하드웨어 번인과 소프트웨어 시험
(Hardware Burn-in and Software Testing)

유 영관, 이 종무
한라대학교 경상학부

Abstract

Burn-in is a test procedure to find and eliminate the inherent initial failure of a product during or at the final stage of production process. Software testing is the validation and verification process which is used to cut off the faults from a software. The two have the common function and objective of "debugging". This article summarizes some significant models on the optimal hardware and software burn-in time, and provides the relevant paper lists. The need for the development of the unified burn-in policy of a hardware-software system is addressed.

1. 서론

번인(burn-in)은 하드웨어에 내재된 고장 요인을 생산 초기에 찾아내어 제거하기 위해 행하는 시험 방법으로서 제조업의 생산 현장에서 유용하게 사용되고 있다. 그러나 번인을 충분한 시간 동안 실시하면 잠재된 초기 고장을 제거할 수 있는 확률이 높아지지만, 번인에 소요되는 비용이 증가하는 문제가 있으므로 적절한 번인 시간의 결정이 필요하다.

소프트웨어 시험은 소프트웨어의 개발 과정 중에 생긴 프로그램의 결함(fault)을 찾아내어 제거하기 위한 것으로, 소프트웨어의 신뢰성 확보를 위해 매우 중요한 과정이다. 그러나 시험기간이 길면 많은 결함을 제거할 수는 있지만 시험 비용이 많이 소요되므로 경제적인 소프트웨어 시험기간의 결정이 요구된다. 이것을 "최적 소프트웨어 릴리즈(release) 문제" 또는 "시험(test)문제"라고 한다.

본 논문에서는 주요한 하드웨어와 소프트웨어의 번인비용 최적화 모델을 정리하고, 관련 논문들의 목록을 제공한다. 또한 그 동안 이 두 문제가 독립적으로 다루어져 왔으나, 하드웨어-소프트웨어 시스템에 대한 통합 시험 모델의 필요성을 제시한다.

2. 하드웨어 번인비용 최적화 모형

번인에 관계된 총 비용을 최소화하는 최적 번인 시간의 결정은 그 동안 신뢰성 분석가들의 주요 관심사가 되어 왔다. 최적 번인시간의 결정에 관련된 비용은 일반적으로 다음과 같다.

(1) 번인 고정 비용
번인 시간이나 번인 중 고장 발생 수와는 무관하게 정해지는 준비비용(set-up cost).
(2) 번인 중 고장 비용
번인 중 발생하는 고장을 처리하고 수리하는데 소요되는 비용.
(3) 번인 시간 비용
번인 시간에 비례하여 발생하는 가변비용.
(4) 번인 후 고장 비용
제품이 고객에게 인도된 후 발생하는 보증 수리비용과 고객의 호감 상실 비용.

다음의 <표 1>에는 번인비용 최적화와 관련된 주요 연구가 정리되어 있다.

<표 1> 주요 번인비용 최적화 모형

저자	특징
NGuyen & Murthy (1982)	보증기간이 설정되어 있는 제품에 대해 제조비용, 번인비용, 보증비용의 합이 최소가 되는 최적 번인 시간을 결정.
Chi & Kuo(1989)	번인비용 외에 최소 신뢰도 요구와 번인 용량의 제약조건 부가.
Mi(1994)	번인비용과 보전비용 동시 고려.
Hui & Lu (1996)	(가속)번인비용, 제조비용, 품질 및 신뢰성 비용의 합을 최소화하는 번인시간과 스트레스 수준.
Mi(1997)	Nguyen-Murthy의 확장 모형
Kar & Nachlas (1997)	다양한 번인정책과 보증정책을 동시에 고려.
Yan & English (1997)	잠복(latent) 고장이 있는 새로운 형태의 욕조곡선 고장률 함수의 번인 비용 최적화.

3. 소프트웨어 최적 시험시간 모형

소프트웨어의 최적 시험시간 결정에 관계되는 비용 요소들은 다음과 같이 하드웨어 번인 문제의 경우와 거의 비슷하다.

(1) 시험시간 비용
시험시간에 비례하여 발생하는 비용. CPU 사용비용, 시험인력의 노무비 등.
(2) 시험 중 고장비용
시험 중 발견한 결함 당 수리 비용.
(3) 시험 후 고장 비용
고객에게 릴리즈(release)된 후 발생하는 고장에 대한 수리비와 호감상실 비용.
(4) 납기 벌과 비용
납기에 늦을 때 발생하는 벌과 비용.

다음의 <표 2>에는 최적 릴리즈 문제에 관한 주요 연구가 정리되어 있다.

<표 2> 주요 최적 소프트웨어 릴리즈 모형

저자	특징
Okumoto & Goel(1980)	최적 릴리즈 문제의 선구적 연구. 시험 비용과 시험 후 비용의 합의 최소화.
Koch & Kubat(1983)	Okumoto-Goel의 확장 모형. 비용 세분화.
Brown et al.(1989)	시험시간 대신 시험사례(test case)의 개수를 결정변수로 사용.
Dalal & Mallows (1990)	결함 발견 분포를 모를 경우의 최적 시험시간.
Singpurwalla (1991)	시험시간 결정 문제를 불확실성하의 의사결정 문제로 재구성. 효용(utility)함수를 사용.
Cho & Park (1994)	2단계 시험. 고객에 의한 무고장(falure-free)기간 요구 설정.
Dalal & McIntosh (1994)	시험 중 코드(code)에 변화가 발생한다고 가정.
Yang & Chao (1995)	동일한 오류가 반복되는 경우(시험과 오류 수정의 분리).
Hou et al.(1996)	초기하 소프트웨어 신뢰도 성장모형 사용.
Littlewood Wright(1997)	무고장(failure-free) 시험 사례 개수와 시험 시간.
Kimura et al.(1999)	Non-homogeneous Poisson pro- cess 소프트웨어 신뢰도 모형 사용.

4. 하드웨어-소프트웨어 시스템의 시험

일반적으로 컴퓨터 응용 시스템은 하드웨어와 소프트웨어로 구성되어 있으므로 하드웨어 시험은 필연적으로 소프트웨어 시험을 수반하게 된다. 예를 들어 새로 개발된 레이더 시스템은 전자 장비와 운영 소프트웨어로 이루어져 있으며, 소프트웨어 시험과 함께 하드웨어 시험이 동시에 이루어지게 된다. 소프트웨어 시험 단계 중 단위(unit) 시험에서는 소프트웨어 단독 시험이 실시되기도 하지만, 통합(integration)시험이나 인수(acceptance) 시험에서는 하드웨어와의 동시 시험이 이루어지게 된다. 따라서 이 경우 소프트웨어나 하드웨어에 대한 개별적인 시험시간의 적용 대신 이 두 하부 시스템(sub-system)을 동시에 고려한 시험시간의 결정이 더욱 바람직할 것이다.

소프트웨어-하드웨어 시스템에 대한 신뢰성 분석에 관한 연구는 일부 수행되고 있으나 (Welke et al. 1995, Goel & Soenjoto 1981, Sumita & Masuda 1986, Tokuno & Yamada 2000), 통합 시스템의 시험에 관한 연구는 아직 이루어지지 않고 있다. 소프트웨어 시험과 하드웨어 번인의 개념을 통합한 새로운 시험 방법의 개발이 요구된다.

[참고문헌]

하드웨어의 최적 번인시간

[1] Ascher, H., "Evaluation of repairable system reliability using the bad-as-old concept" , *IEEE Tr. Reliability*, Vol.17, No.2, pp103-110, 1968.
[2] Canfield, R, "Cost effective burn-in and replacement times", *IEEE Tr. Reliability*, Vol.24, No.2, pp154-156, 1975
[3] Chi, D., Kuo, W., "Burn-in optimization under reliability and capacity restrictions" , *IEEE Tr. Reliability*, Vol.38, No.2, pp193-198, 1989.
[4] Chien, W., Kuo, W., "A nonparametric Bayes approach to decide system burn-in time" , *Naval Research Logistics*, Vol.44, pp655-671, 1997.
[5] Costantini, C., Spizzichino, E., "Explicit solution of an optimal stopping problem: The burn-in of conditionally exponential components" , *J. of Applied Probability*, Vol.34, pp267-282, 1997.
[6] Dishon, M., Weiss, G., "Burn-in programs for repairable systems" , *IEEE Tr. Reliability*, Vol.22, No.5, pp265-267, 1973.
[7] Hui, Y., Lu, W., "Cost optimization of accelerated burn-in" , *International J. of Quality & Reliability Management*, Vol.13, No.7, pp69-78, 1996.
[8] Jensen, F., Petersen, N., *Burn-in*, John-Wiley & Sons, New York, 1982.
[9] Kar, T., Nachlas, J., "Coodinated warranty & burn-in strategies" , *IEEE Tr. Reliability*, Vol.46, No.4., pp512-518, 1997.
[10] Kim, T., Kuo, W., "Optimal burn-in decision making" , *Quality & Reliability Engineering International*, Vol.14, pp417-423, 1998.
[11] Kuo, W., "reliability enhancement through optimal burn-in", *IEEE Tr. Reliability*, Vol.33, No.2, pp145-156, 1984.
[12] Leemis, L., Beneke, M., "Burn-in models and methods: A review" , *IIE Transactions*, Vol.22, No.2, pp172-180, 1990.
[13] Mi, J., "Burn-in and maintenance policies" , *Advances in Applied Probability*, Vol.26, pp207-221, 1994.
[14] Mi, J., "Minimizing some cost functions related to both burn-in and field use", *Operations Research*, Vol.44, No.3., pp497-500, 1996.
[15] Mi, J., "Warranty policies and burn-in", *Naval Research Logistics*, Vol.44, pp199-209, 1997.
[16] Nguyen, D., Murthy, D., "Optimal burn-in time to minimize cost for products sold under warranty", *IIE Transactions*, Vol.14, No.3, pp167-174, 1982.
[17] Plesser. K., Field, T., "Cost optimized burn-in duration for repairable electronic systems", *IEEE Tr. Reliability*, Vol.26, No.3, pp195-197, 1977.
[18] Yan, L., English, J., "Economic cost modeling of environmental-stress-screening and burn-in", *IEEE Tr. Reliability*, Vol.46, No.2, pp275-282, 1997.

소프트웨어의 최적 시험시간

[1] Brown, D., Maghsoodloo, S., Deason, W., "A cost model for determining the optimal number of software test cases", *IEEE Tr. Software Engineering*, Vol.15, No.2, pp218-221, 1989.
[2] Cho, B., Park, K., "An optimal time for software testing under the users requirement of failure-free demonstration before release", *IEICE Tr. Fundamentals*, Vol.77-A, No.3, pp563-570, 1994.
[3] Dalal, S., Mallows, C., "Some graphical aids for deciding when to stop testing software", *IEEE J.*

on selected areas in communications, Vol.8, No.2, pp169-175, 1990.
[4] Dalal, S., McIntosh, A., "When to stop testing for large software systems with changing code", *IEEE Tr. Software Engineering*, Vol.20, No.4, pp318-323, 1994.
[5] Forman, E., Singpurwalla, N., "Optimal time intervals for testing on computer software errors", *IEEE Tr. Reliability*, Vol.28, pp250-253, 1979.
[6] Hou, R., Kuo, S., Chang, Y., "Optimal release policy for hyper-geometric distribution software reliability growth model", *IEEE Tr. Reliability*, Vol.45, No.4., pp646-651, 1996.
[7] Hou, R., Kuo, S., Chang, Y., "Optimal release times for software systems with scheduled delivery time based on the HGDM", *IEEE Tr. Computers*, Vol.46, No.2, pp216-221, 1997.
[8] Kapur, P., Garg, R., "Cost-reliability optimum release policies for a software system under penalty cost", *International J. of Systems Science*, Vol.20, pp2547-2562, 1989.
[9] Kimura, M., Toyota, T., Yamada, S., "Economic analysis of software release problems with warranty cost and reliability requirement", *Reliability Engineering & System Safety*, Vol.66, pp49-55, 1999.
[10] Koch, H., Kubat, P., "Optimal release time of computer software", *IEEE Tr. Software Engineering*, Vol.9, No.3, pp323-327, 1983.
[11] Littlewood, B., Wright, D.,"Some conservative stopping rules for the operational testing of safety-critical software", *IEEE Tr. Software Engineering*, Vol.23, No.11, pp673-683, 1997.
[12] Okumoto, K., Goel, A., "Optimum release time for software systems based on reliability and cost criteria", *J. of System Software*, Vol.1, No.4, pp315-318, 1980.
[13] Ross, S., "Software reliability: The stopping rule problem", *IEEE Tr. Software Engineering*, Vol.11, No.12, pp1472-1476, 1985.
[14] Singpurwalla, N., "Determining an optimal time interval for testing and debugging software", *IEEE Tr. Software Engineering*, Vol.17, No.4, pp313-319, 1991.
[15] Yamada, S., Narihisa, H., Osaki, S., "Optimum release policies for a software system with a scheduled delivery time", *International J. of Systems Science*, Vol.15, pp905-914, 1984.
[16] Yamada, S., Osaki, S.," Optimal software release policies with simultaneous cost and reliability requirements", *European J. of Operational Research*, Vol.31. No.1, pp46-51, 1987.
[17] Yamada, S., Osaki, S., "Cost-reliability optimal release policies for software systems", *IEEE Tr. Reliability*, Vol.34, No.5, pp422-424, 1985.
[18] Yang, M., Chao, A., "Reliability estimation & stopping rules for software testing, based on repeated appearances of bugs", *IEEE Tr. Reliability*, Vol.44, No.2., pp315-321, 1995.
[19] Zeephongsekul, P., Chiera, C., "Optimal software release policy based on a two person game of timing", *J. of Applied Probability*, Vo.42, pp470-481, 1995.

하드웨어-소프트웨어 시스템의 신뢰성 분석

[1] Goel, A., Soenjoto, J., "Models for hardware-software system operational perfor- mance evaluation", *IEEE Tr. Reliability*, Vo.30, No.3, pp232-239, 1981.
[2] Sumita, U., Masuda, Y., "Analysis of software availability/reliability under the influence of hardware failures", *IEEE Tr. Software Engineering*, Vol.12, No.1, pp32-41, 1986.
[3] Tokuno, K., Yamada, S., "Markovian availability modeling for software-intensive systems", *Int. J. of Quality & Reliability Management*, Vo.17, No.2, pp200-212, 2000.
[4] Welke, S., Johnson, B., Aylor, J., "Reliability modeling of hardware/software systems", *IEEE Tr. Reliability*, Vol.44, No.3, pp413-418, 1995.