우선순위를 고려한 BCMP 큐잉 네트워크를 이용한 컴퓨터 시스템의 성능 분석

박 동 준† 이 상 훈†† 정 상 근†††

요 약

본 연구에서는 두개 이상의 프로세서로 구성된 컴퓨터 시스템을 순환 형태의 모델로 가정하여 우선순위를 고려한 BCMP 큐잉 네트워크 이론을 적용하여 성능 분석을 하였다. 터미날, 프로세서, I/O장치 등을 포함하는 컴퓨터 시스템에서 멀티프로그래밍 레벨을 최적화 하여 최대 처리율을 구하여 이 상태에서 성능을 비교하였고, 이러한 상태에서 최적의 서버 수와 터미날의 수를 구하여 , 두개 이상의 프로세서로 구성된 멀티프로세싱 시스템의 상태와 m 개의 멀티프로그래밍 레벨을 동시에 가지고 있는 시스템을 각 서버에 우선순위를 고려하여 분석하였다. 각각의 서버를 터미날, 프로세서, I/O 장치 등의 특성에 맞는 서버의 형태를 적용하고, 각 서버가 갖는 서비스 확률 분포에 따라 분석하였다. 우선순위를 고려하여 컴퓨터 시스템을 모델링하여 최적의 상태를 제시하여 성능 분석을 해서 부하가 많은 상태에서 컴퓨터의 효율을 높이고자 하였다.

Analysis of Performance for Computer System using BCMP Queueing Network with Priority Levels

Dong-Jun Park† Sang-Hun Lee†† and Sang-Geun Chung†††

ABSTRACT

In this paper, We assume that the closed computer system model composed of multiprocessor system is analyzed by BCMP queueing network theory with priority levels. In this system that contains terminals, processors and I/O devices, We show maximum throughput and the number of active terminals in the optimum multiprogramming levels. It is compared the performance with the other. In the result, it is obtained the optimum number of processors and active terminals. Therefore, the system model consisted of the optimum number of processors and multiprogramming level m is analyzed by the servers with a priority level. Each server is applied to the type of server which is characterized terminal, processor or I/O device etc.. This model is analyzed by the server with a probability ditribution. Ideal state is proposed by the modeling for priority levels. Finally, we try to increase the performance in overload system.

1. 서 론

중앙 컴퓨터, 단말기 및 입출력 장치들을 포함한 컴퓨터 시스템의 성능 분석은 개방 형태의 큐잉 이론으로 해석하였으나 최근에는 JACKSON 큐잉 이론의 확장 형태인 BCMP 큐잉 이론에 우선순위를 적용하여 시스템을 해석하고 성능 분석을 한다 [1, 2].

일반적으로 컴퓨터 시스템에서, 서비스되는 정보의 흐름을 기준으로 개방 시스템, 순환 시스템, 복합 시스템으로 분류한다[1, 2]. 개방 시스템은 통신망 외부로부터 입력되는 정보가 서비스 시스템에서 처리되어 다시 통신망 외부로 출력되는 형태이고, 순환 시스템은 통신망 외부로부터

† 정 회 원 : 광운대학교 전자공학과
†† 종신회원 : 광운대학교 전자계산 교육원
††† 종신회원 : 안양전문대학 전자계산학과
논문접수 : 1995년7월7일, 심사완료 : 1995년11월30일

입력되는 정보가 없으며 통신망 외부로 출력되는 정보도 없는 형태의 시스템으로서 일반적으로 온라인 시스템에서 터미날의 서비스 요구를 프로세서가 터미날에 다시 서비스 요구에 대한 서비스를 제공해 줌으로서 정보가 하나의 통신망 내에서 순환하는 형태이며, 그리고, 복합 시스템은 위의 두 가지 형태를 혼합한 형태로 분석하기에 다소 복잡하며 컴퓨터 시스템과 다양한 통신 수단이 동반된 형태이다.

본 연구에서는 순환 시스템을 모델로 설정하고 성능 분석을 위해 평균값 해석을 통해 터미날 및 서버의 최적의 수, 시스템 내에 존재하는 정보의 수, 대기 시간 등을 구한다[1, 4].

일반적으로 개방 시스템에서는 도착 과정을 포아송 분포로 가정하여 분석하지만, 순환 시스템에서는 각 서버로부터 서비스되어 서버에 다시 입력되는 도착율을 처리율로 하여 해석한다.

Buzen이론은 모든 자원에 대한 평균 이용률과 작업의 순환 시간을 구하는 콘볼루션(convolution) 알고리즘 이다. 따라서, 멀티프로그래밍 레벨을 1 로 하였을 때, 즉 전체 시스템의 부하가 적을 경우에 프로세서의 수를 고려하여 이 값을 상수 화하여 멀티프로그래밍의 레벨을 증가시켜 시스템의 부하가 많은 경우에 대해서 터미날 수의 변화에 따른 시스템의 응답 시간, 처리율을 구해서 최적의 멀티프로그래밍 레벨을 구했다[3, 6].

그러나 컴퓨터 시스템이 여러 개의 작업을 동시에 많은 터미날로부터 서비스 요구를 받게 되면 시스템은 과부하 상태가 됨으로, 시스템의 과부하 상태를 효과적으로 극복하기 위해 우선순위 레벨을 고려하여 해석함으로써 성능 향상을 보였다.

2. BCMP 큐잉 네트워크

BCMP 큐잉 네트워크 이론은 1975년에 Baskett, Chandy, Muntz, Palacios에 의해 발표된 이론으로 Jackson 큐잉 이론의 확장 형태로 실제 시스템을 분석하는데 있어서 널리 적용되고 있다[1, 2, 13].

BCMP 큐잉 네트워크에서 항목은 3가지가 있는데 루팅(routing) 변수, 서비스 요구 및 가용한 항목의 수이다.

그리고 서버는 4가지 형태인데, 첫 번째 형태의 서버는 FCFS(First Come First Service)의 형태로 보통 큐(queue)의 형태로 많이 적용되며, 각각의 서비스 요구는 같은 형태의 지수 분포를 가진 것으로 가정하고, 두 번째는 PS(Processor-Sharing)의 형태로 각각의 서비스 요구는 코엑시안(coaxian) 분포를 갖는다고 가정하며 CPU 에 적용하고, 세 번째는 IS(Infinite-Server)의 형태로 도착된 서비스 요구가 즉시 서비스가 시작되는 형태로서 PS 형태와 같은 확률 분포를 가지고 있으며 지연 형태의 서버로 사용되며 일반적으로 터미날에 적용하고, 마지막 형태는 LCFS(Last Come First Service)의 형태로 확률 분포는 PS 형태와 동일하다[5, 7, 8].

각 터미날로부터 입력되는 서비스 요구는 하나의 큐로 받아들여지게 되며 이 큐는 각 서버에 균등하게 서비스 요구를 제공되어진다고 가정한다. 따라서 입력된 요구에 대한 서비스는 균일하게 나누어 작업한다고 가정한다.

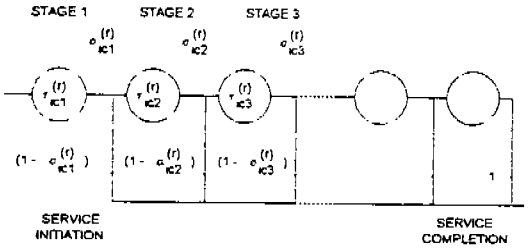
i 노드에서 c 항목내에 존재하는 r 형태의 서비스 요구가 완벽하게 완료된 후 d 항목에 있는 다음 노드 j 로 진행될 확률은 $P_{ic,jd}^{(r)}$ 이고 네트워크 외부로 나갈 확률은 $P_{ic,0}^{(r)}$이다.

$$P_{ic,0}^{(r)} = (1 - \sum_{j=1}^{N} \sum_{d \in C_{jr}} P_{ic,jd}^{(r)}) \qquad (1)$$

여기서 N은 전체 네트워크에서 노드의 수이다. 그리고 C_{jr}은 r형태의 서비스 요구가 j노드에 있을 때 항목의 집합이다. BCMP 큐잉 네트워크에서 외부로부터 입력되는 서비스 요구는 포아송 프로세스이며, c 항목내에 r 형태의 서비스 요구가 i 노드에 도착할 확률은 $\lambda_{ic}^{(r)}$로 표시하며, $c \in C_{ir}$이고 $i \in \{1,..., N\}$ 일 때 $\lambda_{ic}^{(r)} > 0$ 이면 이 루팅 체인(routing chain)을 개방 시스템이라 하고 이 값이 영 이면 순환 시스템이라 한다.

각 노드에서 서비스 요구에 대한 분포는 Laplace 변환의 형태를 갖게 되는데, 이 분포는 지수 분포와 하이퍼 지수(hyperexponential) 분포와 하이포 지수(hypoexponential)분포를 포함한다. 이것을 코엑시안 분포라 하며, 지수 적인 상

태로 다음 그림과 같이 표현될 수 있다.



(그림 1) 서비스 요구에 대한 코엑시안 분포
(Fig. 1) Coaxian distribution for service demand

상태 s에서 c항목내에 존재하는 r형태의 서비스 요구가 i노드에 도착할 평균 서비스 요구는 $\tau_{i,cs}^{(r)}$로 표시한다. 이때 하나의 상태(s)가 완료된 후에 다음 상태(s+1)로 진행할 확률은 $\sigma_{ics}^{(r)}$ 이고 모든 서비스 요구가 완료될 확률은 $(1-\sigma_{ics}^{(r)})$ 이다. 이러한 방식은 전체적으로 동축 분포를 갖고 있는 평형 상태로 보여지며, 상태 변수s 와 항목 변환 특성을 갖는 서비스 요구 항목의 인덱스를 지정할 수 있게 된다. 또 코엑시안 분포를 갖는 서비스 요구를 합성할 수 있게 된다.

3. 우선순위 시스템

일반적으로 우선순위 시스템은 서비스 진행 중에 있는 서비스는 더 높은 우선순위를 갖는 서비스 요구가 있더라도 진행 중인 서비스를 완료하고 다음에 입력된 가장 높은 우선순위를 갖는 서비스를 진행하는 경우와 현재 진행 중인 서비스가 존재할 때 더 높은 우선순위를 갖는 서비스 요구가 입력되면 진행 중인 서비스를 중단하고 높은 우선순위의 서비스 요구를 받아들여 진행하는 경우가 있다.

본 연구에서는 진행 중인 서비스를 완수하는 전자의 경우를 가정한다. 가장 높은 우선순위를 1, 가장 낮은 우선순위를 k로 할 경우 이용율은 다음과 같다.

$$\rho = \rho_1 + \rho_2 + \cdots + \rho_k < 1 \tag{2}$$

가장 높은 우선순위 서비스 요구에 대한 대기 시간 W_{p1}은 다음과 같다.

$$W_{p1} = L + N_{q1} S_1 \tag{3}$$

여기서 L은 전송 중인 서비스의 잔여 서비스 시간이며, S_1은 우선순위 1인 서비스 요구의 서비스 시간이고, N_{q1}은 우선순위 1인 서비스 요구의 수이다. $N_{q1} = \lambda_1 W_{p1}$ 이므로 위 식에 대입하여 정리하면 다음과 같다.

$$W_{p1} = \frac{L}{1-\rho_1} \tag{4}$$

우선순위 레벨이 2인 서비스 요구의 대기 시간 W_{p2}은 우선순위 레벨 1의 전체 서비스 시간과 우선순위 레벨 이이 2인 서비스 요구의 서비스 시간 및 이 시간 동안 도착하는 우선순위 레벨이 1인 서비스 요구의 서비스 시간의 합으로 구해진다.

$$W_{p2} = W_{p1} + N_{q2} S_2 + M_1 S_1 \tag{5}$$

여기서 M1 은 Wp2 동안 도착하는 우선순위 레벨이 1인 서비스 요구의 평균 값 으로 다음과 같다.

$$M_1 = \lambda_1 W_{p2} \tag{6}$$

따라서 W_{p2} 는 다음과 같다.

$$W_{p2} = \frac{L}{(1-\rho_1)(1-\rho_1-\rho_2)} \tag{7}$$

우선순위 레벨이 j인 서비스 요구의 평균 대기 시간은 다음과 같다.

$$W_{pj} = \frac{L}{(1-\sum_{j=1}^{j-1}\rho_j)\ (1-\sum_{j=1}^{j}\rho_j)} \tag{8}$$

여기서 L은 나머지 서비스 시간이므로 다음과 같다.

$$L = \sum_{i=1}^{k} \lambda_i S_i^{(2)} \tag{9}$$

따라서 우선순위 레벨이 j일 때 대기 시간을 구하면 다음과 같다.

$$W_{pj} = \frac{\sum_{i=1}^{k} \lambda_i S_i^{(2)}}{(1-\sum_{j=1}^{j-1}\rho_j)\ (1-\sum_{j=1}^{j}\rho_j)} \tag{10}$$

4. 시스템 모델

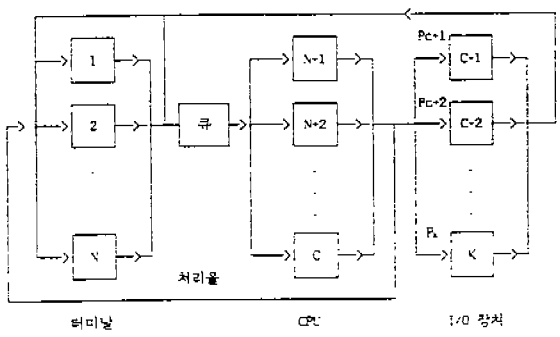
컴퓨터 시스템의 모델을 다음과 같이 가정한다.

먼저 N개의 터미날과 C개의 CPU, K개의 I/O로 구성되어 있다. 각각의 CPU는 서버로 간주하며 I/O 서버는 I/O 서비스를 하기 위한 큐는 없는 것으로 가정한다.

CPU 와 I/O 서버의 서비스 시간 간격은 지수분포로 가정하며 CPU와 I/O는 동시에 서비스가 발생하는 경우는 없으며 단지 I/O는 버퍼에 저장하는 것으로 가정한다.

터미날을 사용하는 사용자가 평균적으로 생각하는 시간을 E[t]=T로 표시한다.

멀티프로그래밍 레벨에 대해서도 최적의 상태로 나누어 작업을 수행한다고 가정한다. 이와 같은 순환 형태의 컴퓨터 시스템에서 동작하는 고정된 프로그램의 수를 멀티프로그래밍 레벨이라 하며, 이 변수를 사용함으로써 컴퓨터 시스템 모델에서 고려하기 어려운 주기억장치를 고려하여 분석할 수 있게 된다. 이때 프로그램은 컴퓨터 시스템을 순환하는 토큰으로 생각 될 수 있으며 임의의 한 프로그램이 수행이 완료된 후에 다음 프로그램을 수행한다고 가정한다.



(그림 2) 컴퓨터 시스템 모델
(Fig. 2) Compuer system model

5. 모델링 분석

멀티프로그래밍 레벨을 m 이라 하고 레벨 1에서 부터 m까지 각각의 레벨에 대한 평균 서비스율을 $\mu[MPL]=\lambda[MPL]$ 로 가정한다. 여기서 $\lambda[MPL]$는 평균 처리율이고, MPL은 $1 \le MPL \le m$ 이다.

또한, MPL>m인 경우에는 평균 서비스율을 $\mu[m]$로 하면 다음과 같다.

$$\mu[MPL]=\lambda[MPL] \quad MPL=1,2,\cdots\cdots,m \tag{11}$$

$$\mu[m]=\lambda[m] \quad MPL=m,m+1,\cdots\cdots,N \tag{12}$$

여기서, 각 프로그래밍 레벨에 대한 평균 서비스율을 구하기 위해 각 서버에 대해 초기 상태에서는 시스템 내에 존재하는 서비스 요구 $L_k[0]$는 없는 것으로 가정한다.

$$L_k[0]=0 \quad k=N+1,N+2,\cdots,K \tag{13}$$

이때 κ는 CPU와 I/O서버 모두를 포함한다.

멀티프로그래밍 레벨 MPL(MPL=1, 2, ... m)에 대해서 κ번째 서버가 서비스하는 시간 $W_k[MPL]$은 다음과 같다.

$$W_k[MPL]=\frac{D_k}{C}(1+L_k[MPL-1]) \quad k=N+1,N+2,\cdots,C \tag{14}$$

$$W_k[MPL]=D_k(1+L_k[MPL-1]) \quad k=C+1,C+2,\cdots,K \tag{15}$$

$$W[MPL]=\sum_{k=N+1}^{C} W[MPL]+\sum_{k=C+1}^{C} W_k[MPL] \quad k=N+1,N+2,\cdots,K \tag{16}$$

$$\lambda[MPL]=\frac{MPL}{W[MPL]} \tag{17}$$

$$L_k[MPL]=\lambda[MPL]\cdot W_k[MPL] \tag{18}$$

여기서 D_k는 κ번째 서버에서 서비스 요구에 대한 평균 서비스 시간이며 W[MPL] 는 멀티프로그래밍 레벨 MPL에 대해 모든 서버에 대한 전체 서비스 시간이다. 따라서 서비스율은 MPL개의 레벨을 전체 서비스 시간으로 나눈 것이며, 이 서비스율과 κ번째 서버가 멀티프로그래밍 레벨 MPL일 때 서비스 시간의 곱으로 이 레벨에서 κ번째 서버가 서비스 해야 할 작업이 큐로 남아 있는 수이다.

FESC(Flow Equivalent Service Center) 내에서 j개의 서비스 요구가 존재할 확률을 q[j]라 정의하고 시스템의 초기 상태에서 서비스 요구가 존재하지 않을 확률 q[0]=1 라고 할 수 있다. 따라서 초기 상태에서의 응답 시간은 없으며 n (n = 1, 2, ... N) 개의 터미날에 대한 응답 시간(W)을 각 서버에 우선순위를 적용하면 다음과 같다.

$$\mu[j]=\mu_1[j]+\mu_2[j]+\cdots+\mu_k[j] \tag{19}$$

$$W_1 = W_1 + \frac{j}{\mu_1[j]}\, q[j-1] \tag{20}$$

$$\lambda[n]=\frac{n}{W_1+T} \tag{21}$$

$$q[j]=\frac{\lambda[n]}{\mu_1[j]}\, q[j-1] \tag{22}$$

식 (19)는 j개의 터미날의 서비스 요구에 대한 서버의 서비스율을 우선순위 레벨 p로 표현한 것이며 W1은 우선순위 레벨이 1일 때 응답 시간이며 식 (21)과 식 (22)는 역시 마찬가지로 우선순위 레벨이 1일 때 평균 처리율과 j개의 서비스 요구가 큐로 존재할 확률이다. 따라서 임의의 터미날 n에 대한 처리율 λ[n] 은 다음과 같다.

$$W_p = W_p + \frac{j}{\sum_{p1=1}^{p-1}\mu_{p1}[j]\ \sum_{p1=1}^{p}\mu_{p1}[j]}\, q[j-1] \tag{23}$$

$$\lambda_p[j]=\frac{n}{W_p+T} \tag{24}$$

$$\lambda_p[j]=\lambda_1[j]+\lambda_2[j]+\cdots\lambda_p[j] \tag{25}$$

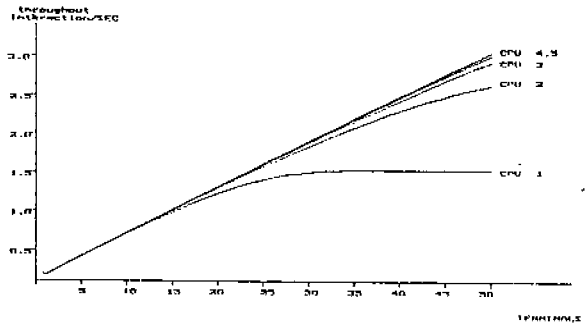
$$q[j]=\frac{\sum_{p1=1}^{p-1}\lambda_{p1}[j]\ \sum_{p1=1}^{p}\lambda_{p1}[j]}{\sum_{p1=1}^{p-1}\mu_{p1}[j]\ \sum_{p1=1}^{p}\mu_{p1}[j]}\, q[j-1] \tag{26}$$

여기서 q[j]는 j 개의 서비스 요구가 시스템 내에 큐로 존재할 확률이며 j=n, n−1, ... ,1 이다. 따라서 시스템 전체의 처리율은 λ[N] 이다.

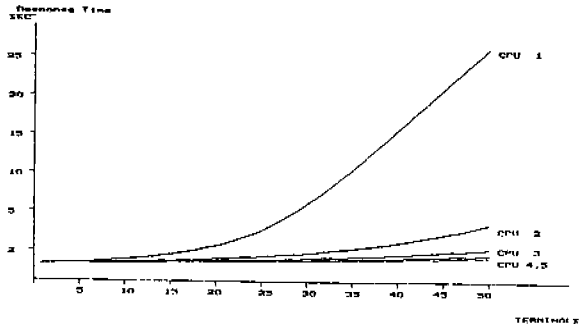
6. 수학적 해석 결과 및 고찰

서비스 요구에 대한 평균 서비스 시간을 모든 서버에 대해 주어져야 하는데 본 연구에서는 중앙 서버와 입출력 서버 2가지로 구분하였다. 중앙 서버에서의 서비스 요구에 대한 서비스 시간은 0.7 SEC로 하였으며, 입출력 서버에서 서비스 요구에 대한 서비스 시간은 70mSEC 로 가정하였다. 터미날 동작시키는데 사용자가 평균적으로 생각하는 시간(E[t])을 15 SEC로 하였고, 입출력 서버는 10개로 가정하였다.

(그림 3)은 컴퓨터 시스템에서 우선순위가 없고 멀티프로그래밍 레벨도 없는 경우를 가정하여 서버의 수에 따른 처리율을 보인 것이다. 서버 즉 CPU가 1개부터 5개까지 처리율을 분석한 것으로 서버가 4개 이상이면 처리율 측면에서 서



(그림 3) MPL이 10이고 우선순위가 없는 경우 터미날 수에 따른 처리율
(Fig. 3) Throughput vs. number of active terminal with no priority and MPL=1



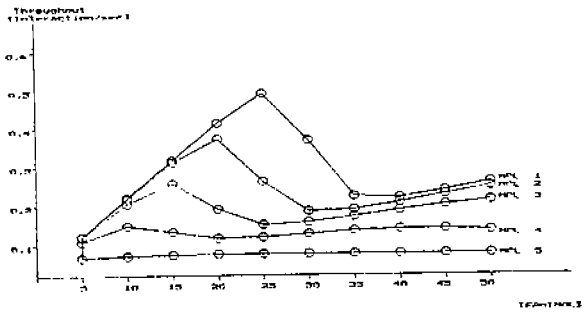
(그림 4) MPL이 10이고 우선순위가 없는 경우 터미날 수에 따른 응답시간
(Fig. 4) Response time vs. number of active terminal with no priority and MPL=1

버의 수에 상관없이 미세한 향상을 보임을 알 수 있었다. 그러나 서버가 1개로 구성된 것에 비교하면 터미날의 수, 즉 서비스 요구가 증가함에 따라 2개 이상의 서버로 구성된 시스템은 1개의 서버로 구성된 경우보다 많은 처리율의 향상이 있음을 알 수 있었다.

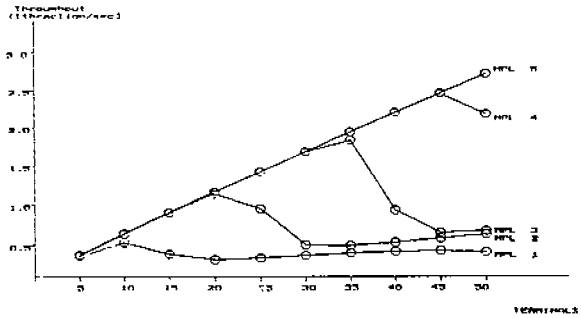
(그림 4)는 (그림 3)에서와 같은 컴퓨터 시스템에서 우선순위를 고려하지 않은 경우로 터미날 수에 따른 응답 시간을 보인 것이다. 이 컴퓨터 시스템에서도 응답 시간이 서비스 요구가 증가함에 따라 1개의 서버로 구성된 시스템보다는 2개 이상의 서버로 구성된 시스템에서 응답 시간이 향상되었으며, 4개이상의 서버로 구성한 경우 서버의 수가 증가되어도 응답시간의 향상이 미세하여 컴퓨터 시스템의 응답시간은 터미날 수에 제한을 받음을 알 수 있었다.

컴퓨터 시스템의 처리율은 서버의 수가 4개를 넘어서면 성능 향상이 다소 둔화됨으로 (그림 5)는 (그림 3)과 (그림 4)의 결과에 따라 서버의 수를 4개로 가정하고 대신에 멀티프로그래밍 레벨을 1부터 5까지 변화시켜 터미날 수에 따른 처리율을 보인 것이다. MPL1에서 MPL5는 각 멀티프로그래밍 레벨에 따른 처리율이며, 각 레벨에서 최대 처리율을 갖는 터미날 수가 서비스 요구에 대한 병목 현상이 발생되었다. 또 멀티프로그래밍 레벨이 2이하일 때 터미날 수의 변화에 관계없이 일정한 처리율을 보임을 알 수 있었다.

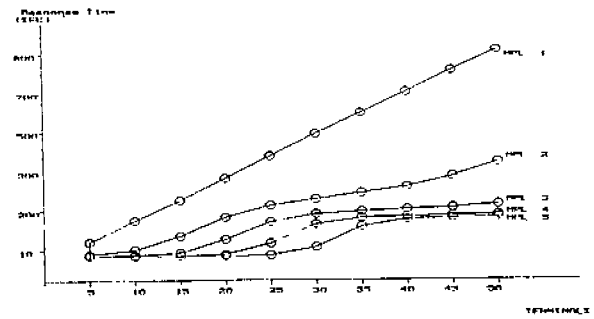
(그림 6)은 (그림 5)에서와 같은 조건에서 서버의 수가 4개일 때 터미날 수에 따른 응답 시간이다. 컴퓨터 시스템의 응답시간에 대해서도 서버의 수가 4이고 멀티프로그래밍 레벨이 3이



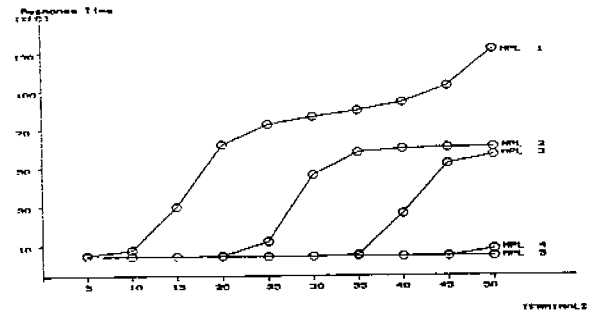
(그림 5) CPU가 4개이고 우선순위가 없는 경우 터미날 수에 따른 처리율
(Fig. 5) Throughput vs. number of active terminal with no priority and CPU=4



(그림 7) CPU가 4개이고 우선순위가 있을 때 터미날 수에 따른 처리율
(Fig. 7) Throughput vs. number of active terminal with priority levels and CPU=4



(그림 6) CPU가 4 개이고 우선순위가 없는 경우 터미날 수에 따른 응답시간
(Fig. 6) Response time vs. number of active terminal with no priority and CPU=4

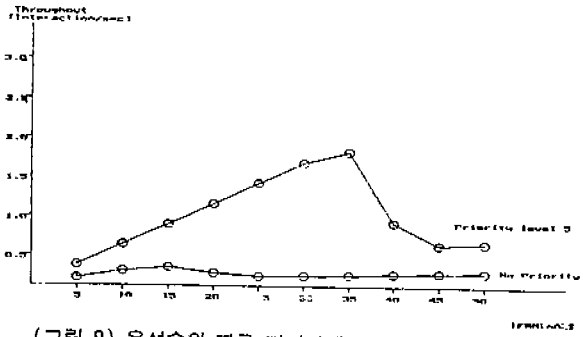


(그림 8) CPU가 4개이고 우선순위가 있을 때 터미날 수에 따른 응답시간
(Fig. 8) Response time vs. number of active terminal with priority level and CPU=4

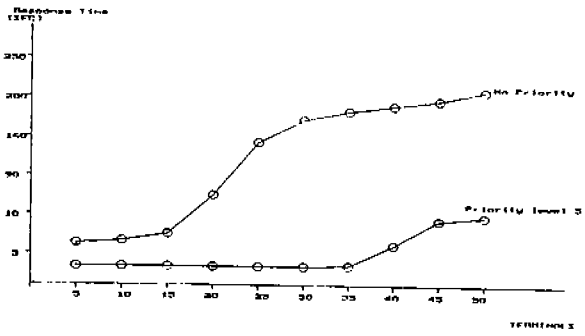
상일 때 터미날이 30개 이상이면 시스템의 효율이 떨어짐으로 터미날이 25-30개가 동시에 사용될 때 컴퓨터 시스템은 최적의 상태가 됨을 알 수 있었다.

(그림 7)은 서버의 수가 4개이고 우선순위 레벨을 5로 가정한 컴퓨터 시스템에서 각 멀티프로그래밍 레벨을 MPL1에서 MPL까지 변화시켰을 때 처리율이다. (그림 5)에서는 MPL의 변화에따라 터미날의 수가 25이상일 때 처리율이 급격히 떨어지나, 우선순위를 고려한 경우 MPL3에서는 터미날의 수가 35 MPL4에서는 터미날의 수가 45정도로 향상되어 우선순위가 없는 경우보다 동시에 사용 가능한 터미날의 수가 증가됨을 알 수 있었다.

(그림 8)은 (그림 7)과 같은 조건에서 터미날 수에 따른 응답시간을 보인 것으로 응답시간의 한계값에서 터미날 수를 살펴보면 멀티프로그래밍 레벨이 증가함에 따라 증가함을 알 수 있었다. MPL1일 때 동시에 사용 가능한 터미날의 수는 25개 정도이고 MPL2일 때 동시에 사용 가능한 터미날의 수는 35개 MPL3일 때 동시에 사용 가능한 터미날의 수는 45개로 향상되었고 MPL4이상에서는 적은 변화를 보였다.



(그림 9) 우선순위 따른 터미날에대한 처리율 (MPL=3)
(Fig. 9) Throughput vs. number of active terminal with priority levels and MPL=3



(그림 10) 우선순위 따른 터미날에 대한 응답 시간 (MPL=3)
(Fig. 10) Response time vs. number of active terminal with priority levels and MPL=3

(그림 9)는 서버가 4개 이고 멀티프로그래밍 레벨을 3으로 하여 우선순위 레벨 5를 가지는 경우와 우선순위를 고려하지 않는 두 시스템을 터미날에 대한 처리율을 비교하였다. 우선순위가 고려되지않은 경우는 최적의 처리율에서 동시에 사용가능한 터미날이 15개 정도이나 우선순위가 고려된 경우 동시에 사용 가능한 터미날이 35개 정도로 향상됨을 보였다.

(그림 10)은 서버가 4개 이고 멀티프로그래밍 레벨을 3으로 하여 우선순위 레벨 5를 가지는 경우와 우선순위를 고려하지 않는 두 시스템의 터미날에 대한 응답시간을 비교하였다. 우선순위가 고려되지않은 응답시간은 터미날이 15개 이상이되면서 응답시간이 증가하기 시작하고 우선순위가 고려된 경우 터미날이 35개 이상부터 응답시간이 증가함을 보였다. 응답시간측면에서 사용 가능한 터미날의 수에 대한 응답시간은 우선순위가 고려되었을 경우 향상됨을 보였다.

7. 결 론

본 연구에서는 2개 이상의 프로세서로 구성된 컴퓨터 시스템을 순환 형태의 모델로 가정하여 우선순위를 고려한 BCMP 큐잉 네트워크 이론을 적용하여 성능 분석을하였다.

멀티프로그래밍 레벨을 고려한 것은 여러 개의 작업을 동시에 처리하는 과부하 측면을 고려하여 각 멀티프로그래밍 레벨에 대해 최대 처리율과 최대 응답 시간에서 터미날의 수를 구해 보았고 이 상태에서 시스템 포화 상태가 되므로, 이때 처리율과 응답 시간의 향상을 위해 우선순위 레벨을 5로 하여 우선순위가 없는 경우와 비교한 결과 처리율과 응답 시간은 5-10 배의 향상을 보였다.

컴퓨터 시스템의 성능 향상을 위해 여러개의 프로세서를 사용함에 있어서, 5개 이상의 프로세

서로 구성할 경우 시스템의 처리율과 응답시간의 성능 변화가 작음을 알 수 있었다. 따라서 컴퓨터 시스템의 성능 향상을 위해 적절한 프로세서의 수와 터미날의 수를 최적화하는 등의 연구가 진행되어져야 할 것이다.

컴퓨터 시스템의 향상을 위해서 시스템 분석과 성능 분석을 위해 수학적 방법과 더불어 앞으로는 시뮬레이션으로 BENCHMARK TEST 를 이용하여 두 가지 방법을 비교 분석이 이루어져야 할 것으로 사료된다.

참 고 문 헌

[1] Arnold O. Allen "Probability, Statistics, and Queueing Theory with Computer Science Application 2nd Edition", ACADEMIC PRESS, 1990.

[2] Adrian E. Conway and Nicolas D. Georganas , "Queueing Network-Exact Computational Algorithms : A Unified Theory Based on Decomposition and Aggregation", The MIT Press, 1989.

[3] Alberto Leon-Garcia "Probability and Random Processes for Electrical Engineering", ADDISON WESLEY, 1989.

[4] I.F. Akyildiz, "Mean Value Analysis for Blocking Queueing Network", IEEE Trans. on Software Eng., 14, 4, pp. 418-428, 1988.

[5] Adrian E. Conway and Nicolas D. Georganas, "A Polynomial Complexity Mean Value Analysis Algorithm for Multiple-Chain Closed Queueing Networks", in Digest of Paper, IEEE Int. Symp. on Information Theory, p 61, Ann Arbor, Michigan, 1986.

[6] Adrian E. Conway and Nicolas D. Georganas , "RECAL : A New Efficient Algorithm for the Exact Analysis of Multiple-Chain Closed Queueing Networks", J. ACM , 33, 4, pp. 768-791, 1986.

[7] Adrian E. Conway and Nicolas D. Georganas, "Decomposition and Aggregation by Class in Closed Queueing Networks", IEEE Trans. on Software Eng., 12, 10, pp. 1025-1040, 1986.

[8] Adrian E. Conway and Nicolas D. Georganas, "A New Method for Computing the Normalization Constant of Multiple Chain Queueing Networks", INFOR, 24, 3, pp. 184-198, 1986.

[9] P.S. Kritzinger, S. van Wyk, and A. E. Krzesinski, "A Generaligation of Norton's Theorem for Multiclass Queueing Networks",Performance Evaluation, pp. 98-107,1982.

[10] P.S. Kritzinger, "A Performance Model of the OSI Communcaition Architecture", IEEE Trans. on Communications, 34, 6, pp. 554-563. 1986.

[11] A. E. Krzesinski, "Multiclass Queueing Networks with State-Dependent Routing Routing", Performance Evaluation, 7, pp. 125-143, 1987.

[12] Kevin Dowd, "High Performance Computing",O'Reilly & Associates, INC, 1993.

[13] Thomas G. Robertazzi, "Computer Network and Systems", Springer-Verlag, 1994.

이 상 훈

983년 광운대학교 응용전자 공학과 졸업(공학사)
987년 광운대학교 대학원 전자공학과 졸업(공학석사)
992년 광운대학교 대학원 전자공학과 졸업(공학박사)
990년~현재 광운대학교 전자계산 교육원 조교수

박 동 준

1989년 광운대학교 전자공학과 졸업(공학사)
1991년 광운대학교 대학원 전자공학과 졸업(공학석사)
1991년~현재 광운대학교 대학원 박사과정 재학중

정 상 근

1971년 광운대학교 통신공학과 졸업(공학사)
1975년 연세대학교 산업대학원 전자공학과졸업(석사)
1992년 경희대학교 대학원 전자공학과(박사)
1971년~77년 동양공고 전자과 교사
1977년~현재 안양전문대학 전자계산과 교수