고속병렬컴퓨터[1](SPAX) 노드간 통신시스템의 성능분석을 위한 대기행렬모형

조 일 연† · 이 재 경† · 김 해 진†

요 약

본 논문에서는 고속병렬컴퓨터와 같이 다중노드로 구성된 시스템에서 사용되는 유한 버퍼를 가지는 노드간 통신시스템의 성능을 대기행렬모형을 사용하여 평가/분석하였다. 전체 시스템을 각 서브 시스템별로 나누어 버퍼크기의 제한으로 발생하는 블로킹을 고려한 안정상태에서의 도착률과 확률 값들을 구한 뒤, 이를 이용하여 전체 시스템의 성능을 분석하는 방법을 사용하였다. 평균 전송시간과 처리율을 구하여 이를 시뮬레이션 결과와 비교 검증하였으며, 제안한 모형을 사용하여 버퍼의 크기가 처리율에 미치는 결과를 분석하였다.

A Queueing Model for Performance Analysis of SPAX Inter-Node Communication System

Ilyeon Cho† · Jaekyung Lee† · Haejin Kim†

ABSTRACT

A queueing model for performance evaluation of finite buffered inter-node communication system is proposed in this paper. Each components are modeled as M/M/1/B queues to obtain the steady state probabilities and arrival rate and to analyze finite buffer behavior. The overall system is integrated as series queues with blocking. Average delay and throughput are the performance measures studied in this analysis. The analytical results are first validated through simulation. Next, the effect of buffer length is discussed using the proposed model.

† 정 회 원: 한국전자통신연구원 시스템 S/W 연구실
논문접수: 1997년 7월 15일, 심사완료: 1997년 11월 7일

1) SPAX(Scalable Parallel Architecture computer based on X-bar network) 시스템은 정보통신부 및 과학기술처의 지원으로 한국전자통신연구원이 주관하여 연구 개발하고 있는 시스템(일명: 주전산기 Ⅳ)의 명칭이다.

2) Xcent-Net은 SPAX 시스템에 사용되는 상호 연결망의 명칭으로 한국전자통신연구원에서 독자 설계/개발한 계층적 크로스바 연결망이다.

3) MISIX(MIcrokernel based Single system Image Unix)는 Unix Ware/MK를 기반으로 하는 운영체제이다.

1. 서 론

한국전자통신연구원에서 개발중인 SPAX[1, 2, 3, 4]는 최대 16개까지의 클러스터로 구성 될 수 있으며, 각 클러스터는 4개의 처리기를 가지는 8개의 노드로 구성된다. 클러스터 내의 노드간 통신과 클러스터간의 통신은 Xcent-Net[2]을 통하여 이루어진다. Xcent-Net은 입출력이 분리된 10×10 크로스바 스위치를 계층적으로 연결하여 구성되며, 전송 패킷의 최대 크기는 64바이트이다.

SPAX에 사용되는 운영체제인 MISIX[3][5, 6]는 마

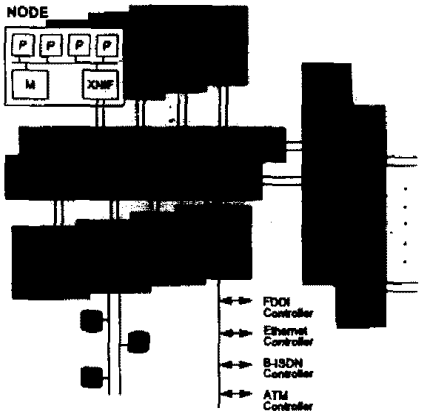
이크로커널을 기반으로 하고 있다. 시스템의 각 노드에는 UNIX의 기본 기능-메모리 관리, 예외 및 인터럽트 처리, 노드간 IPC, 프로세스/쓰레드 실행관리-을 제공하는 마이크로커널이 동작하고 있으며, 그 위에 사용자에게 UNIX 사용환경을 제공하는 시스템 서버들이 동작한다. 대표적인 시스템 서버로는 프로세스 관리 서버(PM:Process Manager), 파일 관리 서버(FM:File Manager), 스트림 관리 서버(STM:STream Manager) 등이 있다. 각 노드에 분산되어 실행되는 시스템 서버들은 마이크로 커널에서 제공하는 IPC를 이용하여 서로 필요한 정보를 주고받는다. 따라서 SPAX 시스템 전체의 성능에 가장 큰 영향을 주는 요소는 노드간 통신 시스템이다.

클러스터링을 기반으로 하는 다중처리기(Multiprocessor) 시스템의 노드간 또는 처리기간 통신 시스템의 성능 평가에 관한 연구로는 다음과 같은 연구들을 찾아 볼 수 있다. [7, 8]에서는 크기가 무한한 버퍼를 가지는 MIN(Multistage Interconnection Network)을 기반으로 하는 시스템에 대한 성능 평가 모델을 제시하였다. 유한한 크기의 버퍼를 가지는 MIN을 기반으로 하는 시스템에 대한 성능 평가에 관한 연구로는 확률적 분석(probabilistic analysis) 방법을 적용한 연구[9]와 노드간 블로킹이 일어나지 않는다는 가정 하에서 대기행렬모형(Queueing model)을 이용한 연구[10]을 찾아 볼 수 있다. 본 논문에서는 SPAX 시스템의 노드간 통신 시스템의 성능을 유한한 크기의 버퍼를 가지는 노드가 직렬로 연결된 블로킹이 존재하는 Tandem 대기행렬모형 형태로 모델링하여 성능을 평가/분석하고 시뮬레이션을 통하여 모형의 정확성을 검증하였다.

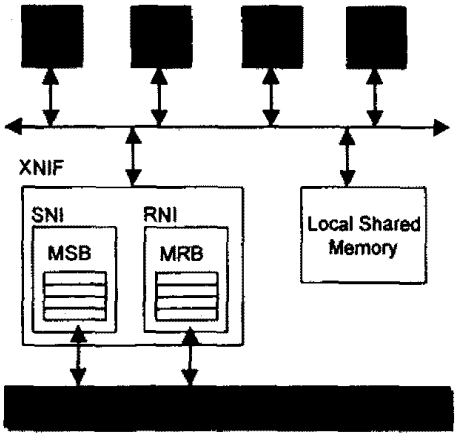
본 논문의 구성은 다음과 같다. 2장에서 SPAX 시스템의 구조와 Xcent-Net와의 인터페이스를 제공하는 XNIF(X-bar Network InterFace)[11, 12]에 대하여 설명한다. 3장에서 성능분석을 위한 대기행렬모형을 제안한다. 4장에서는 본 논문에서 제안한 대기행렬모형의 결과를 시뮬레이션 결과와 비교 검증하고 대기행렬모형을 사용하여 버퍼의 크기가 처리율(throughput)에 미치는 영향을 분석한다. 끝으로 5장에서 결론을 맺는다.

2. 시스템 설명

SPAX 시스템의 전체적인 모습은 (그림 1)과 같다. 한 노드는 대칭형 다중 프로세서(Symmetric Multi-Processor)로 구성된 노드로서 4개의 펜티엄프로 처리기와 이들 프로세서가 공유하는 공유 메모리, 다른 노드와 통신 기능을 제공하는 XNIF 보드로 구성된다. (그림 2)에서와 같이 SPAX 시스템의 상호 연결망 인터페이스인 XNIF는 송신 연결망 인터페이스(SNI: Send Network Interface)와 수신 연결망 인터페이스(RNI:Receive Network Interface)로 구성되어 있다.



(그림 1) SPAX 시스템
(Fig. 1) SPAX system

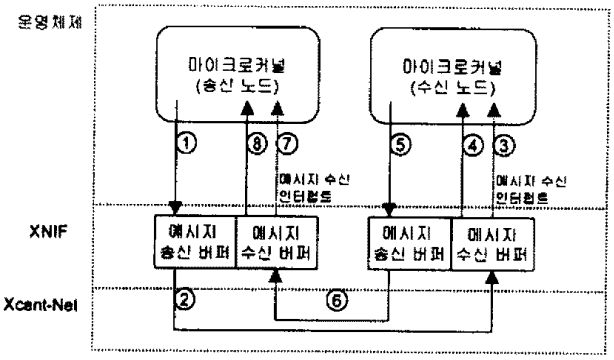


(그림 2) SPAX 노드
(Fig. 2) SPAX node

메시지 버퍼는 SNI에 송신버퍼와 RNI에 수신버퍼가 있다. 메시지 송신버퍼(MSB:Message Send Buffer)는 프로세서가 메시지의 송신을 SNI에 의뢰하기 위

하여 사용된다. MSB는 운영체제에서 읽기/쓰기가 가능한 깊이가 4인 버퍼로 256 바이트의 메시지를 저장할 수 있다. 메시지 수신버퍼(MRB:Message Receive Buffer)는 수신된 메시지를 저장하기 위해 사용된다. MRB는 운영체제에서 읽기만 가능한 깊이가 4인 버퍼로 256 바이트의 메시지를 저장 할 수 있다.

Xcent-Net에 연결된 여러 노드에서 동시에 하나의 노드에 메시지를 전송하고자 할 때 이들 메시지들 간에 충돌(hot-spot)이 발생 할 수 있는데, 이러한 충돌을 Xcent-Net에서는 "2 수준의 우선 순위를 기반으로 하는 라운드 로빈(round robin)" 정책을 사용하여 중재한다.



(그림 3) 동기적 방법의 메시지 전달 흐름
(Fig. 3) Synchronous message transmission flow

(그림 3)에 마이크로커널의 동기적인 메시지 전달 요구가 XNIF의 버퍼를 통하여 처리되는 흐름을 나타내었다. 다음은 (그림 3)에 대한 설명이다.

① 전송할 메시지가 있는 노드의 프로세서는 MSB 제어 레지스터의 값을 읽어 버퍼에 여유가 있으면 메시지를 MSB에 저장함으로써 메시지의 전송을 의뢰한다.

② XNIF는 전송 의뢰된 메시지를 Xcent-Net로 전송한다.

③ 수신노드의 XNIF는 Xcent-Net으로부터 메시지를 수신하여 MRB에 저장하고, 수신 프로세서에게 수신할 메시지가 있음을 인터럽트를 통하여 알린다.

④ 인터럽트를 수신한 프로세서는 MRB에 저장된 메시지를 읽어간다.

⑤⑥⑦⑧은 수신한 메시지에 대한 응답메시지의 처리 순서로 메시지 송신시와 같은 방법으로 처리된다.

3. 대기행렬모형

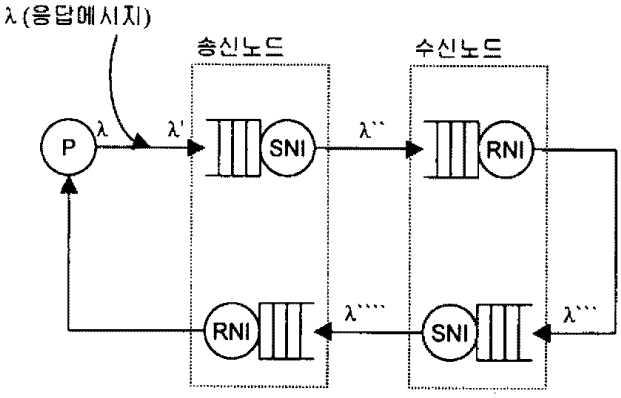
대기행렬모형을 수립하기 위하여 다음과 같은 가정을 하였다.

1) 각 노드의 프로세서에서 생성되는 메시지 생성 간격은 평균 $1/\lambda$의 지수 분포를 따른다. λ는 각 노드의 SNI에 도착하는 메시지의 도착률(arrival rate)이다.

2) SNI와 RNI의 처리시간은 평균 $1/\mu$의 지수분포를 따른다. μ는 서비스율(service rate)이다.

3) 수신노드의 선택은 균등 분포(Uniform distribution)를 따른다.

4) 메시지 송/수신 버퍼에 여유가 없어 메시지의 송신이 블로킹되는 경우에도 메시지는 없어지지 않고 계속해서 재시도 된다.



(그림 4) 입력율의 변화
(Fig. 4) Change of input rate

대기행렬 시스템의 지연시간과 처리율을 구하는 방법을 살펴보면 다음과 같다. 버퍼크기의 제한으로 생기는 블로킹을 고려한 변화된 도착률 λ'은 다음과 같이 구할 수 있다. 시스템에 도착한 작업이 블로킹되지 않고 버퍼에 쓰여지려면, 버퍼에 여유가 있어야한다. 버퍼에 여유가 있을 확률은 전체확률 1에서 버퍼가 꽉 차있을 확률을 빼면 구할 수 있다. 현재 버퍼가 꽉 차있을 확률은 다음과 같다.

$$P_{B+1}=\frac{(1-\rho)\,\rho^{B+1}}{1-\rho^{B+2}} \quad (\rho=\lambda/\mu,\ B=\text{버퍼 크기}) \qquad (1)$$

따라서 실제 도착률은 다음과 같다.

$$\lambda'=2\lambda(1-P_{B+1}) \qquad (2)$$

이러한 블로킹으로 인한 실제 도착률을 고려한 지연시간은 Little의 법칙[13]을 이용하여 다음과 같이 구할 수 있다.

$$\text{지연시간}=\frac{1}{\lambda'}\left[\frac{\rho}{1-\rho}-\frac{(B+2)\,\rho^{B+1}}{1-\rho^{B+2}}\right] \qquad (3)$$

위의 결과를 전체 시스템에 적용하여 근사해를 구하기 위하여 Hillier와 Boling[14]이 제안한 방법을 사용하였다. 이 근사해법에서는 첫 번째 노드는 항상 한 개 이상의 작업이 존재하고, 블로킹은 서비스가 끝난 후에 발생한다고 가정을 하였다. 즉, i번째 노드에서 서비스가 끝난 작업은 $(i+1)$번째 노드의 버퍼에 여유가 있으면 다음 노드로 전이되고 그렇지 않으면 $(i+1)$번째 노드의 버퍼에 여유가 생길 때까지 i번째 노드에서 기다리게 된다. 다음과 같은 기호 및 확률을 정의하자.

기호 및 확률	설명
M	노드 수
T	처리율
m_i	i 번째 노드의 버퍼 크기
n_i	현재 i 번째 노드에 있는 작업 개수
$P_i(0)$	i 번째 노드의 메시지 큐에 처리할 메시지가 없어 서버가 유휴(Idle) 할 확률

각 노드에서 발생하는 블로킹으로 인하여 도착과정과 서비스 과정, 노드의 용량은 다음과 같이 변화된다.

도착률 λ_i $(i=2, 3, \cdots, M)$

$$\lambda_i=\begin{cases}\mu_{i-1}[1-P_{i-1}(0)] & \text{if } n_i \le m_i \\ 0 & \text{if } n_i = m_i+1\end{cases} \qquad (4)$$

시스템의 첫 번째 노드가 비어있을 확률은 가정에 의해 0이 된다. 즉, $P_1(0)=0$이다.

서비스율 $\hat{\mu}_i$ $(i=1, 2, 3, \cdots, M)$

$$\hat{\mu}_i=\frac{T}{1-P_i(0)} \qquad (5)$$

시스템의 마지막 노드인 M번째 노드의 실제 서비스율을 $\mu_M=\hat{\mu}_M$이라 하면, 식 (5)로부터 시스템의 처리율은 다음과 같다.

$$T=\mu_M(1-P_M(0)) \qquad (6)$$

노드의 용량 증가

블로킹된 작업을 표현하기 위하여 각 노드의 용량을 하나씩 증가시킨다. 이러한 처리는 "서비스후의 블로킹"이 발생하는 시스템의 분석을 위하여 필요한 가정이다. 즉, $(i-1)$번째 노드의 서버는 i번째 노드로 인한 블로킹이 발생하는 동안에는 마치 블로킹된 작업을 위한 공간이 하나 더 있는 것처럼 처리한다. 즉, 각 노드에 바로 앞의 노드에서 블로킹된 메시지를 수용할 수 있도록 큐 용량이 하나 더 큰 것으로 모델링을 하게 된다. 그러므로, 블로킹된 작업이 $(i-1)$번째 노드의 맨 앞에 있으면서 실제로는 i번째 노드의 작업대상으로 간주 될 수 있다.

각 노드 $i(i=1, 2, \cdots, M)$를 $M/M/1/m_i+1$으로 보고, 식 (1)을 이용하면 다음과 같은 결과를 얻을 수 있다.

$$1-P_i(0)=\frac{\rho_i(1-\rho_i^{m_i+1})}{1-\rho_i^{m_i+2}}, \quad where\ \rho_i=\frac{\lambda_i}{\hat{\mu}_i} \qquad (7)$$

식 (4)와 (5)를 이용하여 다음과 같은 결과를 얻을 수 있다.

$$\rho_i=\frac{\mu_{i-1}[1-P_{i-1}(0)]}{T/[1-P_i(0)]} \qquad (8)$$

T와 $P_{i-1}(0)$의 값을 식 (8)에 대입하여 ρ_i를 구하여 이를 다시 식 (7)에 대입하면 $P_i(0)$를 구할 수 있다.

다음의 방법으로 노드 i가 유휴 상태에 있을 확률 $P_{i-1}(0)$와 시스템의 처리율 T를 구할 수 있다.

단계 1) $T(0)=2\lambda$

단계 2) 식 (4)에서 λ_i를 구한다.(λ_2인 경우에는 가정

(그림 4)에 메시지의 전달시 각 버퍼로의 도착률이 버퍼의 블로킹으로 인해 변화되는 것을 나타내었다. 메시지 송신요구는 지역 프로세서의 메시지 송신요구와 다른 노드로부터의 서비스 요구에 대한 응답 메시지가 있다. 지역 프로세서에서의 메시지 송신 요구는 λ의 비율로 생성되며, 응답 메시지는 자기 자신을 제외한 나머지 $(K-1)$개의 노드에서 λ의 비율로 생성된 메시지 중 $1/(K-1)$ 만큼이 이 노드에 서비스를 요구하게 되므로 λ가 된다. 따라서 임의의 한 노드에서 발생되는 메시지 송신 요구율은 2λ가 된다. 그러나 송신노드의 메시지 송신버퍼가 유한하므로 실제로 SNI에 입력되는 비율은 버퍼의 블로킹을 고려한 λ'이 된다. 유한크기 B의 버퍼를 가지는 SNI와 RNI를 대기행렬모형으로 표현하면 M/M/1/B +1이 되며 전체 시스템은 이러한 노드가 직렬로 연결된, 개방형 대기행렬 네트워크(Open Queueing Network) 형태의, 블로킹이 존재하는 Tandem 대기행렬모형이 된다. 이러한 대기행렬모형의 처리율은 송신노드의 RNI로의 도착률인 λ''''이 되며, 총 지연시간은 각 노드의 지연시간을 합한 것과 같다.

전체 시스템의 해를 구하기 전에 먼저 M/M/1/B +1 $P_1(0)=0$을 고려하면 $\lambda_2=\mu_1$이 된다.)

단계 3) 단계 2)의 결과를 식 (8)에 대입하여 ρ_i를 구한다.

단계 4) 단계 3)에서 구한 ρ_i를 식 (7)에 대입하여 $P_i(0)$를 구한다.

[i를 증가시키면서 단계 2)에서 단계 4)를 반복하여 $P_M(0)$을 구한다.]

단계 4) 단계 4)에서 구한 $P_M(0)$을 식(6)에 대입하여 시스템의 처리율 T를 구하여 이를 $T(1)$이라 한다.

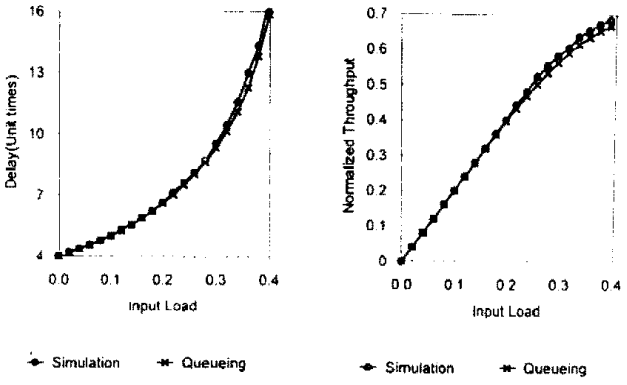
단계 5) $|T(0)-T(1)|$이 정한 허용한계치 보다 작으면 이를 시스템의 처리율로 선택하고 그렇지 않으면 $T(0)=T(1)$이라 하여 단계 2)부터 다시 반복한다.

식 (2)을 통하여 얻어진 첫 번째 노드의 도착률과 위의 알고리듬을 통하여 얻어진 각 노드의 안정상태에서의 도착률과 처리율을 식(3)에 대입하여 각 노드에서의 지연시간을 구한 뒤 이를 모두 합하면 원격노드에 대한 서비스 요구 시에 소요되는 총 시간을 구할 수 있다.

4. 성능 분석

4.1 모형의 검증

본 논문에서 제안한 대기행렬모형을 검증하기 위하여 128개 노드로 구성된 SPAX 시스템에 대한 시뮬레이션을 SIMSCRIPT II.5를 사용하여 수행하였다. 시뮬레이션에서는 송/수신 버퍼의 크기를 4로 하고 각 노드의 서비스율을 1로 고정한 상태에서 도착률을 증가시켜 시스템의 작업부하 변화에 따른 송/수신시에 발생되는 지연시간과 처리율의 변화를 대기행렬모형과 비교하였다. 각 노드에서 발생되는 메시지 전송요구는 도착률 λ인 포아송 프로세스로 도착하고 각 노드의 서비스 시간은 평균 $1/\mu$인 지수분포를 따른다. 생성된 메시지의 수신노드는 균등분포를 사용하여 선택하게 하였으며, 시스템의 처리율을 구하기 위하여 단위시간동안 처리되는 작업의 개수를 구하였다. (그림 4)에 128개 노드로 구성된 시스템의 성능을 대기행렬모형의 결과와 비교하였다. 그림을 통하여 본 논문에서 제안한 대기행렬모형이 비교적 정확함을 알 수 있다. 그림에 제시된 구성 외에도 많은 입력 값들에 대하여 시뮬레이션을 수행하여 대기행렬모형과 비교하였다.

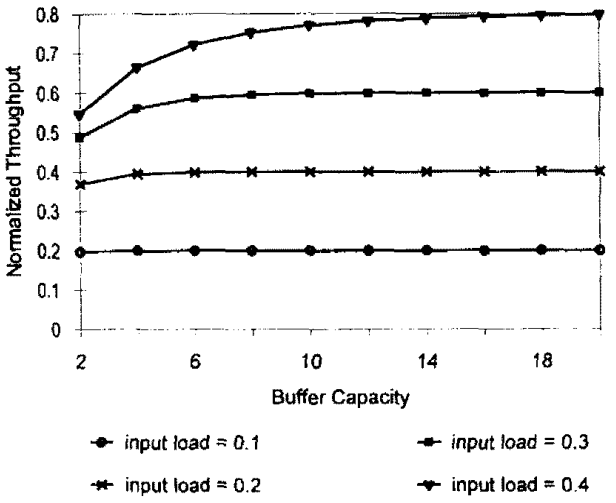


(그림 4) 시뮬레이션과의 비교
(128개 노드, 송/수신 버퍼 크기 = 4, Input Load = $\rho=\lambda/\mu$, $\mu=1$)
(Fig. 4) Performance comparison with simulation

4.2 버퍼크기에 따른 처리율의 변화

(그림 5)에 128개 노드로 구성된 시스템에서 메시

지 수신버퍼의 크기를 변화시킬 때 처리율의 변화를 나타내었다. 버퍼크기가 작고 시스템의 부하가 경부하인 경우에는 마치 무한대 크기의 버퍼를 가진 것과 같은 성능을 보여 주고 있음을 그림을 통하여 알 수 있다. 즉, 입력부하가 0.2보다 작은 범위에서 버퍼크기가 4이상이 되면 이와 같은 효과를 얻을 수 있다. 이러한 성능분석의 결과를 이용하면 시스템의 부하에 따른 최소한의 버퍼크기, 즉 경제적인 버퍼크기를 구할 수 있다. 예를 들어 시스템의 입력부하가 0.3정도인 경우에 시스템이 무한대 크기의 버퍼를 가진 것과 같은 성능을 내기 위한 최소한의 버퍼크기는 대략 10 정도가 됨을 알 수 있다. 또한 시스템의 처리율은 버퍼의 크기가 클수록 높음을 알 수 있다. 따라서 처리율을 우선으로 하는 시스템이라면 큰 버퍼를 가지는 것이 유리할 것이다. 그러나 앞서 언급한 것처럼 시스템의 입력부하에 따라서 마치 무한대 버퍼를 가진 것과 같은 성능을 내어 주는 크기 이상의 버퍼를 가지는 것은 불필요 할 것이다.



(그림 5) 버퍼 크기의 변화에 따른 처리율의 변화
(128개 노드, 송신 버퍼 크기 = 4, Input Load = $\rho = \lambda/\mu$, $\mu = 1$)
(Fig. 5) Effect of buffer length

5. 결 론

본 논문에서는 고속병렬컴퓨터에서 사용되는 유한 버퍼를 가지는 노드간 통신시스템의 성능분석을 위한 대기행렬모형을 제안하였다. 모형의 정확성 검증을 위하여 시뮬레이션을 수행하여 결과를 비교하였다. 대기행렬모형의 단순함에 비하여 비교적 정확한 결과를 쉽게 얻을 수 있었다.

본 모형을 사용하여 버퍼의 크기가 처리율에 미치는 영향을 분석하였다. 버퍼의 크기가 커짐에 따라서 처리율이 증가 하기는 하지만 일정 크기 이상에서는 더 이상의 처리율 증가가 없음을 알 수 있었다. 이러한 적절한 버퍼크기를 본 논문에서 제안한 대기행렬모형을 통하여 근사적으로 손쉽게 구할 수 있다.

참 고 문 헌

[1] 김양우, 최창열, 임기욱, "*고속병렬 컴퓨터의 소개 및 개발현황*", 타이컴월드, pp. 237-243, 1995. 10.

[2] 한우종, 윤석한, 임기욱, "*상용응용을 위한 병렬처리 구조 설계*", 대한전자공학회 논문지 33B권, 5호, pp. 897-907, 1996. 5.

[3] W. J. Hahn, K. W. Rim, S. W. Kim, "*SPAX: A New Parallel Processing System for Commercial Applications*", Proc. of 11th IPPS, pp. 744-749, 1997. 4.

[4] Y. W. Kim, S. W. Oh, and J. W. Park, "*Design issues and system architecture of TICOM IV, A highly parallel commercial computer,*" The 3rd Euromicro Workshop on Parallel & Distributed Processing, pp. 219-226, January 1995.

[5] Chorus Systems, "*Chorus kernel v3 r5 specification and interface,*" CS/TR-91-62.9, 1991.

[6] H. J. Kim, J. K. Lee, and K. W. Rim, "*Parallel Operating System for MPP System: Design and Implementation,*" Lecture Notes in Computer Science, pp. 413-422, 1996.

[7] C. P. Krusal and M. Snir. "*The performance of multistage interconnection networks for multiprocessor,*" IEEE trans. Comput., pp. 1091-1098, Dec. 1983.

[8] H. Jiang, L. N. Bhuyan, and J. K. Muppala, "*MVAMIN: Mean value analysis algorithms for multistage interconnection networks,*" J. Parallel Distrib. Comput., pp. 189-201, July 1991.

[9] T. Lin and L. Kleinrock, "*Performance analysis*

of finite-buffered multistage interconnection networks with a general traffic pattern," In Proc. ACM SIGMETRICS Conf. Meas. Model. Comput. Syst., pp. 68-78, May 1991.

[10] T. N. Mudge and B. A. Makrucki, "*An approximate queueing model for packet switched multistage interconnection networks,*" In Proc. Int. Conf. Distrib. Comput. Syst., pp. 556-562, Oct. 1982.

[11] K. Park, J. S. Hahn, S. W. Sim, and W. J. Hahn, "*Xcent-Net : A hierarchical crossbar network for a cluster based parallel architecture,*" Proc. of 8th PDCS, 1996. 10.

[12] 모상만, 신상석, 윤석한, 엄기욱, "*고속병렬컴퓨터(SPAX)에서 효율적인 메시지 전달을 위한 메시지 전송 기법,*" 대한전자공학회 논문집, pp. 1299-1304, 제 18권, 제 1호. 1995. 6.

[13] D. Gross and C. Harris, *Fundamentals of Queueing Theory*, Wiley, 1985.

[14] F. S Hillier and R. W. Boling, "*Finite Queue in series with exponential or Erlang service times-a numerical approach,*" Oper. Res. 15, pp. 286-303, 1967.

조 일 연

1991년 성균관대학교 산업공학과(공학사)
1993년 성균관대학교 산업공학과(공학석사)
1993년~현재 한국전자통신연구원 연구원(컴퓨터연구단 시스템연구부)
1995년~1996년 미국, OSF RI(Open Software Foundation Research Institute) 파견 근무
관심분야 : 병렬운영체제, 성능평가

이 재 경

1981년 경북대학교 전자공학과(공학사)
1997년~현재 충남대학교 컴퓨터공학과 석사과정
1983년~1984년 (주)금성반도체 사내 전산과 근무
1985년~현재 한국전자통신연구원 책임연구원(컴퓨터연구단 시스템연구부)
관심분야 : 병렬운영체제, 펌웨어, 병렬프로그래밍

김 해 진

1983년 경북대학교 컴퓨터공학과 졸업(공학사)
1995년 충남대학교 전산학과 졸업(공학석사)
1983년~현재 한국전자통신연구원 책임연구원(컴퓨터연구단 시스템연구부 시스템 S/W 연구실장)
관심분야 : 운영체제, 병렬 처리, 실시간 처리, 고장 감내