韓國 OA學會誌
第8卷 第2號, 2001. 6

NVP 신뢰도 분석을 위한 새로운 접근방법에 관한 연구 (A Study on Method a New Approach f The Analsis of NVP Reliablity)

신 경 애*

요 약

소프트웨어 신뢰성을 향상시키는 방법에는 소프트웨어 결함 허용기법중에서 가장 객관적이고 정량적으로 평가받는 것이 NVP(N-Version Programming)기법이다. 이 기법에서 신뢰도를 추정하는 모델로 이항분포를 사용하는데 이 모델은 각 컴포넌트 신뢰도의 값들이 동일하다는 한계점이 있었다. 본 연구에서는 기존 모델의 한계점을 해결하기 위하여 NVP 신뢰도 분석을 위한 새로운 접근 방법으로 유전자 알고리즘(Genetic Algorithms)을 적용하였고, 또한 적용 모델과 기존 모델를 서로 비교 검토하였다. 그 결과 전체시스템 신뢰도를 일정 수준이상 유지하면서 각 컴포넌트 신뢰도의 값들을 최적화할 수 있었고, 또한 비용을 최소로 하는 최적의 수를 추정할 수 있었다. 그리고 적용 모델과 기존 모델를 비교 및 평가하여 타당성을 증명하였다.

I. 서 론

컴퓨터 시스템의 사용이 일반화 됨에 따라서 고 신뢰성은 매우 중요한 의미를 갖는다. 왜냐하면 컴퓨터 시스템의 고장이 경제나 사람의 생명에 치명적인 타격을 주기 때문이다. 이러한 고 신뢰도에 대한 요구를 만족시키기 위한 방법이 바로 결함허용 컴퓨터 시스템이다. 결함허용 컴퓨터 시스템은 구성 요소 중에서 하나의 고장이 발생해도 계속 수행을 하도록 설계된 시스템을 말한다[1][8]. 결함허용 컴퓨터 시스템을 구현하기 위하여 하드웨어 신뢰도를 증가시키기 위한 노력은 오래 전부터 계속되어 왔는데 특히 그 결함율을 감소시키는 결함허용 구조를 개발하는 방향으로 발전되었다. 소프트웨어 신뢰성이란 소프트웨어가 규정된 환경 하에서 의도하는 기간 동안 고장이 발생하지 않고 동작이 가능한 성질이나 정도를 나타낸다. 이러한 소프트웨어 신뢰성을 확률로서 표현한 것이 소프트웨어 신뢰도이다[4]. 지금까지 소프트웨어 결함허용 기법들 중에서 객관적이고 정량적이라고 평가받는 것이 NVP 기법이다. 그러나 이 기법은 신

* 동주대학 컴퓨터정보통신계열 부교수

뢰도 추정에서 컴포넌트의 신뢰도 값들이 동일하다라는 가정을 가지고 이항분포 모델을 사용하여 전체 시스템 신뢰도를 추정하였다. 그러나 현실적으로 볼 때 컴포넌트의 신뢰도의 값들이 다른 경우가 대부분이다. 즉, 팀의 능력이나 개발환경 등에 따라서 컴포넌트 값들이 달라질 수 있다. 따라서 본 연구에서는 이런 문제점을 해결하기 위해서 NVP 신뢰도 분석을 위한 새로운 접근 방법으로 유전자 알고리즘(Genetic Algorithms)을 적용하고자한다. 또한 적용한 모델이 기존의 모델보다 타당한지를 비교해 본다.

Ⅱ. 관련연구

2.1 결함허용 기법 개념 및 종류

결함허용 기법은 하드웨어, 소프트웨어 또는 정보의 결함이 발생하더라도 주어진 기능을 수행하여 올바른 결과를 산출할 수 있도록 하는 방법을 의미한다. 결함허용 시스템에는 결함탐지(fault detection), 결함위치(fault location), 결함분리 또는 봉쇄(fault isolation or containment), 결함 복구 및 시스템 재구성(fault recovery and reconfiguration) 등과 같은 구성 요소를 갖추어 설계해야 한다[1]. 그리고 결함허용 기법의 종류에는 하드웨어 결함허용, 소프트웨어 결함허용, 정보 결함허용이 있다[12]. 하드웨어 결함허용 기법에는 TMR(triple modular redundancy), Watchdog timer, Pair and spare, Duplication with can parison, Stand-by sparing, 정보 결함허용 기법에는 Parity check code, m of n code, Checksum, Berger code, Hamming error correcting code, 소프트웨어 결함허용 기법에는 Check pointing, Recovery block, Conversation, Distributed recovery block, N Self-checking Programming, N-Version Programming 등이 있다.

2.2 소프트웨어 결함 허용 기법 성능 평가

Hudak은 소프트웨어 결함 허용 기법의 성능 평가를 〈표 1〉과 같이 비교 실험하였다[8].
〈표 1〉에서 보면 NVP가 다른 소프트웨어 결함 허용 기법에 비해 우수한 것으로 평가되었다.

표 1. 소프트웨어 결함허용 성능비교

FT	ED	ER	Ab	Cr	Av	MTTF	MTTF L
NVP	5	1		1	1	1	1
Recovery Block	2	4	3	5	4	4	5
Concurrent Error-Detection	4		1	2	4	4	4
Algorithmic FT	1	2	3	2	3	4	4
Baseline				5	5	3	5

1 2 3 4 5
better <- ->worse

여기서, ED : error detection, ER : error recovery, Ab : aborting, Cr : correctness,
Av : availability, MTTF : mean time to failure, MTTF L : MTTF w/o luck 이다.

2.3 NVP기법 개념

소프트웨어 결함허용 방법중의 하나가 NVP 기법이다. 이는 소프트웨어 모듈을 N번 설계하고 코드화하여 이 모듈들에 의해 생성된 N개의 결과를 비교(Voting)하는 것이다[5]. 즉, NVP는 하나의 요구 사양을 가지고 N개의 설계팀이 서로 독립된 상태에서 프로그램을 개발하는 것을 뜻한다. NVP를 N-version voting or Parallel redundancy[4], 또한 Multiversion programming[5]이라고도 한다.

2.4 기존 NVP 기법 신뢰도 추정방법

기존 NVP 기법에서 신뢰도를 추정하기 위하여 다음과 같이 가정한다.

◉ 결함은 독립적으로 발생한다.

◉ 각 단위 프로그램의 신뢰도는 같다.

◉ $[\frac{N}{2}+1]$개 이상의 단위 프로그램이 정상이면 시스템은 결함이 없다.

이때, 신뢰도 추정 $R(N)$은 식(1)과 같다.

$$R(N)=\sum_{j=k}^{N} {}_{N}C_{j}\cdot P^{j}\cdot(1-P)^{N-j},\ k=[\frac{N}{2}+1] \qquad \text{식(1)}$$

여기서, ${}_{N}C_{j}\cdot P^{j}\cdot(1-P)^{N-j}$ 는 N개중에서 j개만 성공할 확률이다. 즉, N개 중에서 j개의 단위 프로그램이 정상일 때 시스템이 작동할 확률을 말한다.

2.5 유전자 알고리즘 개념

유전자 알고리즘은 생물의 진화 메카니즘을 모방한 탐색 알고리즘이다. 또한 생물 진화의 원리로부터 착안된 것으로서 확률적 탐색이나 학습 및 최적화를 위한 한가지 기법이다[11]. GA는 Holland의 저서 『Adaptation in Natural and Artificial Systems』 (1975년)에 처음 소개되었으며 포겔(Fogel)은 진화방식의 모형화를 시도하여 간단한 유한상태 시스템의 최적화를 수행하였다[6][11]. 유전자알고리즘에 대한 연구가 본격적으로 수행된 것은 최근의 일이며 아직 초보단계에 있는 학문이다.

2.5.1 유전자 알고리즘 오퍼레이션

유전자 알고리즘은 선택(selection), 교차(crossover), 돌연변이(mutation)와 같은 3단계 유전자 오퍼레이션(genetic operations)을 사용하여 확률적 탐색이나 최적화를 만들어 낸다[6]. 즉, 3단계 오퍼레이션을 반복함으로써 환경에 대응하는 평가값이 높은 문자열을 만들어 내어 문자열의 집단 전체의 평가값을 향상시켜가는 것이다.

2.5.2 유전자 알고리즘 처리과정

유전자 알고리즘의 처리 순서는 〈그림 1〉과 같다. 〈그림 1〉에서 보면 제일 먼저 초기집단(initial population)을 생성한다. 초기집단은 일반적으로 결정된 개체수의 염색체를 임의로 생성한다. 주로 초기집단의 크기(size)는 10개에서 100개 사이다. 이와 같이 초기집단이 생성되고 나면 각각의 개체에 대한 적응도 평가(fitness evaluation)를 하고 그것을 기초로 하여 교차시키는 조작을 한다. 이때 기본적으로 적응도가 높은 개체가 보다 많은 자손을 남기도록 한다. 이와 같이 함으로써 보다 좋은 개체를 형성하는 유전자가 확산하게 된다. 그리고 선택교차를 수행할 개체의 쌍이 결정되면 염색체의 교차를 수행한다. 교차 방법도 여러 가지가 있지만 기본적으로 쌍방의 염색체로부터 일부분씩 취하여 자손의 염색체를 만든다. 마지막으로 돌연변이를 수행한다. 돌연변이는 어떤 확률로 염색체의 일부 값을 변경시키는 조작이다.

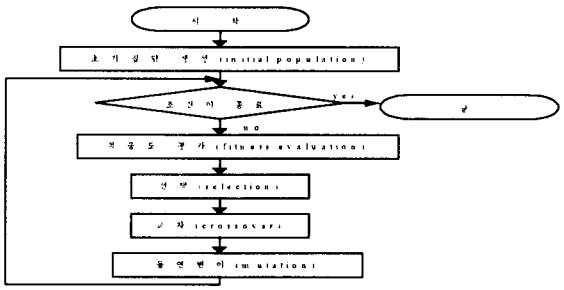

그림 1. 유전자 알고리즘 처리과정

Ⅲ. 수학적 모델 제안 및 유전자알고리즘 적용

3.1 신뢰도 추정방법을 위한 수학적 모델 제안

본 연구는 유전자 알고리즘을 적용하기 위하여 다음과 같이 가정한다.

◉ 각 컴포넌트의 신뢰도는 다르다.

◉ 각 컴포넌트들은 두가지 상태를 가진다. 즉, 0은 결과 불일치, 1은 결과 일치

◉ 각 컴포넌트들은 각 상태 비트마다 특정 신뢰도 가중치를 가진다.
즉, 0.5, 0.6, 0.7, 0.8, 0.9

◉ N개 버전중 $[\frac{N}{2}+1]$개 이상은 일치하는 것으로 한다.

이때, 신뢰도 추정 $R(N)$은 식(2)와 같다.

$$R(N)=\prod_{i=1}^{k} P_i^{xi}\cdot(1-P_i)^{1-xi} \qquad \text{식(2)}$$

여기서, N은 버전의 수, k는 $[\frac{N}{2}+1]$, xi는 각 컴포넌트 상태값이다. 즉, 0또는 1이다. 또한 P_i는 i번째 컴포넌트 신뢰도의 값을 말한다. 그리고 $P_i^{x_i}$는 성공할 확률, $(1-P_i)^{1-x_i}$는 실패할 확률이다. 식(2)은 병렬 시스템에서 전체 N개 중에서 $[\frac{N}{2}+1]$ 개 이상이 성공하는 경우에 대한 시스템 신뢰도를 추정하는 모델이다.

3.2 유전자 알고리즘 적용과정 및 결과

본 연구에서는 5개의 버전을 가지고 유전자 알고리즘을 시험적으로 적용해 그 결과를 고찰했다.

3.2.1 초기세대

유전자 알고리즘을 적용한 초기세대는 〈표 2〉와 같다. 〈표 2〉에서 살펴보면 5개의 비트 스트링(즉, 각 컴포넌트 신뢰도 값)의 형태 중에서 1의 개수가 3개인 것을 선택했다. 왜냐 하면Ⅲ. 3.1. 가정 3에서 N개 버전 중에서 $n/2+1$개 이상 일치하는 것으로 했기 때문이다. 또한 5개에서 3개가 일치한다는 것은 전체 시스템 신뢰도를 일정 수준이상

유지하는 각 컴포넌트 수이다. 그리고 x값은 각 비트위치에서 가중치를 더한 값이며, 가중치는 0.5, 0.6, 0.7, 0.8, 0.9로 하였다. 전체 시스템 신뢰도를 구하기 위하여 일반화 수식를 적응도 함수(fitness function)로 사용하였다. 적응도 함수의 전체 시스템 신뢰도는 식(3)과 같다.

$$f(x)=R(N)=\sum_{j=2}^{N} N\cdot X\cdot P^{j}\cdot(1-P)^{n-j} \qquad \text{식(3)}$$

여기서, N은 버전의 수, x는 각 컴포넌트 신뢰도 값의 합, P^{j}는 성공할 확률, $(1-P)^{(n-j)}$는 실패할 확률이다. 그리고 pselect는 선택될 확률이며 식(4)와 같다.

$$pselect=\frac{fi}{\sum_{f}} \qquad \text{식(4)}$$

여기서, $\sum_{f}$는 $f(x)$의 합, fi는 각 적응도 함수값이다. 그리고 actual count는 다음세대에 존재할 스트링의 반복 개수이고 expected count값을 반올림하여 사용하였다. 기대값(expect count)는 식(5)와 같다.

$$\exp ected\ count=\frac{f_i}{f} \qquad \text{식(5)}$$

여기서, f는 $f(x)$의 평균, fi는 각 적응도 함수값이다. 초기세대의 합(sum)의 값은 8872.80, 평균(average)값은 887.28이다.

표 2. 초기세대 적용과정

string No.	initial population					x value	$f(x)=\sum_{i=2}^{n} n\cdot x\cdot p^{3}\cdot(1-p)^{(n-j)}$	pselect	Expected count	Actual count
	0.5	0.6	0.7	0.8	0.9					
1	0	0	1	1	1	2.4	1592.53	0.18	1.79	2
2	0	1	0	1	1	2.3	1287.27	0.15	1.45	1
3	0	1	1	0	1	2.2	1030.73	0.12	1.16	1
4	0	1	1	1	0	2.1	816.82	0.09	0.92	1
5	1	0	0	1	1	2.2	1030.73	0.12	1.16	1
6	1	0	1	0	1	2.1	816.82	0.09	0.92	1
7	1	0	1	1	0	2.0	640.00	0.07	0.72	1
8	1	1	0	0	1	2.0	640.00	0.07	0.72	1
9	1	1	0	1	0	2.0	640.00	0.07	0.72	1
10	1	1	1	0	0	1.8	377.91	0.04	0.43	0
Sum							8872.80	1.00	10.00	10
Average							887.28	0.10	1.00	1
Max							1592.53	0.18	1.79	2

3.2.2 1세대와 2세대

1세대는 〈표 3〉와 같다. 〈표 3〉에서 mating pool after reproduction은 초기세대의 actual count를 가지고 생성하였고, mate는 임의로 정해주었다. mate를 할때 될 수 있으면 상위 비트가 0인것과 1를 mate 시켰고, crossover site도 임의로 mate를 시켜서 새로운 집단을 생성하였다. 또한 초기세대와 마찬가지 방식으로 x값을 가지고 전체시스템 신뢰도 값, Pselect, Expect count, Actual count를 구했다. 1세대의 합(sum)의 값은 11395.41, 평균(average)값은 1139.54이다. 2세대는 〈표 4〉와 같다. 2세대도 1세대와 동일한 방법으로 구했다. 2세대의 합(sum)의 값은 22119.70, 평균(average)값은 2211.97이다.

표 3. 1세대 적용과정

string No.	M.P. after Reproduction	Mate	Crossover site	New population	x value	$f(x) = \sum_{j=2}^{n} n \cdot x \cdot p^3 \cdot (1-p)^{(n-j)}$	pselect	Expected count	Actual count
1	00111	6	0011|1	00111	2.4	1592.53	0.14	1.40	1
2	00111	7	1001|1	10011	2.2	1030.73	0.09	0.90	1
3	01011	9	0011|1	00111	2.4	1592.53	0.14	1.40	1
4	01101	8	1010|1	10101	2.1	816.82	0.07	0.72	1
5	01110	10	010|11	01001	1.5	151.88	0.01	0.13	0
6	10011	1	110|01	11011	2.8	3442.07	0.30	3.02	3
7	10101	4	01|101	01110	2.1	816.82	0.07	0.72	1
8	10110	3	10|110	10101	2.1	816.82	0.07	0.72	1
9	11001	2	101|10	10110	2.0	640.00	0.06	0.56	1
10	11010	5	110|10	11010	1.9	495.22	0.04	0.43	0
sum						11395.41	1.00	10.00	10
Average						1139.54	0.10	1.00	1
Max						3442.07	0.30	3.02	3

*M.P. : mating pool

표 4. 2세대 적용과정

string No.	M.P. after Reproduction	Mate	Crossover site	New population	x value	$f(x) = \sum_{j=2}^{n} n \cdot x \cdot p^3 \cdot (1-p)^{(n-j)}$	pselect	Expected count	Actual count
1	00111	10	00|111	00110	1.5	151.88	0.01	0.07	0
2	10011	8	10|110	10111	2.9	4102.23	0.19	1.85	2
3	00111	9	10|011	10110	2.0	640.00	0.03	0.29	0
4	10101	6	01|110	01011	2.3	1287.27	0.06	0.58	1
5	11011	7	001|11	00101	1.6	209.72	0.01	0.09	0
6	11011	4	101|01	10111	2.9	4102.23	0.19	1.85	2
7	11011	5	101|01	10111	2.9	4102.23	0.19	1.85	2
8	01110	2	110|11	11001	2.0	640.00	0.03	0.29	0
9	10101	3	110|11	11011	2.8	3442.07	0.16	1.56	2
10	10110	1	110|11	11011	2.8	3442.07	0.16	1.56	2
sum						22119.70	1.00	10.00	10
Average						2211.97	0.10	1.00	1
Max						4102.23	0.19	1.85	2

3.2.4 비교 분석 및 평가

본 연구에서는 5개의 버전을 가지고 초기세대의 비트 스트링을 만들었다. 그리고, 가우스함수에 의하여 초기세대의 비트열은 3개 이상이 1로 구성되었다. 초기세대를 5비트의 비트 스트링으로 했기 때문에 2세대까지 기술하고 있으나 버전 수(즉, 비트)가 증가할수록 표현하는 비트 스트링이 많아지기 때문에 보다더 다양한 세대를 생성할 수있다. 따라서 추정하고자 하는 전체시스템 신뢰도 값은 더 정확해질 것이다. 그러나 본 연구에서 사용한 5비트의 비트스트링으로 표현한 세대만으로 그 성향을 확실하게 알 수 있으므로 본 연구에서는 가능성만 제시했다. 본 연구에서 설정한 환경에 적합한 비트스트링이 다음 세대에 생존할 확률이 높고 결국 수치의 합과 평균값이 점점 더 커지는 형태로 나타나게 된다. 이는 전체 시스템의 신뢰도가 점점 최대화 됨에 따라 더 적합함을 의미한다. 결국 전체 시스템 신뢰도 값은 계속 증가할 것이며 일정한 수준이 되면 수렴하는 경향을 띄게된다. 또한 최하위 2~3비트가 1인 비트스트링이 높은 적응도 함수값을 가진다는 것을 알 수 있다. 이것은 각각 달리 설정한 컴포넌트 신뢰도의 가중치가 높을수록 전체 시스템 신

뢰도를 향상시킨다는 사실을 예측 할 수 있다.

3.3 기존 이항분포 모델과 유전자 알고리즘모델의 비교

NVP 기법을 가지고 신뢰도를 추정할 때 이항분포 모델과 유전자 알고리즘 모델을 비교해 보면 〈표 5〉와 같다. 〈표 5〉에서 알 수 있는 사실은 유전자 알고리즘 모델이 이항분포 모델보다 각 컴포넌트 신뢰도를 다르게 하면서 원하는 전체 시스템 신뢰도를 예측할 수 있고, 또한 최적화 기능을 가지고 있다는 점이 특징이다.

표 5. 이항분포모델과 유전자 알고리즘 비교

	이항분포 모델	유전자알고리즘 모델
버전 수	동일	동일
$[\frac{N}{2}+1]$의 수	동일	동일
각 컴포넌트 신뢰도	동일	다름
비용 예측	버전수와 c에 의해서 예측함	버전수와 pi에 의해서 예측함
최적화 기능	없음	있음

Ⅵ. 결론 및 향후 연구 과제

본 연구에서는 기존의 NVP 신뢰도 추정모형의 한계점을 해결하기 위하여 NVP 신뢰도 분석을 위한 새로운 접근 방법으로 유전자 알고리즘(Genetic Algorithms)을 적용하는 방안을 제안하였다. 이 유전자 알고리즘을 이용한 추정 방안을 이용하여 각 컴포넌트 신뢰도들을 0.5 ~ 0.9 까지 달리 지정하였다. 그 결과 컴포넌트 상태 비트 중에서 하위 비트에 1이 나타날수록 다음 세대에 살아남을 가능성이 높은 것으로 나타났다. 이 특징은 높은 가중치를 가지는 컴포넌트가 많을수록 전체 시스템의 신뢰도가 향상되는 것을 의미한다. 그러므로, 전체 시스템의 신뢰도를 일정 수준 이상 유지하면서 각 컴포넌트들에 대한 최적의 신뢰도를 추정함으로써 전체 시스템 개발 비용을 감소시킬 수 있을 것으로 추정된다.

본 연구에서 얻은 결론은 다음과 같이 두가지로 요약할 수 있다.

첫째, 전체 시스템의 신뢰도를 일정 수준 이상 유지하면서 각 컴포넌트 신뢰도를 최적화 할 수 있다.

둘째, 시스템 신뢰도에 가장 적합한 최적의 수를 추정할 수 있다.

한편 본 논문에서 적용시킨 유전자 알고리즘은 기초적인 단계이므로 보다 높은 신뢰성을 보장하기 위해 다양한 기법의 유전자 알고리즘 적용이 요구되며 아울러 이를 기반으로 한 시뮬레이터의 개발이 중요한 향후 연구과제이다.

참고문헌

[1] 양승민, "결함허용 소프트웨어 신뢰도 모델링 기법", 정보과학회지, pp. 48-57, 1993.

[2] 신경애, 한판암, "GA를 이용한 NVP신뢰도 분석에 관한 연구", 한국정보처리학회 논문지 제6권 제2호, pp. 85-94, 1999.2.
[3] 신경애, "NVP신뢰도 분석을 위한 유전자 알고리즘 적용", 한국컴퓨터산업교육학회 논문지, "2000, 10, Vol. 1., No. 1, October.
[4] Ann Marie Neufelder, "Ensuring Software Reliability", Marcel Dekker, Inc., NewYork, 1993.
[5] Fevzi Belli and Piotr Jedrzejovicz, "An Approach to the Reliability Optimization of software with Redundancy", IEEE Transactios on Software Engineering, Vol. 17, No. 3, 1991.
[6] Goldberg. D., "Gentic Algorithms in Search Optimazation & Machine Learning", Addision-Wesley, 1989.
[7] Joanne Bechta Dugan, Michael R. Lyu, "System Reliability Analysis of an N-Version Programming Application", IEEE Transactions Reliability, Vol. 43, No. 4, 1994.
[8] John Hudak et al, "Evaluation & Comparision of Fault Tolerant Software Techniques", IEEE Transactions On Software Engineering, Vol. 42, No. 2, 1993.
[9] Karama Kanoun, Mohamed Kaaniche, Christian Beounes, Jean-Claude Laprie, Jean rlat, "Reliability Growth of Fault-Tolerance Software", IEEE Reliability, Vol. 42, No. 2, 1993.
[10] Kishor. Trivedi, "Reliability & Statistics With Reliability", Queuing, And Computer Science Applications, 1995.
[11] Syswerda, G. and Palmucci, J. : "The Application of Genetic Algorithms to Resoure Scheduling", Proc. of ICGA-91,1991.
[12] Whitely, D. and Hnson, T. : "Optimizing Neural Networks Using Faster, More Accurate Genetic Search", Proc. of ICGA-89, 1989.

□ 著者紹介

신 경 애
1986 한국 방송대학교 경영학과 졸업(경영학사)
1991 한국 방송대학교 전자계산학과 졸업(이학사)
1989 계명대학교 교육대학원 전자계산전공(교육학 석사)
1999 경남대학교 대학원 컴퓨터공학과(공학박사)
1983~1989 동주여자상업고등학교 컴퓨터교사 재직
1990~현재 컴퓨터통신계열 부교수
2000.3~현재 동주대학 정보통신연구소 소장

관심분야 : 소프트웨어 신뢰도 성장모델, 결함허용, CAI