論文
3-2-12

(n,k) 클러스터 웹서버 시스템의 작업전이를 고려한 소프트웨어 재활기법의 가용도 분석

(Availability Analysis of (n,k) Cluster Web Server System using Software Rejuvenation Method over Switchover)

강 창 훈*
(Chang-Hoon Kang)

요 약

클러스터 웹서버 시스템에서는 다수 서버의 장애로 인해 발생하는 가용도 저하 문제와 소프트웨어의 노화로 인하여 높은 성능과 가용도를 제공하기 쉽지 않다. 본 연구에서는 n대의 주 서버와 k대의 여분서버로 구성되는 클러스터 웹 서버 시스템에 성능과 작업전이 시간을 고려한 소프트웨어 재활 모델을 제시하였고, 다양한 시스템 운영 파라미터에 기초하여 소프트웨어의 재활정책에 대한 평가를 위해 평형 상태에서의 확률, 가용도, 손실비용 등을 계산하였다. 수학적 분석을 통해 다양한 시스템 운영 상태에 대한 실험을 통해 검증하였으며, 소프트웨어의 재활 정책에 의한 예방적 결함허용 기법이 시스템의 안정성에 중요한 요소임을 확인하였다.

ABSTRACT

An cluster web server system, one has the problem that does the low availability occured by the high chance of the server failures and it is not easy to provide high performance and availability occuring software aging. In this paper, running cluster web servers consists of n primary servers and k backup servers, propose software rejuvenation model on performance and switchover time. Based on the various system operational parameters, we calculate to evaluate the rejuvenation policy such steady-state probabilities, availability, and downtime cost. And we validate the solutions of mathematical model by experiments based on various operation parameters and find that the software rejuvenation method can be adopted as prventive fault tolerant technique for stability of system.

* 정회원 : 극동정보대학 멀티미디어과 조교수

논문접수 : 2002. 2. 4.
심사완료 : 2002. 2. 23.

1. 서론

최근 사회, 문화, 교육 등 여러 분야에서 인터넷의 활용이 급증하면서 사용자들의 다양한 요구로 인해 소프트웨어의 복잡도는 날로 증가하고 있다. 이같이 복잡도가 증가됨에 따라 소프트웨어의 설계, 구현 또는 그 밖의 여러 가지 원인과 소프트웨어와 관련된 결함으로 인해 전체 시스템의 오동작 또는 수행 중단으로 이어지는 사례가 늘어나고 있다. 현재 정보화 사회의 특성상 대다수의 업무들이 컴퓨터 시스템에 대한 의존도가 절대적이고, 많은 업무에서 중단 없는 서비스가 요구되므로 컴퓨터 시스템 장애로 인한 피해는 이루 상상할 수 없을 정도로 커지게 된다. 최근에는 하드웨어적인 장애보다는 소프트웨어의 장애가 더 큰 것으로 보고되었으며 앞으로 소프트웨어 결함에 의한 장애는 비중이 점차로 늘어날 전망이다[4]. 특히 무정지 서비스를 해야 하는 인터넷 서버에서 사용되는 소프트웨어는 빈번한 통신 두절과 데이터 유실로 인해 소프트웨어의 결함이 더욱 심각할 가능성이 높다[4,18].

소프트웨어는 시간이 지남에 따라 소프트웨어 노화(Software Aging) 현상으로 인해 시스템의 성능 저하 및 일시적 결함의 발생 가능성이 커지게 되며, 점차로 소프트웨어의 성능이 저하되어, 결국에는 시스템 작동이 멈추는 현상이 발생 할 수 있게 된다[17]. 이러한 문제를 해결하기 위해 소프트웨어 재활(Software Rejuvenation) 기법을 사용한다. 소프트웨어 재활은 결함을 미연에 방지하는 능동적 차원의 결함 허용 기법이다[2,18].

한편 인터넷 사용의 증가로 웹서버, 전자상거래서버, 게임서버 등을 포함한 여러 분야에서 고성능 서버에 대한 수요가 증가하고 있다. 그러나 고성능 서버는 가격이 고가이므로 구입 및 활용에 상당한 어려움이 따른다. 이에 대한 대안으로 저가의 PC나 워크스테이션을 여러 대 연결하여 고성능을 발휘할 수 있는 클러스터 시스템의 사용이 증가하고 있다[7,19]. PC/워크스테이션 클러스터는 슈퍼 컴퓨팅 분야 외에도 다양한 분야에 활용되고 있는데, 그 중 하나가 인터넷과 월드 와이드 웹(World Wide Web) 분야이다. 이처럼 웹서버의 역할을 대신 할 수 있는 클러스터형 웹서버가 중요한 수단으로 대두되고 있다[3,5,7,10,19].

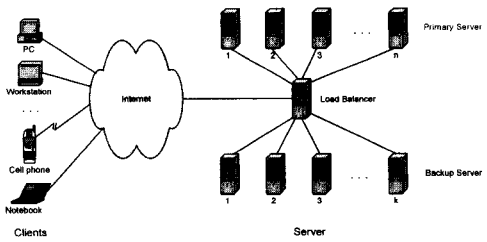
현재까지 소프트웨어적 결함으로 인해 시스템의 성능이 감소하거나 다운되는 소프트웨어 노화(Software Aging) 현상에 대한 연구[1,11,18,20]와 클러스터 시스템에서 소프트웨어 재활을 적용한 가용도 및 성능에 대한 많은 연구가 이루어지고 있다. 한 대의 서버로 구성되는 단일계(Simplex) 시스템에서의 재활에 연구가 이루어졌다[12,13,14,15,16,18]. 또한 백업 서버로 구성되는 다중계(Multiplex) 시스템에서의 소프트웨어 재활에 관한 연구도 이루어 졌다[8,9]. 이 연구에서는 주 서버의 수를 하나로 고정하여 재활을 수행하였다. 그러나 여러 대의 서버로 구성되는 시스템에서 가용도의 척도를 단순히 한 대의 서버라도 가동되면 클러스터 시스템 전체가 가용하다 라는 가정이 많이 포함된 연구들이 많았다. 단순히 가용도 만을 고려하지 않고 성능을 고려한 연구가 있었다[17]. 또한 성능을 고려하여 사용자 측면에서 서비스를 받는데 까지 대기한 시간을 고려한 연구가 있었다. 그러나 이 연구에서는 재활을 수행하지 않은 상태에서의 가용도와 성능을 분석하였다.

본 연구에서는 n대의 주 서버와 k대의 여분 서버를 가진 클러스터 시스템에 재활 기법을 적용하여, 주 서버가 불안정한 상태가 일 때 임의의 여분 서버로 교체하는 작업전이(Switchover) 시간을 시스템 모델에 포함하여 임의의 상태에서 클러스터 시스템의 성능을 나타내는 척도를 포함하여, (n, k) 클러스터 시스템에서 수행되는 소프트웨어의 재활주기, 재활소요시간, 서버의 고장율, 수리율, 불안정률, 작업전이시간, 주 서버의 수, 여분 서버의 수, 서버의 가동시간 등의 시스템 운영파라미터에 기초하여 소프트웨어 재활 정책을 수학적 방법으로 계산하였다.

본 논문의 2장에서는 재활기법에 의한 (n, k) 클러스터 시스템 모델을 정의하고, 3장에서는 제안된 모델의 수학적 해석 방법의 정확성을 검증하기 위하여 다양한 운영 조건에 대한 실험 및 분석을 수행하였다. 마지막으로 결론에서 제안된 소프트웨어 재활 기법의 활용 방안 및 향후 연구에 관하여 논하였다.

2. (n,k) 클러스터 시스템 모델

본 연구에서 분석하고자 하는 (n,k)클러스터 시스템은 [그림 1]과 같다. (n,k) 클러스터 시스템은 n대의 주 서버(Primary Server)와 k대의 여분서버(Backup Server)로 구성되어 있으며 임의의 주 서버에 고장이 날 경우 k대의 여분서버중 한 대가 주 서버의 역할을 대신하는 시스템이다.



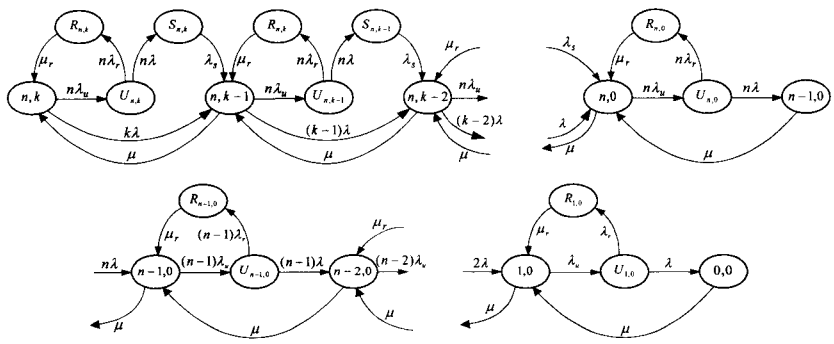
[그림 1] (n,k) 클러스터 시스템의 구성도

[Fig. 1] The configuration diagram of (n,k) Cluster system

클러스터 시스템을 구성하는 서버들 중 임의의 서버가 로드 밸런서(Load Balancer)의 역할을 수행한다. 로드 밸런서는 주 서버들에게 작업을 분배하며, 주 서버의 재활 작업을 수행하고, 주 서버가 고장날 경우 백업 서버로 작업전이 작업을 수행한다. 백업 서버들은 주 서버가 불안정 상태이거나 고장났을 때 주 서버로 곧 바로 대치 될 수 있는 대기(Standby) 상태로 존재한다.

본 연구에서 (n,k) 클러스터 시스템에 사용될 가정을 다음과 같이 정의한다.

- n 대의 주 서버가 클러스터로 연결되어 있으며, 각 서버는 서로 독립적으로 작업을 수행한다.
- k 대의 여분 서버가 존재하며($n \geq k$), 임의의 주 서버에 고장이 날 경우 이의 역할을 Active/Standby 방식으로 대체한다.
- 주 서버와 여분 서버의 고장률(λ), 수리률(μ)은 상수이다. 즉 고장 및 수리 시간 간격은 지수분포를 따른다.
- 여분서버는 주 서버 고장 시에만 사용되며, 고장 난 서버는 수리된다.
- 주 서버에서 여분서버로 작업전이 시에 일정 시간이 소요되며, 작업전이 시간은 평균값이 $1/\lambda_s$인 지수분포를 따른다.
- 재활에 들어갈 경우, k대의 여분 서버를 제외한 n대의 주 서버가 작업 대상이며, 재활 작업시간 ($1/\mu_r$)은 여분 서버 수에 무관하다.
- 장시간 가동으로 인한 서버의 불안정율(λ_u)은 모든 가동 상태에서 동일하다.
- 서버의 가동을 주기적으로 멈추는 재활률(λ_r)은 모든 가동상태에서 동일하다.
- 여분서버가 없을 때(k=0) 결함을 감지하여 클러스터로부터 제거하는 결함제거 시간($1/\lambda_d$)은 상수이며 지수분포를 따른다.



[그림 2] (n, k) 클러스터 시스템의 고장-재활-수리-작업전이 상태 전이도

[Fig. 2] The fault-rejuvenation-repair-switch state transition diagram of (n, k) cluster system

본 논문에서 제시한 (n,k) 클러스터 시스템의 상태 모델을 [그림 2]에 나타내었다. 정상상태에서 가동중인 서버는 (n,k), $(n,k\text{-}1)$,…, $(n,0)$, (0,0) 등의 가동중인 (주 서버, 여분 서버)의 수를 상태 변수로 가지고 있으며, k개의 여분 서버가 모두 사용될 때까지는 시스템의 가동 서버는 n개로 유지된다. (0,0)인 상태는 주 서버와 여분서버의 가동이 모두 중지된 상태로 고장상태를 의미한다.

정상상태에서 가동중인 서버가 장시간 가동으로 인해 성능이 저하된 불안정 상태(Ustable State)는 $U_{n,k}$, $U_{n,k\text{-}1}$, …, $U_{n,0}$, …, $U_{1,0}$, $U_{0,0}$로 나타내었다. 정상상태에서 불안정상태로 변화율을 λ_u로 표시하며, 이는 소프트웨어의 장기간 가동으로 인한 시스템의 불안정성을 반영한다. 불안정상태(Unstable State)에서 λ_r의 변화율로 재활작업에 들어가거나, $n\lambda$(여분의 서버가 존재할 경우 n대의 주서버가 가동함)나 $i\lambda$(여분서버가 없을 경우 i대의 주서버가 가동함)의 변화율로 고장이 발생하게 된다. 재활상태(Rejuvenation State)는 $R_{n,k}$, $R_{n,k\text{-}1}$, …, $R_{n,0}$, …, $R_{1,0}$, $R_{0,0}$로 표시하며, 시스템의 가동을 고의로 중지시켜 재 부팅하는 상태(State)를 나타낸다. 불안정 상태에서 주 서버가 고장이 발생하여 고장난 서버를 여분서버로 대치하는 상태를 $S_{n,k}$, $S_{n,k\text{-}1}$, …, $S_{n,1}$로 나타내었다. 여분서버가 없을 경우(k=0) i대의 클러스터 중 임의의 서버에서 결함이 발생할 경우 고장난 주 서버는 수리 상태에 들어가고 클러스터 시스템은 하나 감소한 상태로 서비스를 수행한다. 그림 2의 평형상태(Steady State)에서의 균형식(Balance Equation)은 다음과 같다.

$$(k\lambda + n\lambda_u)P_{n,k} = \mu_r P_{R_{n,k}} + \mu P_{n,k-1} \quad (1)$$

$$(\mu + n\lambda_u + j\lambda)P_{n,j} = \lambda_s P_{S_{n,j+1}} + \mu_r P_{R_{n,j}} + \mu P_{n,j-1} + (j+1)\lambda P_{n,j+1}$$

$$j = 1,\cdots,k-1 \quad (2)$$

$$(\mu + n\lambda_u)P_{n,0} = \lambda_s P_{S_{n,1}} + \mu_r P_{R_{n,0}} + \mu P_{n-1,0} + \lambda P_{n,1} \quad (3)$$

$$(\mu + i\lambda_u)P_{i,0} = (j+1)\lambda P_{U_{i+1,0}} + \mu_r P_{R_{i,0}} + \mu P_{i-1,0}$$

$$i = 1,\cdots,n-1 \quad (4)$$

$$\mu P_{0,0} = \lambda P_{U_{1,0}} \quad (5)$$

$$n(\lambda + \lambda_r)P_{U_{n,j}} = n\lambda_u P_{n,j}$$

$$j = 1,\cdots,k \quad (6)$$

$$i(\lambda + \lambda_r)P_{U_{i,0}} = i\lambda_u P_{i,0} \quad i = 1,\cdots,n \quad (7)$$

$$\mu_r P_{R_{n,j}} = n\lambda_r P_{U_{n,j}} \quad j = 1,\cdots,k \quad (8)$$

$$\mu_r P_{R_{i,0}} = i\lambda_r P_{U_{i,0}} \quad i = 1,\cdots,n \quad (9)$$

$$\lambda_s P_{S_{n,j}} = n\lambda P_{U_{n,j}} \quad j = 1,\cdots,k \quad (10)$$

$$\sum_{j=1}^{k} P_{n,j} + \sum_{i=0}^{n} P_{i,0} + \sum_{j=1}^{k} P_{U_{n,j}} + \sum_{i=1}^{n} P_{U_{i,0}} + \sum_{j=1}^{k} P_{R_{n,j}} + \sum_{i=1}^{n} P_{R_{i,0}} + \sum_{j=1}^{k} P_{S_{n,j}} = 1 \quad (11)$$

위의 균형식(Balance Equation)과 각 상태(State)에서 머물 확률의 총합이 1이되는 식을 결합한 연립방정식을 풀면, 시스템이 평형일 때 각 상태에서 머물 확률을 얻을 수 있다.

$$P_{n,k} = \left[\left(1 + \frac{\lambda_u}{\lambda+\lambda_r} + \frac{n\lambda_r}{\mu_r}\frac{\lambda_u}{\lambda+\lambda_r} + \frac{n\lambda}{\lambda_s}\frac{n\lambda}{\lambda+\lambda_r}\right) \sum_{j=1}^{k} \left(\frac{\lambda}{\mu}\right)^{k-j} \frac{\left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)!}{\left(\frac{n\lambda_u}{\lambda+\lambda_r}+j\right)!} + \left(\frac{\lambda}{\mu}\frac{\lambda_u}{\lambda+\lambda_r}\right)^n n! \left(\frac{\lambda}{\mu}\right)^k \left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)! + \left(1 + \frac{\lambda_u}{\lambda+\lambda_r}\right) \sum_{i=1}^{n} \left(\frac{\lambda}{\mu}\right)^{n-i} \left(\frac{\lambda_u}{\lambda+\lambda_r}\right)^{n-i} \frac{n!}{i!} \left(\frac{\lambda}{\mu}\right)^k \left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)! + \left(\frac{\lambda_r}{\mu_r}\frac{\lambda_u}{\lambda+\lambda_r}\right) \sum_{i=1}^{n} \left(\frac{\lambda}{\mu}\right)^{n-i} \left(\frac{\lambda_u}{\lambda+\lambda_r}\right)^{n-i} \frac{n!}{(i-1)!} \left(\frac{\lambda}{\mu}\right)^k \left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)! \right]^{-1}$$

$$n \geq k,\ n = 1,2 \quad (12)$$

$$P_{n,0} = \left(\frac{\lambda}{\mu}\right)^k \left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)! P_{n,k} \quad (13)$$

$$P_{n,j} = \left(\frac{\lambda}{\mu}\right)^{k-j} \frac{\left(\frac{n\lambda_u}{\lambda+\lambda_r}+k\right)!}{\left(\frac{n\lambda_u}{\lambda+\lambda_r}+j\right)!} P_{n,k} \quad (14)$$

$$j = 0,\cdots,k$$

$$P_{i,0} = \left(\frac{\lambda}{\mu}\right)^{n-i} \left(\frac{\lambda_u}{\lambda+\lambda_r}\right)^{n-i} \frac{n!}{i!} P_{n,0} \quad (15)$$

$$i = 0,\cdots,n-1$$

$$P_{U_{n,j}} = \frac{n\lambda_u}{n(\lambda+\lambda_r)} P_{n,j} = \frac{\lambda_u}{\lambda+\lambda_r} P_{n,j} \quad (16)$$

$$j = 1,\cdots,k$$

$$P_{U_{i,0}} = \frac{i\lambda_u}{i(\lambda+\lambda_r)} P_{i,0} = \frac{\lambda_u}{\lambda+\lambda_r} P_{i,0} \quad (17)$$

$$i = 1,\cdots,n$$

$$P_{R_{n,j}} = \frac{n\lambda_r}{\mu_r} P_{U_{n,j}} = \frac{n\lambda_r}{\mu_r} \frac{\lambda_u}{\lambda+\lambda_r} P_{n,j} \quad (18)$$
$j=1,\cdots,k$

$$P_{R_{i,0}} = \frac{i\lambda_r}{\mu_r} P_{U_{i,0}} = \frac{i\lambda_r}{\mu_r} \frac{\lambda_u}{\lambda+\lambda_r} P_{i,0} \quad (19)$$
$i=1,\cdots,n$

$$P_{S_{n,j}} = \frac{n\lambda}{\lambda_s} P_{U_{n,j}} = \frac{n\lambda}{\lambda_s} \frac{\lambda_u}{\lambda+\lambda_r} P_{n,j} \quad (20)$$
$j=1,\cdots,k$

3. 실험 및 분석

1) 성능을 고려하지 않은 가용도 : 클러스터 시스템을 구성하는 주 서버중 한 대의 서버라도 가동중일 때 전체 웹 클러스터 시스템이 가용 하다고 간주하여 가용도를 계산한다. 따라서 모든 서버가 고장난 상태 ($P_{0,0}$)와 한 대의 주 서버가 가동중인 상태에서 재활 작업을 수행($P_{RI,0}$)하는 상태를 제외한 경우에는 한 대 이상의 주 서버가 서비스를 제공함으로 두 상태를 제외함으로써 다음과 같이 가용도를 계산할 수 있다.

$$Avail = 1-(P_{0,0}+P_{R_{1,0}})$$

2) 성능을 고려하지 않은 손실비용 : 시스템의 모든 서버가 고장 상태여서 클러스터 시스템서비스를 제공하지 못하는 상태로 가동기간(T)에 대한 함수로 다음과 같다.

$$DownCost(T) = (P_{0,0} * C_F + P_{R_{1,0}} * C_R) * T$$

3) 성능을 고려한 가용도 : 시스템이 임의의 시점에서 실제로 가동되고 있는 서버의 대수를 고려하여 가용도를 계산하였다. 정상상태 ($P_{n,j}$)와 불안정상태 ($P_{U_{n,j}}$)에는 모든 서버가 서비스를 제공한다. 그러나 재활상태 ($P_{R_{n,j}}$)와 작업전이 ($P_{S_{n,j}}$) 상태에서는 1대의 서버가 서비스를 하지 못하므로 100%의 성능을 내지 못하게 된다. 이러한 점을 고려하여 다음과 같이 가용도를 계산 할 수 있다.

$$Avail_R = \sum_{j=1}^{k} P_{n,j} + \sum_{i=1}^{n} P_{i,0} + \sum_{j=1}^{k} P_{U_{n,j}} + \sum_{i=1}^{n} P_{U_{i,0}}$$

4) 성능을 고려한 손실비용 : 시스템이 100% 정상적으로 가동되지 않는 경우 즉, 재활상태, 작업전이 상태와 불안정 상태에서 서버의 불시의 고장으로 인해 발생하는 손실비용 등을 고려하여 다음과 같이 전체 손실비용을 계산할 수 있다.

$$Downcost_R = \sum_{j=1}^{k} P_{R_{n,j}} * C_R + \sum_{i=1}^{n} P_{R_{i,0}} * C_R + \sum_{j=1}^{k} P_{S_{n,j}} * C_S + \sum_{i=0}^{n-1} P_{i,0} * C_F$$

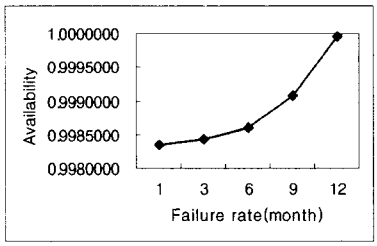
5) 시스템 운영 파라미터 : 본 논문에서 사용한 시스템 운영 파라미터는 [그림 3]과 같다[18]. 3대의 주 서버와 1대의 여분 서버의 가동기간(T) 1년으로 하며, 서버의 고장 (λ)은 6개월에 1회, 고장수리에 (μ)에 2시간이 소요되며, 한 달에 한 번씩 재활 작업 (λ_r)을 수행하며 30분이 소요된다. 시스템 불시 정지로 인해 발생하는 비용 (C_f)은 재활 작업으로 발생하는 비용 (C_r)의 1000배로, 작업전이 시간 (λ_s)은 5분이다. 또한 이와 같은 파라미터 값들은 여러 가지 실험에서 다양하게 변화 시켜가며 실험하였다.

주 서버의 수(n)	number of primary servers	3
여분 서버의 수(k)	number of backup server	1
가동기간(T)	1years	12*30*24*3600
고장률(λ)	2 time / years	1/(6*30*24*3600)
수리율(μ)	1 time / month	1/(60*120)
재활율(λ_r)	1 time / month	1/(30*24*3600)
재활작업시간(μ_r)	2 times / hour	2/(60*60)
작업전이시간(λ_s)	5 minute	12/(3*24*3600)
불안정률(λ_u)	every 3 days	1/(3*24*3600)
불시정지로 인한 손실비용(C_F)	downtime cost	1
재활로 인한 손실비용(C_R)	rejuvenation cost	0.001
작업전이로 손실비용(C_S)	switchover cost	0.001

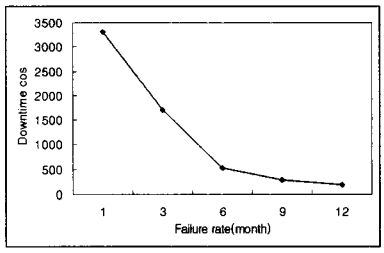
[그림 3] 시스템 운영 파라미터[18]

[Fig. 3] System operation parameters

[그림 4]는 고장률에 따른 변화, 즉 시스템의 고장이 얼마나 자주 일어나느냐에 따른 결과로, 고장이 적을수록 가용도는 증가하는 추세를 보이며, 이에 따른 손실비용도 감소하고 있다. [그림 5]는 시스템의 수리율에 따른 변화로 수리하는데 걸리는 시간이 길면 길수록 가용도는 감소하는 추세를 보이며, 손실 비용은 증가하는 추세를 보이고 있다. [그림 6]은 재활작업을 수행하는 기간에 따른 변화로, 재활작업을 빈번하게 수행하게 되면 가용도가 감소하며, 손실비용은 증가하는 추세를 보이고 있다. 시스템의 불안정 정도에 따른 변화로, 시스템이 빨리 불안정 상태로 변하게 되면 가용도는 감소하고 안정적인 시스템일수록 가용도는 높아지는 추세를 보이고 있으며, 손실 비용도 시스템이 안정적일수록 낮아지는 추세를 보이고 있다.



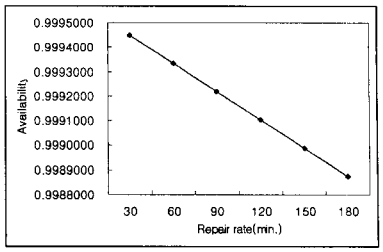
(a) 가용도(Availability)



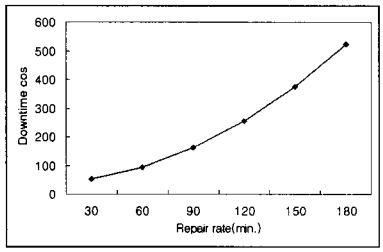
(b) 손실비용(Downtime cost)

[그림 4] 고장률 (λ)에 따른 가용도 및 손실비용

[Fig. 4] Availability and Downtime cost according to failure rate (λ)



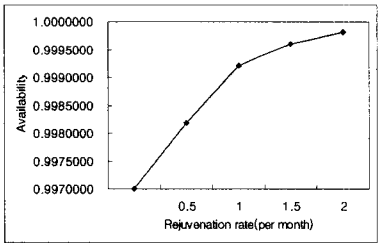
(a) 가용도(Availability)



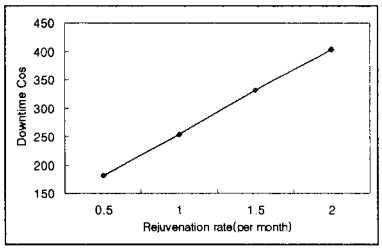
(b) 손실비용(Downtime cost)

[그림 5] 수리율 (μ)에 따른 가용도 및 손실비용

[Fig. 5] Availability and Downtime cost according to repair rate (μ)



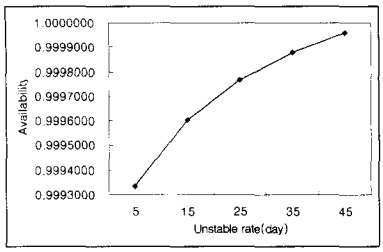
(a) 가용도(Availability)



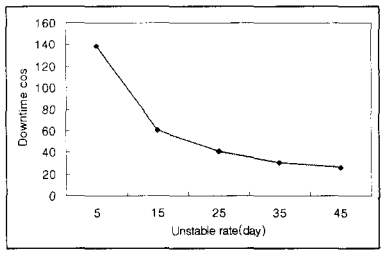
(b) 손실비용(Downtime cost)

[그림 6] 재활률 (λ_r)에 따른 가용도 및 손실비용

[Fig. 6] Availability and Downtime cost according to rejuvenation rate (λ_r)



(a) 가용도(Availability)



(b) 손실비용(Downtime cost)

[그림 7] 불안정률 (λ_u)에 따른 가용도 및 손실비용

[Fig. 7] Availability and Downtime cost according to unstable rate (λ_u)

4. 결 론

클러스터 웹서버 시스템에서는 다수 서버의 장애로 인해 발생하는 가용도 저하 문제와 소프트웨어의 노화로 인하여 높은 성능과 가용도를 제공하기 쉽지 않다. 본 연구에서는 n대의 주 서버와 k대의 여분 서버로 구성되는 클러스터 웹 서버 시스템에 성능과 작업전이 시간을 고려한 소프트웨어 재활 모델을 제시하였고, 시스템에서 가동되는 서버의 수, 여분서버의 수, 소프트웨어의 재활주기, 재활소요시간, 서버의 고장률, 서버의 수리율, 서버의 불안정률, 작업전이 시간 등의 시스템 운영 파라미터에 기초하여, 소프트웨어의 재활정책에 대한 평가를 위해 평형 상태에서의 확률, 정지시간, 가용도, 손실비용 등을 계산하였다. 수학적 분석을 통해 다양한 시스템 운영 상태에 대한 실험을 통해 검증하였으며, 소프트웨어의 재활 정책에 의한 예방적 결함허용 기법이 시스템의 안정성에 중요한 요소임을 확인하였고, 여분 서버를 두어 서버를 운영하는 것이 가용도를 높이는데 중요한 요소임을 파악하였다. 따라서 무정지로 운영되어야 하는 웹서버의 경우 소프트웨어의 빠른 노화로 가용도가 급격히 감소하게 되는데, 이에 대한 예방적 차원에서 가용도를 높이며, 유지보수 비용을 상당히 줄일 수 있는 방법이라 본다.

※ 참고문헌

[1] A. Pfening, S. Garg, A. Puliafito, M. Telek and K. Trivedi, "Optimal Rejuvenation for Tolerating Soft Failures," 27th & 28th Perfo rmance Evaluation, pp. 491-506, October 19 96.

[2] B.W. Johnson, *Design and Analysis of Fault -Tolerant Digital Systems*. p. 584, Addison-Wesley Publishing Company, 1989.

[3] D. Anderson, T. Yang and O.H. Ibarra, "To ward a Scalable Distributed WWW Server o n Workstation Clusters," Journal of Parallel and Distributed Computing, Vol. 42, pp. 91-100, 1997.

[4] J. Gray and D. Siewiorek, "High-Availabilit y Computer Systems," IEEE Computer, pp. 39-48, September 1991.

[5] 권세오, 김상식, 김동승, "리눅스 클러스터형 웹 서버 설계," 정보과학회지, 제18권, 제3호, pp. 48-56, 2000. 3.

[6] M. Nabe, M. Murata and H. Miyahara, "An alysis and Modeling of World Wide Web T raffic for Capacity Dimensioning of Internet Access Lines," Performance Evaluation, Vol. 34, pp. 249-271, 1998.

[7] 오수철, 정상화, "클러스터 시스템 기술 동향," 정보과학회지, 제18권, 제3호, pp. 4-10, 2000. 3.

[8] 박기진, 김성수, 김재훈, "소프트웨어 재활 기법을 적용한 다중계 시스템의 가용도 분석," 한국정보과학회논문지(시스템및이론), 제27권, 제8호, pp. 730-740, 2000. 8.

[9] 박기진, 김성수, "고가용도 Cold Standby 클러스터 시스템 성능 분석," 한국정보과학회

논문지(시스템및이론), 제28권, 제3·4호, pp. 173-180, 2001. 4.

[10] R. Buyya, *High Performance Cluster Computing Volume 1: Architectures and Systems.* p. 849, Prentice-Hall, 1999.

[11] S. Garg, A. Moorsel, K. Vaidyanathan and K. Trivedi, "A Methodology for Detection and Estimation of Software Aging," Proceedings of 9th International Symposium on Software Reliability Engineering, pp. 282-292, November 1998.

[12] S. Grag, Y. Huang, C. Kintala and K. Trivedi, "Time and Load Based Software Rejuvenation: Policy, Evaluation and Optimality," Proceedings of the First Conference on Fault Tolerant Systems, pp. 22-25, December 1995.

[13] S. Garg, Y. Huang, C. kintala and K. Trivedi, "Minimizing Completion Time of a Program by Checkpointing and Rejuvenaion," ACM SIGMETRICS Conference, pp. 252-261, May 1996.

[14] S. Garg, A. Puliafito, M. Telek and K. Trivedi, "Analysis of Software Rejuvenation Using Markov Regenrative Stochastic Petri Net," Proceedings of the Sixth International Symposium on Software Reliability Engineering, pp. 180-187, October 1995.

[15] S. Garg, A. Puliafito, M. Telek and K. Trivedi, "On the Analysis of Software Rejuvenation Policies," Annual Conference on Computer Assurance(COMPASS), pp. 16-20, June 1997.

[16] S. Garg, A. Puliafito, M. Telek and K. Trivedi, "Analysis of Preventive Maintenance in Transactions Based Software Systems," IEEE Transactions on Computers, Vol. 47, No. 1, pp. 96-107, January 1998.

[17] V. Mainkar, "Availability Analysis of Transaction Processing Systems Based on User- Perceived Performance," Symposium on Reliable Distributed Systems, pp. 10-16, October 1997.

[18] Y. Huang, C. Kintala, N. Kolettis and N. Fulton, "Software Rejuvenaion: Analysis, Module and Applications," IEEE Intl. Symposium on Fault Tolelant Computing, FTCS 25, pp. 381-390, June 1995.

[19] 유찬수, "리눅스 클러스터링," 정보과학회지, 제18권, 제2호, pp. 33-39, 2000. 2.

[20] Y. Wang, Y. Huang, K. Vo. P. Chung and C. Kintala, "Checkpointing and Its applications," Proceedings of 25th IEEE Fault-Tolerant Computing Symposium, pp. 22-31, June 1995.

강 창 훈

1986 충남대학교
계산통계학과 졸업(이학사)
1988 충남대학교 대학원
계산통계학과 졸업(이학석사)
1999 아주대학교 대학원
컴퓨터공학과 수료
1994~현재 극동정보대학
멀티미디어과 조교수
관심분야 : 결함허용,
성능분석, 클러스터컴퓨팅,
멀티미디어시스템