

# 입력단 큐잉 방식의 ATM 스위치를 위한 효율적 셀 중재 방식에 관한 연구

김용웅\*\*, °김진상\*, 박영근\*  
연세대학교 전기공학과\*  
LG 정보통신 전송연구소 다중통신실\*\*

## An Effective Cell Scheduling Algorithm for Input Queueing ATM Switch

Kim Yong-Woong, Kim Jin Sang°, and Young-Keun Park\*  
Dept of Computer Communications, Yonsei University  
LG IC Transmission Lab Mux Team\*\*

### 요 약

본 논문에서는 광대역 종합 정보 통신망을 구현하는데 필수적인 ATM 스위치의 방식 중 입력단 큐잉 방식에 적용할 수 있는 셀 중재 기법으로 입력단 큐잉방식에서의 HOL (Head-of-Line) 블럭킹과 출력단 충돌을 개선하여 입력단과 출력단의 매칭이 최대가 되도록 하는 기법인 MUCS(Matrix Unit Cell Scheduler)를 개선한 WMUCS(Weighted Matrix Unit Cell Scheduler)를 제안한다. WMUCS는 MUCS의 장점인 단순한 알고리즘과 높은 처리율에 대한 특성은 그대로 지니면서 더 좋은 특성을 보여주었다. 그리고 MUCS의 문제점인 기근(starvation)현상을 보완하여 최대 처리율을 거의 100%로 출력단 큐잉 방식에 근접하는 뛰어난 결과를 얻어내었다. WMUCS의 성능 분석을 위해 소프트웨어로 시뮬레이션하였다. 가장 중요한 세 가지 파라미터는 최대 처리율과 평균 지연, 그리고 셀 손실률이다. 최대 처리율은 예상한대로 순수한 MUCS보다 다소 개선되었다. 평균 지연은 버스티 트래픽의 경우에 개선 효과가 뚜렷했다. 셀 손실률도 MUCS보다 우수한 수준이다.

### I. 서 론

ATM은 하나의 독립된 네트워크 기술로서 매우 광범위한 연구 분야를 갖고 있으며, 그 중에서도 155.52Mbps정도의 고속 데이터 전송 속도에서 빠르게 데이터를 교환해 줄 수 있는 ATM스위치는 매우 중요한 위치를 갖고 있다고 볼 수 있다. 이러한 위치의 설계에 있어서 버퍼의 위치는 매우 중요한 이슈이다.

버퍼의 관점에서 ATM 스위치는 크게 출력단 큐잉방식 공유 메모리 방식, 입력단 큐잉 방식으로 나눌 수 있다<sup>[1][3]</sup>. 이 중 입력단 큐잉 방식은 스위치 패브릭(switch fabric)의 구현이 쉽고, 구조가 간단하며, 포트나 메모리의 속도를 올릴 필요가 없고, 셀 손실률(CLR : Cell Loss Rate)이 낮은 장점이 있다. 그리고 주어진 셀 손실률을 보장하기 위해 필요한 버퍼의 크기도 다른 방식에 비해 적다. 이런 점 때문에 확장성이 다른 방식에 비해 우수하다. 또한 셀 지연에 관한 특성이 우수하여 최근에 중요성이 커지고 있는 시간에 민감한 멀티미디어 트래픽에 대한 서비스 품질 보장이 용이하다. 그러나, HOL 블럭킹(Head Of Line blocking) 문제, 입력단 충돌, 출력단 충돌로 인하여 스위치의 처리율(throughput)이 다소 떨어진다는 문제점이 있다<sup>[3]</sup>. 이런 한계를 극복하는 보편적인 방법으로, FIFO(First In First Out) 큐잉과 바이패스 큐잉(bypass queueing) 혹은 FIRO(First In Random Out)이 있다. 이 방식은 한 타임 슬롯에 얼마나 많은 셀을 전송할 수 있는가 하는 문제와 셀 손실률 및 평균 지연을 개선하기 위해 준최적화된 셀 중재(cell scheduling or cell arbitration)가 매우 중요하다고 볼 수 있다<sup>[2]</sup>.

이런 관점에서 그 동안 여러 가지 셀 중재 기법이 발표되었고, 처리율 면에서는 이상적인 경우라고 할 수 있는 출력단 큐잉 방식에 버금가는 성능을 보여준 것도 여러 가지가 있다. 그 중에서도 MUCS(Matrix Unit Cell Scheduler)는 일종의 최대수 매칭(maximum size matching) 방법으로 아주 뛰어난 처리율을 보여준다<sup>[4][5][6]</sup>. 이 기법은 입력단 충돌과 출력단 충돌이 될 수 있는 한 적게 되도록 입력 포트와 출력 포트를 선택하는 과정을 반복하여 이후 과정에서 경합에 참여하는 입출력단 쌍의 수가 최대가 되게 하는 것이다. 본 논문에서는 MUCS를 기초로 셀 손실률이나 평균 지연 등의 특성을 향상시킬 수 있는 개선된 알고리즘을 제안한다.

## II. Weighted Matrix Unit Cell Scheduler

그 동안 발표된 여러 가지 기법들은 나름대로 장단점을 가지고 있었다. 근래의 것으로는 PIN[(Parallel Iterative Matching) 계열이나 2DRR(Two Dimensional Round Robin) 알고리즘이 있다<sup>[7][8]</sup>. 본 논문에서는 최근에 발표되어 HOL 블럭킹과 출력단 충돌을 해소하여 대역폭의 이용률을 높여 최대 처리율을 거의 출력단 큐잉에 가깝게 하면서 하드웨어 구현이 비교적 용이한 것으로 알려진 MUCS(Matrix Unit Cell Scheduler)를 고찰하고 이를 개선시킨 알고리즘을 제안한다.

## A. Matrix Unit Cell Scheduler 기법

셀 중계 초기에 트래픽 행렬  $A$ 를 만든다. 스위치의 크기가  $N \times N$ 짜리 일 때, 행렬의 크기는  $N \times N$ 이 된다. 각각의 파라미터  $a_{ij}$ 는 입력 포트  $i$ 에서 출력 포트  $j$ 로 가는 셀이 버퍼에 있는지의 유무를 나타낸다. 그러므로  $a_{ij}$ 는 음이 아닌 정수 값을 가지며 0 혹은 1이다.  $a_{ij} > 0$  일 때는, 입력 포트  $i$ 에서 출력 포트  $j$ 로 가는 셀이 존재함을 나타낸다.  $a_{ij} = 0$  일 때는, 입력 포트  $i$ 에서 출력 포트  $j$ 로 향하는 셀이 존재하지 않음을 나타낸다<sup>[9]</sup>. 그 다음 트래픽 행렬  $A$ 로부터 가중치 행렬(weight matrix)  $W$ 를 만든다. 가중치 행렬의 크기도 역시  $N \times N$ 이다. 이 행렬의 파라미터  $w_{ij}$ 를 계산하고 나서,  $W$ 의 요소 중 가장 큰  $w_{ij}$ 를 선택하여 차례로 반복하고(iteration) 갱신하여 행렬의 크기를 줄여  $A'$ 를 만든다. 다음 단계에서도 같은 방법으로 트래픽 행렬을 줄여나가서 입출력단을 선택하여 최적화된 셀 선택 집합을 선택한다. 이 기법의 핵심은 입력단 충돌과 출력단 충돌이 가장 작아지는 입출력단 쌍을 선택하여 그 다음 단계에서 고를 수 있는 대상이 되는 입출력단 쌍이 최대가 되도록 하는 것이다 이런 일련의 알고리즘은 하드웨어로 구현하여 동시에 병렬적으로 동작하도록 구현이 가능하다<sup>[6]</sup>. 이제 MUCS 알고리즘의 과정과 특성을 알아보면,  $N \times N$ 을 스위치의 크기라 하고  $M \times M$ 을 크기가 줄어든 행렬  $A'$ 의 크기라고 하자. 알고리즘의 단계는 다음과 같다.

1) 초기에는  $M=N$ 이고  $A'=A$ 이다.

2) 각각의 반복 과정(iteration round)에서 가중치 행렬의 요소인  $w_{ij}$ 를 매 과정마다  $a_{ij}$ 로부터 다음의 식을 이용하여 계산한다.

$$w_{ij} = \frac{1}{wr_i} + \frac{1}{wc_j}, \quad \text{if } a_{ij} = 1 \quad (1)$$

$$w_{ij} = 0, \quad \text{if } a_{ij} = 0 \quad (2)$$

이 때,

$$wr_i = \sum_{j=1}^M a_{ij} \quad (3)$$

$$wc_j = \sum_{i=1}^M a_{ij} \quad (4)$$

3)  $M^2$  개의  $w_{ij}$  중에서 큰 제일 것을 선택한다. 이때 제일 큰 값이 복수개일 경우  $i$ 와  $j$ 가 겹치지 않는다면 여러 개를 동시에 선택해도 된다.  $i$ 가 겹치지 않도록 하는 것은 한 입력 포트에서 하나만 선택하기 위함이다.  $j$ 가 겹치지 않도록 하는 것은 출력단 충돌을 방지하기 위함이다. 일단 이렇게 선택된  $a_{ij}$ 에 대하여  $i$ 행과  $j$ 열을 모두 제거하고 크기가 줄어든 새로운 트래픽 행렬  $A'$ 를 만든다. 그리고 나서 역시 크기가 줄어든  $A'$ 에 대한 가중치 행렬  $M$ 를 만든다. 이제는 같은 과정을 반복하는데  $N$ 개의  $a_{ij}$ 가 선택되면 중단한다.

4) 만일 가중치가 같고  $i$ 와  $j$ 가 달라 선택 가능한 것이 여러 개 있다면 무작위로(randomly) 선택한다.

MUCS는 버스티 트래픽 환경에서 MUCS의 성능이 별로 저하되지 않는다는 장점을 가지고 있다. 특히 중요한 것은 이런 우수한 성능이 포트 수에 거의 제한을 받지 않고 유지된다는 것이다. 이는 MUCS의 확장성이 다른 알고리즘에 비해 매우 우수하다는 것을 보여준다<sup>[6]</sup>. 따라서 대용량 ATM 스위치의 구현에 응용하기에 적합하다. 그러나 이 MUCS 알고리즘은 다음과 같은 문제점을 가지고 있다. MUCS에서 트래픽 행렬을 만드는 과정을 보면 각 입력 단에서 출력단으로 가는 패킷의 유무만을 따지기 때문에 각 입력 포트의 길이를 고려하지 않는다. 이는 큐 길이의 변이가 커져서 버퍼 넘침으로 인한 각 손실률의 특성이 나빠질 위험이 크다. 이는 기근(starvation)이 발생하기 때문이다. 무작위로 선택하는 알고리즘에서는 기근이 발생하지 않지만 MUCS는 최대 수 매칭을 지향하기 때문에 기근이 발생한다. 이는 큐간의 공평성에 심각한 문제를 야기할 뿐더러 입력단 큐잉의 장점인 셀 손실률에도 나쁜 영향을 미친다. 본 논문에서는 이런 여러 가지 단점을 보완하고 장점을 살릴 수 있도록 MUCS를 개선한 새로운 알고리즘을 제안한다.

### B. 제안하는 Weighted MUCS 기법

위와 같은 문제를 해결하기 위해 본 논문에서는 트래픽 행렬을 생성하는 과정에 변화를 주었다. MUCS의 트래픽 행렬에서 각각의 파라미터  $a_{ij}$ 는 입력 포트  $i$ 에서 출력 포트  $j$ 로 가는 셀이 버퍼에 있는지의 유무를 나타낸다. 그러므로  $a_{ij}$ 는 음이 아닌 정수 값을 가지며 0 혹은 1이다. 이 부분에서, 셀이 없을 때 0으로 하는 것은 같지만 셀이 있을 경우에 단순히 유무를 1로 표시하는 것이 아니라, 그 개수를 세서 음이 아닌 정수로 표현한다.

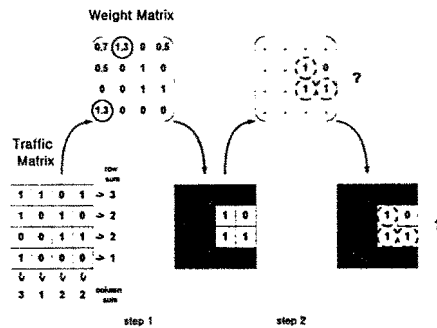


그림 6 MUCS 알고리즘 적용

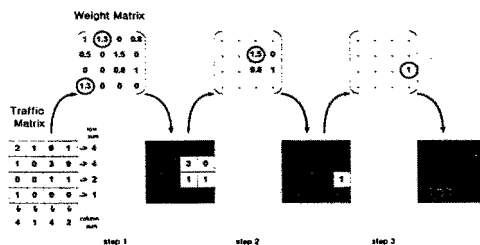


그림 2 WMUCS 알고리즘 적용

즉, 위의 식 (1)과 (2)에서  $w_j$ 의 계산이 달라진다.

$$w_{ij} = \frac{a_{ij}}{wr_i} + \frac{a_{ji}}{wr_j}, \quad \text{if } a_{ij} > 1 \quad (5)$$

$$w_{ij} = 0, \quad \text{if } a_{ij} = 0 \quad (6)$$

이런 개선이 구체적으로 최대수 매칭에 어떻게 기여하는지 알아보면, 트래픽 행렬을 만드는 방법으로 그림 1은 순수한 MUCS를 적용하였고 그림 2는 개선된 MUCS를 적용한 것이다. MUCS를 적용하면 2단계에서 가중치가 1인 파라미터가 3개가 나온다. 이런 경우에는 무작위로 선택해야 하는데, 운 좋게도  $w_{23}$ 과  $w_{32}$ 를 선택할 경우에는 3단계에서  $w_{33}$ 을 선택하여 4개의 입출력단 쌍을 선택할 수 있다. 그러나  $w_{33}$ 을 선택하면 더 이상 진행하지 못하여 3개의 입출력단 쌍을 고를 수밖에 없다. 개선된 MUCS는 각 입력 포트에서 기다리는 셀의 수를 트래픽 행렬에 반영하기 때문에 가중치 행렬이 약간 다르게 나온다. 이 경우에 가중치 행렬의 파라미터 중에서 같은 값을 가질 확률은 버퍼가 안정되어 있는 한 거의 없다. 2단계의 가중치 행렬을 보면 MUCS에서 같은 값을 가졌던  $w_{23}$ ,  $w_{32}$ ,  $w_{33}$ 이 비로소 다른 값을 가진다. 이제 2 단계에서  $w_{23}$ 을 선택하고 나서 3단계에서  $w_{32}$ 를 선택하면 4개의 입출력단 쌍을 고를 수 있다. 이와 같이 WMUCS(Weighted MUCS)라 이름 붙인 알고리즘은 가중치 행렬에서 같은 값이 나와 선택할 수 있는 기회를 놓칠 수 있는 가능성을 없앴다. 또한 앞에서 언급한 기근문제에 대해 개선 효과를 기대할 수 있다. 왜냐하면, 기근으로 인해 선택받지 못한 입력 버퍼는 그만큼 셀이 쌓일 것이므로 타임 슬롯이 경과할수록 가중치 행렬에서 같이 높아질 것이기 때문이다. 물론 기존의 이산화된 (즉, 0 또는 1) 수치를 쓴 알고리즘보다 복잡도가 다소 늘어나기는 하지만 MUCS 알고리즘이 본래 단순하고 계산이 적은 편이기 때문에 최근의 하드웨어 속도라면 별 무리 없이 구현 가능하다고 예측할 수 있다. 이상으로 Weighted MUCS에 대한 이론적인 검토를 마치고 다음 장에서는 컴퓨터 시뮬레이션을 통해 다양한 트래픽 유형이나 파라미터의 변화에 따른 MUCS의 성능을 평가하도록 하겠다.

### III. WMUCS의 성능 고찰

#### A. 시뮬레이션 환경

본 논문에서는 제안한 WMUCS 기법의 성능을 C++ 언어로 작성한 사건 발생(Event-Driven) 시뮬레이션 프로그램을 이용하여 검증하였으며, 스위치에 부가하는 트래픽은 크게 균일 무작위 트래픽(uniform random traffic)과 버스티 트래픽(bursty traffic)이다. 또 확장성에 대한 성능을 측정하기 위해 스위치의 크기에 변화 주었다. 이런 각 요소에 대하여 셀 도착률을 변화시켜가면서 성능 측정 파라미터로 최대 처리율, 셀 손실률과 평균 셀 지연을 측정한다.

#### B. 시뮬레이션 결과 및 분석

먼저 스위치의 가장 기본적인 성능 파라미터인 최대 처리율에 대하여 살펴보자. 그림 3은 포트 수에 따른 최대 처리율을 보여주고 있다.

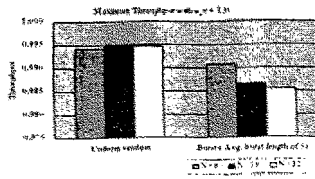


그림 3 포트 수 N에 따른 최대 처리율

포트 수는  $N=8$ ,  $N=16$ ,  $N=32$ 로 하였다. 왼쪽은 균일 무작위 트래픽을 인가한 것이고 오른쪽은 평균 버스티 길이가 5인 버스티 트래픽을 인가한 것이다. 트래픽에서 보듯이 균일 무작위 트래픽일 때는 포트 수  $N$ 에 상관없이 0.990을 상회하는 높은 처리율을 나타내었다. 버스티 트래픽을 때에는 균일 무작위 트래픽에 비해 다소 낮은 값이 나오지만 역시 0.985를 상회한다. 주목할 것은 버스티 트래픽의 경우에는 포트 수  $N$ 이 커짐에 따라 처리율이 영향을 받는다는 것이다.

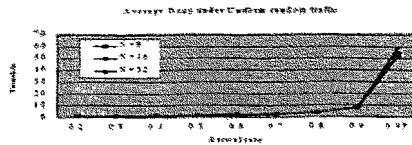


그림 4 셀 도착률에 대한 평균 지연[균일 무작위 트래픽]

그림 4는 셀 도착률에 따른 평균 지연을 나타내고 있다. 균일 무작위 트래픽을 인가한 것이다. 셀 도착률이 0.7이하일 경우에는 지연이 거의 완만하게 유지되는데 0.7을 넘어서면 지연이 커진다. 그림 5는 버스티 트래픽에 대한 평균 지연에 대한 것이다. 예측했던 대로 균일 무작위 트래픽보다는 버스티 트래픽에서 지연이 큰데, 0.7 이하에서는 대략 2배 정도이지만 0.7을 넘어서면 지연이 3배정도의 차이가 난다. 그림 4와 5에서 보면 스위치 포트 수의 증가가 평균 지연에 미치는 영향은 미미하다. 이는 WMUCS의 확장성을 확인할 수 있는 부분이다.

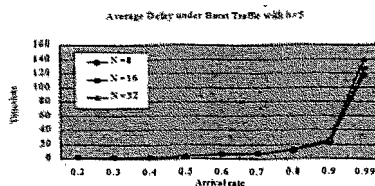


그림 5 셀 도착률에 대한 평균 지연[버스티 트래픽]

그림 6은 균일 무작위 트래픽에 대한 셀 손실률을 보여준다.  $10^{-6}$  이하의 셀 손실률을 유지하기 위해서  $N=8$ 일 경우에는 큐 길이가 11정도 되어야 하고  $N=16$ 일 경우에는 큐 길이가 15정도 되어야 한다. 포트 수가 두 배인데 추가로 필요한 큐 길이가 4라면 그리 손해보는 것은 아니다.

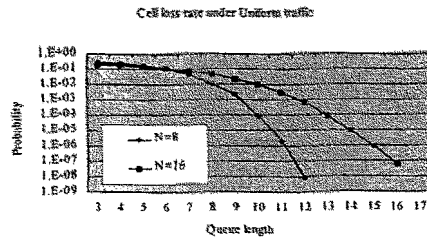


그림 6 균일 무작위 트래픽에 대한 셀 손실률

그림 7에는 버스티 트래픽에 대한 셀 손실률을 나타내었다.  $10^{-6}$  이하의  $N=8$ 일 경우에는 큐 길이가 23정도 되어야 하고  $N=16$ 일 경우에는 큐 길이가 28정도 되어야 한다.

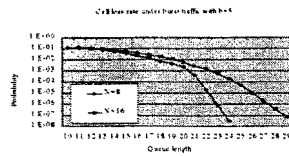


그림 7 버스티 트래픽에 대한 셀 손실률

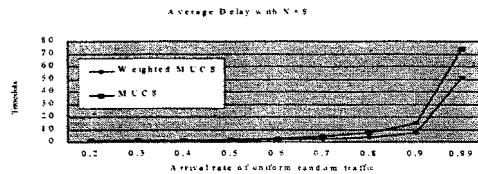


그림 8 평균 지연으로 본 WMUCS와 MUCS

그림 8에서 평균 지연에 대하여 Weighted MUCS와 MUCS를 비교해 보았다. 셀 도착률이 0.6 이하일 때는 별 차이가 없는데 0.7 근처에서부터 차이가 나기 시작해서 0.99의 셀 도착률에서는 20 슬롯 정도의 차이가 난다. 이로부터 트래픽 부하가 높으면 Weighted MUCS의 평균 지연 특성이 MUCS보다 뚜렷하게 양호한 결과를 보인다. 이는 기근 문제의 해소와 공평성 문제를 해소했기 때문인 것으로 보인다. 그림 9에서는 최대 처리율의 측면에서 WMUCS와 MUCS를 비교하였다. 균일 랜덤 트래픽일 때보다 버스티 트래픽일 때 처리율의 개선폭이 더 크다. 즉, 버스티 환경에서는 WMUCS가 MUCS에 비해서 특히 최대 처리율이 더 우수하다.

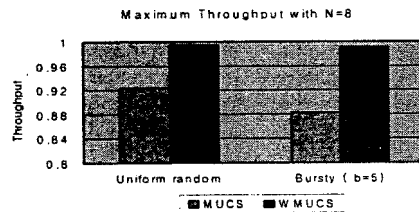


그림 9 최대 처리율로 본 WMUCS와 MUCS

## IV. 결 론

본 논문은 처리율이 뛰어나고 알고리즘이 간단한 Matrix Unit Cell Scheduler에 주목하고, 이를 분석하여 문제점을 개선한 Weighted MUCS를 제안하였다. MUCS 기법에 저장된 셀의 길이를 고려한 가중치를 두어 입력단과 출력단의 쌍이 최대로 선택되는 가능성을 더욱 높이는데 중점을 두었다. 시뮬레이션을 통한 다양한 성능 분석 결과는 최대 처리율은 이상적인 경우라 할 수 있는 출력단 큐잉 방식과 거의 비슷한 결과가 나왔다. 이런 처리율의 특성은 트래픽 부하가 100%인 경우에도 유지됨을 알 수 있었다. 또 스위치의 크기가 증가하더라도 최대 처리율은 여전히 우수한 수준을 나타내는 결과를 보여주었다. 셀 손실률은 입력단 큐잉 방식답게 만족할만한 결과를 보여주었다. 시뮬레이션 결과에서 특히 주목할 것은 스위치의 크기가 커져도 성능 인자에는 영향을 주지 않는다는 사실이다. MUCS 알고리즘이 하드웨어적으로 구현이 용이한 것으로 알려져 있는데, WMUCS 또한 간결한 알고리즘에 속하기 때문에 하드웨어적으로 구현하는데 어려움이 없을 것으로 보인다.

## 참고 문헌

- [1] Perter Newman, "ATM technology for corporate networks", *IEEE Communications Magazines*, pp. 90-101, April, 1992
- [2] Young-Keun Park, Gyunho Lee, "Neural network-based ATM cell scheduling with queue length-based priority scheme", *IEEE Journal on Selected Areas in Communications-Computational and Artificial Intelligence in High Speed Networks*, Vol.12 No.2 1997.2.
- [3] Michael G. Hluchyj, Mark J. Karol, "Queueing in high-performance packet switching", *IEEE JSAC in communication*, vol.6, December 1988
- [4] H. Duan, "Design and development of cell queueing, processing, and scheduling modules for iPOINT input-buffered ATM testbed", *Ph.D. Dissertation*, Univ. Illinois at Urbana-Champaign 1997.
- [5] H. Duan, J.W. Lockwood, S.M. Kang, "A High-performance OC-12/ OC-48 queue design prototype for input-buffered ATM switch", *IEEE Inform'97*, pp.20-28, 1997.
- [6] H. Duan, J.W. Lockwood and S.M. Kang, "Matrix Unit Cell Scheduler (MUCS) for Input-Buffered ATM Switches", *IEEE Communication Letters*, vol.2, no.1, pp.20-23
- [7] T.E. Anderson, "High Speed Switch Scheduling for Area Networks", *ACM Trans. on computer system*, Vol/11. NO.4, pp. 319-352, Nov. 1993
- [8] R.O. Lemaire, D.N. Serpanos. "Two-Dimensional Round-Robin Schedulers for Packet Switches with Multiple Input Queues", *IEEE/ACM Trans. on Networking*, Vol. 2, No.3, pp. 471-482, Oct. 1994.
- [9] H. Duan, J.W Lockwood, S.M. Kang, "A 3-dimensional queue (3DQ) for practical ultra-broadband input buffered ATM switches", *IEEE Tran. VLSI Syst*, submitted for publication.