Designing Efficient Benes and Banyan Based Input-Buffered ATM Switches

Rajendra V. Boppana*
Computer Science Division
The Univ. of Texas at San Antonio
San Antonio, TX 78249-0667
boppana@cs.utsa.edu

C. S. Raghavendra
The Aerospace Corporation
P. O. Box 92957
Los Angeles, CA 90009-2957
raghu@aero.org

Abstract. Multistage network based input-buffered ATM switches, which have been studied extensively in the recent past, are cheaper compared to crossbar designs but suffer from elaborate cell selection methods or expensive network setup. In this paper, a fast cell selection method is proposed to avoid slow cell selection and costly network setup for these designs. In particular, we propose network hardware specific selection techniques for cell selection in input buffered Banyan network with internal speed twice that of the external links. Our simulation results show that cell selection by looking at up to 10 cells in each input queue for switch sizes up to $N = 64$ yields 95% or higher switch utilization.

1 Introduction

Many types of switching fabrics have been proposed for use in ATM networks [12, 15, 16]. These include input buffered switches, output buffered shared memory switches, crossbar-based switches, and multi-stage network based switches. The simplest non-blocking, input-buffered switch is the crossbar switch, where, cells arriving at each input are queued at the respective input-buffers. The queues can be served by the switch in sending cells from an input to a desired output. However, in each cycle, an output can only receive one input cell. If the input queues are served in FIFO order, then among all the Head-of-Line (HOL) cells, we can choose those cells that are going to distinct outputs. It is shown in [5] that a crossbar serving HOL cells only can achieve a maximum throughput of about 58%.

Although output queueing can achieve high throughput, there are some drawbacks. Since cells from inputs going to the same output get queued in one queue, cell loss could occur if there is no room in the buffer due to bursty arrival of cells to that output. This is alleviated to some extent by using shared buffer memory rather than partitioning the memory among the queues. In such output-buffered switches, the memory bandwidth must be high as N reads and N writes (for an $N \times N$ switch) to memory need to be performed in each cycle. Another drawback is providing fairness to the users. Since the output queues are served in FIFO order it is possible for some inputs to experience long delays due to bursty arrivals at other inputs.

For these reasons and to support rate based services for inputs, there is significant interest in using input-buffered switches in ATM networks [1, 6, 9, 11]. It has been shown that the HOL blocking problem can be eliminated and arbitrarily close to 100% throughput can be achieved if multiple cells in each input are examined and selected such that there is at most one cell from each input and at most one cell to each output [1]. To facilitate searching of multiple cells at each input, one queue for each output could be maintained at each input. Alternatively, one queue is maintained at each input, but first k, $k > 1$, cells may be searched [2, 8], or input ports may be grouped [6].

The problem of searching and selecting cells in input buffers to maximize the number of cells that can be routed in a cycle can be modeled as a bipartite graph matching problem. A deterministic algorithm for this matching problem takes O($N^{2.5}$) time, which is quite expensive, even for $N = 16$, to run in each step of routing ATM cells. Several researchers have used randomized algorithm for speeding up this search problem. It is shown in [1, 11] that by using $\log N$ iterations it is possible to achieve maximal matching with high probability. The advantage of randomized algorithms is that it is very simple to implement and gives very good results on the average. These randomized schemes use at least N-depth lookup.

Recently, a low-cost non-blocking switch, the Benes network, with input buffers was proposed as an alternative to crossbar based switches [8]. The authors use a sequential procedure with each input buffers size varying from $N/2$ to $5N/2$ for cell selection and show that 95% or better throughput can be achieved in switches of size up to $N = 64$. Their searching considers both output contention and network link contention. The complexity of this search procedure is $O(N^2)$ and there is no guarantee of obtaining maximal match in each cycle with this sequential search.

In this paper, we investigate methods that efficiently select cells, while greatly simplifying the network control issue when a multistage network rather than a crossbar is

*Boppana's research has been partially supported by DOD/AFOSR grant F49620-96-1-0472 and NSF grant CDA 9633299.

used for switching. We consider using a single copy of the Banyan network as the switch, but operating with internal speeds twice that of external speeds, for example as in [14]. In each cycle we select and route two partial permutations. For this purpose, we developed network specific hardware selection to obtain two partial permutations that are passable in the given blocking network. Our simulation results show that with our hardware selection schemes we can achieve 95% or better throughput. Therefore, we can use a single blocking network operating at twice the speed to obtain similar performance compared to Benes network which requires expensive routing.

2 Cell Selection and Routing in Input-Buffered Switches

Consider an $N \times N$, $N = 2^n$, non-blocking ATM switching fabric with input buffers. A buffer is associated with each input for queueing the incoming cells. The switch can determine by table lookup using VCI to which output link a cell needs to be routed. In one cycle, we can route, possibly, one cell from each input to its desired output link. Each output link can only receive one cell in a cycle, and therefore, we need to carefully select cells from the input buffers for routing. Of course, our goal is to maximize the throughput while keeping the ordering of cells going from a particular input link to a given output link.

By using a non-blocking network, such as the crossbar or the Benes network, we will avoid network conflicts. Thus, in switch routing, we only need to consider output conflicts. If we look at only the HOL cells from each input buffer, the throughput will be limited. The idea is to look deeper into each input buffer to select non-conflicting cells for routing. Several researchers have used this approach to studying the throughput performance of input buffered switches. At ATM link speeds, the cycle time for routing cells is very short, and previous studies have used fast randomization algorithms for selecting sub-optimal solutions.

Anderson et.al., [1] show that the problem of selecting the cells with distinct destinations from the input queues can be modeled as a bipartite graph matching problem. From the input queued data, one can construct a bipartite graph with nodes as the input and output ports and the edges denoting the destinations desired by the input cells. Such a graph constructed in [1] will have $2N$ nodes and at most N^2 edges. By using a randomized iterative matching algorithm with O($\log N$) iterations, one can achieve high throughput. Another recent work [11] also used randomized algorithm approach for cell scheduling to provide bandwidth guarantees in input buffered crossbar switch.

The Benes network is known to be topologically equivalent to a cascade of two copies of Banyan network. Banyan networks are known to be self routable, although only for the set of passable permutations through that network. The two copies of Banyan networks can be used in cascade with all lines operating at same speed, or in time with a single copy of the Banyan network and recirculating the outputs back to inputs and clocking the internal links twice as fast as the external links. This will reduce the hardware cost further and such an approach was used in the design of Phoenix switch [14]. With such a Banyan network, we can also perform two independent switching steps within the switch and match with the line speed. With this mode of operation, in each cell switching time, we can select and route two separate routable sub-permutations through the Banyan network.

Our goal is to use the Banyan network with internal links twice as fast as the line speed as input-buffered ATM switches. For efficient operation of the switch, we would like to take advantage of the self-routing capabilities of these networks. However, the Benes network can self-route only a few subsets of permutations, such as the linear-complement class. So, the question is how to select input cells from the buffers that have no output conflicts and are routable by the self routing algorithm of the Benes network. The Banyan network can self route even smaller subset of permutations. The problems of selecting the cells that area routable without conflict in the network still remains. If we can find a fast approach to choosing large number of cells from the input buffers for self routing through these networks, then we can achieve high throughput.

Our idea is to use a copy of the network of interest (Benes or Banyan) as the control network to efficiently determine the routable set of cells from the input buffers. Although this control network is topologically identical to the routing network, its switching elements can be simpler as it is used only for making decisions by just moving the destination bits through them. The idea is to attempt routing the tags through the control network and randomly select inputs to proceed when there are conflicts in a switching stage. We can repeat this procedure several times to increase the number of cells that can be routed in the next step through the routing copy of the network. This essentially implements the randomization schemes used in crossbar on a Benes or Banyan network. In the next section, we describe the details of this hardware specific cell selection techniques and evaluate the performance vis simulation.

3 Network Specific Hardware Selection Methods

We first describe the hardware selection method for both rearrangeable networks such as the Benes network and blocking networks such as the Omega networks. We explain

how the technique applies to Benes and Omega using examples. Next we present some simulation results on the performance of the proposed selection method.

3.1 Selection Method

We propose to use a copy of the underlying network itself as the control network to aid the destination selection process, thereby cell selection and routing are achieved simultaneously. However, this control copy of the network needs to be bit routable only as we only route destination address information and do not route data packets through this network. Hence, each stage of this network could operate in a clock step and several cycles of selection/routing could be performed in a single ATM cell data switching time. Our scheme works as follows. In each cell time slot, the selection process is conducted to determine the inputs and cells that they send in the following time slot. This technique works for Benes, Banyan or any other multistage network.

There are multiple rounds of contest for destination selection. In the first round, all inputs have equal chance of winning. In a round of contest, each input that has not won earlier, will send the destination address of the cell it intends to send in the next time slot; each destination that has won in an earlier round of the current selection process, will send the address of the destination it won. Each input chooses its output request independently of the other inputs. The destination bits propagate through the network, stage by stage. At each stage, a switch uses an appropriate routing digit to route inputs to outputs. Conflicts are resolved as follows. If there is an input with a winning destination address, then it is routed to the output in consideration. (It is noteworthy that there can be at most one input with a winning destination address that competes for an output of any switch.) Otherwise, one of the inputs is randomly chosen and routed to the output. If the network is a unique path network, such as the Banyan, the losers can be dropped at this point. Otherwise, the remaining inputs are randomly assigned the remaining outputs. A round concludes after routing the destination addresses through the last stage of the network and inputs that have the winning addresses are notified. It can be easily shown that the number of inputs with winning destination addresses is a monotonically increasing function of the number of rounds. After a few such rounds we have a set of destinations that can be routed in the next time slot, on the actual data network, *without conflicts* using the paths selected in the selection process.

3.2 Selection of Cells for Blocking Networks

To see how the selection process works for blocking multistage networks, let us consider an 8×8 Omega network designed using 2×2 switches. Figure 1 gives an example of the cell selection. Initially, each input requests for the output specified by its first cell in the queue. It is helpful to view the destination addresses as 3-bit numbers. In each stage, a particular bit (called the routing bit) of the destination addresses is used to determine how the 2×2 switches in that stage are setup. For the Omega network, the left most bit, middle bit, and right most bit are the routing bits for the left, middle, and right stages, respectively. If the routing bit of a request is a 0, then that request should be routed to the upper output of the 2×2 switch; otherwise it should be routed to the lower output. This is the so-called Omega self-routing method. If both inputs of a 2×2 switch have the same value for their routing bits, then only one of them can be connected to the output of that switch correctly; the other can be dropped, since once a path is misrouted, it will never reach the correct destination.

Using these rules, the input connection requests are propagated through the Omega network. At each stage, the requests that are not routed correctly due to contention are dropped. Three inputs win in the first round, two in the second round, and one more in the third round. In this example, further rounds do not improve the matching. The paths that are used for winning inputs' requests are not disturbed in later rounds. For example, inputs 0, 3, and 6 (when counted from top starting with 0), win in the first round and have paths to their destinations 6, 4, and 0, respectively, established. These paths are not disturbed in rounds 2 and 3.

3.3 Selection of Cells for Rearrangeable Networks

A commonly used rearrangeable multistage network is the Benes network. In the Benes network, an input's connection request that is misrouted in the first $\log N - 1$ stages may still reach its correct output provided it is not misrouted in the final $\log N$ stages. For example, in an 8×8 Benes network, we allow connection requests to be misrouted when there is contention in the first two stages of 2×2 switches, unless both requests are to the same destination. The network setup for the last three stages is exactly as in the case of the Omega network. More details are given in the preliminary version of this paper, available at http://www.cs.utsa.edu/faculty/boppana/papers.htm.

3.4 Enhancements to the Selection Method

This scheme can be enhanced in several ways. We can make the inputs that lost in the previous selection round choose destinations that are not taken already by a winning input. A bit vector, one bit for each destination to indicate its availability, can be easily maintained for this purpose. Each input still selects its cell independently of the other inputs; it merely avoids choosing a cell with a destination that has the corresponding bit in the bit vector set.

Another enhancement is to run the network twice as

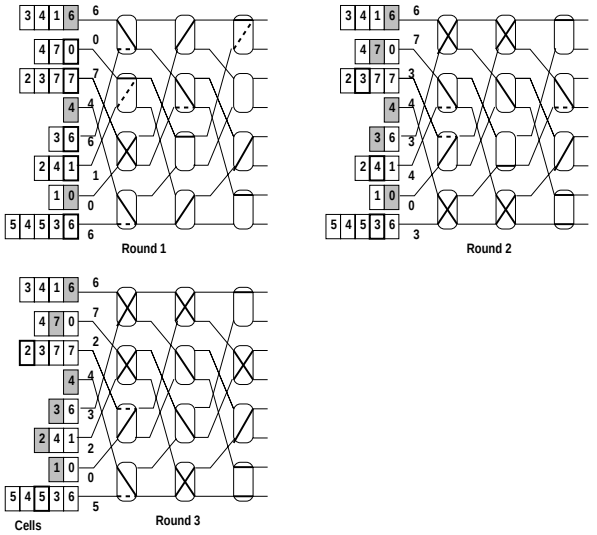

Figure 1: Example of Omega network based cell selection. The rectangular blocks indicate the cells (with numbers indicating their destinations) queued at switch inputs. The numbers at the inputs of the first stage of switches indicate the destinations of the cells that inputs wish to send. Shaded cells indicate winning inputs and the cells selected; thick-lined cells indicate inputs with rejected requests. In a switch, dashed lines indicate rejected connections and solid lines accepted connections. Inputs with rejected connections select new outputs for future rounds. Inputs with accepted connections use the same outputs for later rounds.

fast as the ATM switch being designed. This is a feasible method and used, albeit for lower speeds, in the design of the Phoenix switch [14]. Since the number of rounds that can be done in a time slot is finite and determined by the cell transmission time, we can split the rounds for selecting input-output pairs between the two passes through the network. An input may send (based on the selection process) a cell either in the first pass or in the second pass only. Alternatively, we can use two copies of the multistage network, each running at the same speed of the ATM switch being designed.

We have conducted several simulations using the Benes and Banyan networks with the selection technique described above. Our simulations indicate that, the Banyan network performs as well as the Benes. This may seem counter intuitive because the Benes network is more powerful than the Banyan when its first half is set up using sophisticated routing algorithms. Since our selection method obviates this, there is no appreciable performance difference between the Benes and Banyan networks.

It may appear that the proposed design is expensive to implement. Let us say we use three copies of the Banyan network: two copies for cell routing and one copy for cell selection. Then we need at most $3 \log N$ stages of switches. Since the cell selection network handles only $\log N$ bits of information for comparison and routing, the switch sizes for this network will be very small. Many of the previous designs use the Benes network or two or more copies of a Banyan to switch cells. Compared to these designs, our methods requires an extra copy of the Banyan to aid the selection process. In return, we simplify the selection process, make the network control trivial, and obtain high throughputs comparable to those achieved with full crossbars.

3.5 Evaluation of the Selection Method Using Simulations

We have conducted extensive simulations to evaluate the performance of the proposed technique. Because of the selection method used, even a conflict-prone network such as the Banyan performs as well as the Benes network. We present below performance results for simulations of ATM switches with the Banyan as the switching network. We use two copies of the Banyan with a total of 10 rounds (5 rounds for each copy of the network) of contest to determine inputs and the cells they can send for the subsequent time slot. Figure 2 presents the delay results and Figure 3 presents the throughput results. As in the case of crossbar based designs, the switch utilization is over 95% and queueing delays are very small until just before the point of saturation. For each point in the graph, the half-width of the corresponding 95% confidence interval is within 5% of the mean value reported.

4 Conclusions

In this paper we investigated various schemes for cell scheduling in input buffered switches. This problem can be solved as a maximum matching in bipartite graph, although it can be expensive for larger switches. We showed that more efficient cell scheduling is possible by exploiting the self routing capabilities of Benes and Banyan networks. We presented network specific hardware selection methods so that very simple routing algorithms can be used and we show that high throughput can be achieved with Omega (or Banyan) network with either two copies of the network or operate the network at twice the speed compared to input/output links. Table 1 gives a comparison of various approaches in the ATM switch design.

Simulations are used to evaluate the performance of these methods under various traffic conditions. The results show that high throughput can be achieved and practical switches can be built using smaller input-buffered switches. Such low cost networks can be used in very high-speed input-buffered ATM switches or in IP switches. We are currently applying the selection methods to handle multicast

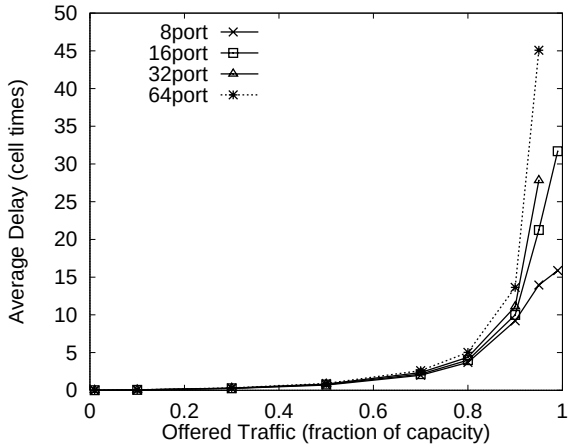

Figure 2: Performance of Banyan Network with Uniform Traffic. The curve with label $d$port indicates the performance of a $d \times d$ port switch.

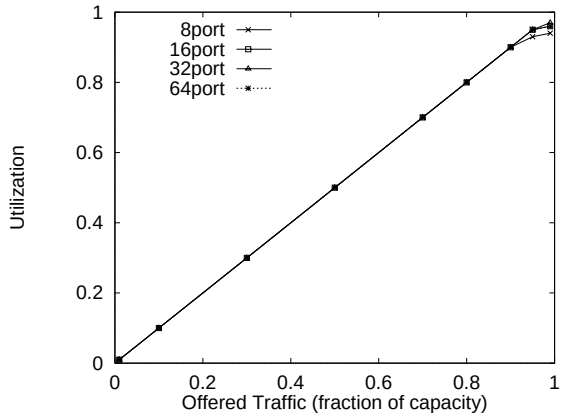

Figure 3: Throughput of Banyan Network with Uniform Traffic. The curve with label $d$port indicates the performance of a $d \times d$ port switch.

traffic and to provide fair access to all outputs by inputs.

References

[1] T. E. Anderson, S. S. Owicki, J. B. Saxe and C. P. Thacker, "High Speed Switch Scheduling for Local-Area Networks," ACM transactions on Computer Systems, Vol 11, No. 4, November 1993,pp. 319-352.

[2] M. Chen, N. D. Georganas, O. W. W. Yang, "A Fast Algorithm for Multi-Channel/Port Traffic Assignment," Proc. ICC 94, pp. 96-100.

[3] E. A. Dinic, "Algorithm for Solution of a Problem of Maximum Flow in a network with Power Estimation," Soviet Math. Dokl.,Vol 11,1970, pp 1277-1280.

[4] T. Hanawa, H. Amano, Y. Fujikawa, "Multistage Interconnection Networks with multiple Outlets," in Proc. of International Conference on Parallel Processing, 1994.

Table 1: Comparison of ATM switch designs

Approach	Cell Selection	Network Setup
Hardware selection + Crossbar (see [1])	Hardware implementation; fast	Standard crossbar setup
Examine inputs + Benes network (see [8])	Algorithmic or processor based; N/2 to 5N/2 cells per input checked; $O(N^2)$ time	Careful network setup needed; $O(NlogN)$ sequential time
Hardware selection + Banyan network (this paper)	Hardware implementation; fast; more expensive than crossbar schemes	Self-routing or standard Banyan network setup

[5] M. J. Karol, M. G. Hluchyj, S. P. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch," IEEE Transactions on Communications, December 1987, pp. 1347-1356.

[6] M. J. Karol, K. Y. Eng, H. Obara, "Improving the Performance of Input-Queued ATM Packet Switches," Proc. IEEE INFOCOM 1992, pp. 110-115.

[7] T. T. Lee, S. Y. Liew, "Parallel Routing Algorithms in benes-Clos Networks," Proc. INFOCOM 96, pp. 279-286.

[8] J. F. Lin, S. D. Wang, "High-Performance Low-Cost Non-Blocking Switch for ATM," Proceedings of the INFOCOM'96, pp. 818-822.

[9] N. McKeown, V. Anantharam, J. Walrand, "Achieving 100% Throughput in an Input-Queued Switch," Proc. INFOCOM 96, pp. 296-302.

[10] D. C. Opferman, N. T. Tsao-Wu, "On a Class of Rearrangeable Switching Networks Part I: Control Algorithm," Bell Syst. Tech. Jour., May-June 1971, pp. 1579-1600.

[11] D. Stiliadis, A. Varma, "Providing Bandwidth Guarantees in an Input-Buffered Crossbar Switch," Proc. INFOCOM 95.

[12] F. A. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Network," Proc. IEEE, Jan. 1990, pp. 133-167.

[13] S. Urushidani, "Rerouting Network: A High-performance Self Routing Switch for B-ISDN," IEEE Journal on Selected Areas in communications. Vol 9 No 8 October 1991.

[14] V. P. Kumar, J. G. Kneuer, D. Pal, and B. Brunner, "Phoenix: A building Block for Fault-tolerant Broadband Packet Switches," in *Proc. IEEE GLOBECOM'91*, Phoenix, Arizona, pp. 228-233, Dec. 1991.

[15] P.C. Wong, M.S. Yeung, "Design and Analysis of a Novel Fast Banyan Switch - Pipeline Banyan," in IEEE/ACM transactions on Networking, Vol. 3 No.1 February 1995, pp 63-69.

[16] Y. Yeh, M. Hluchyj, and A. Acampora, "The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching," IEEE Journal on Selected Areas in Communications, Vol SAC-5, No. 8, October 1987, pp. 1274-1283.