Active/Active 클러스터 시스템의 가용도 모델

박 기 진† · 김 성 수††

요 약

하드웨어 기술의 발전으로 인해 컴퓨터 하드웨어의 결함 발생률은 상수 값이거나 점차 작아지는 경향이 있다. 반면에 하드웨어에 탑재된 소프트웨어의 복잡성 및 크기는 이전에는 상상할 수 없을 정도로 방대해져가고 있기 때문에, 소프트웨어의 결함 발생으로 인한 컴퓨터 시스템의 장애 발생 가능성은 점차 더 높아지고 있다. 본 논문에서는 Active/Active 클러스터 시스템의 가용도 개선을 위해서 소프트웨어적인 결함 발생을 미연에 방지할 수 있는 능동적 결함허용 기법인 소프트웨어 재활(rejuvenation) 방법에 대하여 연구하였다. 소프트웨어 재활 과정 및 여분서버로 작업전이(switchover) 과정을 semi-Markov 프로세스로 모델링 한 후, 수학적 분석을 통해 구한 Active/Active 클러스터 시스템의 평형상태 확률을 이용하여, 다양한 운영 조건하의 가용도 및 손실비용을 계산하였으며, 이를 통하여 소프트웨어 재활을 통한 Active/Active 클러스터 시스템의 가용도 개선 가능성을 확인하였다.

An Availability Model for Active/Active Cluster Systems

Kie-Jin Park† · Sung-Soo Kim††

ABSTRACT

For the fast increase of a hardware reliability and the size or complexity of application softwares, the ratio of hardware-origin system failures becomes much smaller than that of software-origin system failures. To improve the availability of an Active/Active cluster systems, we study software rejuvenation method that is a proactive fault-tolerant approach to handle software-origin system failures. In this paper, we model software rejuvenation and switchover processes as the problem of a semi-Markov chain and get mathematical steady-state solution of the process. We calculate the availability and the downtime of an Active/Active cluster systems using software rejuvenation method and find that the method can be improve the availability of the systems.

키워드 : 고가용도(High Availability), 클러스터 시스템(Cluster Systems), 소프트웨어 재활(Software Rejuvenation)

1. 서 론

고객과 판매자가 인터넷을 통하여 복잡하게 연결되어 있는 전자 상거래(electronic commerce) 환경을 통해 상품이나 서비스를 구매하는 일이 점차 보편화되고, 그 규모가 더욱더 확대되는 추세에 있다[1]. 전자 상거래를 지원하는 클러스터 컴퓨팅 시스템은 장시간에 걸쳐 대량의 트랜잭션을 처리해야 하며, 빈번한 통신 두절 및 데이터 유실 가능성이 높기 때문에, 현재 이와 같은 시스템의 가용도를 개선·확보하기 위한 연구가 활발하다[2, 3]. 일반적으로 컴퓨팅 시스템의 가용도를 높이기 위해서 가동중인 주(primary)서버의 역할을 대신 수행할 수 있는 여분(backup)서버를 두는 방법이 주로 사용되며, 주서버와 여분서버간의 작업전이(switchover) 방식에 따라 Active/Active 시스템과 Active/ Standby 시스템으로 구분하고 있다[4]. 위의 두 가지 방법 중에서 주서버와 여분서버가 동시에 가동되는 Active/ Active 방식은 주서버의 역할을 대체하는 시간이 매우 짧지만 여러 대의 기계를 동시에 가동해야 하므로, 시스템 운영비용이 많이 들며, 또한 장시간 가동으로 인해 서버들의 고장 발생 가능성이 커지게 된다. 반면에 특정 시간에 주서버만 가동되는 Active/Standby 방법의 경우는 Active/Active 방법에 비해서 작업전이 시간은 길지만 그 대신 비용이 적게 들고 시스템의 하드웨어적인 고장 발생 가능성이 상대적으로 낮다[5].

최근 클러스터 기술을 이용하여 고가용도 시스템을 구축하려는 연구가 활발히 진행되고 있으며, 고가용도 클러스터 시스템의 성공 여부는 하드웨어 및 소프트웨어의 복잡성으로 인한 클러스터 시스템 가용도 저하 문제 해결에 있다[6-11]. 고가용도 시스템을 구축하기 위해서는 시스템의 고

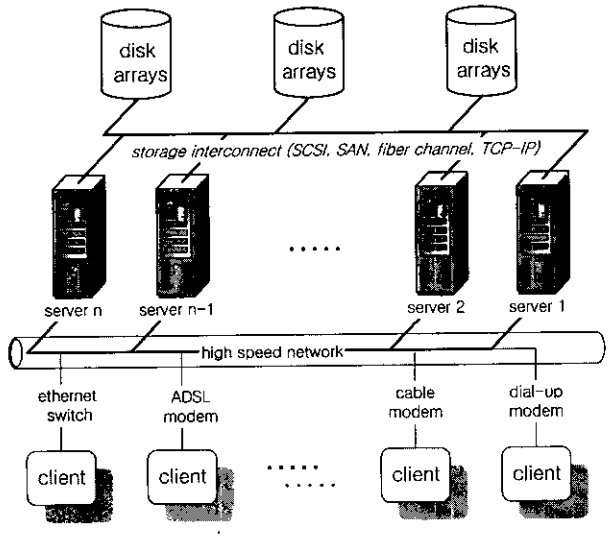
※ This work is supported in part by the Ministry of Information & Communication of Korea("Support Project of University Foundation Research<'00>" supervised by IITA).

†준 회 원 : 아주대학교 대학원 컴퓨터공학과
††정 회 원 : 아주대학교 정보통신전문대학원 교수

논문접수 : 2000년 11월 13일, 심사완료 : 2001년 2월 6일

장 발생 가능성을 미리 예측해서 대비하는 것이 중요하며, 고가용성 시스템에서 고장이 날 경우, 이를 얼마나 신속하게 복구할 수 있는지가 핵심 요소이다. 일반적으로 시스템의 가용도는 주로 1년 동안의 가동시간으로부터 계산되며, 만약 1년에 5분 내외의 다운타임(가용도 : 0.99999)을 가지면 고가용성 시스템이라고 말한다. 예를 들면, 소규모 인터넷 서비스 시스템의 가용도를 확보하기 위해서 2대의 서버로 구성된 이중계(duplex) 클러스터 시스템을 구축할 경우, 1년에 9시간 이하(가용도 : 0.999)의 다운타임 성능으로 서비스를 제공할 수 있다[12]. 따라서 가용도가 엄격한 시스템(교환기, CTI 서버, 멀티미디어 서버)에서는 Active/Active 방식을 사용해야 하지만 어느 정도까지 작업전이 지연을 허용하는 시스템(전자상거래서버, 웹서버)에서는 Active/Standby 방식을 사용하는 것이 시스템 운영비용 측면에서 볼 때 바람직하다.

(그림 1)은 일반적인 Active/Active 클러스터 시스템의 구성을 나타낸다. 클라이언트와 서버는 ADSL(asymmetric digital subscriber line), 케이블 및 랜 등의 고속 가입자망을 통해 연결되며, 서버의 데이터는 SCSI(small computers system interface)나 광 채널 인터페이스 등을 통해 하드디스크 등의 저장 장치에 저장된다. 고가용성 서비스 시스템 분야(인터넷 뱅킹, 증권, 교환기 등)의 가용도를 확보하기 위해, 여분서버의 대수를 늘리는 하드웨어적인 결함허용 방법과 하드웨어의 증설이 불필요한 비용 효율적인 소프트웨어 재활(rejuvenation) 방식의 소프트웨어 결함허용 기법[13-20]이 고려될 수 있으며, 본 논문에서는 고가용도 클러스터 시스템의 결함을 사전에 예방할 수 있고, 별도의 하드웨어 자원을 요구하지 않으며, 결함으로 인해 발생하는 손실비용을 최소화 할 수 있는 소프트웨어 재활 결함허용 기법에 관하여 기술하였다.

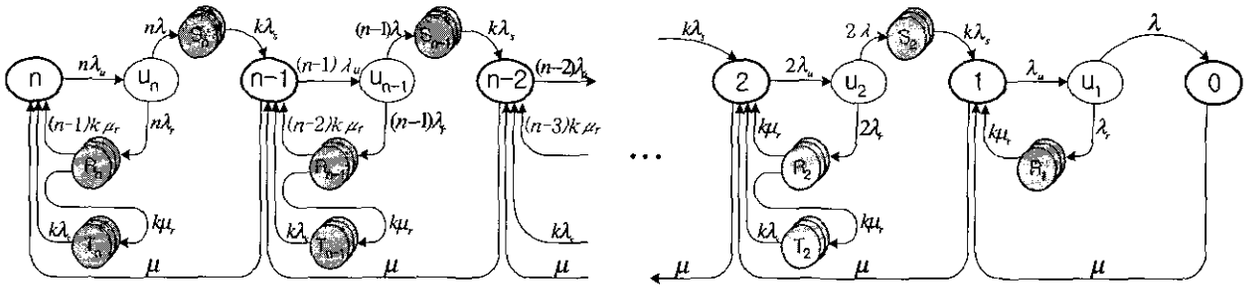


(그림 1) Active/Active 클러스터 시스템의 구성

2. 관련 연구

하드웨어 기술의 발전으로 인해 컴퓨터 하드웨어의 결함 발생률은 상수 값이거나 점차 작아지는 경향이 있다. 반면에 하드웨어에 탑재된 소프트웨어의 복잡성 및 크기는 이전에는 상상할 수 없을 정도로 방대해져가고 있기 때문에, 소프트웨어의 결함 발생으로 인한 컴퓨터 시스템의 장애 발생 가능성은 점차 더 높아지고 있다. 여러 대의 서버가 동시에 가동 상태에 있는 Active/Active 클러스터 시스템의 설계, 구현, 운영 또는 그 밖의 여러 가지 원인과 관련된 결함으로 인하여 클러스터 시스템 서비스의 오동작 또는 수행 중단 가능성이 점차 커지고 있다. 조사된 연구에 따르면 소프트웨어적 원인으로 인한 서비스 장애가 전체의 62%를 차지하는 것으로 보고되었고[21] 앞으로 소프트웨어의 결함에 의한 컴퓨터의 장애는 그 비중이 점차로 늘어날 전망이다. 클러스터 시스템의 서비스 중단으로 인해 야기되는 가용도 저하 및 손실 비용 발생을 최소화하기 위해, 장시간 시스템 가동으로 인한 소프트웨어 노화 현상(memory leak, accumulated round off error, buffer overflow 등)의 진행을 고의로 정지시켜, 클러스터 시스템이 결함 발생 가능성이 적은 건강한 상태에서 재출발하도록 하는 소프트웨어 재활에 의한 결함 예방 방법은 고가용성이 요청되는 Active/Active 클러스터 시스템 응용 분야에서 사용될 가능성이 높다고 볼 수 있다.

본 논문은 이미 발표된 본 저자들의 연구 테마[22]에 대한 계속 연구로서, Active/Active 클러스터 시스템의 가용도 개선을 위해서 소프트웨어적인 결함 발생을 미연에 방지할 수 있는 능동적 결함허용 기법인 소프트웨어 재활 방법에 대한 내용을 포함한다. 구체적으로, Active/Active 클러스터 시스템의 소프트웨어 재활 과정 및 여분서버로의 작업전이 과정을 semi-Markov 프로세스로 모델링 하였다. Markov 프로세스는 모든 상태에서 머무는 시간을 지수 분포로 가정하고 있다. 하지만 확정적으로 머무는 시간(deterministic sojourn time)이 소요되는 재활 과정 및 작업 전이 과정은 지수 분포의 memoryless property 적용이 적합하지 않으므로, 임의의 상태에서 머무는 시간의 분포가 지수 분포가 아닌 경우를 표현할 수 있는 semi-Markov 프로세스로 모델링 하였다. 수학적 분석을 통해 구한 시스템의 평형상태 확률을 이용하여, 다양한 운영 조건하의 가용도 및 손실비용을 계산하였다. 이를 통하여 소프트웨어 재활을 통한 Active/Active 클러스터 시스템의 가용도 개선 가능성을 확인하였다. 본 논문의 제2장에서는 관련 연구 결과와 문제를 정의하였으며, 제3장에서는 소프트웨어 재활 기법을 적용한 고가용도 Active/Active 클러스터 시스템 모델을 정의하고, 제4장에서는 제안된 모델의 수학적 해석 방법의 정확성을 검증하기 위한 다양한 시스템 운영 조건에 대한 실험을 수



(그림 2) Active/Active 클러스터 시스템 상태 전이도(state transition diagram)

행하였다. 마지막으로 결론부에서는 제안된 소프트웨어 재활 기법의 활용 방안 및 향후 연구에 관하여 논하였다.

3. Active/Active 클러스터 시스템 모델

소프트웨어 재활을 고려한 Active/Active 가동 방식에 따른 고가용성 클러스터 시스템의 상태 모델을 (그림 2)에 나타내었으며, 시스템 모델링에 사용된 기본적인 가정들은 다음과 같다.

- n개의 가동중인 서버로 구성된 Active/Active 클러스터 시스템에서 각 서버의 고장률(λ)은 모든 가동 상태에서 동일하다.
- 고장난 서버의 수리률(μ)은 모든 가동 상태에서 동일하다.
- 서버의 가동을 주기적으로 멈추는 재활률(λ_r)은 모든 가동 상태에서 동일하다.
- 장시간 가동으로 인한 서버의 불안정률(λ_u)은 모든 가동 상태에서 동일하다.
- 재활 작업 시간은 평균값 $1/\mu_r$인 k-stage Erlangian 분포를 따른다.
- 주서버에서 여분서버로의 작업 전이시간은 평균값 $1/\lambda_s$인 k-stage Erlangian 분포를 따른다.
- 재활이 완료된 여분서버들 중 한 대는 가동중인 주서버의 작업을 전이 받아 주서버 역할을 수행한다.
- 재활 상태와 작업전이 상태를 제외한 모든 상태에서 머무는 시간은 지수분포를 따른다.

정상 상태(normal state)에서 가동되고 있는 서버는 n, n-1, …, 1, 0 등의 가동 중인 서버의 수를 상태 변수로 가지고 있으며, 장시간 가동으로 인해 성능이 저하된 불안정 상태(unstable state)의 서버는 u_n, u_{n-1}, …, u_2, u_1로 표시된다. 정상 상태에서 불안정 상태로의 변화율은 $i*\lambda_u$(i : 가동중인 서버의 수)로 표시되며, 이는 소프트웨어의 장기간 가동으로 인한 시스템의 불안정률을 반영한다. 불안정 상태에서는 $i*\lambda_r$의 변화율로 재활 상태에 들어갈 경우, 소프트웨어 노화 현상으로 인한 일시적 결함 발생 가능성은 대부분 제거될 수 있다. Active/Active 클러스터 시스템의 경우, 특정 시점을 기준으로 모든 서버가 가동되고 있기 때문에 불안정 상태(u_i)에서 고장 발생률 $i*\lambda$ 또한 가동 중인 서버 대수와 비례하도록 모델링 된다.

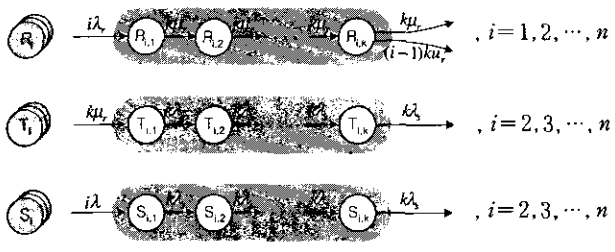
서버의 결함 발생과 여분서버로의 작업 대체 수행을 표시하는 작업전이 상태를 회색의 겹쳐진 원으로 표시된 S_n, S_{n-1}, …, S_3, S_2로 나타냈다. 서버의 대수가 1대인 단일계(simplex) 시스템에서 서버간 작업전이 과정은 불가능하므로 모델링에서 제외하였다. 또한 회색의 겹쳐진 원모양의 R_n, R_{n-1}, …, R_2, R_1은 재활 상태를 표시하며, 시스템의 가동을 고의로 중지시켜, 시스템이 결함 발생 가능성이 작은 건강한 상태로 되돌리는 것을 나타낸다. 재활이 완료된 여분 서버들 중 한 대가 가동중인 주서버의 작업을 전이 받아 주서버 역할을 수행하는 상태를 T_n, T_{n-1}, …, T_3, T_2로 모델링 하였다. (그림 2)의 모델링은 재활 상태와 작업전이 상태에서 머무는 시간(sojourn time)의 분포가 memoryless 성질을 만족하지 않는 k-stage Erlangian 분포를 따르기 때문에, semi-Markov 프로세스 문제로 분류되어 정확한 해를 구하는 간단한 방법이 존재하지 않는다[23, 24].

3.1 Method-of-stage 기법

임의의 상태에 머무는 시간이 확정적이라는 조건을 만족시켜야 하는 프로세스 상태 모델링에서, 머무는 시간 분포(sojourn time distribution)의 분산을 작게 할 경우, 머무는 시간을 확정적으로 표현할 수 있다. 분산이 큰 임의의 한 상태에 머무는 시간 분포를 여러 개의 stage로 나누어서 모델링(method-of-stage)할 경우, 결국에는 stage의 수에 따라 머무는 시간 분포의 평균은 동일해지고, 분산이 감소하게 된다. 일반적으로 시스템을 재활하여, 깨끗한 상태로 만드는데 소요되는 시간(rebooting time 혹은 purging time)은 일정한 상수 값을 가지며, 주서버에서 여분서버로의 작업전이 시간 또한 일정 시간 이내에 이루어져야 하는 조건을 가지고 있다. 이와 같이 현실적인 고가용성 클러스터 시스템의 운영 상태를 반영하는 모델의 수학적 해를 구하기 위해서, (그림 2)에서 회색으로 칠해진 재활 상태와 작업전

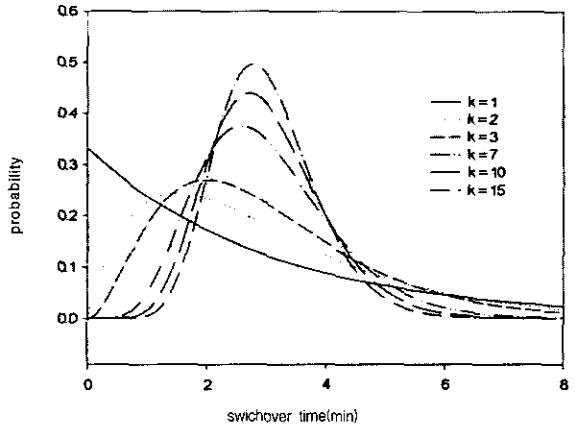
이 상태에서 머무는 시간을 임의의 수 k개로 나눈 분할 상태 전이도로 모델링 하였다((그림 3) 참조).

예를 들면, 임의의 작업전이 상태 S_i는 k개의 분할 상태를 가지는 부분 작업전이 상태(sub-switchover state: $S_{i,j}$)들로 표현되며, 각 부분 작업전이 상태에서의 작업전이 소요 시간은 평균 $1/k\lambda_s$인 지수분포를 따른다.



(그림 3) sojourn time의 분할 상태 전이도

(그림 3)의 분할 상태 전이도는 k개의 지수분포의 합, 즉 k-stage Erlangian 분포로 표현되며, k-stage Erlangian 분포의 성질 중 상태 분할 개수(k)가 증가함에 따라 확률 분포의 분산 값이 점점 작아져서 결국에는 1의 확률로 unit impulse 값을 가지는 사실을 적용하여((그림 4) 참조)[23], 본 모델링에서 요구되는 재활 상태와 작업전이 상태에서의 확정 값을 가지는 sojourn time 문제를 해결하였다.



(그림 4) k 값에 대한 Erlangian 분포의 분산 변화 추세
(평균 작업전이 시간 : 3분)

(그림 2)의 모든 상태에서 입력과 출력 비율이 일정해지는 평형(steady-state)일 때의 균형 방정식(balance equation)은 다음과 같다.

$$n\lambda_u P_n = \mu P_{n-1} + (n-1)ku_r P_{R_{n,k}} + k\lambda_s P_{T_{n,k}}$$

$$(u+i\lambda_u)P_i = \mu P_{i-1} + (i-1)ku_r P_{R_{i,k}} + k\lambda_s P_{T_{i,k}} + k\lambda_s P_{S_{i+1,k}}$$

$$i = 2,3,\cdots,n-1$$

$$(u+\lambda_u)P_1 = \mu P_0 + ku_r P_{R_{1,k}} + k\lambda_s P_{S_{2,k}}$$

$$\mu P_0 = \lambda P_{u_1}$$

$$(\lambda+\lambda_r)P_{u_i} = \lambda_u P_i,\ i=1,2,\cdots,n$$

$$k\mu_r P_{R_{i,j}} = i\lambda_r P_{u_i},\ i=1,2,\cdots,n,\ j=1,2,\cdots,k-1$$

$$k\mu_r P_{R_{i,k}} = \lambda_r P_{u_i},\ i=1,2,\cdots,n$$

$$\lambda_s P_{T_{i,j}} = \mu_r P_{R_{i,k}},\ i=2,3,\cdots,n,\ j=1,2,\cdots,k$$

$$k\lambda_S P_{S_{i,j}} = i\lambda P_{ui},\ i=2,3,\cdots,n,\ j=1,2,\cdots,k$$

$$\sum_{i=0}^{n} P_i + \sum_{i=1}^{n} P_{u_i} + \sum_{i=1}^{n}\sum_{j=1}^{k} P_{R_{i,j}} + \sum_{i=2}^{n}\sum_{j=1}^{k}(P_{S_{i,j}} + P_{T_{i,j}}) = 1$$

; conservation 방정식

각 state 확률의 의미는 아래와 같다.

P_i : 시스템이 평형일 때, 정상 상태 i에 머물 확률

P_{u_i} : 시스템이 평형일 때, 불안정 상태 i에 머물 확률

$P_{R_{i,j}}$: 평형 시스템의 서버 개수가 i이고 재활 작업 시에, 추가로 거쳐야 할 부분재활 상태의 개수가 k-j개 남아 있을 확률

$P_{S_{i,j}}$: 평형 시스템의 서버 개수가 i일 때, 서버의 고장으로 인한 작업전이 시에, 추가로 거쳐야 할 부분작업전이 상태의 개수가 k-j개 남아있을 확률

$P_{T_{i,j}}$: 평형 시스템의 서버 개수가 i일 때, 재활이 완료된 서버 중 하나가 기존에 가동중인 서버의 작업을 대신 수행하기 위한 작업전이 시에, 추가로 거쳐야 할 부분작업전이 상태의 개수가 k-j개 남아있을 확률

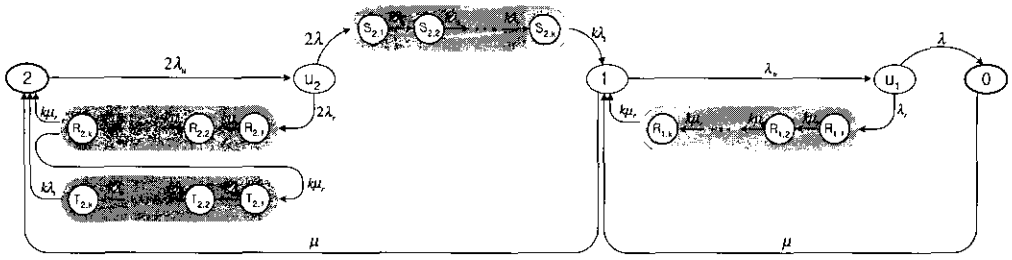
위의 균형 방정식과 각 상태에서 머물 확률의 총합이 1이 되는 conservation 방정식을 결합한 연립 방정식을 풀면, 시스템이 평형일 때, 각 상태에 머물 확률을 얻을 수 있다.

- 이중계 시스템 이상일 때($n \geq 2$)

$$P_n = \Bigg[\, n!\Bigg\{\Bigg(1+\frac{\lambda_u}{\lambda+\lambda_r}\Bigg(1+\frac{\lambda_r}{ku_r}\Bigg)\Bigg)\sum_{i=1}^{n}\frac{1}{i!}\Bigg(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg)^{n-i} + \frac{(k-1)\lambda_r}{k\mu_r}\frac{\lambda_u}{\lambda+\lambda_r}\sum_{i=1}^{n}\frac{1}{(i-1)!}\Bigg(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg)^{n-i} + \frac{\lambda_r}{\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}\sum_{i=2}^{n}\frac{1}{i!}\Bigg(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg)^{n-i} + \frac{\lambda}{\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}\sum_{i=2}^{n}\frac{1}{(i-1)!}\Bigg(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg)^{n-i} + \Bigg(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg)^{n}\Bigg\}\Bigg]^{-1}$$

- 단일계 시스템일 때($n=1$)

$$P_1 = \Bigg[\, 1+\frac{\lambda_u}{\lambda+\lambda_r}\Bigg(1+\frac{\lambda_r}{ku_r}\Bigg) + \frac{(k-1)\lambda_r}{k\mu_r}\frac{\lambda_u}{\lambda+\lambda_r} + \frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\Bigg]^{-1}$$



(그림 5) Active/Active 이중계 클러스터 시스템의 상태 전이도

$$=\left[1+\frac{\lambda u}{\lambda+\lambda_r}\left(1+\frac{\lambda_r}{u_r}+\frac{\lambda}{u}\right)\right]^{-1}$$

$$P_i=\frac{n!}{i!}\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)^{n-i}P_n,\quad i=0,1,2,\cdots,n$$

$$Pu_i=\frac{\lambda_u}{\lambda+\lambda_r}P_i,\quad i=1,2,\cdots,n$$

$$P_{R_{i,j}}=\frac{i\lambda_r}{ku_r}\frac{\lambda_u}{\lambda+\lambda_r}P_i,\quad i=1,2,\cdots,n,\ j=1,2,\cdots,k-1$$

$$P_{R_{i,k}}=\frac{\lambda_r}{ku_r}\frac{\lambda_u}{\lambda+\lambda_r}P_i,\quad i=1,2,\cdots,n$$

$$P_{S_{i,j}}=\frac{i\lambda}{k\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}P_i,\quad i=2,3,\cdots,n,\ j=1,2,\cdots,k$$

$$P_{T_{i,j}}=\frac{\lambda_r}{k\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}P_i,\quad i=2,3,\cdots,n,\ j=1,2,\cdots,k$$

즉 시스템 운영 파라미터를 이용하여 P_n을 계산하면, 그 외의 모든 정상 상태(P_i), 불안정 상태(P_{u_i}), 재활 상태($P_{R_{i,j}}$) 및 작업전이 상태($P_{S_{i,j}}$, $P_{T_{i,j}}$)에서의 확률을 차례로 계산할 수 있다. (그림 5)는 서버의 중복도를 2로 한 Active/Active 이중계 클러스터 시스템의 상태 전이도이며 시스템의 가동이 평형 상태일 때의 확률은 다음과 같다.

3.1.1 정상 상태 :

$$P_2+P_1=\left(1+2\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)\right)P_2$$

$$=\left(1+2\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)\right)$$

$$\left[2\left\{\left(1+\frac{\lambda_u}{\lambda+\lambda_r}\left(1+\frac{\lambda_r}{ku_r}\right)\right)\sum_{i=1}^{2}\frac{1}{i!}\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)^{2-i}\right.\right.$$

$$+\frac{(k-1)\lambda_r}{ku_r}\frac{\lambda u}{\lambda+\lambda_r}\sum_{i=1}^{2}\frac{1}{(i-1)!}\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)^{2-i}$$

$$\left.\left.+\frac{1}{2}\frac{\lambda_r}{\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}+\frac{\lambda}{\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}+\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)^2\right\}\right]^{-1}$$

3.1.2 불안정 상태 :

$$P_{u_2}+P_{u_1}=\frac{\lambda_u}{\lambda+\lambda_r}P_2+\frac{\lambda_u}{\lambda+\lambda_r}P_1$$

3.1.3 재활 상태 :

$$\sum_{j=1}^{k}(P_{r_{2,j}}+P_{r_{1,j}})=\sum_{j=1}^{k-1}\left(\frac{2\lambda_r}{ku_r}\frac{\lambda_u}{\lambda+\lambda_r}P_2+\frac{\lambda_r}{ku_r}\frac{\lambda_u}{\lambda+\lambda_r}P_1\right)$$

$$+\frac{\lambda_r}{ku_r}\frac{\lambda_u}{\lambda+\lambda_r}(P_2+P_1)$$

3.1.4 작업전이 상태 :

$$\sum_{j=1}^{k}P_{S_{2,j}}=\frac{2\lambda}{k\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}P_2,\ \sum_{j=1}^{k}P_{T_{2,j}}=\frac{\lambda}{k\lambda_s}\frac{\lambda_u}{\lambda+\lambda_r}P_2$$

3.1.5 고장 상태 :

$$P_0=2\left(\frac{\lambda}{u}\frac{\lambda_u}{\lambda+\lambda_r}\right)^2P_2$$

불안정 상태에서 가동되고 있는 서버(ui)는, $i*\lambda$의 변화율로 하드웨어적인 고장이 발생하며, 서버 두 대의 가동이 모두 중지된 고장 상태(0)에서는 μ의 변화율로 고장이 수리된다. 서버의 장시간 가동으로 인한 소프트웨어 노화로 인해 서버의 성능이 저하되는 불안정 상태에 있을 경우, 고의로 시스템의 가동을 멈추는 재활 상태(R_{2j}, R_{1j})로 가거나 혹은 고장이 발생하게 된다. 주서버의 고장으로 인해 발생한 여분서버로의 작업전이는 작업전이 상태(S_{2j})를 통해 이루어진다. 한 대의 여분 서버가 재활 작업을 마친 후에 가동 중인 주서버로부터 작업을 전이 받는 과정은 작업전이 상태(T_{2j})를 통해 이루어진다. 서버가 한 대인 단일계 시스템의 경우에는 여분서버가 없으므로, 작업전이 상태가 모델링에 포함되지 않는다

4. 실험 및 성능 평가

Active/Active 클러스터 시스템 응용 분야의 특성에 적합한 소프트웨어 재활 방식을 결정하기 위하여, 재활률, 고장률, 수리률, 가동 시간 및 손실 비용 등 시스템 운영 변수를 복합적으로 고려하였다. 서버의 가동이 중지되는 down-time의 절대적 크기보다는, 이로 인해 발생하는 손실 비용이 더욱 중요한 요소임을 고려하여 가용도 및 손실 비용을 계산하였다.

4.1 가용도 및 손실 비용

소프트웨어 재활 기법을 적용한 Active/Active 클러스터 시스템의 가용도(Avail)는 모든 서버의 동작이 멈춘 고장

상태(P0), 시스템 서비스가 일시적으로 중지되는 단일계 시스템의 재활 상태 및 모든 작업전이 상태에서 머물 확률을 제외함으로써 계산된다.

$$Avail = 1 - \left(P_0 + \sum_{j=1}^{k} P_{R_{1,j}} + \sum_{i=2}^{n} \sum_{j=1}^{k} (P_{T_{i,j}} + P_{S_{i,j}}) \right)$$

클러스터 시스템의 갑작스런 가동 정지로 인한 단위 시간당 손실 비용을 C_f, 재활 작업으로 인한 단위 시간당 손실 비용을 C_r, 작업전이로 인한 단위 시간당 손실 비용을 C_s라 할 경우, 일반적으로 예상 가능한 시스템 정지 비용은 불시 정지로 인한 손실 비용에 비해 훨씬 저렴하게 들므로, $C_f >> C_r$의 관계가 성립하며, 작업전이 비용은 C_f 보다는 작다고 가정한다. Active/Active 클러스터 시스템의 가동 시간에 대한 손실 비용(Cost)은 서버의 가동시간(T)의 함수로 아래와 같이 정의된다.

$$Cost(T) = \left[C_f \times P_0 + C_r \times \sum_{j=1}^{k} P_{R_{1,j}} + C_s \times \sum_{i=2}^{n} \sum_{j=1}^{k} (P_{T_{i,j}} + P_{S_{i,j}}) \right] \times T$$

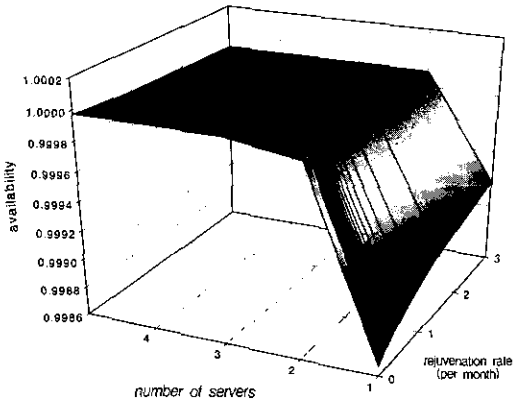
4.2 성능 분석

실험에 사용된 Active/Active 이중계 클러스터 시스템의 운영 파라미터는 <표 1>과 같다[22]. 클러스터 시스템의 연속 가동기간(T)은 1년으로 하며, 서버의 고장(λ)은 1년에 1회, 고장 수리(μ)에 12시간이 소요되고, 한 달에 한번씩 재활작업(λ_r)을 수행하며, 이때 평균 10분($1/\mu_r$)이 소요된다. 주 서버의 고장으로 인해 발생하는 여분서버로의 작업전이 시간($1/\lambda_s$)은 평균 3분이 소요되며, 시스템의 불시정지로 인해 발생하는 비용(C_f)은 재활 작업시 발생하는 비용(C_r)의 1000배가 되며, 작업전이 비용(C_s)은 불시정지 비용의 10%이다.

〈표 1〉 시스템 운영 파라미터[22]

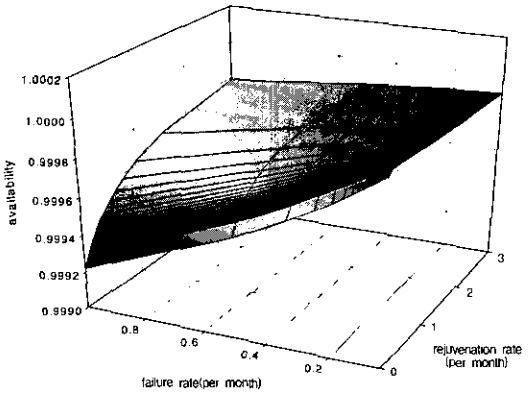
시스템 운영 방식(n)	2 (이중계)
연속 가동기간(T)	1년
서버 고장률(λ)	1회/년
서버 수리률(μ)	2회/일
서버 불안정률(λ_u)	1회/15일
재활률(λ_r)	1회/월
재활작업 시간($1/\mu_r$)	10분
작업전이 시간($1/\lambda_s$)	3분
downtime 비용(C_f)	1000
rejuvenation 비용(C_r)	1
작업전이 비용(C_s)	100
분할 stage 수(k)	20

소프트웨어 재활의 효과를 검증하기 위해서, Active/Active 클러스터 시스템을 구성하고 있는 서버의 재활률에 대한 가동 대수(단일계, 이중계 및 다중계), 고장률, 수리 시간, 손실비용률, 작업전이 시간 및 불안정률에 따른 가용도와 손실비용 변화 추세를 입체적으로 파악함으로써, 시스템 운영 변수들에 대한 가용도와 손실비용 변화의 민감도(sensitivity)를 효과적으로 나타내었다.



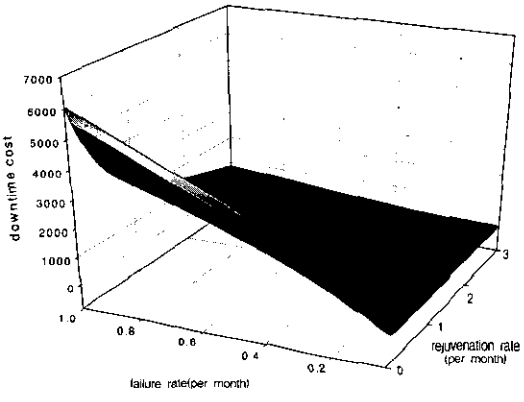
(그림 6) 재활률과 서버 수에 대한 가용도 변화

(그림 6)은 Active/Active 클러스터 시스템에서 소프트웨어 재활을 하지 않는 경우(재활률 0)와 재활 실시 주기를 10일 간격으로 약 1년까지(360일) 변화시키면서, 동시에 시스템을 구성하는 서버의 대수를 단일계에서 다중계(n = 5)까지로 증가시켰을 때, 시스템의 가용도를 변화 추세를 나타낸다. 재활을 자주 한다는 의미는 시스템의 불안정성을 사전에 제거하는 것을 의미하며, 이중계 이상의 Active/Active 시스템에서는 재활 작업으로 인한 서비스 중단이 발생하지 않기 때문에, 가용도가 재활률에 비례하는 추세, 즉 재활을 자주 할수록 가용도가 증가하는 현상을 나타냈다. 한편 한 대의 서버만이 가동되는 단일계 시스템의 가용도에 비해서 이중계 이상의 고가용성 클러스터 시스템의 가용도가 높게 나오는 것을 확인할 수 있으며, 이중계 이상의 서버 구성을 통해 개선될 수 있는 가용도의 상승폭은 미미함을 그래프에서 확인할 수 있다. 이는 일반적으로 고가용도를 확보하기 위해 이중계 클러스터 시스템 구성이 가장 비용 효율적임을 나타낸다.



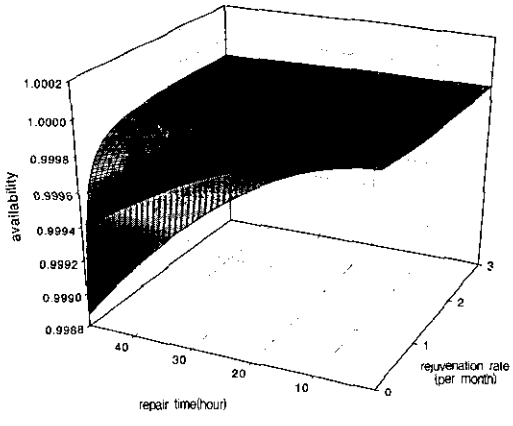
(그림 7) 재활률과 고장률에 대한 가용도 변화

서버의 하드웨어적 고장 간격(1개월~2년)과 재활률이 클러스터 시스템의 가용도와 손실비용에 미치는 영향을 (그림 7)과 (그림 8)에 나타내었다. 이중계 이상의 클러스터 시스템을 구성할 경우, 하드웨어 고장률이 시스템의 가용도에 미치는 영향의 변화 폭(0.9993~0.9994)이 작았으나, 재활률의 변화에 대해서는 비교적 민감하게(0.9992~0.9998) 반응하였다. 이는 서론부에서도 밝힌 바와 같이, 하드웨어의 신뢰도보다는 소프트웨어의 신뢰도가 클러스터 시스템의 가용도 개선에 더 중요한 역할을 수행하고 있는 실례라 할 수 있다.



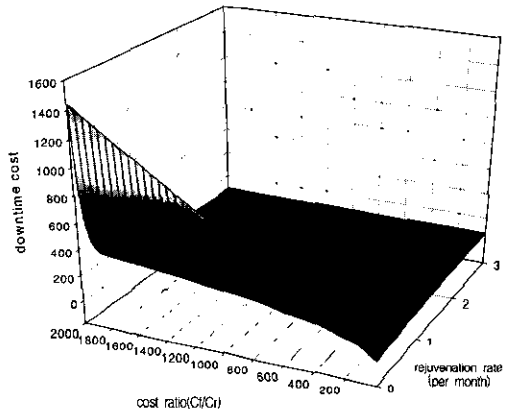
(그림 8) 재활률과 고장률에 대한 손실비용 변화

(그림 8)은 소프트웨어 재활 기법의 현실 문제 적용 가능성의 예를 보여주는 그래프로서, 재활 및 작업전이로 인해 발생하는 시스템 정지 비용에 비해, 불시의 시스템 정지로 인해 발생하는 비용이 클 경우, 소프트웨어 재활을 자주 하면 할수록 손실 비용이 감소하는 추세를 나타내고 있다. 이 그래프에서 소프트웨어 재활을 수행할 경우 시스템 운영자에게 더욱 중요한 성능 평가 요소인 손실비용 절감의 가능성을 제시하고 있다. 일반적으로 고장이 자주 발생하면 발생할수록 손실비용이 증가하는 반면에, 재활률과 손실비용은 서로 반비례하는 추세를 확인하였다.



(그림 9) 재활률과 수리 시간에 대한 가용도 변화

(그림 9)는 서버의 고장수리 능력(1시간~48시간)과 재활률이 시스템의 가용도에 미치는 영향을 표시한다. 고장수리 시간이 빠르면 빠를수록 시스템의 가용도가 급격히 증가함을 표시하고 있으며, 또한 재활률이 높은 경우, 고장 수리 능력이 가용도에 미치는 영향이 작음을 나타내고 있다. 고장 수리 능력이 12 시간이내의 경우에는 가용도의 개선 정도가 그리 크지 않으므로, 시스템의 하드웨어적인 고장 수리 능력을 결정할 때에 (그림 9)는 중요한 참고 지표로서 사용될 수 있다.

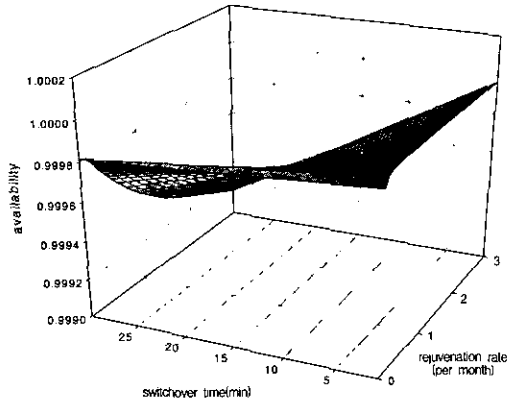


(그림 10) 재활률과 손실비용률에 대한 손실비용 변화

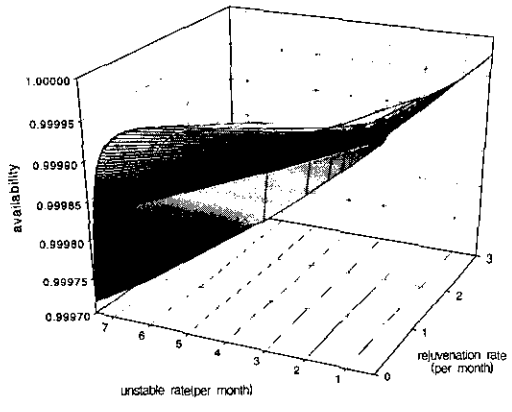
(그림 10)은 불시의 고장으로 인한 단위 손실비용(C_l)과 소프트웨어 재활에 의한 예정된 서비스 중단으로 인한 단위 손실비용(C_r)의 비율과 재활률 변화에 대한 손실비용의 변동 추세를 나타낸다. 재활을 자주할 경우에는 시스템 가용도의 증가로 인해 손실비용의 변화가 거의 없으나, 재활률이 작을 경우, 손실비용률의 변화에 대해 손실비용이 상당히 급격하게 변화하는 것을 나타내고 있다. 또한 재활률이 작을 경우, 손실비용이 손실비용률에 선형적으로 증가하는 추세를 보이고 있다.

(그림 11)은 작업전이 시간과 재활률이 클러스터 시스템에 미치는 영향을 나타낸다. 작업전이 시간을 0초에서 30분까지 변화 시켰을 때, 가용도의 변화는 재활률과 밀접한 관련이 있음을 보이고 있다. 즉 클러스터 시스템을 소프트웨어 재활 정책을 사용하여 운영할 경우, 작업전이 시간 약 15분을 기준으로, 작업전이 시간이 15분 이내일 경우 재활을 하면 할수록 가용도가 높아지는 반면에 작업전이 시간이 15분 이상인 경우, 재활을 하면 할수록 가용도가 감소하는 것을 알 수 있다. 작업전이 시간이 짧으면, 시스템에 하드웨어적 고장 혹은 소프트웨어적인 재활 작업이 발생하여도, 바로 서비스가 가능하게 되므로, 소프트웨어 재활의 효과는 작업전이 시간과 밀접한 관련이 있다. 작업전이 시간이 평균 10분이 소요되고 작업전이 상태의 분할 개수가 20일 때, 각 부분 작업전이 상태의 평균작업전이 시간은 30초

가 된다. 20개의 부분작업전이 시간을 모두 합한 20-stage Erlangian 분포의 평균은 지수 분포(k = 1)와 동일한 10분이 되지만, 분산값은 지수 분포일 때(100분)의 5%인 5분으로 줄게 된다. 만약 k를 더욱 증가시킬 경우, k-stage Erlangian 분포의 분산값이 k의 비율로 줄게되어 원하는 확정시간을 가지는 작업전이 시간을 표현할 수 있게 된다.



(그림 11) 재활률과 작업전이 시간에 대한 가용도 변화



(그림 12) 재활률과 불안정률에 대한 가용도 변화

(그림 12)는 시스템의 장시간 가동으로 인한 시스템의 불안정성과 재활률이 가용도에 미치는 영향을 나타낸다. 불안정률이 높을 경우(6 이상)에는 재활을 자주하면 할수록 오히려 가용도가 감소하는 현상을 나타내고 있으며, 이는 불안정률과 재활률이 상승 작용을 하여, 시스템 서비스가 불가능한 고장 상태(P_0)의 확률을 높게 만들기 때문이라 생각된다.

5. 결 론

본 연구에서는 Active/Active 클러스터 시스템의 가용도 개선을 위해, 소프트웨어 재활 결함허용 기법을 적용하였으며, 현실에 좀더 근접한 시스템 가용도를 모델링하기 위해서, method-of-stages 개념을 적용하였다. 즉 재활 상태와 작업전이 상태에서 머무는 시간을 지수분포들의 합으로 표현한 k-stage Erlangian 분포를 사용하고, 각 상태 분할 개수(k)를 증가시킴으로서 재활 작업과 작업전이 시간을 확정적으로 표현하였다. 본 논문의 상태 전이도는 임의의 상태에서 머무는 시간분포가 memoryless 성질을 만족하지 않는 semi-Markov 프로세스 문제이며, method-of-stages 방법을 통해 구한 Active/Active 클러스터 시스템 재활 모델의 closed-form 해는 다양한 시스템 운영 상태에 대한 실험을 통해 검증하였으며, 소프트웨어 재활을 통한 Active/Active 클러스터 시스템의 가용도 개선이 가능하다는 것을 확인하였다. 따라서 소프트웨어 재활 방법은 컴퓨터 자원의 추가 비용이 없고, 최근 시스템 비용 중 가장 커다란 비중을 차지하는 소프트웨어 유지보수 비용을 상당히 줄일 수 있는 최선의 해결책으로 판단된다. 추후에는 고가용성 부하분산 클러스터 시스템의 성능 평가 모델을 연구할 예정이고, 그 밖에 소프트웨어 재활 기법을 적용한 시스템의 부하 및 가동능력(performability)을 동시에 고려할 수 있는 방안[25, 26]에 대한 연구가 요청된다.

참 고 문 헌

[1] 김춘길, "전자상거래의 개념과 발전방향", 정보과학회지, 제16권 제5호, pp.5-10, 1998. 5.

[2] Y. Maya and A. Ohtsuji, "High-availability Scheme Using Data Partitioning for Cluster Systems," IEICE Transactions on Information & Systems, Vol. E82-D, No.11, pp.1457-1465, Nov. 1999.

[3] M.R. Lyu and V.B. Mendiratta, "Software Fault Tolerance *in a Clustered Architecture: Techniques and Reliability* Modeling," Proceedings of the 1999 IEEE Aerospace Conference, Vol.5, pp.141-149, Mar. 1999.

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

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

[6] H. Zhu, T. Yang, Q. Zheng, D. Watson, O. Ibarra and T. Smith, "Adaptive Load Sharing for Clustered Digital Library Servers," Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, pp. 225-235, July 1998.

[7] N. Talagala and D. Patterson, "An Analysis of Error Behavior in a Large Storage System," Annual IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems, San Juan, Puerto Rico, USA, April, 1999.

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

[9] G. F. Pister, "In Search of Cluster," Prentice-Hall, 1998.
[10] 오수철, 정상화, "클러스터 시스템 기술 동향", 정보과학회지, 제18권 제3호, pp.4-10, 2000. 3.
[11] 유찬수, "리눅스 클러스터링", 정보과학회지, 제18권 제2호, pp.33-39, 2000. 2.
[12] R. Jain, *The Art of Computer Systems Performance Analysis.* p.685, John Wiley & Sons Inc., 1991.
[13] S. Garg, A. Puliafito, M. Telek and K. Trivedi, "On the Analysis of Software Rejuvenation Policies," Proc. 12th Annual Conference on Computer Assurance (COMPASS), June 1997.
[14] 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, Jan. 1998.
[15] A. Pfening, S. Garg, M. Telek, A. Puliafito and K. Trivedi, "Optimal Rejuvenation for Tolerating Soft Failures," Performance Evaluation, Vol 27 & 28, North-Holland, pp.491-506, Oct. 1996.
[16] Y. Huang, C. Kintala, N. Kolettis and N. Fulton, "Software Rejuvenation : Analysis, Module and Applications," Proceedings of the 25th International Symposium on Fault Tolerant Computing (FTCS-25), Pasadena, CA, pp.381-390, June 1995.
[17] K. Vo, Y. Wang, P. Chung, and Y. Huang, "Xept : A Software Instrumentation Method for Exception Handling," in Proc. Int. Symp. on Software Reliability Engineering, Nov. 1997.
[18] S. Garg, Y. Huang, C. Kintala and K. Trivedi, "Time and Load Based Software Rejuvenation : Policy, Evaluation and Optimality," Proc. of the First Conference on Fault Tolerant Systems, Madras, India, Dec. 1995.
[19] Y. Huang, C. Kintala and Y. Wang, "Software Tools and Libraries for Fault Tolerance," Bulletin of the Technical Committee on Operating Systems and Application Environment (TCOS), Vol.7, No.4, pp.5-9, Winter 1995.
[20] K. Trivedi, K. Vaidyanathan and K. Goseva-Popstojanova, "Modeling and Analysis of Software Aging and Rejuvenation," Proceedings of the 33rd Annual Simulation Symposium, pp.270-279, Apr. 2000.
[21] I. Lee and R. Iyer, "Software Dependability in the Tandem GUARDIAN System," IEEE Transactions on Software Engineering, Vol.21, No.5, pp.455-467, May 1995.
[22] 박기진, 김성수, 김재훈, "소프트웨어 재활 기법을 적용한 다중계 시스템의 가용도 분석", 한국정보과학회논문지(시스템및이론), 제27권 제8호, pp.730-740, 2000. 8.
[23] L. Kleinrock, *Queueing Systems Volume 1 : Theory.* pp. 417, John Wiley & Sons Inc., 1975.
[24] K. Trivedi, *Probability and Statistics with Reliability, Queueing, and Computer Science Applications.* pp.624, Prentice-Hall, 1982.
[25] M. Sereno and G. Balbo, "Mean Value Analysis of Stochastic Petri Nets," Performance Evaluation, Vol.29, No.1, pp.1-28, Feb. 1997.
[26] Z. Liu, "Performance Analysis of Stochastic Timed Petri Nets Using Linear Programming Approach," IEEE Transactions on Software Engineering, Vol 24, No.11, pp.1014-1030, Nov. 1998.

박 기 진

e-mail : kiejin@yahoo.co.kr
1989년 한양대학교 산업공학과(공학사)
1991년 포항공과대학교 산업공학과(공학석사)
1991년~1996년 삼성종합기술원 기반기술 연구소(전임연구원)
1996년~1997년 삼성전자(주) 소프트웨어센타 선임연구원
1997년~현재 아주대학교 컴퓨터공학과 박사과정
관심분야 : 고가용성 클러스터 시스템, 멀티미디어 시스템, 결함허용 시스템, 성능 평가, 시뮬레이션

김 성 수

e-mail : sskim@madang.ajou.ac.kr
1982년 서강대학교 전자공학과(공학사)
1984년 서강대학교 전자공학과(공학석사)
1995년 Texas A&M University, Computer Science Dept.(공학박사)
1983년~1986년 삼성전자(주) 종합연구소 컴퓨터연구실(주임연구원)
1986년~1996년 삼성종합기술원(수석연구원)
1991년~1992년 Texas Transportation Institute(연구원)
1993년~1995년 Texas A&M University, Computer Science Dept.(T.A. & R.A.)
1997년~1998년 한국정보처리학회, 한국정보과학회 논문지 편집위원
1996년~현재 아주대학교 정보통신전문대학원 부교수
관심분야 : 클러스터 시스템, 결함허용, 성능 평가, 이동컴퓨팅, 멀티미디어 등