대수 선형 위험함수 학습효과에 근거한 NHPP 신뢰성장 소프트웨어 모형에 관한 비교 연구

김희철* · 신현철**

요 약

본 연구에서는 소프트웨어 제품을 개발하여 테스팅을 하는 과정에서 소프트웨어 관리자들이 소프트웨어 및 검사 도구에 효율적인 학습기법을 이용한 NHPP 소프트웨어 모형에 대하여 연구 하였다. 적용모형은 로그 형 위험함수 모형을 적용한 유한고장 NHPP에 기초하였다. 소프트웨어 오류 탐색 기법은 사전에 알지 못하지만 자동적으로 발견되는 에러를 고려한 자동에러탐색요인과 사전 경험에 의하여 세밀하게 에러를 발견하기 위하여 테스팅 관리자가 설정해놓은 요인인 학습효과의 특성에 대한 문제를 비교 제시 하였다. 그 결과 학습요인이 자동 에러 탐색요인보다 큰 경우가 대체적으로 효율적인 모형임을 확인 할 수 있었다. 본 논문의 소프트웨어 고장 자료 분석에서는 고장 간격 시간 자료를 적용하고 모수추정 방법은 최우추정 법을 이용하고 추세분석을 통하여 자료의 효율성을 입증한 후 평균제곱오차와 R^2 (결정계수)를 이용하여 효율적인 모형을 선택 비교하였다

The Comparative Study for NHPP Software Reliability Model based on the Property of Learning Effect of Log Linear Shaped Hazard Function

Kim Hee Cheul* · Shin Hyun Cheul**

ABSTRACT

In this study, software products developed in the course of testing, software managers in the process of testing software and tools for effective learning effects perspective has been studied using the NHPP software.

The log type hazard function applied to distribution was based on finite failure NHPP.

Software error detection techniques known in advance, but influencing factors for considering the errors found automatically and learning factors, by prior experience, to find precisely the error factor setting up the testing manager are presented comparing the problem.

As a result, the learning factor is greater than autonomous errors-detected factor that is generally efficient model could be confirmed. This paper, a failure data analysis of applying using time between failures and parameter estimation using maximum likelihood estimation method, after the efficiency of the data through trend analysis model selection were efficient using the mean square error and R^2(coefficient of determination).

Key words : Learning Effects, Non-Homogeneous Poisson Process, Log Shaped Type Hazard function.

접수일(2012년 5월 1일), 수정일(1차: 2012년 6월 15일), 게재확정일(2012년 6월 29일)

* 남서울대학교 산업경영공학과
** 백석문화대학 인터넷정보학부

1. 서 론

소프트웨어 고장으로 인한 컴퓨터 시스템의 고장은 우리 사회에 엄청난 손실을 유발 할 수 도 있다. 따라서 소프트웨어 개발 과정에서 소프트웨어 신뢰성은 중요한 문제이다. 이 문제는 사용자의 요구조건과 테스팅 비용을 만족시켜야 한다. 소프트웨어 테스팅(디버깅)면에서 비용을 줄이기 위해서는 소프트웨어의 신뢰성의 변동과 테스팅 비용을 사전에 알고 있어야 효율적이다. 따라서 신뢰도, 비용 및 방출 시간의 고려사항을 가진 소프트웨어 개발 과정은 필수 불가결 하다. 결국 소프트웨어 제품의 결함내용을 예측하기 위한 모형 개발이 필요하다. 지금까지 많은 소프트웨어 신뢰성 모형이 제안 되었다. 이 중에서 비동질적 포아송 과정(NHPP; non-homogeneous Poisson process)에 의존한 모형은 에러 탐색 과정측면에서는 우수한 모형이고[1] 이 모형은 결함이 발생하면 즉시 제거되고 디버깅 과정에서 새로운 결함이 발생되지 않는다는 가정을 하고 있다.

Gokhale과Trivedi [1]은 고양된 비동질적인 포아송 과정 모형(enhanced NHPP) 모형을 제시하였고 Goel과 Okumoto [2]은 지수적 소프트웨어 신뢰성 모형(exponential software reliability growth model)을 제안 하였다. 이 모형은 결함의 누적수가 S 형태나 지수적 형태(S-shaped or exponential-shaped)를 가진 평균값 함수(mean value function)를 이용하였다. 이러한 모형에 의존한 일반화 모형은 Yamada 와 Ohba [3]에 의해 지연된 S 형태 신뢰 성장모형(delayed S-shaped reliability growth model)과 변곡된 S 형태 신뢰 성장 모형(inflection S-shaped reliability growth model)이 제안되었다. Zhao [4]는 소프트웨어 신뢰도에서 변환점 문제를 제시하였고 Shyur [5]는 변환점을 이용한 일반화한 신뢰도 성장 모형을 제안하였다. Pham와 Zhang[6]는 테스팅 커버리지(coverage)를 측정하여 소프트웨어 안정도를 평가 할 수 있는 소프트웨어 안정도 모형을 제시했다. 비교적 최근에, Huang [7]은 일반화 로지스틱 테스팅 노력 함수 (generalized logistic testing-effort function)와 변환점 모수(change-point parameter)를 통합하여 효율적인 소프트웨어 신뢰성 예측 기술을 제시하기도 하였다. 그리고 최근에는 S-형태 모형은 소프트웨어 관리자들이 소프트웨어 및 검사 도구에 익숙해지는 학습 과정을 설명할 수 있다고 하였다[8].

또한 국내에서는 김희철과 신현철[15]은 이 분야에서 와이블 분포를 이용하여 영향요인을 분석한 결과 학습요인이 높을수록 효율적인 모형을 확인 할 수 있었다. 그러나 와이블 분포는 위험함수가 비교적 높은 편이다[9]. 이를 해결하기 위한 하나의 방법으로 로그형 모형을 적용하면 보다 위험함수의 패턴을 줄 일 수 있다.

따라서 본 연구는 로그형 위험함수 모형을 이용한 NHPP 소프트웨어 모형에 대하여 자동적으로 발견되는 에러를 고려한 자동에러 탐색요인(autonomous errors-detected factor) 과 사전에 설정되는 학습요인(learning factor)으로 구성된 영향요인(influential factor)의 특성에 대한 문제를 비교 제시 하였다.

2. 관련 연구

2.1 유한고장 NHPP 모형

신뢰도에서 관측시간 $(0, t]$ 사이에 발견된 고장 수 $N(t)$을 모형화 하는데 비동질적 포아송 과정이 널리 사용하여 왔다.

이 과정(process)에서 강도함수(intensity function) 혹은 고장 발생률(ROCOF; rate of occurrence of failure) $\lambda(t) = dE[N(t)]/dt$ 은 t 에 대한 단조(monotonic) 함수로 흔히 가정 한다[3]. 이 범주에서 지금까지 알려진 모형들은 Goel-Okumoto 모형, Weibull 모형 그리고 Cox-Lewis 모형 등이 있는데 이 모형들에 대한 강도함수는 각각의 시간에 의존한 함수, 멱(power) 함수, 대수 선형(log-linear) 함수를 가정하였다[9].

NHPP 모형에서 평균값 함수 $m(t)$ (mean value function)와 강도 함수 $\lambda(t)$ 는 다음과 같은 관계로 표현할 수 있다.

$$m(t) = \int_0^t \lambda(s)ds, \quad \frac{dm(t)}{dt} = \lambda(t) \qquad (1)$$

$N(t)$는 모수 $m(t)$을 가진 포아송 확률밀도함수 (probability density function)로 알려져 있다. 즉,

$$P\{N(t)=n\}=\frac{[m(t)]^n \cdot e^{-m(t)}}{n!},\ n=0,1,2,...\infty \quad (2)$$

이처럼 시간관련 모형(time domain models)들은 NHPP에 의해서 확률 고장 과정으로 설명이 가능하다. 이러한 모형들은 고장 강도 함수 $\lambda(t)$가 다르게 표현됨으로서 평균값 함수 $m(t)$도 역시 다르게 나타난다. 이러한 NHPP 모형들은 유한 고장 모형(finite failure)과 무한 고장(infinite failure) 범주로 분류한다[10]. 유한 고장 NHPP 모형들은 충분한 테스트 시간이 주어지면 결함들(faults)의 기대 값이 유한 값 $\left(\lim_{t\to\infty} m(t)=\theta<\infty\right)$을 가지고 반면에 무한 고장 NHPP 모형들은 무한 값을 가진다고 가정 된다. 유한 고장 NHPP 모형에서 충분한 테스트 시간이 주어졌을 때 탐색되어 질 수 있는 결함의 기대 값을 θ라고 표현하고 $F(t)$를 분포함수라고 표현하면 유한 고장 NHPP모형의 평균값 함수는 다음과 같이 표현 할 수 있다[9][10].

$$m(t)=\theta F(t) \quad (3)$$

(3)식으로 부터 순간고장 강도함수(instantaneous failure intensity) $\lambda(t)$는 다음과 같이 유도된다.

$$\lambda(t)=\theta F'(t) \quad (4)$$

(4)식을 다음과 같이 변형하여 표기 할 수도 있다.

$$\lambda(t)=[\theta-m(t)]\frac{F'(t)}{1-F(t)}=[\theta-m(t)]\,h(t) \quad (5)$$

단, $h(t)=\frac{F'(t)}{1-F(t)}$ 는 위험함수(hazard function, 고장률 함수)로서 소프트웨어 결함 당 고장 발생률을 의미하고 $[\theta-m(t)]$은 t 시점에서 소프트웨어에 남아있는 결함들의 기대값을 나타낸다. $[\theta-m(t)]$의 값은 시점 t에 대한 단조 비증가 함수(monotonically non-increasing function)가 된다. 즉, 시간이 지남에 따라 그 고장을 찾아 제거하는 디버깅 과정을 거치면서 제거되기 때문에 감소 성을 가진다. $\lambda(t)$는 $h(t)$의 속성에 따라 달라지며 그 추세는 상수나 증가, 혹은 감소(증가)하다가 증가(감소)하는 패턴을 가질 수도 있다.

시간 $(0,t]$까지 조사하기 위한 시간 절단(time truncated)모형은 n번째 까지 고장시점 자료를

$$x_k=\sum_{i=1}^{k} t_k\ (k=1,2,\cdots,n;\ 0\le x_1\le x_2\le\cdots\le x_n) \quad (6)$$

이라고 하면 데이터 집합 D_t는 $\{n,x_1,x_2\cdots x_n;t\}$와 같이 구성된다. n번째까지 고장시점이 관찰된 고장 절단 모형일 경우에 데이터 집합 D_{x_n}은 $\{x_1,x_2\cdots x_n\}$으로 구성되며 이 시간 절단 모형에서의 Θ를 모수공간이라고 표시하면 우도함수는 다음과 같이 알려져 있다[9, 10].

$$L_{NHPP}(\Theta|\underline{x})=\left(\prod_{i=1}^{n}\lambda(x_i)\right)\exp(-m(x_n)) \quad (7)$$

단, $\underline{x}=(x_1,x_2,x_3,\cdots,x_n]$

이 분야의 기본적 모형인 Goel-Okumoto 모형은 $h(t)$가 정수 패턴을 가짐으로서 시점 t에 독립이고 잘 알려진 Yamada, Ohba-Osaki 모형은 단조 비감소 패턴을 가진다[1].

NHPP 모형에서 테스트 시점 x_n에서 소프트웨어 고장이 일어난다고 하는 가정 하에서 신뢰구간 $(x_n, x_n+t]$(단, t는 임무시간(mission time))사이에서 소프트웨어의 고장이 일어나지 않을 확률인 신뢰도(reliability) $\hat{R}(t\mid x_n)$는 다음과 같이 됨이 알려져 있다[9].

$$\hat{R}(t\mid x_n)=e^{-\int_{x_n}^{x_n+t}\lambda(\tau)d\tau}=\exp[-\{m(t+x_n)-m(x_n)\}] \quad (8)$$

2.2 학습효과를 고려한 강도함수와 누적함수

소프트웨어 테스팅 작업에 있어서 학습효과는 테스팅 관리자에 의해 동일한 혹은 조작 가능한 작업이 될 수 있으므로 이러한 효과들을 어떠한 방법으로 반영하는가는 소프트웨어 신뢰성에 중요한 과정이 된다.

소프트웨어 에러들을 발견하기 위하여 자동 에러 탐색요인γ 과 학습요인 η이 포함된 영향요인들이 고려될 수 있다. 따라서 $f(t)$을 t 시점에서 에러를 발견된 확률을 나타내는 확률밀도함수이고 $F(t)$ 을 $(0,t]$ 시점까지의 누적분포함수라고 가정하면 영향요인들을 고려한 모형은 다음과 같이 표현 된다[8].

$$f(t)=(\gamma+\eta F(t))\,(1-F(t)) \tag{9}$$

단, $\gamma,\eta>0,\ t>0$.

(9)식에서 자동 에러 탐색요인 γ 는 사전에 알지 못하지만 테스팅 과정에서 테스팅 관리자가 자동적으로 에러를 발견하는 요인이지만 학습 요인 η 은 과거에 발견된 에러패턴을 바탕으로 세밀하게 에러를 발견하기 위하여 테스팅 관리자가 설정해 놓은 요인을 의미한다.

한편 (9)을 다음과 같이 위험함수 형태로 변경 할 수 있다.

$$h(t)=(\gamma+\eta F(t)) \tag{10}$$

단, $h(t)=\dfrac{F'(t)}{1-F(t)}=\dfrac{f(t)}{1-F(t)}$

(10)식에서 누적분포함수와 확률밀도함수는 다음과 같이 수정 가능하다.

$$F(t)=\frac{h(t)-\gamma}{\eta},\ f(t)=F'(t)=\frac{h'(t)}{\eta} \tag{11}$$

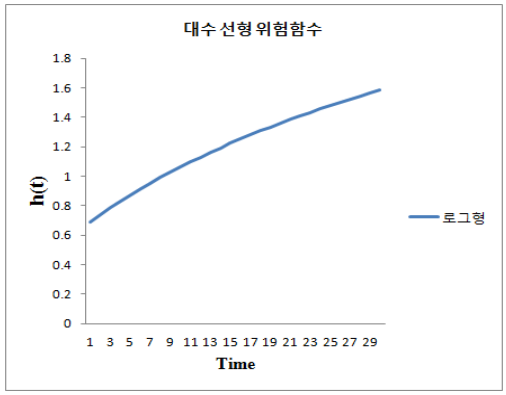
3. 제안한 대수 선형 위험 함수 학습 효과를 이용한 NHPP

대수 선형 위험함수 다음과 같이 정의 된다[9].

$$h(t)=\ln(a+bt) \tag{12}$$

단, $a,b>0$

위 위험함수의 패턴은 (그림 1)에서 보여 주듯이 위험함수가 시간에 따라 상승하는 패턴을 가지고 있다.



(그림 1) 위험함수

따라서 학습효과를 고려한 분포함수는 (11)식을 이용하면 로그형 분포함수는 다음과 같이 유도 된다.

$$F(t)=\frac{h(t)-\gamma}{\eta}=\frac{\ln(a+bt)}{\eta}-\frac{\gamma}{\eta} \tag{13}$$

결과적으로 학습효과를 고려한 확률밀도함수는 각각 다음과 같이 유도 된다.

$$f(t)=F'(t)=\frac{h'(t)}{\eta}=\frac{b}{\eta\,(a+bt)} \tag{14}$$

따라서 (3)식과 (5)식을 이용하면 학습효과를 고려한 유한 고장 NHPP 모형의 로그형 평균값 함수와 강도함수는 다음과 같이 표현할 수 있다.

$$m(t)=\theta F(t)=\theta\left(\frac{\ln(a+bt)}{\eta}-\frac{\gamma}{\eta}\right) \tag{15}$$

$$\lambda(t)=\theta F'(t)=\theta\frac{b}{\eta\,(a+bt)} \tag{16}$$

이 경우의 우도함수는 (7)식에 (15)식과 (16)식을 대입하면 다음과 같다.

$$L_{NHPP}(\Theta \mid \underline{x}) = \prod_{i=1}^{n}\left(\frac{\theta b}{\eta (a+bx_i)}\right)\exp\left[-\theta\left(\frac{\ln(a+b x_n)-\gamma}{\eta}\right)\right] \quad (17)$$

단, $\underline{x} = (x_1, x_2, x_3, \cdots, x_n]$

모수 추정방법은 최우추정법(MLE: maximum likelihood estimation)을 사용하였고 최우추정법을 이용하기 위한 로그 우도함수는 (18)식과 관련하여 다음과 같이 유도된다.

$$\ln L_{NHPP}(\Theta \mid \underline{x}) = n\ln\theta + n\ln b - n\ln\eta - \sum_{i=1}^{n}\ln(a+bx_i) - \theta\left(\frac{\ln(a+b x_n)-\gamma}{\eta}\right) \quad (19)$$

로그형 모형에서 특성을 유지하면서도 보다 간결하게 하기 위하여 (19)식에서 형상모수 $a=0.5$으로 가정한 모형을 사용하고자 한다. 즉, θ와 b에 대하여 편미분하여 다음과 같은 식을 만족하는 $\hat{\theta}_{MLE}$ 와 $\hat{b}_{MLE}$을 수치 해석적 방법으로 추정할 수 있다[9].

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \theta} = \frac{n}{\theta} - \left(\frac{\ln(a+b x_n)-\gamma}{\eta}\right) = 0 \quad (20)$$

$$\text{즉,} \quad \hat{\theta} = \frac{n\eta}{\ln(a+b x_n)-\gamma}$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial b} = \frac{n}{b} - \sum_{i=1}^{n}\frac{x_i}{a+bx_i} - \frac{\theta\, x_n}{\eta(a+b x_n)} = 0 \quad (21)$$

그리고 (8)식을 이용한 신뢰도는 각각 다음과 같다.

$$\hat{R}_{\ln}(t \mid x_n) = \exp\left[-\left(\frac{\theta}{\eta}(\ln(a+b(x_n+t))-\gamma)\right)+\left(\frac{\theta}{\eta}(\ln(a+b x_n)-\gamma)\right)\right] \quad (22)$$

4. 관측 자료에 대한 모형 비교

최근에 모형에 대한 효율성을 조사하기 위한 기준으로서 MSE(평균제곱오차)와 R^2(결정계수)를 사용한다[8].

4.1 평균제곱오차

평균제곱오차는 실제 관찰 값과 예측 값에 대한 차이를 측정하는 도구로서 다음과 같이 정의 된다.

$$MSE = \frac{\sum_{i=1}^{n}\left(m(x_i)-\hat{m}(x_i)\right)^2}{n-k}$$

단, $m(x_i)$은 시간$(0, x_i]$까지 나타난 에러들의 누적함수를 의미하고 $\hat{m}(x_i)$는 x_i 시점까지 평균값 함수로부터 추정된 에러의 누적개수를 의미한다. 그리고 n은 관찰 값의 수이고 k는 모수의 수를 의미한다.

4.2 결정계수

결정계수는 관찰 값의 차이에 대한 설명력을 나타내는 도구로서 다음과 같이 정의 된다.

$$Rsq = 1 - \frac{\sum_{i=1}^{n}\left(m(x_i)-\hat{m}(x_i)\right)^2}{\sum_{i=1}^{n}\left(m(x_i)-\sum_{j=1}^{n}m(x_j)/n\right)^2}$$

5. 소프트웨어 고장 자료 분석

이 장에서 소프트웨어 고장 간격 시간 자료[12](failure interval time data)를 가지고 제시하는 신뢰모형들을 분석하고자 한다. 이 자료의 고장 시간은 18.735 시간단위에 30번의 고장이 발생된 자료이며 <표 1>에 나열 되어 있다 이 고장간격자료의 기초통계량은 <표 2> 에 요약되었다. 이 표에서 왜도와 첨도는 각각 양수로 나타나 정규분포와 비교하여 좀 더

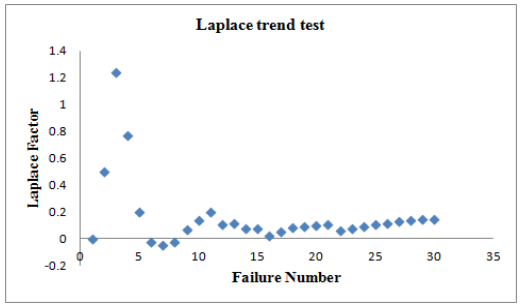
뾰족하고 오른 쪽 꼬리를 가지는 분포 속성을 가지고 있다.

〈표 1〉 고장 자료

Failure Number	Failure Interval (second)	Failure Time (second)
1	0.479	0.479
2	0.266	0.745
3	0.277	1.022
4	0.554	1.576
5	1.034	2.610
6	0.949	3.559
7	0.693	4.252
8	0.597	4.849
9	0.117	4.966
10	0.170	5.136
11	0.117	5.253
12	1.274	6.527
13	0.469	6.996
14	1.174	8.170
15	0.693	8.863
16	1.908	10.771
17	0.135	10.906
18	0.277	11.183
19	0.596	11.779
20	0.757	12.536
21	0.437	12.973
22	2.230	15.203
23	0.437	15.640
24	0.340	15.980
25	0.405	16.385
26	0.575	16.96
27	0.277	17.237
28	0.363	17.600
29	0.522	18.122
30	0.613	18.735

<표 2> 고장 간격 시간자료의 기술통계량

기술 통계량	
평균	0.6245
중앙값	0.5005
첨도	3.8572
왜도	1.8807
범위	2.113
관측 수	30
신뢰 구간(95.0%)	(0.4406, 0.8084)



(그림 2) 라플라스 추세 검정

또한 제시하는 신뢰모형들을 분석하기 위하여 우선 자료에 대한 추세검정이 선행 되어야 한다[13].

추세분석에는 일반적으로 라플라스 추세 검정(Laplace trend test)을 사용한다. 이 검정을 실시한 결과 (그림 3)에서 라플라스 추세 검정의 결과는 라플라스 요인(factor)이 -2와 2사이에 존재함으로서 신뢰성장(reliability growth) 속성을 나타내고 있다. 따라서 이 자료를 이용하여 신뢰 성장모형을 제시하는 것이 효율적임을 시사하고 있다[13, 14].

모수 추정은 최우추정법을 이용하고 비선형 방정식의 계산방법은 수치 해석적 기본 방법인 이분법(bisection method)을 사용하였다.

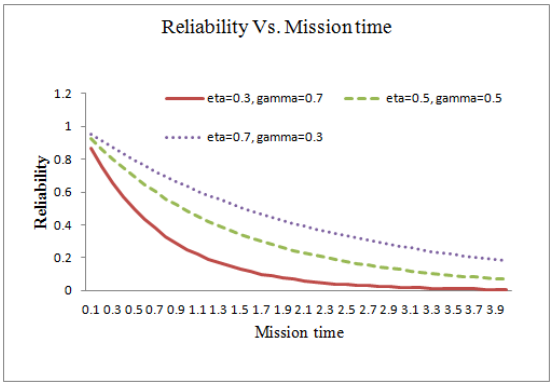
〈표 3> 각 모형의 모수 추정값 및 MSE와 R^2

영향 요인		로그 형		
η	γ	MLE	MSE	R^2
0.1	0.9	$\hat{b}_{MLE}=0.1044$ $\hat{\theta}_{MLE}=5.7799$	52024.3	0.833
0.2	0.8	$\hat{b}_{MLE}=0.0921$ $\hat{\theta}_{MLE}=5.9127$	11090.8	0.834
0.3	0.7	$\hat{b}_{MLE}=0.0808$ $\hat{\theta}_{MLE}=6.0568$	4149.9	0.835
0.4	0.6	$\hat{b}_{MLE}=0.0706$ $\hat{\theta}_{MLE}=6.2136$	1385.13	0.836
0.5	0.5	$\hat{b}_{MLE}=0.0613$ $\hat{\theta}_{MLE}=6.3854$	1010.83	0.837
0.6	0.4	$\hat{b}_{MLE}=0.0529$ $\hat{\theta}_{MLE}=6.5761$	622.692	0.837
0.7	0.3	$\hat{b}_{MLE}=0.0454$ $\hat{\theta}_{MLE}=6.7900$	363.257	0.838
0.8	0.2	$\hat{b}_{MLE}=0.0385$ $\hat{\theta}_{MLE}=7.0331$	216.091	0.840
0.9	0.1	$\hat{b}_{MLE}=0.0323$ $\hat{\theta}_{MLE}=7.3154$	**160.267**	**0.841**

단, Influential factors :η, γ ; MLE : Maximum likelihood

estimation; *MSE*: Mean square error; R^2 : Coefficient of determination

모수추정을 용이하게 하기 위하여 원 자료를 변수변환($failure\ time \times 10^2$)하여 사용하였다. 이러한 계산은 초기 값을 0와 10을, 허용 한계(tolerance for width of interval)는 10^{-5} 을 주고 수렴 성을 확인하면서 충분한 반복 횟수인 100번을 C-언어를 이용하여 모수 추정을 수행하였다. 그 결과는 <표 3>에 요약되었고 또한 학습요인 η과 자동 에러 탐색요인 γ은 각각 0.1부터 0.9까지 고려한 MSE와 R^2의 값도 이 표에서 학습요인 η이 증가 할수록 효율적인 모형으로 나타나고 있다. 그리고 설명력 측면에서도 유사하게 결정 계수 값이 0.8보다 크게 나타나 실제 값과 추정된 값에 대한 차이의 대한 설명력이 높고 역시 학습요인 η이 증가 할수록 설명력이 높게 나타나고 있다. (그림 3)의 신뢰도 그림에서 보여 주듯이 임무시간이 지남에 따라 완만히 감소하는 형태를 보여 주지만 학습요인 η의 값이 클수록 높은 신뢰도를 보이고 있다.



단 η (eta): learning factor,
γ (gamma): autonomous errors-detected factor

(그림 3) 임무시간에 대한 각 경우의 신뢰도

결과적으로 자동탐색요인 γ보다 큰 경우가 대체적으로 효율적인 모형임을 확인 할 수 있어서 이 분야에서 학습요인을 고려한 모형이 효율적 모형으로 선택 될 수 있음을 보여주고 있다.

6. 결 론

대용량 소프트웨어가 수정과 변경하는 과정에서 결점의 발생을 거의 피할 수 없는 상황이 현실이다. 따라서 소프트웨어 관리자들이 소프트웨어 및 검사 도구에 효율적인 학습 과정을 이용한 NHPP 소프트웨어 모형에 대하여 연구 하였다. 사전에 알지 못하지만 자동적으로 발견되는 에러를 고려한 영향요인과 사전 경험에 의하여 세밀하게 에러를 발견하기 위하여 테스팅 관리자가 설정해놓은 요인인 학습효과의 특성에 대한 문제를 비교 제시 하였다.

따라서 학습요인이 자동 에러 탐색요인보다 큰 경우가 대체적으로 효율적인 모형임을 확인 할 수 있어서 이 분야에서 효율적 모형으로 선택 될 수 있음을 보여주고 있다.

경우에 따라서는 왜도와 첨도 측면에서 효율적인 카파분포, 지수화지수분포 등 업데이트된 분포에 대한 적용 문제를 비교 분석하는 연구도 가치 있는 일이라 판단되고 이 연구를 통하여 소프트웨어 개발자들은 위험함수에 의한 학습요인을 파악 하는데 어느 정도 도움을 줄 수 있으리라 사료 된다.

참고문헌

[1] Gokhale, S. S. and Trivedi, K. S. “A time/structure based software reliability model”, Annals of Software Engineering. 8, pp. 85-121. 1999.

[2] Goel AL, Okumoto K, “ Time-dependent fault detection rate model for software and other performance measures”, IEEE Trans Reliab 28, pp.206-11, 1978.

[3] Yamada S, Ohba H. “ S-shaped software reliability modeling for software error detection", IEEE Trans Reliab, 32, pp.475-484, 1983.

[4] Zhao M. "Change-point problems in software and hardware reliability”, Commun. Stat Theory Methods, 22(3), pp.757-768, 1993.

[5] Shyur H-J. "A stochastic software reliability model with imperfect debugging and change-point", J Syst. Software 66, pp.135-141, 2003.
[6] Pham H, Zhang X. "NHPP software reliability and cost models with testing coverage", Eur J Oper Res, 145, pp.445-454, 2003.
[7] Huang C-Y. "Performance analysis of software reliability growth models with testing-effort and change-point". J Syst Software 76, pp. 181-194, 2005.
[8] Kuei-Chen, C., Yeu-Shiang, H., and Tzai-Zang, L. "A study of software reliability growth from the perspective of learning effects". Reliability Engineering and System Safety 93, pp. 1410 - .1421, 2008.
[9] J. F. Lawless. Statistical Models and Methods for Lifetime Data. John Wiley & Sons, New York, 1981.
[10] L. Kuo and T. Y. Yang."Bayesian Computation of Software Reliability".Journal of the American Statistical Association, Vol.91, pp. 763-773, 1996.
[11] Alaa Sheta, "Parameter Estimation of Software Reliability Growth Models by Particle Swarm Optimization", AIML Journal, Volume (7), Issue (1), pp. 55-61, June, 2007.
[12] Y. HAYAKAWA and G. TELFAR "Mixed Poisson-Type Processes with Application in Software Reliability", Mathematical and Computer Modelling, 31, pp. 151-156, 2000.
[13] K. Kanoun and J. C. Laprie, "Handbook of Software Reliability Engineering", M.R.Lyu, Editor, chapter Trend Analysis. McGraw-Hill New York, NY: 1996; p.401-437.
[14] Hee-Cheul Kim and Hyoung-Keun Park. "The Comparative Study for ENHPP Software Reliability Growth Model Based on Mixture Coverage Function". Communications in Computer and Information Science, Springer-Verlag Berlin, Heidelberg. pp. 187-194, 2011
[15] 김희철, 신현철"학습효과 기법을 이용한 NHPP 소프트웨어 신뢰도 모형에 관한 연구", 정보, 보안 논문지, 제 11권3호, pp. 26-32, 2011년 3월

[저 자 소 개]

김 희 철 (Hee-cheul Kim)

1992년 2월 동국대학교 통계학과 졸업(이학석사)
1998년 8월 동국대학교 통계학과 졸업(이학박사)

email : kim1458@nsu.ac.kr

신 현 철 (Hyun-cheul Shin)

1990년 2월 광운대학교 전자계산학과 졸업(공학석사)
2002년 2월 원광대학교 컴퓨터공학과 졸업(공학박사)

email : hcshin@bcc.ac.kr