이들 테스트 케이스의 비용 중 최대값이다. 이 최대값 4를 100으로 놓아 테스트 케이스 C는 100을 상대적인 비용으로 가지며, 다른 테스트 케이스들은 이에 대한 상대적인 비율에 의한 값을 상대적인 비용의 값으로 갖는다. 즉, 테스트 케이스 A는 비용이 2이고 이는 C의 비용 4의 절반이기에 테스트 케이스 A의 상대적인 비용은 C의 상대적인 비용 값의 절반인 50이 된다.

결함 심각도의 상대적인 값은 테스트 수행 비용에서와 다르게 산정된다. 다음의 표는 결함과 이에 대한 심각도의 예를 보여주기 위한 것이다.

표 2 상대적인 결함 심각도를 설명하기 위한 예제

Faults	Fault Severity
1	4
2	2
3	5
4	1
5	2
6	2
7	4
8	2
9	2
10	1

예를 들어, 해당 소프트웨어 시스템 P는 10개의 결함을 갖고 있으며, 이에 대해 1부터 10까지 번호를 붙여놓았다. 이들 각각은 1부터 5까지의 결함 심각도를 가진다. 만약 테스트 케이스 A의 수행을 통해 결함 1과 5를 발견할 수 있다면, 테스트 케이스 A는 전체적으로 6의 결함 심각도, 즉 6의 전체 결함 심각도를 탐지할 능력을 가진 것이다. 각각의 테스트 케이스가 갖는 전체 결함 심각도가 다음의 표와 같다고 가정하며, 이들의 상대적인 값이 이들 전체 결함 심각도 값의 최대값에 대한 상대적인 값으로 표현되게 된다.

표 3 상대적인 전체 결함 심각도를 설명하기 위한 예제

Test cases	Total Fault Severity	Relative Total Fault Severity
A	6	60
B	4	40
C	10	100
D	3	30
E	2	20

즉, 테스트 케이스 A는 60의 상대적인 전체 결함 심각도를 가지며, 이는 최대 전체 결함 심각도를 갖는 테스트 케이스 C의 전체 결함 심각도 100에 대한 상대적인 값이다. 이와 유사하게 다른 테스트 케이스들은 최대 전체 결함 심각도 값에 대한 비율로서 상대적인 전체 결함 심각도 값을 갖게 된다.

i번째 테스트 케이스 수행에서의 테스트 케이스 수행 비용에 대한 가중치 wC_i는 다음과 같이 정의된다.

$$wC_i = \frac{\overline{C_i}}{\overline{C_i} + \overline{FS_i}}$$

이와 유사하게, i번째 테스트 케이스 수행에서의 결함 심각도에 대한 가중치 wFS_i는 다음과 같이 정의된다.

$$wFS_i = \frac{\overline{FS_i}}{\overline{C_i} + \overline{FS_i}}$$

$\overline{C_i}$는 테스트 수행 비용 C의 i번째 수행에서의 평균값이며, $\overline{FS_i}$는 전체 결함 심각도 FS의 평균값이다. 이러한 가중치는 두 개의 다른 종류의 값들에 대한 밸런싱을 맞춰주기 위하여 사용되며, 이러한 종류의 가중치의 사용이 요구사항 기반의 테스트 케이스 우선순위화 기법에 대한 연구에서 이미 소개되었다[12]. 기본적으로 히스토리컬 가치가 C와 FS의 합이기에, 이러한 가중치 wC_i와 wFS_i가 각각의 히스토리컬 가치에서의 값의 비율로서 사용된다.

테스트 케이스 t의 i번째 수행에서의 히스토리컬 가치 $HV(t, i)$는 다음의 공식으로 표현된다.

$$HV_{(t,\,i)} = [(100 + \min(C_{(t,\,1)}, C_{(t,\,n)}) - C_{(t,\,i-1)}] \times wC_{i-1} + FS_{(t,\,i-1)} \times wFS_{i-1}$$

$HV_{(t,\,i)}$는 크게 두 개의 부분으로 구성된다. 즉, wC_i에 의해 조정된 상대적인 테스트 케이스 수행 비용에 대한 부분과 wFS_i에 의해 조정된 테스트 케이스 t에 의

해 발견되는 결함에 대한 상대적인 전체 결함 심각도에 대한 부분이다. 테스트 케이스 수행 비용에 대한 실제 값이 상대적인 부분에 대해 역으로 표현되기에, 테스트 케이스 수행 비용의 최대값 100과 테스트 스위트에서의 테스트 케이스들의 수행 비용 중 최소값을 더한 값에서 해당 테스트 케이스의 수행 비용을 뺀 값을 사용한다. 다음의 그림은 히스토리컬 가치를 계산하여 테스트 케이스의 우선순위를 도출하기 위한 알고리즘을 보여주고 있다.

Input: Test suit T with n test cases (from t_1 to t_n), the current time of testing i, cost of *test case* t at i-1th time $C_{(t,(i-1))}$, fault severity of *test case* t at i-1 time $FS_{(t,(i-1))}$
Output: *Test case* t's historical value $HV_{(t,i)}$ at ith test execution

```
1: begin
2:   set T empty
3:   calculate wC_i
4:   calculate wFS_i
5:   for each test case t ∈ T do
6:     get C_(t,(i-1)) and FS_(t,(i-1))
7:     calculate an historical value HV_(t,i) of t
8:   end for
9: end
```

그림 5 히스토리컬 가치, $HV_{(t,i)}$를 계산하기 위한 알고리즘

해당 알고리즘에서 t_1부터 t_n까지의 n개의 테스트 케이스로 구성된 테스트 스위트 T에 대해, 현재의 테스팅 시점을 i라 가정하고 테스트 케이스 t의 i-1 시점에서의 비용과 결함 심각도를 $C_{(t,(i-1))}$, $FS_{(t,(i-1))}$라고 정의하고 이를 알고리즘에 대한 입력으로 한다면, 이에 대한 출력은 테스트 케이스 t의 i번째 테스트 수행에서의 히스토리컬 가치 $HV_{(t,i)}$로서 산출된다.

비용에 대한 가중치 wC_i 와 결함 심각도에 대한 가중치 wFS_i를 우선 계산하고, 테스트 스위트 T에 속하는 각 테스트 케이스 t에 대한 비용 $C_{(t,(i-1))}$와 결함 심각도 $FS_{(t,(i-1))}$를 구하고 이를 바탕으로 각 테스트 케이스 t에 i시점의 히스토리컬 가치 $HV_{(t,i)}$를 구한다. 구해진 테스트 케이스들의 히스토리컬 가치 HV 값을 테스트 케이스 우선순위화의 기준으로 삼아 HV값이 높은 테스트 케이스부터 테스트 케이스 수행을 진행해 나가며 회귀테스트를 진행한다.

4. 결론 및 향후 연구

본 논문에서 우리는 비용인식 기반의 테스트 케이스 우선순위화 기법을 위한 히스토리컬 가치기반의 접근법을 제안하였고, 이는 히스토리컬 정보의 이용을 통한 테스트 케이스 수행 비용과 결함 심각도의 경향을 예측하기 위한 모델과 알고리즘을 포함한다.

이러한 접근법은 다음과 같은 측면에서 회귀테스트를 위한 테스트 케이스 우선순위화 기법에 공헌하고 있다.

- 히스토리컬 정보에 기반한 현재의 테스트 케이스 수행 비용과 결함 심각도에 대한 추정법의 제공
- 제안된 접근법이 다른 테스트 케이스 우선순위화 기법에 결합하여 사용할 수 있기에, 제안된 접근법은 기존의 다른 테스트 케이스 우선순위화 기법을 보충하기 위하여 사용될 수 있다.

위와 같은 공헌을 통해 본 논문에서 제안하는 히스토리컬 가치기반의 접근법은 회귀테스트 수행시의 효율성 향상에 기여할 수 있을 것이다. 이러한 주장을 검증하기 위한 방안으로서, 제안된 접근법을 검증하기 위한 실험을 고려하고 있다. 테스트 케이스 우선순위화 기법에 대한 실험 환경에 대한 구성 및 실험의 수행, 실험 결과의 분석에 대한 연구가 이미 해외의 여러 연구자에 의해 수행되었다[13-18]. 이러한 실험 환경의 구성 및 실험 수행, 실험 결과의 분석은 본 논문에서 제안된 히스토리컬 가치기반 접근법의 회귀테스트 수행에서의 효과성과 유용성을 증명할 수 있을 것이다.

참 고 문 헌

[1] Gregg Rothermel, R. Untch, C. Chu, and Mary J. Harrold, "Prioritizing Test Cases for Regression Testing," IEEE Transactions on Software Engineering, Vol.27, No.10, pp. 929-948, October 2001.

[2] Gregg Rothermel, Sebastian Elbaum, A.G. Malishevsky, P. Kallakuri, and X. Qiu, "On Test Suite Composition and Cost-Effective Regression Testing," ACM Transactions on Software Engineering and Methodology, Vol.13, No.3, pp. 227-331, July 2004.

[3] Jung-Min Kim and Adam Porter, "A History-Based Test Prioritization Technique for Regression Testing in Resource Constrained Environments," Proceedings of the International Conference on Software Engineering (ICSE'02), May 2002.

[4] Alexey G. Malishevsky, Joseph R. Ruthruff, Gregg Rothermel, and Sebastian Elbaum, "Cost-cognizant Test Case Prioritization," Technical Report TR-UNL-CSE-2006-0004, University of Nebraska-Lincoln, March 2006.

[5] Institute of Electrical and Electronics Engineers (IEEE). IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.

[6] Jung-Min Kim, Adam Porter, and Gregg Rothermel, "An Empirical Study of Regression Test Application Frequency," Proceedings of the International Conference on Software Engineering (ICSE'00), June 2000.

[7] A. Srivastava and J. Thiagarajan, "Effectively Prioritizing Tests in Development Environment," Proceedings of the International Symposium on Software Testing and Analysis, July 2002.
[8] W.E. Wong, J.R. Horgan, S. London, and H. Agrawal, "A Study of Effective Regression Testing in Practice," Proceedings of the International Symposium on Software Reliability Engineering, November 1997.
[9] Gregg Rothermel and M.J. Harrold, "Analyzing Regression Test Selection Techniques," IEEE Transactions on Software Engineering, Vol.22, No.8, pp. 529-551, August 1996.
[10] Sebastian Elbaum, Alexey G. Malishevsky, and Gregg Rothermel, "Test Case Prioritization: A Family of Empirical Studies," IEEE Transactions on Software Engineering, Vol.28, No.2, pp. 159-182, Feburary 2002.
[11] Sebastian Elbaum, Alexey Malishevsky, and Gregg Rothermel, "Prioritizing Test Cases for Regression Testing," Proceedings of the International Symposium on Software Testing and Analysis, August 2000.
[12] Hema Srikanth and Laurie Williams, "On the Economics of Requirements-Based Test Case Prioritization," Proceedings of the International Workshop on Economics-Driven Software Engineering Research (EDSER'05), July 2005.
[13] Hyunsook Do and Gregg Rothermel, "A Controlled Experiment Assessing Test Case Prioritization Techniques via Mutation Faults," Proceedings of the International Conference on Software Maintenance (ICSM'05), 2005.
[14] Hyunsook Do and Gregg Rothermel, "On the Use of Mutation Faults in Empirical Assessments of Test Case Prioritization Techniques," IEEE Transactions on Software Engineering, Vol.32, No.9, September 2006.
[15] Hyunsook Do, Sebastian Elbaum, and Gregg Rothermel, "Supporting Controlled Experimentation with Testing Techniques: An Infrastructure and Its Potential Impact," Empirical Software Engineering, Vol.10, No.4, pp. 405-435, 2005.
[16] Software-artifact Infrastructure Repository, http://sir.unl.edu
[17] Hyunsook Do, Gregg Rothermel, and A. Kinneer, "Prioritizing JUnit Test Cases: An Empirical Assessment and Cost-Benefits Analysis," Empirical Software Engineering, Vol.11, No.1, pp. 33-70, Mar. 2006.
[18] Hyunsook Do, Gregg Rothermel, and Alex Kinneer, "Empirical Studies of Test Case Prioritization in a JUnit Testing Environment," Proceedings of the International Symposium on Software Reliability Engineering (ISSRE'04), November 2004.

박 현 철

2001년 숭실대학교 컴퓨터학부 학사 2008년 한국정보통신대학교 공학부 석사 2008년~현재 삼성탈레스 기술연구소 선임연구원. 관심분야는 지휘무장통제체계, 모델링 & 시뮬레이션, 소프트웨어 프로세스, 소프트웨어 메트릭, 소프트웨어 측정, 소프트웨어 테스팅

류 호 연

1998년 경상대학교 전산학 학사. 2000년 경상대학교 정보시스템학 석사. 2005년 경상대학교 정보시스템학 박사. 2005년~2006년 한국과학기술연구원 박사후 연구원. 2006년~2008년 한국정보통신대학교 공학부 연구교수. 2008년~현재 ㈜포스데이타. 관심분야는 소프트웨어 테스팅, 소프트웨어 신뢰성, 소프트웨어 프로세스 개선, 소프트웨어 보안, 온톨로지 공학

백 종 문

1993년 조선대학교 컴퓨터과학 및 통계학과 학사. 1996년 미국 University of Southern California Computer Science 석사. 2000년 미국 University of Southern California Computer Science 박사 2001년~2005년 미국 모토로라 SSERL (Software and System Engineering Research Lab) 수석연구원. 2005년~2009년 한국정보통신대학교 공학부 부교수. 2009년~현재 한국과학기술원 정보과학기술대학 부교수 관심분야는 소프트웨어 신뢰성, 소프트웨어 메트릭스, 소프트웨어 비용추정, 소프트웨어 동적모델링, 소프트웨어 프로세스 개선, 소프트웨어 품질보증, 소프트웨어 식스 시그마