다중입력영역시험에서의 대형 소프트웨어 고장률 추정 연구

문 숙 경
목원대학교, 정보통계학과

Estimating the Failure Rate of a Large Scaled Software in Multiple Input Domain Testing

Soog-Kyung Moon
Dept. of Information and Statistics, Mokwon University

Key Words : Failure Rate, Bayesian Rule, Multiple Input Domain

Abstract

In this paper we introduce formulae for estimating the failure rate of a large scaled software by using the Bayesian rule when a black-box random testing which selects an element(test case) at random with equally likely probability, is performed. A program or software can be treated as a mathematical function with a well-defined (input)domain and range. For a large scaled software, their input domains can be partitioned into multiple subdomains and exhaustive testing is not generally practical. Testing is proceeding with selecting a subdomain, and then picking a test case from within the selected subdomain. Whether or not the proportion of selecting one of the subdomains is assumed probability, we developed the formulae either case by using Bayesian rule with gamma distribution as a prior distribution.

1. 서 론

1.1 머리말

소프트웨어라 함은 완벽성이 요구되는 항공모함을 제어하는 최첨단 소프트웨어에서부터 인간의 생명력과 직결되는 원자력 발전소나 항공기, 비행기 등을 제어하는, 그리고 각종 중, 대형 컴퓨터에 내장된 프로그램에 이르기까지 이루 헤아릴 수 없을 정도의 다양한 기능을 수행하는 프로그램 등을 말한다. 이 중에서 절대적 완벽성이 요구되는 제품일수록, 나아가 인간의 생명과 직결되는 제품일수록 고도의 신뢰성이 요구된다.

성공적인 소프트웨어는 그 소프트웨어에 에러가 없음을 나타내는 것이다. 소프트웨어

시험이란 에러를 발견할 의도로 그 소프트웨어를 실행시키는 과정이며, 성공적인 시험은 개발 과정 중에 혹은 이 전에 발견하지 못한 에러를 찾아내는 것이다. 그러므로 시험의 목표는 최소한의 시간과 노력으로 다른 부류의 에러를 체계적으로 검출하려는 방법들을 설계하는 것이다.

기존의 소프트웨어 신뢰성 연구 분야에서는 소프트웨어 제품을 시험하는 도중에 발견된 고장들에 대한 정보를 바탕으로 몇가지 가정을 세우고 통계적 추정 모형에 따라 신뢰도, 고장률 및 잔존 에러 수 등을 예측하는 연구들을 하여왔다[1,3]. 본 고에서는 시험 기간 중 예상과 다르게 나온 시험 결과를 고장이라 간주하고 이런 고장 발생 빈도 즉, 고장률에 대한 연구를 하려한다.

대부분의 소프트웨어 시험에 있어서 시험 항목(test case)의 종류는 이루 헤아릴 수 없을 정도로 많기 때문에, 현실적으로 모든 시험 항목들에 대하여 시험을 실시한다는 것은 거의 불가능하다. 특히, 규모가 큰 대형 소프트웨어인 경우는 더욱 불가능하다. 본 고에서는 여러 가지 소프트웨어를 시험하는 방법 중에서도 주로 임의로 시험 항목을 선택하는 블랙박스(black-box) 식의 시험 방법을 채택한다. 소프트웨어를 시험한다는 행위는 입력 공간과 출력 공간을 가지는 함수로 규정할 수 있으며, 소프트웨어마다 하나의 입력 공간 및 출력 공간을 정의할 수 있는데, 규모가 큰 소프트웨어는 대부분 큰 입력 공간으로 정의되므로, 이를 동질의 몇 개의 부분으로 나누는데 이를 다중 입력영역이라 한다. 대규모 소프트웨어들은 대부분 다중 입력 영역을 가지고 있다고 볼 수 있다.

베이지안적인 추정 방법은 단순히 임의 표본에서 얻어진 정보만을 이용하여 고장률을 추정하는 대신에 과거의 경험과 주관적인 판단을 고장률에 대한 사전 정보의 형태(분포형태)로 고장률의 추정에 이용한다는 것이다. 본 고에서는 고장률에 대한 사전 분포(prior distribution)를 감마분포로 가정하며, 시험을 실시할 때, 시험 항목들은 임의로 선택하여 시험을 실시하는 랜덤 시험이라 가정한다.

고장률을 추정하는 기존의 연구 동향 및 기본 용어들을 2장에서 간략히 설명하고, 고장률을 추정하는 기법들은 3장에서 논하고자 한다. 현실적으로 규모가 큰 소프트웨어가 신뢰도나 고장률 등을 추정하는 신뢰성 연구의 실제 대상들이며, 이들 대규모 소프트웨어들은 대부분 다중 입력 영역을 가지고 있다고 볼 수 있으므로, 3장에서는, 이 다중 입력 영역을 가진 소프트웨어에 대하여 고장률을 추정하는 방법을 제시하였다.

1.2 기존 사례 연구

Thayer et al.[5]의 연구에서 그들 역시 베이지안적인 방법으로 접근하였는데 사전 분포(prior distribution)를 일양분포(uniform distribution)를 사용하여 고장률을 추정하였다.

Weiss and Weyuker[6]는 랜덤하지 않게 시험을 실시하여 시험 항목마다 가중치를 다르게 부여하는 방법으로 고장률을 추정하였다.

Becker and Camarinopoulos[2] 역시 베이지안 방법으로 고장률을 추정하였는데 그들은, 소프트웨어 제품의 앞 버전을 시험한 결과 발생된 고장 정보를 이용하여 고장률을 추정하였으나 다중 입력 영역 시험과 같은 확장된 이론은 전개하지 못하였다.

Keith W. Miller and etc.[4] 또한 베이지안적인 방법을 사용하였는데 사전 분포(prior distribution)를 베타 분포(Beta distribution)라 가정하고 고장률을 추정하였으나 시험 항목 수에 의한 이산적인 방법을 사용하였다. 그리고 연구 대상이 대형 소프트웨어이며, 특히 고장 발생이 거의 없는 고품질의 소프트웨어를 시험하는, 즉, 고장률이 0에 가까운 경우의 고장률 추정 방법에 관한 연구를 하였다.

2. 기본 가정 및 용어 정리

본 고에서 소프트웨어라 함은 하나의 프로시져 또는 몇 개의 모듈이 합쳐진 나아가서 하나의 완전한 시스템을 지칭하는 프로그램을 말한다. .

대상 소프트웨어를 시험함에 있어서 고장이 발생하였다함은 예상된 결과 혹은 반응치가 나오지 않을 경우를 지칭하는 것으로, 흔히 고장률(the probability of failure)은 발생된 총 고장 수를 지금까지 시험된 총 시험 항목 수나 총 시험 시간으로 나누어 계산되어 진다.

본 고에서는 소프트웨어 제품의 앞 버전을 시험한 결과 발생된 고장 정보를 이용하려는, 즉, 사전에 이와 비슷한 상황에서의 시험 결과에 대한 정보들을 이용하려는 베이지안적인 방법을 이용하여 고장률을 제시하려한다. 본 고에서는 고장률에 대한 사전 분포를 감마분포로 가정한다.

임의의 소프트웨어 제품은 시험 단계에 따라 소프트웨어 내부흐름을 볼 수 없다고 간주하면서 시험하는 블랙 박스(black-box) 시험이나 소프트웨어 내부흐름을 볼 수 있다고 가정하는 화이트 박스(white-box) 시험 방식 중 하나를 선택하여 시험한다. 본 고에서는 소프트웨어를 내부가 보이지 않는 검은 상자로 간주하는 블랙 박스 시험을 랜덤하게 실시함을 원칙으로 한다. 랜덤하게 시험을 실시한다함은 예정되어진 순서에 따라 시험을 실시하는 것이 아니라, 하나 하나의 시험 항목을 임의적으로 무작위로 선별하여 시험을 실시하는 것을 말한다.

각각의 시험 항목을 시험하는데 걸리는 시간은 같다고 간주하며, 또한 고장이 발생하였을 때의 수정 시간은 무시한다고 가정하자.

한편 소프트웨어를 시험하는 행위를 하나의 수학적인 함수로 비유되어질 수 있다. 즉, 시험을 실시할 때, 소프트웨어에 입력을 가하여 소프트웨어가 제대로 된 결과를 보일 수도 있고 그렇지 못할 수가 있다. 전자는 시험이 성공하여 고장이 발생하지 않은 경우이고, 후자는 고장이 발생했다고 말한다. 이때, 시험 데이터 등을 비롯한 다양한 입력들을 모아놓은 하나의 집합을 입력 공간(정의역)으로 그리고 예상된 혹은 예상 밖의 기대치들을 모아놓은 집합을 출력 공간(치역)으로 정의하는 하나의 수학적 함수로 묘사할 수 있다.

t 시간까지 발생된 고장 수를 나타내는 확률변수를 N(t)로 표기하며, 일정한 고장률 λ를 갖는 포아송분포(Homogeneous Poisson process)로 가정한다.

3. 고장률 추정

3.1 점 추정과 구간 추정

t 시간까지 발생된 고장 수가 n(N(t)=n),이라 할 경우, 일정한 고장률 λ를 갖는 포아송분포(Homogeneous Poisson process)로 가정하였고, 고장률에 대한 사전 분포를 감마분포로 가정하였으므로,

$$P(N(t)=n \mid \Lambda=\lambda)=\frac{(\lambda t)^n}{n!}e^{-\lambda t}$$
$$\text{for} \quad n=0,1,\ldots \quad .$$

고장률에 대한 사전 분포를 감마분포로 가정하였으므로,

$$f_\Lambda(\lambda)=\frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda} \ ,$$
$$\text{for} \quad \lambda>0,\ \alpha>0\ ,\beta>0 \ .$$

그러므로,

$$P(N(t)=n)$$
$$=\int_0^\infty P(N(t)=n \mid \Lambda=\lambda)\cdot f_\Lambda(\lambda)d\lambda$$
$$=\int_0^\infty \frac{(\lambda t)^n}{n!}e^{-\lambda t}\cdot\frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}d\lambda$$
$$=\frac{\beta^\alpha t^n}{\Gamma(\alpha)n!}\int_0^\infty \lambda^{\alpha+n-1}e^{-(\beta+n)\lambda}d\lambda$$
$$=\frac{\beta^\alpha t^n}{\Gamma(\alpha)n!}\ \frac{\Gamma(n+\alpha)}{(\beta+t)^{n+\alpha}} \ .$$

따라서, N(t)가 주어진 상황에서의 Λ에 대한 사후 확률 분포는 다음과 같다.

$$f_{\Lambda\mid N(t)}(\lambda \mid n)$$
$$=\frac{(\beta+t)^{\alpha+n}}{\Gamma(\alpha+n)}\lambda^{\alpha+n-1}e^{-(\beta+t)\lambda}.$$

따라서 λ에 대한 베이지안적 추정치 $\hat{\lambda}$은 다음과 같다.

$$\hat{\lambda}=E(\Lambda \mid N(t)=n)=\frac{\alpha+n}{\beta+t} \ .$$

확률변수 Z를 다음과 같이 정의하고

$$Z=2(\beta+t)\Lambda \ ,$$

$N(t)=n$이 주어졌을 때의 Z의 조건부 분포함수를 구하면 다음과 같다.

$$f_{Z\mid N(t)}(z \mid n)$$
$$=\frac{1}{2^{\alpha+n}\Gamma(\alpha+n)}z^{\alpha+n-1}e^{-z/2}, \ \text{for } z>0 \ .$$

즉, 자유도 $2(\alpha+n)$를 갖는 Chi-square 분포를 갖는다. 그러므로,
양방향의 신뢰구간을 구하면 다음과 같다.

$$1-\varepsilon$$
$$=P(\frac{z_{1-\varepsilon/2,2(\alpha+n)}}{2(\beta+t)}<\Lambda<$$
$$\frac{z_{\varepsilon/2,2(\alpha+n)}}{2(\beta+t)} \mid N(t)=n)$$

, 여기서 $z_{\varepsilon,\nu}$는 자유도 ν를 갖는 Chi-square 분포의 $100\,\varepsilon$ % percentile을 나타낸다.
그리고, 한 방향으로의 신뢰구간은 다음과 같다.

$$P(\Lambda<\frac{Z_{\varepsilon,2(\alpha+N)}}{2(\beta+t)} \mid N(t)=n)=1-\varepsilon \ .$$

한 예로, $\alpha=3, \beta=1\cdot10^{-4}$ 인 경우 즉, 사전 확률분포함수인 감마분포에서의 모수

α, β값을 안다고 가정하자. 그리고, 총 시험에 소요된 시간(t)은 $5 \cdot 10^3$ 시간이며, 이 기간 동안 발견된 고장 수(n)는 2이라 할 때, λ에 대한 베이지안 추정치 $\hat{\lambda}$은 다음과 같다.

$$\hat{\lambda} = \frac{\alpha + n}{\beta + t} = \frac{3+2}{10^4 + 5 \cdot 10^3} \approx 3.33 \cdot 10^{-4}$$

그리고 90%신뢰구간을 구하면 다음과 같다.

$$P\left(\frac{z_{0.95,10}}{2(\beta+t)} < \Lambda < \frac{z_{0.05,10}}{2(\beta+t)} \mid N(t)=2\right) = 0.90$$

$$P(1.31 \cdot 10^{-4} < \Lambda < 6.10 \cdot 10^{-4} \mid N(t)=2) = 0.90$$

3.2 다중 입력영역 시 고장률의 추정

현실적으로 신뢰도나 고장발생 확률 등을 추정하려한다면 상당히 규모가 큰 것들이 대부분이다. 그러므로 이 장에서는 전체 입력 공간이 대체로 너무 규모가 커서 다루기 어려운 대형 소프트웨어를 그 대상으로 하여 고장률을 구하여 볼 것이다. 전체 입력 공간을 k개의 부 입력 공간으로 나누고, 각각의 부 입력 공간들 안에 수많은 시험 항목들이 속해있으며, 이런 부 입력 영역 공간을 모두 모아 다중 입력 영역이라 한다. 이렇듯 소프트웨어를 시험하는 행위를 입력 공간과 출력 공간을 가지는 함수로 간주할 때, 다중 입력 영역을 입력 공간으로 간주하고 소프트웨어를 시험하는 것을 다중 입력 영역 시험이라 한다. 한편, 시험을 실시하기 위해 시험 항목을 선택하는 과정을 살펴보면, 우선, 부 입력 영역 공간들 중에서 하나를 선택하고 나서, 선택된 부 입력영역 공간에서 하나의 시험항목을 랜덤하게 선택한다고 한다.

시험이 임의로 하나의 부 입력 영역을 우선 선택하는데 i번째 부 영역을 선택할 확률(p_i)은 i번째 부 영역에 속해 있는 항목 수에 비례한다고 가정한다. 즉, 전체 시험 항목 수는 y개, 그리고 k개의 각 부 입력 공간에서 i번째 부 영역에서의 시험 항목 수는 y_i라 가정하자. 즉, $y = y_1 + y_2 + \cdots + y_k$ 라 가정할 때, 임의로 하나의 부 입력 영역을 선택할 확률(p_i)은 다음과 같다고 가정한다.

$$p_i = \frac{y_i}{y}, \quad i = 1, 2, \cdots, k$$

다시 말하자면 각기 다른 수의 흰공, 검은 공이 섞여 들어있는 여러 항아리에서 공을 하나 뽑을 때, i번째 항아리를 선택하고 난 후, 이 항아리 속에 들어 있는 공을 하나 랜덤하게 뽑는데, i번째 항아리를 선택할 확률이 p_i이며, 이는 항아리 속에 들어있는 공의 수에 비례한다고 가정한다는 것이다. 선택되어진 부 입력영역 공간에서 하나의 시험 항목을 선택하는 경우는 모든 시험항목들이 동일한 확률을 갖는다고 본다. 즉, 랜덤하게 하나의 시험 항목을 선택하여 시험을 실시하고, 각각의 뽑혀진 하나 하나의 시험 항목을 시험하는데 소요되는 시간 또한 동일하다고 가정한다.

각각의 i번째 부 영역에서의 고장률(모수)를 λ_i 라하고, 총 시험 시간 t시간 동안 i번째 부 영역에서 발견된 고장 수를 n_i, 즉, $N_i(t) = n_i$ 라 하자. 그러면 발견된 전체 고장 수 n은 다음과 같다.

$$n_1 + n_2 + \cdots + n_k = n.$$

한편, 총 시험 시간 t시간 동안 k개의 각 부 영역에서 시험 결과 각각 $z_1, z_2, \cdots, z_k$ 개의 시험 항목들이 실제 뽑혀져 시험되어졌고, 이들 시험 항목들을 모두 합한 수를 $z(= z_1 + z_2 + \ldots + z_k)$라 가정하자. 이 경우 대상이 대형 소프트웨어일수록 모든 항목들에 대하여 시험을 완전히 이행할 수 없는 경우가 대부분이므로, 각각의 z_i들은 y_i보다 작은 수가 일반적이다. 그러면 이 경우의 각각의 z_i들에 대한 기대치는 zp_i와 같게 될 것이다.

실제 시험은 각 i번째 부 영역을 확률 p_i로 선택하고 선택되어진 각 부 영역 내에서 랜덤하게 시험항목들을 선택하여 시험하게 되는데, 각 i번째 부 영역을 가정되어진 확률 p_i에 준해서 뽑을 수도 있고, 실제 시험을 실시하면서 이것을 지키지 못할 수도 있을 것이다. 이 두 경우를 구분하여 각각의 경우마다 고장발생 확률을 추정하는 방법을 제시하려 한다.

3.2.1 각 i번째 부 영역에서 각 원소들이 가정되어진 확률 p_i로 뽑혀지는 경우

1) 각각의 i번째 부 영역에서 사전 분포에 대한 정보를 모르는 경우

비록 여러 부 영역들이 합하여져 있지만 하나의 프로그램으로 간주할 수 있을 것이다. 그러므로 전체 시험된 시험 항목 수 z를 이용하여 $\hat{\lambda}$를 다음과 같이 구할 수 있다.

$$\hat{\lambda} = \frac{\alpha + n}{\beta + t}$$

2) 각각의 i번째 부 영역에서 사전 분포에 대한 정보를 아는 경우

각각의 i번째 부 영역에서 사전 분포는 α_i, β_i를 모수로 갖는 감마 분포로 가정함으로, i번째 부 영역에서의 고장률의 점 추정치 ($\hat{\lambda_i}$)는 다음과 같고,

$$\hat{\lambda_i} = \frac{\alpha_i + n_i}{\beta_i + t} \qquad i = 1, 2, \ldots, k$$

총 영역에서의 전체 고장률에 대한 점 추정치($\hat{\lambda}$)은 다음과 같다.

$$\hat{\lambda} = \sum_{i=1}^{k} p_i \hat{\lambda_i} .$$

이는 마치 검은 공과 흰 공의 수가 각기 다르게 섞여 있는 k개의 보자기 중에서 임의로 각 항아리를 선택할 확률이 각각 $p_1, p_2, \ldots, p_k$이고, 각각의 항아리에서 검은 공을 선택할(고장이 발생할) 확률을 각각 $\lambda_1, \lambda_2, \ldots, \lambda_k$이라 할 때, 이들 k개의 전체 항아리 중에서 검은 공을 선택할 확률을 λ라 하면 다음과 같이 표현될 수 있다.

$$\lambda = \Pr\{black\}$$

$$= \sum_{i=1}^{k} \Pr\{i\text{번째 항아리선택}\} \cdot \Pr\{black \mid i\text{번째 항아리선택}\}$$

$$= \sum_{i=1}^{k} p_i \lambda_i$$

3.2.2 각 i번째 부 영역에서의 가정되어진 확률 p_i와 다를 경우

각 i번째 부 영역에서 z_i개의 시험 항목들이 뽑혀져 시험되어졌지만 각각의 부 영역들은 가정되어진 p_i 비율이 아닌 다른 비율 $q_1, q_2, \ldots, q_k$로 뽑혀져야 된다는 가정을 하여 보자. 이처럼 시험되어진 결과와 가정된 비율이 다르게 된 결과를 초래하게된 이런 상황은 또한 다음과 같이 크게 2가지로 나누어 생각할 수 있다. 부 입력 영역들은 같은데 단순히 뽑히는 비율만 $p_i \rightarrow q_i$로 변경된 경우와 부 입력 영역들이 가정되어진 것과 다르게 분할되어져서 각 부 영역을 선택하는 비율이 달라진 결과를 초래할 수 있는 2가지 경우이다. 전자는 비교적 다루기 쉬우나 후자는 조금 복잡해진다. 본 고에서는 전자에 대하여서만 고장률을 구하는 방법을 제시하고자 한다.

단순히 뽑히는 비율만 $p_i \rightarrow q_i$로 변경된 경우, 시험 항목 수가 많은 부 영역에서 보다 오히려 시험항목 수가 비교적 적은 부 영역에서의 시험이 더 많이 이루어져야한다는 것이다. 부연하자면, 실제 상황에서 소프트웨어를 직접 사용할 사용자 그룹들이 빈번하게 사용하거나 중요하게 여기는 부 영역들에 대해, 이 부 영역에 속해있는 시험항목의 수도 많아야될 것 같으나 실제적으로 그렇지 않는 경우가 현실적으로 더 많이 발생할 수 있다. 다시 말해, 특정 부 영역에 대한 사용자 그룹의 중요도와 시험 항목 수는 반드시 비례한다고 볼 수는 없는 것이다. 이 경우, 대부분 시험을 추가로 더 실시하여야만 하므로, 이에 대한 구체적인 방법을 제시하고자 한다.

현재까지 시험되어진 총 시험항목 수 z를 이용하여 전체적으로 더 필요한 시험 시간(항목 수)를 추정한 후, 각각의 i번째 부 영역에서 더 추가로 시험되어져야할 항목 수(시험 시간)를 배정시키고 시간을 t가 아닌 t'로 조정한 고장 수를 이용하여 $\hat{\lambda}$을 구할 수 있을 것이다. 우선, 전체적으로 시험되어져야할 총 시험 항목 수(z')는 다음과 같이 구할 수 있을 것이다.

$$z' = z \max_i (p_i/q_i) .$$

각각의 시험항목을 시험하는데 걸리는 시간은 동일하다는 가정 하에, 총 시험된 항목 z개를 시험한 시간이 t이므로, z'개의 시험항목을 시험할 시험 시간인 t'을 비례식을 이용하여 구할 수 있다. 뿐만 아니라, 각각의 i번째 부 영역에서 t'시간 동안 발생된 고장 수가 n_i'($N_i(t') = n_i'$)이고, 이들을 모두 합해 놓은 것이 n'이라 하면, 이 역시 다음과 같다.

$$\hat{\lambda} = \frac{\alpha + n'}{\beta + t'}$$

여기서 $n' = n_1' + n_2' + \cdots + n_k'$.

그리고, 각 부 영역에서 사전 정보를 안다면 다음과 같이 구할 수 있을 것이다.

$$\hat{\lambda}_i = \sum_{i=1}^{k} q_i \frac{\alpha_i + n_i'}{\beta_i + t'} .$$

또한, 각 부 영역에서 추가로 시험되어져야할 비율 $r_1, r_2, \ldots, r_n$ 들은 다음과 같이 정의할 수 있을 것이다.

$$r_i = (t'q_i - tp_i)/(t' - t) .$$

윗 식에서 $t' - t$는 전체적으로 추가로 필요한 시험시간(항목 수)이며, $t'q_i\ (= t_i')$는 각 i 번째 부 영역에서의 시험되어져야할 총 시험 시간(항목 수)에 대한 기대치이며, tp_i는 i 번째 부 영역에서 지금까지 실제적으로 시험되어진 총 시험 시간(항목 수)이다.

4. 논의 및 결론

본 고에서는 소프트웨어를 시험하는데 있어, 소프트웨어를 입력 영역과 출력 영역을 가지는 함수로 규정하였고, 현실적으로 신뢰도나 고장발생 확률 등을 추정하려한다면 상당히 규모가 큰 것들이 대부분이다. 규모가 큰 소프트웨어일수록 입력 영역의 규모가 크므로, 이를 동질의 성격으로 몇 개의 부분으로 나누는데 이 부분들을 부 입력 영역이라 하고, 이들을 모두 합쳐 다중 입력영역이라 하였다. 그러므로 이 장에서는 전체 입력 공간이 대체로 너무 규모가 커서 다루기 어려운 대형 소프트웨어를 그 대상으로 하였고, 각각의 부 입력 공간들마다 각기 다른 수많은 시험항목들을 포함하고 있다.

시험은 우선, 부 입력영역 공간들 중에서 하나를 선택하고 나서, 선택된 부 입력영역 공간에서 하나의 시험항목을 랜덤하게 선택한다고 가정하였다. 실제 시험은 각 i번째 부 영역을 확률 p_i로 선택하고 선택되어진 각 부 영역 내에서 랜덤하게 시험항목들을 선택하여 시험하게 되는데, 각 i번째 부 영역에 속해있는 전체 항목 수에 비례하는, 가정되어진 확률 p_i에 준해서 뽑을 수도 있고, 이것을 지키지 못할 수도 있을 것이다. 지키지 못하는 경우라 함은, 특정 부 영역에 대한 사용자 그룹의 중요도와 시험 항목 수는 반드시 비례한다고 볼 수 없다는 것이다. 이 경우, 대부분 시험을 추가로 더 실시하여야만 하므로, 이에 대한 구체적인 방법을 제시하였다. 부연하자면 본 고에서는 다중 입력영역 시험을 실시할 수 있는 대규모 소프트웨어에 대한 시험 자료를 정보로 하여 고장률을 추정하는데 그 의의가 있다고 할 수 있다.

개발된 소프트웨어 프로그램을 다중 입력영역 방법으로 시험하던 중 발생된 고장들에 대한 고장률 추정의 골격은 과거의 판단이나 주관적인 판단을 사전 정보로 베이지안 룰을 이용하였다. 본문에서 감마 분포를 사전 분포(prior distribution)로, 고장 발생 빈도를 나타내는 활률 변수는 포아송 분포를 따른다고 하였는데, 본문에서는 이런 분포에 대한 것은 그리 중요하게 취급하지 않았다. 베타 분포를 사전 분포로, 고장 발생 빈도를 나타내는 것은 이항 분포 등, 다양한 분포를 사용하여도 좋을 것이다. 오히려, 전체 입력 공간을 대상으로 한 사전 정보인 α, β 값에 대한 정보만 알고 있을 경우, 이를 각 부 입력 영역 공간으로의 값 α_i, β_i의 할당에 대한 방법론적 연구가 더 중요하고, 여기에 관한

연구를 뒤에 남겨 두려한다.

그리고, 전체 입력 영역을 여러 개의 부 영역으로 나누는 체계적인 방법론에 대한 연구도 심도있게 진행되어져야할 것이다. 본 문에서 구체적으로 언급하지는 않았지만 대상 소프트웨어가 대형임에도 불구하고 여러 개의 부 영역간에 시험 항목에 대한 중복된 내용이 없어서 단순하게 합하기만 하면, 하나의 전체 입력 영역이 되는 비교적 단순한 구조를 가진 소프트웨어임을 가정하고 있기 때문에, 다소 복잡한 구조와 제어 흐름을 갖고 있는 소프트웨어를 대상으로 하는 연구도 차후 수행되어야 연구 과제로 남겨 두려한다.

나아가 여러 개의 부 영역을 단순하게 합하는 것이 아니면 전체 소프트웨어 고장률을 추정하는 문제 또한 더욱 복잡한 형태로 전개될 것이므로, 이에 대한 연구도 병행되어져야할 것이다.

참고문헌

[1] J. D. Musa, A. Iannino, and K. Okumoto, Software Reliability : Measurement, Prediction, and Application, New York : McGraw-Hill, 1987.

[2] G. Becker and L.Camarinopoulos, "A Bayesian estimation method for the failure rate of a possibly correct program," IEEE Trans. Software Eng., vol. 16, pp. 1307-1310, Nov. 1990.

[3] G. J. Schick and R. W. Wolverton, "An analysis of competing software reliability models," IEEE Tran. Software Eng., vol. SE-8, pp. 104-12, Mar. 1978.

[4] Keith W. Miller, Larry J. Morell, Robert E. Noonan, Stephen K. Park and D. M. Nieol, "Estimating the Probability of Failure When Testing Reveals No Failures, "IEEE Tran. Software Eng., vol. 18, No.1, Jan. 1992.

[5] T.A. Thayer, M.Lipow, and E.C.Nelson. Software Reliability(TRW Series of Software Techn., vol. 2). New York: North-Holland, 1978.

[6] S.N. Weiss and E.J. Weyuker, "An extended domain-based model of software reliability," IEEE Trans. Software Eng., vol. 14, pp. 1512-1524, Oct. 1988.