# SRGM용 테스트 자원의 할당

최규식, 김용경
건양대학교
{che}@konyang.ac.kr

# Test Resources Allocation for SRGM

Che Gyu Shik, Kim Yong Kyung
Konyang University

**요 약**

소프트웨어 시스템을 집적하는데 있어서 컴퍼넌트 중심의 소프트웨어 개발 접근 방법이 큰 경향이다. 집적 소프트웨어 시스템의 전체 신뢰도를 확보하기 위해 소프트웨어 컴퍼넌트의 테스트 및 자원의 유한성 내에서 소프트웨어가 요하는 조건을 만족해야만 한다. 기지의 비용, 신뢰도, 테스트 노력, 시스템 컴퍼넌트의 기타 다른 공헌 인자를 가지고 순열조합의 최적화 문제로서 시스템 테스트의 최적화 문제 효율을 공식화할 수 있다. 본 연구에서는 그 각각이 사전에 명세화한 신뢰도 요건을 가진 단일 또는 다중 실용화 시스템에 대한 "소프트웨어 컴퍼넌트 테스트 자원 할당"을 고려한다. 이것은 내고장(fault-tolerant) 시스템에도 확장해서 실용화할 수 있다. 테스트 자원 할당문제에 체계적으로 접근하는 절차를 논하고자 한다.

## 1. 서론

현대의 컴퍼넌트 기준 소프트웨어 공학은 단기간 시장 수요를 만족시키기 위해 경제적이고도 신뢰성 높은 소프트웨어를 개발하는데 큰 관심을 두고 있다. 그러한 모듈과 컴퍼넌트를 가진 시스템 통합에 대해서 시스템 테스트 문제가 기지의 비용, 노력, 컴퍼넌트의 기타 다른 공헌 인자를 가지고 조합적 최적화 문제로서 시스템 테스트 문제를 공식화할 수 있다. 이런 형태로서 가장 잘 알려진 시스템 신뢰도 문제는 직병렬 다중성 할당이며, 이는 시스템 신뢰도가 최대로 되든지 테스트 비용/노력이 최소로되는 경우이다.

본연구에서는 다중 실용화 환경에 있는 여러 형태의 소프트웨어 신뢰도 모델에 근거한 일반적인 소프트웨어 컴퍼넌트 신뢰도 할당 문제를 고찰하였다. 단일 실용화 환경, 일반적인 연속 분포 환경에 대해서 해법을 제공하고 있다.

## 2. 프로젝트 실용화

컴퍼넌트 기준 기법들에 대한 몇 가지 실제 프로젝트를 통하여 이러한 검토를 동기화시켰다. 아래에 3개의 케이스를 설명한다.

### 2.1 분산 소프트웨어 시스템

분산 원거리 통신시스템은 여러 다양한 소프트웨어 요건을 만족시키기 위해 다양한 소프트웨어 컴퍼넌트를 수행함으로써 다중 실용화 형태를v 실용화하게 된다.

그러한 시스템을 테스트중일 때는 신뢰도가 최우선 관심사이며 그러므로 테스트와 자원 할당을 적절하게 하는 것이 매우 중요하다.

### 2.2 내결함(fault-tolerant) 시스템

그림 1에서는 많은 시스템에 실용화되었던 계층내 결함 소프트웨어 구조 모델을 보여주고 있다. 각 층은 여러 소프트웨어 컴퍼넌트를 포함하고 있다. 모든 시스템이 모든 층을 포함하고 있는 것은 아니다.

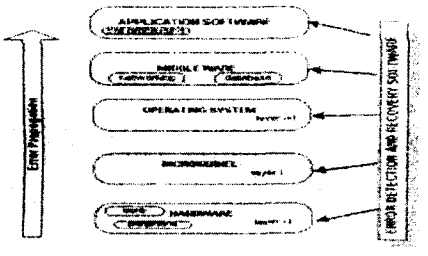

그림 1. 계층소프트웨어구조모델

## 3. 고정 고장율 제한치

테스트 시간을 컴퍼넌트에 할당하여 그 실용화가 신뢰도 요건에 만족하고 테스트 시간이 최소화하기 위함이다. 따라서, 다음과 같이 가정한다.

1) 컴퍼넌트의 고장율은 지수 함수분포를 통하여 신뢰도 실용화와 관련을 짓는다.

$R_i(t) = \exp(-\delta_i t)$

2) 컴퍼넌트 j에서 산출한 테스트시간 Dj는 신뢰도 성장 모델에 따라고 장율 λj를 감소시킨다.

3) 소프트웨어가 일단 발행되면 그들 고장율은 상수로 남아 있다.(실용화 개발자는 사용중인 소프트웨어를 디버깅하거나 변경시킬 수 없기 때문이다)

4) 컴퍼넌트는 그들의 고장 거동에 대하여 s독립이다. (소프트웨어가 상호간에 작용을 할 때, 잠재적으로 다른 고장을 일으킬 수가 있어서 이 가정이 적절하지 않을 수

도 있다)

객체함수는 다음과 같다.

$\sum_{j=1}^{N}\sigma_{ij}\cdot\lambda_j\le\delta_i(A_i$실용화$)$

$A_i$를 사용하면 $\sigma_{ij}=1$, 기타는 $\sigma_{ij}=0$

$\lambda_j\ge0;\ D_1,\cdots,D_N\ge0;\ \delta_1,\cdots,\delta_M\ge0$ 인 조건하에서

$$D=\sum_{j=1}^{N}D_j \tag{1}$$

를최소화하는 것이다.

### 4. 고정된 테스트 예산

테스트 시간을 컴퍼넌트에 할당하여 각각의 실용화가 기껏해야 명세화된 양만큼 할당하여 실용화 신뢰도가 최대가 되도록 한다. 결국, 컴퍼넌트의 총 고장은 최소화해야 하는 객체함수이며, 수학적 프로그래밍 문제로서 아래의 공식에 이르게 한다.

객체함수는

$A_i$에 대해여 $\sum_{j=1}^{N}\sigma_{ij}\cdot D_j\le d_i,\ i=1,\cdots,M$

인 조건하에

$$\lambda\equiv\sum_{j=1}^{N}\lambda_j \tag{2}$$

를 최소화한다.

### 5. 단일 실용화 환경에 대한 해법

시스템에 하나만의 실용화가 있을 때에는 신뢰도 할당에 대한 외부해를 발견할 수 있다.

#### 5.1 지수함수적 신뢰도 성장 모델

지수함수적 신뢰도 성장 모델은 λj를 투자된 Dj와 관련시킨다.

$$\lambda_j=\lambda_{j0}\cdot\exp(-\mu_jD_j) \tag{3}$$

무한시간 전체에 걸쳐서 λj0, μj가 발견된다.

1) 고정된 고장율 제한치 : 이 문제는 지수함수적 신뢰도 성장곡선을 가정하여 단일 실용화 환경에서 공식화한다.

$\sum_{j=1}^{N}\lambda_j\le\delta$

인 조건하에서

$$D=\sum_{j=1}^{N}\frac{1}{\mu_j}\cdot\log(\frac{\lambda_{j0}}{\lambda_j}) \tag{4}$$

를 최소화한다.

(3)을 풀기 위해 래그랜지법을 사용할 수 있다. 최적화 문제는 아래 방정식의 최소치를 구하는 것과 등가이다.

$$F(\lambda_1,\cdots,\lambda_j)=D+\theta\left[\left(\sum_{j=1}^{N}\lambda_j\right)-\delta\right] \tag{5}$$

그 해는

$$\lambda_1=\frac{\delta}{1+\sum_{j=1}^{N}\frac{\mu_1}{\mu_j}},$$

$$\lambda_j=\frac{\mu_1}{\mu_j}\cdot\lambda_1,\ j=2,\cdots,N \tag{6}$$

소프트웨어에 할당된 테스트 시간은 (6)의 값을 (4)에 치환하여 유도한다. 예를 들면

$$D_1=\frac{1}{\mu_1}\cdot\log(\frac{\lambda_{10}}{\lambda_1}) \tag{7}$$

이고, λ1≥λ10이면 D1은 음의 값이다. 어떠한 불가능한 해가 발견되지 않도록 확보하기 위해 검증조건을 점검하고 최적해법이 합당한전략을 따른다는 것을 보증하기 위해 그 절차를 제안한다.

예제 1 :

시스템이 3개의 컴퍼넌트 C1, C2, C3를 가지며 모든 컴퍼넌트를 사용하는 하나의 실용화 A를 가진다. λ10=λ20=λ30=5/year이다. 실용화 요건은 δ=6/year이고 μ1=μ2=μ3=1 이다. 그러면 λ1=λ2=λ3=2/year이고 D1=D2=D3=log(2.5)이다.

그러므로, 초기 고장율과 디버깅이 진행됨에 따른 고장율의 감소율은 모든 컴퍼넌트에 동일하다. 그러면 모든 컴퍼넌트 고장율이 동일하게 되는 곳에서 평균 테스트 전략은 최소 테스트시간의 실용화 요건에 맞는 해법을 제공한다.

예제2:

시스템이 3개의 컴퍼넌트를 가지며, 이 컴퍼넌트는 상이한 초기 고장율을 가지고 있다. λ10=5/yearλ20=6/yearλ30=7/year δ=6/year μ1=μ2=μ3=1. 그러면 λ1=λ2=λ3=2이다. 예제 1에서와 같이 평균 테스트 전략은 최소의 테스트 시간을 소모하면서 실용화 요건을 만족하는 해법을 제공한다. 그러나 각 컴퍼넌트의 테스트 시간이 초기 고장율이 다르므로 각 컴퍼넌트의 테스트 시간이 다르다. D1= log(5/2)=0.3979, D2=log(6/2)=0.4771, D3=log(7/2)= 0.5441. 각 컴퍼넌트의 테스트 시간은 초기 고장율의 대수함수에 비례한다.

예제3:

시스템이 3개의 컴퍼넌트를 가지며 동일한 초기 고장율을 가진다. λ10=λ20=λ30=5/year δ=6, μ1=1, μ2=2, μ3=3. 테스트 시간을 최소화하기 위한 값은 다음과 같다. λ1=6/1.834=3.273, λ2=3/1.834=1.636, λ3=2/1.834=1.091이다. 이러한 고장율에 이르도록 하는 최적테스트 수법은 총 테스트시간

$$\frac{1}{1}\log(\frac{5\times1.834}{6})+\frac{1}{2}\log(\frac{5\times1.834}{3})$$

$$+\frac{1}{3}\log(\frac{5\times1.834}{2})=1.49$$

을 요한다.

λ1=λ2=λ3=2을 할당하는 평균 테스트 수법은 총 테스트 시간이 log(5/2) · 1.834=1.68에 이르게 하며, 이는 최적테스트 수법보다 더 큰 것이다.

#### 5.2 고정된 테스트예산

고정 테스트 예산 문제는 지수함수적 신뢰도 곡선에서처럼 단일 실용화 환경에서 공식화할 수 있다.

$$\sum_{j=1}^{N} D_j \le D$$

의 제한하에서

$$\lambda = \sum_{j=1}^{N} \lambda_j \qquad (8)$$

를 최소화한다.

다시 말해서, 이는 래그랜지법으로 풀 수 있으며, 최적화 문제를

$$F(D_1, \cdots, D_N) = \lambda + \theta(\sum_{j=1}^{N} D_j - D) \qquad (9)$$

의 최소값을 구하는 것과 등가인 것으로 취급한다.

그 해는

$$\lambda_{j0} = [-\mu_j \cdot \exp(-\mu_j \cdot \exp(-\mu_j D_j)] \qquad (10)$$

이 모든 j=1,2, ···, N에 대해서 동일하다.

$$D_1 = \frac{D - \sum_{j=2}^{N} \frac{1}{\mu_j} \log(\frac{\lambda_{j0}\mu_j}{\lambda_{10}\mu_1})}{\sum_{j=1}^{N} \frac{\mu_1}{\mu_j}},$$

$$\frac{1}{\mu_j} \log(\frac{\lambda_j \mu_j}{\lambda_{10}\mu_1}) \frac{+\mu_1}{\mu_j} D_1 \quad , \; j=2, \cdots, N \qquad (11)$$

방정식 (9)-(11)은 컴퍼넌트에 테스트 시간이 어떻게 할당되어야 하는지를 결정한다. 테스트 시간에 대한 값으로부터 최소 λ가 뒤따른다.

λj0와 μj값들이 j에 대해서 독립일 때만 가용한 테스트 시간이 컴퍼넌트중에 동일하게 분포되는 곳에서 평균 테스트 수법이 최적해를 제공한다. λj0나 μj값이 j에 대해서 동일하지 않으면 테스트 시간의 최적 할당을 구하기 위해 (9)-(11)을 계산해야 한다.

## 6. 일반적인 신뢰도성장모델

이 장에서는 일반적인 신뢰도 성장 모델에 대한 폐쇄형태의 해법을 구하는 절차를 제공하는 것이다. 성장 모델의 단 하나의 제약이란 첫 번째 및 두 번째 도함수에 관한 것이다.

고정 고장율 제한치의 경우를 고려해보기로 하자. 고장율과 테스트 시간 사이의 관계는 fj의 함수이다.

$$D_j = f_j(\lambda_j) \qquad (12)$$

$$abs(\frac{d}{d\lambda_j} f_j(\lambda_j)|_{\lambda_j = \lambda_{j0}})$$

$$\ge abs(\frac{d}{d\lambda_{j+1}} f_{j+1}(\lambda_{j+1})|_{\lambda_{j+1} = \lambda_{j+1,0}}),$$

$$j=1,2, \cdots, N-1 \qquad (13)$$

폐쇄형 해를 얻기 위한 알고리즘은 이 (13)을 사용한다.

## 7. 다중 실용화 환경에 대한 해법

시스템에서 다중적으로 실용화하여야 할 경우가 생기면 신뢰도할당이 외형적으로 풀기에 극히 어렵게 된다. 그러나, AMPL과 같은 비선형 프로그래밍 소프트웨어를 사용하면 그 해를 구할 수 있다.

## 8. 소프트웨어고장의 종속성 및 내결함 시스템

기본 신뢰도 할당문제를 공식화하는 것은 여러 가지 방향으로 확장할 수 있다. 두 가지의 확장에 대해서 논하고자 한다.

원래 문제의 제한치를 다음과 같이 수정한다.

Ai에대해서 $\sum_{j=1}^{N} \sigma_{ij}\lambda_j + \sum_{\omega(i,j)} \lambda_{ij} \le \delta$ , i=1, ···, M

$$\omega(i,j) \equiv \{\forall(i,j) | \sigma_{1j} = 1\} \qquad (14)$$

(14)의 제한치 하에 $D = \sum_{j=1}^{N} D_j$

를 최소화한다.

그러므로 고장율문제는 객체함수에서 모든 각개 및 쌍둥이형 컴퍼넌트 테스트 시간을 포함하여 고장율 제한치 하에서 쌍둥이형 고장율을 추가하여 구할 수 있다.

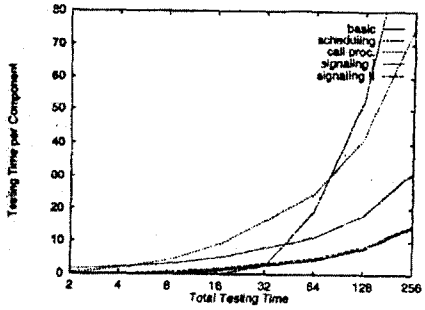


그림 2. 총 테스트 시간에 대한 컴퍼넌트용 최적할당

시스템이 내결함 속성을 소유하고 있을 때에는 본래의 문제에 커버하는 인자를 도입한다. 커버리지는 결함이 컴퍼넌트에서 활성화될 때 시스템 고장을 일으키지 않고 검출하여 복구하는 조건확률로 정의한다. pj≡1-cj를 이용하여 고정 고장율 제한치의 경우 공식을 재정비하면

Ai에대해서 $\sum_{j=1}^{N} \sigma_{ij}\rho_j\lambda_j \le \delta_1$ i=1,2,···,M

인 조건하에서

$$D = \sum_{j=1}^{N} D_j \qquad (15)$$

를 최소화하는 것과 같다.

감사의 글

이 논문은 2003학년도 건양대학교 학술연구비 지원에 의하여 이루어진 것임