민감도 분석을 이용한 소프트웨어 최적방출시기에 관한 연구

신 현 철*

요 약

소프트웨어 제품을 개발하여 테스팅을 거친 후 사용자에게 인도하는 시기를 결정하는 일은 아주 실제적이고 흥미 있는 일이 된다. 이러한 문제를 소프트웨어 최적 방출시기라고 한다. 본 연구에서는 이러한 방출문제에 대하여 연구되었다. 수명분포는 감마족 분포에서 대표적인 어랑 문포 모형을 이용한 최적 방출시기에 관한 문제를 다루었다. 소프트웨어 요구 신뢰도를 만족시키고 소프트웨어 개발 및 유지 총비용을 최소화 시키는 최적 소프트웨어 방출 정책에 대하여 논의 되었고 민감도 분석을 이용하여 효율적 최적 방출시기가 논의되었다.

The Study of Software Optimal Release Using Sensitivity Analysis

Hyun-Cheul Shin*

ABSTRACT

It is of great practical interest to decide when to stop testing a software system in development phase and transfer it to the user. This decision problem called an optimal release policies. In this paper discussed to specify an optimal release policies. In this paper, propose an optimal release policies of the life distribution applied Erlang distribution of special pattern of Gamma distribution. In this paper, discuss optimal software release policies which minimize a total average software cost of development and maintenance under the constraint of satisfying a software reliability requirement. From Sensitivity Analysis, make out estimating software optimal release time.

Key words : Software Release Policies, Total Average Cost, Sensitivity Analysis

* 백석문화대학 컴퓨터정보학부

1. 서 론

소프트웨어 신뢰성은 컴퓨터 시스템에 대한 적용과 이에 대한 연구 분야에서 중요한 역할을 담당해 오고 있다. 소프트웨어 고장으로 인한 컴퓨터 시스템의 고장은 우리 사회에 엄청난 손실을 유발 할 수 도 있다. 따라서 소프트웨어 신뢰도는 현대의 소프트웨어 생산품 개발에서 중요한 문제 가운데 하나이다.

소프트웨어 신뢰도 엔지니어링에서의 연구 활동은 지난 30년 전부터 행해져 오고 있고 많은 신뢰도 성장 모형들이 소프트웨어에 남아 있는 고장들의 수와 소프트웨어 신뢰도의 추정에 관한 문제들을 제안해 왔다. 일반적으로 소프트웨어 개발 과정은 설계단계, 디자인, 코딩 그리고 테스팅 단계를 거친다. 이러한 과정을 거친 후 소프트웨어 제품을 방출하게 되는데 방출이후에 발견되지 않은 고장들이 나타난다면 이것들에 대한 보전 비용(Maintenance cost)은 크게 증가 할 것이다. 결국, 소프트웨어 시스템 시험을 끝내고 그것을 사용자에게 넘기는 시기 결정은 매우 중요한 사항이 된다. 이러한 소프트웨어 방출시간에 대한 연구들은 대부분 유한 고장 NHPP(Non-Homogeneous Poisson Process)모형을 사용하였다[1-4]. 이러한 유한(finite) 고장 NHPP 모형은 소프트웨어가 유한개의 고장이 있고 고장 제거 단계에서는 새로운 고장이 발생하지 않는다는 가정을 한 모형이다. 그러나 실제 고장 제거 단계에서도 새로운 고장이 발생 할 수 있다. 따라서 본 연구에서는 무한(infinite) 고장 NHPP 모형을 이용하여 최적 방출시기에 대한 문제를 제안하고자 한다. 이 분야에서는 Musa-Okumoto의 대수 포아송 실행시간 모형[5]과 Log-power 모형[6]을 이용한 방출 문제에 대한 문제 들이 이미 연구되었고 최근에도 이와 관련된 문제에 대한 연구는 Yang and Xie(2000)와 Huang(2005)에 의해 연구되고 있다[7, 8]. 본 연구에서는 일반적 감마분포 특수한 경우인 어랑분포[9, 10]을 적용한 유한고장 NHPP 모형을 이용하여 최적 방출시기에 관한 문제를 다루고자 한다. 본 논문의 제 2장에서는 관련 연구로서 무한고장과 기록값 통계량, 어랑 분포[19] 및 모수추정에 대하여 약술하였고 제 3장에서는 요구 신뢰도와 비용 최소화를 고려한 방출시간에 대하여 서술하고 제 4장과 제 5장에서는 각각 민감도 분석 결과와 그 결론을 나열 하였다.

2. 무한 고장 NHPP와 어랑 분포 모형

유한(finite) 고장 NHPP(Non-Homogeneous Poisson Process)모형은 소프트웨어가 유한개의 고장이 있고 고장 제거 단계에서는 새로운 고장이 발생하지 않는다는 가정을 한 모형이다. 그러나 실제 고장 제거 단계에서도 새로운 고장이 발생 할 수 있다. 이러한 상황을 해소시켜 주는 모형이 무한(infinite) 고장 NHPP 모형이라고 하고 유한 고장 NHPP 모형의 평균값 함수와 고장 강도 함수는 각각 다음과 같이 알려져 있다[3].

$$m(t) = -\ln(1-F(t)) \tag{2.1}$$

$$\lambda(t) = m'(t) = f(t)/(1-F(t)) \tag{2.2}$$

수명 분포가 감마분포 $\Gamma(\alpha, \beta)$(단, $\alpha>0$)일 경우 확률 밀도 함수는 다음과 같다.

$$f(t) = \frac{\beta(\beta t)^{\alpha-1}}{\Gamma(\alpha)} exp(-\beta t) \tag{2.3}$$

위 식 (2.3)에서 본 논문에서 사용하고자 하는 어랑분포는 $\alpha=2$인 특수한 경우로 확률밀도함수는 다음과 같다.

$$f(t) = \beta^2 t \exp^{-\beta t} \tag{2.4}$$

따라서 식 (2.1)과 식 (2.2)와 연관하면 다음과 같이 표현 가능하다.

$$\lambda(t) = \beta_2^2 t/(\beta_2 t+1),\ m(t) = t\ \beta - \ln(1+\beta t) \tag{2.5}$$

3. 신뢰도 및 비용 최소화를 고려한 방출 시간

NHPP 모형에서 테스트 시점 x_n(마지막 고장시점)에서 소프트웨어 고장이 일어난다고 하는 가정하에서 신뢰구간 $(x_n, x_n+x]$ (단, x는 임무시간(Mission time))사이에서 소프트웨어의 고장이 일어나지 않을 확률인 신뢰도 $\hat{R}(x \mid x_n)$는 다음과 같이 됨이 알려져 있다[14, 15, 18].

$$\hat{R}(x \mid x_n) = \exp\left(-\int_{x_n}^{x_n+x} \lambda(\tau)d\tau\right) = \exp[-\{m(x+x_n)-m(x_n)\}] \tag{3.1}$$

어랑 분포 모형[18]에 대한 신뢰도는 식 (2.5)와 식 (3.1)를 이용하고 $t=x_n$라고 하면 다음과 같이 표현된다.

$$R(x \mid t) = \exp[-(t+x)\beta+t\beta-\ln(1+\beta(x+t)) +\ln(1+\beta t)] \tag{3.2}$$

따라서 소프트웨어 방출시간 T_R이 신뢰도 $R_0 = \hat{R}(x \mid t)$을 확보해야 한다면 다음 방정식을 만족해야 한다(방정식의 해).

$$\ln R_{0} = -(T_R+x)\beta+T_R\beta-\ln(1+\beta(x+T_R) +\ln(1+\beta T_R) \tag{3.3}$$

비용 최소화와 관련된 최적 방출시간은 신뢰도와 함께 비용모형에 의해서 결정된다. 소프트웨어 방출시간을 T로 표현하고 $m(T)$와 $m(\infty)$을 각 각 $(0, T)$와 $(0, \infty)$의 기간에 발견된 기대 고장수라고 표현하고 $C(T)$을 소프트웨어 라이프사이클(life cycle) 동안에 기대되는 소프트웨어 비용이라고 하면 $C(T)$는 다음과 같이 표현된다[5, 6].

$$C(T) = c_1 m(T)+c_2[m(\infty)-m(T)]+c_3 T \tag{3.4}$$

위 식에서 c_1는 테스팅 동안에 하나의 고장을 수리하는 비용이고 c_2는 가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$), 그리고 c_3는 단위 시간당 테스팅 비용을 나타낸다. 이와 관련하여 총비용의 최소화는 무한고장 평균값 함수를 가진 NHPP 모형에 대하여 발생 할 수 있다. 무한 수명에 대한 비용함수 $C(T)$ 식 (3.4)에서 $m(\infty)$은 직접 구체화 되어 있지 않기 때문에 이 식을 사용하기 위해서는 소프트웨어 수명시간 인 T_{LC}을 지정하여 분석한다[6]. 이러한 T_{LC}는 소프트웨어마다 서로 다른 임의의 값이기 때문에 유한 고장 NHPP 모형이라고 할 수는 없다.

따라서 비용함수를 고려하여 소프트웨어의 모든 수명에서 총비용을 최소화함으로서 최적 테스팅 시간을 결정 할 수 있고 다음과 같은 식을 만족하면 비용함수 $C(T)$는 유일한 최소값을 가진다[5].

$$\frac{dC(T)}{dT}=0, \quad \frac{d^2C(T)}{d^2T}>0 \tag{3.5}$$

결국 소프트웨어 지정된 수명 T_{LC}을 이용한 어랑 분포 모형 평균 비용 $C(T)$는 다음과 같이 유도된다.

$$\begin{aligned} C(T) &= c_1 m(T)+c_2[m(\infty)-m(T)]+c_3 T \\ &= (c_1-c_2)(T\beta-\ln(1+\beta T)) \\ &\quad +c_2(T_{LC}\beta-\ln(1+\beta T_{LC})+c_3 T \end{aligned} \tag{3.6}$$

T에 관해서 비용함수 $C(T)$을 미분하면 다음과 같은 방정식을 만족하는 방출시간 T_C를 계산 할 수 있다.

$$(c_1-c_2)\left(\beta-\frac{\beta}{1+\beta T_C}\right)+c_3=0 \tag{3.7}$$

위 식에서 방출시간은 소프트웨어 지정 수명시간인 T_{LC}와 의존하지 않는다는 것을 알 수 있다. 이러한 사실은 무한 고장 평균값 함수를 가진 NHPP 모형들이 새로운 결점들이 발생함으로서 몇 개의 고장이 야기 될 수 있는 점을 고려한 모형으로 적

합 시킬 수 있다.

실제로 만족할 만한 신뢰도가 부여되고 동시에 시스템 고장과 연계된 기대 총 비용을 최소화시키기 위하여 필요하다면 충분한 테스팅을 계속해야 한다. 따라서 신뢰성 요구를 만족하고 총 비용을 최소화하는 상황이 최적 방출 시간이다. 따라서 어랑 분포 모형을 사용한 최적 방출시간 T_{OP}는 T_R과 T_C에 대하여 다음을 만족한다[6].

$$T_{OP} = Max(T_C, T_R) \tag{3.8}$$

식 (3.8)에서 T_R과 T_C는 다음 두 방정식에 의해서 계산 될 수 있다.

$$\ln R_0 = -(T_R + x)\beta + T_R\beta - \ln(1+\beta(x+T_R)) + \ln(1+\beta T_R) \tag{3.9}$$

$$(c_1 - c_2)\left(\beta - \frac{\beta}{1+\beta T_C}\right) + c_3 = 0 \tag{3.10}$$

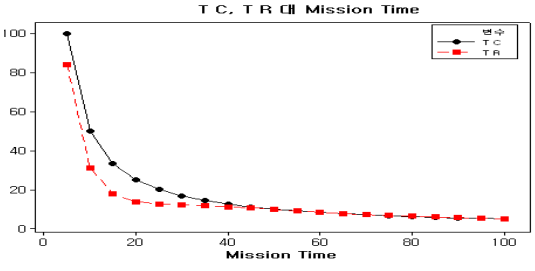
단, 신뢰도 $R_0 = \hat{R}(x \mid t)$

4. 민감도 분석

소프트웨어 신뢰성은 개발의 최종단계에 있는 테스트 공정이나 실제 사용단계에 있어서 소프트웨어 내에 존재하는 고장 수나 고장 발생 시간에 의해서 효과적으로 평가할 수 있는 상황으로 그 평가 기술이 중요하게 된다. 실제로 만족할 만한 신뢰도가 부여되어야 소프트웨어 방출된다. 따라서 최적방출시기는 신뢰도와 비용에 만감한 반응을 보인다, <표 1>은 c_1(테스팅 동안에 하나의 고장을 수리하는 비용) c_2(가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$)), c_3(단위 시간당 테스팅 비용)을 각각 증가추세로 놓고 또한 모형의 모수인 β도 증가 추세를 하는 경우에 최적방출시기(T_{op}) 및 신뢰도 관련 방출시기(T_R), 비용관련 방출시기(T_C)의 민감성을 분석하고자 한다[20]. <표 1>에서 신뢰도를 95%로 일 경우보다 90%인 경우가 소프트 최적방출시기는 짧아짐을 알 수 있다. (그림 1)에서는 임수시간에 다 T_R과 T_C는 둘 다 급격한 감소추세를 보이고 있으며 임무시간이 55시점 까지는 T_C가 최적 방출시기가 되고 60시점 부터는 T_R이 최적 방출시기가 된다. 모수값이 증가하면 증가 할수록 최적방출시기는 짧아지고 c_3가 큰 값에 대하여는 T_C가 민감한 반응을 보이고 모수 β은 T_R에 민감한 반응을 보이고 있다.

〈표1〉 95% 신뢰도를 부여한 최적방출시기(T_{op})

Mission Time	c_1	c_2	c_3	β	T_C	T_R	T_{op}
5	10.00	100	1000	0.01	99.91	53.10	99.01
10	20.0	200	2000	0.02	49.91	15.00	49.91
15	30.00	300	3000	0.03	33.24	10.29	33.24
20	40.00	400	4000	0.04	24.91	10.10	24.91
25	50.00	500	5000	0.05	19.91	10.65	19.91
30	60.0	600	6000	0.06	16.58	11.08	16.58
35	70.00	700	7000	0.07	14.20	11.16	14.20
40	80.00	800	8000	0.08	12.41	10.89	12.41
45	90.00	900	9000	0.09	11.02	10.35	11.02
50	100.0	1000	10000	0.10	9.91	9.68	9.91
55	110.00	1100	11000	0.11	9.00	8.97	9.00
60	120.0	1200	12000	0.12	8.24	8.29	8.29
65	130.00	1300	13000	0.13	7.60	7.68	7.68
70	140.00	1400	14000	0.14	7.05	7.14	7.14
75	150.00	1500	15000	0.15	6.58	6.67	6.67
80	160.0	1600	16000	0.16	6.16	6.25	6.25
85	170.00	1700	17000	0.17	5.79	5.88	5.88
90	180.00	1800	18000	0.18	5.47	5.56	5.56
95	190.00	1900	19000	0.19	5.17	5.26	5.26
100	200.0	2000	20000	0.20	4.91	5.00	5.00



(그림1) 임무시간에 따른 방출시간

5. 결 론

본 연구는 감마분포의 특수한 형태인 어랑분포를 적용한 무한고장 NHPP 모형을 이용하여 최적 방출시기에 관한 민감도 문제를 알아보았다. 즉, 대용량 소프트웨어가 수정과 변경하는 과정에서 결점의 발생을 거의 피할 수 없는 상황이 현실이다. 실제로 만족할 만한 신뢰도가 부여되고 동시에 시스템 고장과 연계된 기대 총비용을 최소화시키기 위하여 필요하다면 충분한 테스팅을 계속해야 한다. 따라서 어랑 분포를 적용한 민감도 문제에 있어서는 단위 시간당 테스팅 비용이 비용관련 방출시기에 민감성을 보이고 모수의 값과 신뢰도는 신뢰도 관련 방출시기에 민감성을 보이고 있다. 최적 방출 시간을 조절하기 위해서는 무엇보다도 단위시간당 테스팅 비용과 신뢰도를 적절히 보안하는 점이 최중요하다고 본다. 본 연구에서는 어랑 분포를 적용하였지만 방출시기에 효율적인 카파분포, 지수화지수분포 등 업데이트된 분포에 대한 방출 시기 문제를 비교 분석하는 연구도 가치있는 일이라 판단되고 이 연구를 통하여 소프트웨어 개발자들은 방출최적시기를 파악 하는데 어느 정도 도움을 줄 수 있으리라 사료 된다.

참 고 문 헌

[1] J. F. Lawless, “Statistical Models and Methods for Lifetime Data”, John Wiley and Sons,” New York, 1981.

[2] 김희철, “일반화감마분포를 이용한 NHPP 소프트웨어 신뢰도 모형에 관한 연구”, 한국 컴퓨터정보학회 논문지, 제10권, 제6호, pp. 27-35, 2005.

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

[4] 김희철, “지수화 지수 분포에 의존한 NHPP 소프트웨어 신뢰성장 모형에 관한 연구”, 한국컴퓨터정보학회논문지, 제11권, 제5호, pp. 9-18, 2006.

[5] 김대경, “Musa-Okumoto의 대수 포아송 실행시간 모형에 근거한 비용-신뢰성 최적정책”, 품질경영학회지, 제26권, 제3호, pp. 141-149, 1998.

[6] M. Xie and G. Y. Homg, “Software release time determination based on unbound NHPP model”, Proceeding of the 24th International Conference on Computers and Industrial Engineering. pp. 165-168, 1999.

[7] B. Yang and M. Xie. “A study of operational and testing reliability in software reliability analysis”, RELIABILITY ENGINEERING & SYSTEM SAFETY, Vol, 70, pp, 323-329, 2000.

[8] C. Y. Huang, “Cost-Reliability-optimal release policy for software reliability models incorporating improvements in testing efficiency, The journal of Systems and Software. Vol, 77, pp, 139-155, 2005.

[9] L. Kuo, and T. Y. Yang, “Bayesian Computation of Software Reliability”, Journal of the American Statistical Association, Vol. 91, pp. 763-773, 1996.

[10] 간광연, 이재곤, 김희철, “기록값 통계량 모형에 기초한 NHPP 소프트웨어 신뢰성에 관한 연구”, 한국통신학회논문지, 제30권, 제12T호, pp. 356-363, 2005.

[11] 김희철, 신현철, 김경수, “기록값 통계량에 기초한 무한고장 NHPP 소프트웨어 혼합신뢰성장모형에 관한 연구”, 정보보안논문지, 제7권, 제3호, pp. 51-60, 2007.

[12] N. Glick, “Breaking Records and Breaking Boards”, American Mathematical Monthly,

Vol. 85, pp. 2-26, 1978.

[13] S. L. Resnick, "Extreme Values, Regular Variation, and Point Process", Berlin : Springer-Verlag, 1987.

[14] H. Pham, L. Nordmann, and X. Zhang, "A General Imperfect-Software-Debugging Model with S-Shaped Fault-Detection Rate", IEEE Trans. on reliability, Vol. 48, No. 2, pp. 169-175, 1999.

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

[16] M. R. Lyu, "Handbook of Software Reliability Engineering", McGraw-Hill, New York, NY, pp. 3-25, 1996.

[17] K. Kanoun and J. C. Laprie, "Handbook of Software Reliability Engineering, Lyu, M. R. Editor, chapter Trend Analysis, McGraw-Hill, New York, NY, pp. 401-437, 1996.

[18] 김희철, "유한 고장 NHPP 어랑 소프트웨어 신뢰성 모형에 관한 연구", 남서울대 논문집, 제11권, 제2호, pp. 99-113, 2006.

[19] 김희철, "어랑분포에 근거한 소프트웨어 최적 방출시기에 관한 연구", 남서울대학교 논문집, 제14권, pp. 59-74, 2008.

[20] D. R. Prince Williams, "Study of the Warranty Cost Model for Software Reliability with an Imperfect Debugging Phenomenon", Turk, J. Elec. Engin, Vol. 15, No. 3, pp, 369-381, 2007.

신 현 철

2002년 원광대학교 컴퓨터공학과(공학박사)

1994년～현재 백석문화대학 컴퓨터정보학부 부교수

2004년 (주)아이비루션 자문위원

2005년～현재 한국정보처리학회 이사

한국사이버테러정보전학회 부회장