FORTH-ICS / TR-171 July 1996

# Credit-Flow-Controlled ATM versus Wormhole Routing

*M. Katevenis, D. Serpanos, E. Spyridakis*

**ABSTRACT:** ATM has been adopted as the main high speed technology in both wide and local area networks. When ATM is combined with credits –the flow control mechanism that is particularly suitable for local data communication– it becomes appropriate for *multiprocessor* interconnection networks as well. Actually, credit-flow-controlled ATM has similarities with wormhole routing, one of the most popular architectures for MP networks: they both use credits and fixed size cells/flits, and their hardware complexity is comparable. In this paper, we show that ATM with credits performs better than wormhole routing, because ATM uses lanes more efficiently: ATM provides high throughput and low latency with much less buffer space than that required by wormhole routing; also, ATM demonstrates little sensitivity to bursty traffic, and, unlike wormhole, it is fair in terms of latency in hot-spot configurations. Our simulation uses detailed and realistic switch models, which operate at clock-cycle granularity and track closely "ATLAS I", a single-chip ATM switch under development.

**KEYWORDS:** *ATM, wormhole routing, credit-based flow control, multi-lane back-pressure, bursty traffic, hot-spot traffic.*

URL: file://ftp.ics.forth.gr/tech-reports/1996/1996.TR171.ATM_vs_Wormhole.ps.gz

# Credit-Flow-Controlled ATM versus Wormhole Routing

*Manolis Katevenis†, Dimitrios Serpanos††, and Emmanuel Spyridakis†*

Technical Report FORTH-ICS / TR-171 – July 1996

Institute of Computer Science (ICS)
Foundation for Research and Technology – Hellas (FORTH)
Science and Technology Park of Crete, Vassilika Vouton,
P.O.Box 1385, Heraklion, Crete, GR 711 10 Greece
E-mail: katevenis@ics.forth.gr Tel.: +30 (81) 391664 Fax: +30 (81) 391661



URL: file://ftp.ics.forth.gr/tech-reports/1996/1996.TR171.ATM_vs_Wormhole.ps.gz

**ABSTRACT:** ATM has been adopted as the main high speed technology in both wide and local area networks. When ATM is combined with credits –the flow control mechanism that is particularly suitable for local data communication– it becomes appropriate for *multiprocessor* interconnection networks as well. Actually, credit-flow-controlled ATM has similarities with wormhole routing, one of the most popular architectures for MP networks: they both use credits and fixed size cells/flits, and their hardware complexity is comparable. In this paper, we show that ATM with credits performs better than wormhole routing, because ATM uses lanes more efficiently: ATM provides high throughput and low latency with much less buffer space than that required by wormhole routing; also, ATM demonstrates little sensitivity to bursty traffic, and, unlike wormhole, it is fair in terms of latency in hot-spot configurations. Our simulation uses detailed and realistic switch models, which operate at clock-cycle granularity and track closely "ATLAS I", a single-chip ATM switch under development.

**KEYWORDS:** *ATM, wormhole routing, credit-based flow control, multi-lane back-pressure, bursty traffic, hot-spot traffic.*

† Institute of Computer Science, FORTH; also with the Dept. of Computer Science, University of Crete, Heraklion, Crete Greece.

†† IBM, T.J.Watson Research Center, Yorktown Heights, NY USA; work performed at FORTH while on leave of absence from IBM.

## 1. Introduction

*Asynchronous Transfer Mode (ATM)* [LeBo92] is an architecture that quickly becomes dominant in both wide and local area networks. We argue that ATM, extended with credit flow control, is also appropriate for *multiprocessor* interconnection networks. One factor pushing in that direction is the recent blurring of the boundaries between multiprocessors (MP) and computers connected through local area networks (LAN): *networks of workstations (NOW),* or clusters of workstations (COW), or local area multiprocessors (LAMP) may become the parallel computers of tomorrow [ACPN95]. The other factor is that, as we show in this paper, credit-flow-controlled ATM is similar to but performs better than wormhole routing. *Wormhole Routing* [DaSe87] is an interconnection network architecture that was frequently used in multiprocessors, e.g. in Torus [DaSe87], iWarp [Bork90], iPSC/2 [Nuge88], Alewife [KuAg93], C104 [Inmo93], Myrinet [Bode95].

There are notable analogies between ATM and wormhole routing. In an ATM network, variable size packets are fragmented into fixed size *cells* before entering the network. Cells are transmitted over virtual circuits (VCs), which are fixed routes over the network. All cells of a packet are assigned the same VC and are transmitted in order. In a wormhole network, variable size packets are fragmented into fixed size *flits* before entering the network. Flits of the same packet are routed in-order, following the same path from source to destination. Thus, ATM cells are analogous to wormhole flits. Next, we consider flow control: wormhole routing uses credit flow control; for ATM, the situation is more complicated.

ATM needs a flow control mechanism; rate-based and credit-based schemes have been considered for that purpose. The network may lose cells under rate protocols, while no cell loss occurs with credits. Recently, the ATM Forum adopted rate flow control [Ohsa95], because the credit mechanism was considered to require too much buffer space in wide area networks. However, credit flow control performs better, and is thus preferable in local area networks, where its buffer requirements are modest. The battle between rate and credit based flow control may be far from over, and mixed schemes may eventually appear [RaNe95]. A commercial ATM switch that implements credit flow control is DEC's GIGAswitch/ATM [Dec94]. Under credit flow control, data are only transmitted if the transmitter knows that there exists available buffer space at the receiver; thus, data are never dropped. The receiver buffer space and the credits for its use can be dedicated to one packet at a time, or to different packets, or per VC, or it can be shared among VCs, indiscriminately or in a controlled fashion. These methods, which have to do with the notion of "lanes", are reviewed in section 2.

In this paper, we evaluate the lane and buffer memory requirements of credit-flow-controlled ATM and of wormhole routing in local area and multiprocessor networks. We show that in ATM a small buffer size is enough in the case of uniform traffic, and a few lanes suffice in the presence of bursts, while wormhole requires significantly larger buffer space and more lanes for similar performance, especially when large packets (bursts) exist. In the case of hot-spot traffic, a number of ATM lanes which is slightly larger than the number of hot spots is enough for the non-hot-spot traffic to stay virtually unaffected by the presence of the hot spots; in wormhole, even a quite larger number of lanes cannot offer satisfactory isolation.

The performance of wormhole routing was studied in [Dally90], but that study was restricted to a single packet size and to uniformly-destined traffic. For ATM, efficient credit-based flow control schemes have been studied for long-distance links [KuBC94] [OzSV95].

This paper differs from the above studies in the following points. For wormhole routing, we extend the study of [Dally90] by simulating the effect of various packet sizes and of hot-spots. For ATM, we study credit flow control with dynamic buffer sharing for *short links,* like those typically found in LANs and multiprocessor networks, rather than the long links of WANs studied elsewhere; in our case, the throughput-roundtrip-time product of the links is less than one cell time. Also, our study of hot-spot traffic, of burst-size dependence, and our network topology and size differentiate ours from previous studies. In addition, we perform the same simulations for wormhole and ATM, and we compare and contrast the results.

Our simulation models are detailed and realistic; they operate at clock cycle granularity, and they track closely two real chips. We have implemented an ATM-like switch, the Telegraphos II chip, that transfers fixed-size cells and includes credit flow control with dedicated per-VC buffers [KVES95]. In that chip, credit flow control occupies 13 % of the (core) silicon area, thus showing that this flow control method has low cost, considering the advantages that it offers. We are currently developing *ATLAS I,* a single-chip ATM switch [KaSV96], that implements optional credit-based flow control of the same kind as the one simulated in this paper. This single-chip CMOS 16×16 ATM switch, which is being developed within the "ASICCOM" project[1], features 20 Gigabits/s aggregate I/O throughput, sub-microsecond cut-through latency, IEEE Standard 1355 "HIC/HS" links, on-chip shared buffering (256 cells), on-chip translation (routing) table, three service classes / priority levels, multicasting, and accelerated load monitoring hardware assistance.

The rest of this paper is organized as follows. Section 2 reviews credit-based flow control, and defines the credit protocol assumed for ATM in this paper. Section 3 points out the similarities and the differences between credit-flow-controlled ATM and wormhole routing. Section 4 presents the assumptions and models used in our simulations; the detailed simulation models are given in an Appendix. Finally, section 5 presents and interprets the results of the simulation experiments, followed by our conclusions.

---

[1] "Atm Switch for Integrated Communication, COmputation, and Monitoring" – the "Gigabit Switching" task project of the European Union *ACTS* (Advanced Communication Technologies and Services) Programme. The ASICCOM Consortium consists of industrial partners (INTRACOM, Greece; SGS THOMSON, France and Italy; BULL, France), telecom operators (TELENOR, Norway; TELEFONICA, Spain), and research institutes (FORTH, Greece; SINTEF, Norway; Politecnico di Milano, Italy; NCSR Democritos, Greece).

## 2. Credit-Based Flow Control

This section reviews credit-based flow control. Credit flow control, also called back-pressure, or hop-by-hop window, or virtual channel (circuit) flow control, dates back to Tymnet [Rind77], GMDNET [Gies78], and Transpac [SiDa79], in packet switched networks; see also the survey [GeKl80]. Under credit flow control, a cell or a flit is only transmitted to the downstream neighbor if the transmitter knows that buffer space is available for it at the receiver. This is implemented as illustrated in figure 1, using a count of available receiver buffer space. This "credit count" is decremented every time a cell or flit departs and is incremented when a credit token is received; credits are sent back every time a unit of buffer space becomes available.

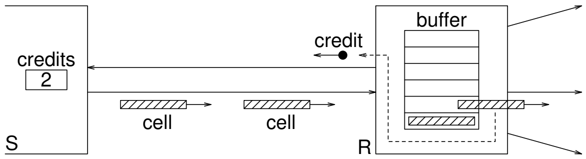


*Figure 1: Credit-based flow control*

In this way, cells are never dropped and (expensive) communication resources are never wasted by transmitting information that will later be discarded. Back-pressure allows stable network operation at full throughput utilization, so low-priority traffic can "fill in" the entire available capacity. Also, credit flow control allows each transmission to start at full-speed; throttling will only occur when necessary and to the degree that is necessary. By contrast, rate flow control is an end-to-end protocol with delayed response to network load fluctuations.

For a given link peak throughput, the receiver buffer space that is necessary and sufficient is equal to this *throughput* times the *round-trip time* (RTT) of data and credits. This buffer space, which is reminiscent of the braking or accelerating distance of a vehicle, is equal to the maximum number of cells and credits that can be in transit over the link. When the link operates at full speed and then *R*'s output suddenly stops, all these cells must be held in the given buffer space. Conversely, when *R*'s output suddenly starts, *R* must have that many cells available to sustain full speed operation until new cells have had the time to arrive from *S*.

### *2.1 Single-Lane Back-Pressure*

In single-lane wormhole routing, the flits of each packet are never interleaved with flits of other packets on a link, as illustrated in figure 2 (H and T markers identify head and tail flits of packets). This creates a behavior similar to *head-of-line blocking* in input queueing [KaHM87], leading to performance degradation [Dally90]. If a packet is stopped because of contention and not all of its flits fit in the buffer, then the packet holds its incoming link, thus blocking all other packets from using it. An analogy in every-day life is single-lane streets (where the name came from):

cars waiting to turn left block cars headed straight. There are commercial wormhole switches that use a single-lane architecture, e.g. C104 [Inmo93], Myrinet [Bode95].

In ATM, cells belonging to different VCs (A or B in figure 2) are interleaved on the network links, so the exact equivalent of single-lane wormhole does not exist. Yet, in the simplistic credit flow control scheme described above, hot-spot traffic creates a similar effect. Cells going to hot-spot (congested) destinations move slower and thus accumulate in the buffers, hence blocking cells destined elsewhere from proceeding at full speed; the result is link underutilization and excess delays.

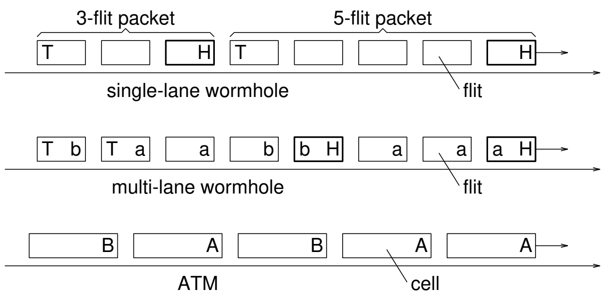


*Figure 2: Wormhole flits or ATM cells on a link*

## *2.2 Multi-Lane Credit Flow Control*

Multi-lane back-pressure solves this performance problem by selectively allowing unblocked traffic to proceed, while preventing congested connections from occupying all available buffer space thus delaying all other traffic as well. Multi-lane (virtual-channel) wormhole routing was introduced in [DaSe87] in order to prevent deadlocks, and in [Dally90] in order to improve performance. The latter version works as follows: a number $L$ of lanes exist on a link. When a new packet arrives at the link entrance, it waits until a free lane exists, and then it acquires such a lane. Flits of packets that have acquired a lane can be transmitted, in any interleaving fashion, on the link; each flit carries the lane identifier of the packet to which it belongs, as illustrated with the a and b markers in figure 2. The receiver buffer is partitioned into separate space for each lane; each credit is for one particular lane; a flit can only be transmitted if it acquires a credit for its lane. After the last flit of a packet is transmitted, its lane is freed. Multi-lane wormhole routing has been implemented in hardware in Torus [DaSe87] and iWarp [Bork90].

In ATM, the simplest way to introduce VC credit flow control (multi-lane back-pressure) is to dedicate separate buffer space to each connection (VC), and hence to use a separate credit counter for each VC; the buffer space per VC must be equal to the VC's peak throughput times the round-trip-time. This *static* buffer allocation ensures complete independence of each VC from all others, but it is also quite wasteful in terms of memory space; however, we can do better than that. The first step is to recognize that all VCs that have a common destination can be flow-controlled together, because their cells need never overtake each other (their common destination is assumed to be a single-threaded device, i.e. one that does not process cells of one VC preferentially over those of another). At each switch in our networks, we treat all such VCs that have a common destination as a single *flow group,* regardless of the different sources that they may have. The static buffer allocation scheme dedicates separate buffer space to each different flow group, i.e. to each destination.

The required buffer space can be further significantly reduced by *dynamically sharing* the receiver's buffer among the flow groups. Such dynamic buffer allocation schemes have been developed for wide area networks [OzSV95] [KuBC94], where the amount of memory needed and the expected savings justify the use of protocols with some complexity. Our interest in this paper is in local area and multiprocessor networks, using single-chip switches, so we consider a simpler sharing protocol. Figure 3 illustrates the credit flow control protocol used for ATM in this paper. For each link, the downstream switch (statically) allocates a buffer pool of size $B$. In order for this buffer not to overflow, the upstream switch maintains a credit count (per output port), *poolCr*, corresponding to the available buffer pool space at the receiver.

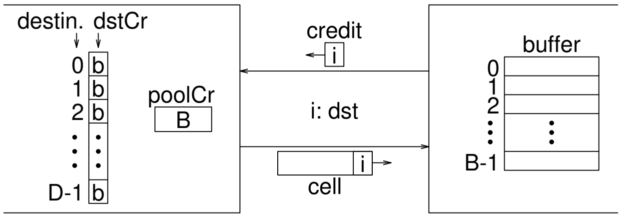


*Figure 3: Multi-lane ATM: D destinations sharing L lanes/link with b cells/lane (B=L·b)*

This buffer is shared among cells according to their flow group (destination), rather than according to their packet as in wormhole; this has important positive implications on performance. Separate credit counts, *dstCr*, are maintained for each destination[2], initialized to the value $b$ of the desired peak throughput times the round-trip-delay. Here, we are interested in short links, where this product is on the order of one ATM cell or less (with peak throughput per destination equal to the link capacity); our simulator assumes this product to be less than 1 cell. Hence, the best value is $b = 1$, which also simplifies implementation; however, we also study values greater than 1. Since at most $b$ cells of a given destination are allowed in the buffer at once, cells of at least $L = B/b$ different destinations can always fit in the buffer at the same time. We call $L$ the *number of lanes.* When $b = 1$, $L$ is precisely the maximum number of distinct destinations for which cells can fit in the buffer simultaneously. Notice that for $b > 1$, more than $L$ distinct destinations may appear in the buffer at once.

[2] all simulations in this paper assume the existence of one permanent VC from each network source to each network destination.

A cell destined to $i$ can depart if and only if $dstCr[i] > 0$ and $poolCr > 0$. When such a cell departs, $dstCr[i]$ and $poolCr$ are decremented by 1, each. When that cell leaves the down-stream switch, a credit carrying its destination ID, $i$, is sent up-stream. When credit $i$ is received, $dstCr[i]$ and $poolCr$ are incremented by 1, each. This back-pressure mechanism avoids the problem of single-lane (section 2.1): bursts of identically destined cells are prevented from uselessly monopolizing the receiver buffer space. In the Telegraphos II ATM-like switch chip that we have implemented [KVES95], $b = 1$ and $D = L = B$. In the more advanced ATM switch chip that we are currently developing, ATLAS I, $b = 1$ and $D$ is much larger than $L$.

## 3. Comparing and Contrasting ATM and Wormhole

This section outlines the similarities and differences between credit-flow-controlled ATM and wormhole routing, with particular emphasis on their relation to performance and cost. As discussed earlier, ATM cells are analogous to wormhole flits, and multi-lane credit flow control is similar in the two architectures (even though it is standard in wormhole while for ATM it is an extension to the standard). Each packet injected into the network results in a *burst of cells or flits* that all follow a common route, traveling back-to-back, except when they happen to get interleaved with other similar bursts of other lanes or connections. Below, we compare the size of cells and flits, we discuss the difference in the way lanes are used, we compare the hardware complexity, and we discuss static versus adaptive routing.

### *3.1 Cell Size versus Flit Size*

Traditionally, the size of wormhole flits was a few bytes, while the ATM cell size is 48 bytes plus a 5-byte header. The factors in favor of small flits are reducing the buffer requirements and reducing the fragmentation overhead for small packets. However, as technology evolves, the flit size is pushed to higher values. First, concerning buffer size, this should not be reduced below one throughput-times-RTT per lane (section 2). So, there is no gain to have flits smaller than the throughput-RTT product, because then multi-flit-per-lane buffer space is needed. With time, throughputs increase; as link lengths also increase with the gradual transformation of multiprocessors into networks of workstations (NOW), the throughput-RTT product grows larger. For multi-Gigabit/second links of a few tens of meters, this product is as high as one or a few ATM cell sizes, so smaller flits are no longer useful. Second, the control circuits of a switch must operate at the flit rate (especially if back-to-back single-flit packets are to be handled correctly). For flit sizes much smaller than the ATM cell size, and for multi-Gb/s rates, this control clock frequency becomes uncomfortably high; thus, larger flits become desirable. Third, the link throughput overhead for transmitting credits (and lane ID's in flits) is reduced when the flit size increases. And fourth, the minimum packet size tends to increase, rarely being very much smaller than one ATM cell any more. In 64-bit processors, a packet containing as little as one address and one data word is already one third of an ATM cell in size; in cache-coherent multiprocessors, small packets frequently contain an entire cache lines, thus being larger than one ATM cell. It follows that, as technology progresses, all the factors that have led to wormhole flits being much smaller than ATM cells are gradually disappearing.

### *3.2 Use of Lanes*

Credit-flow-controlled ATM and wormhole routing use lanes (section 2.2) in very different ways, and this is the reason for their performance differences seen in section 5. In networks with one or few lanes, the most important disadvantage of wormhole routing is that it dedicates each lane to a single packet for the entire duration of that packet's transmission. When the packet's progress is blocked because of contention, if the entire packet does not fit in one buffer, its tail blocks the lane that it has acquired on one or more links. When few lanes exist, all of them can get blocked in this way, causing the link to stay idle. By contrast, ATM allows a lane to be used by any VC after the departure of the cell (not packet) that previously occupied it.

In networks with many lanes, the most important negative characteristic of wormhole is that it allows packets that go to the same destination to occupy multiple lanes per link at the same time. Such similarly-destined packets may originate from the same or from different sources. No advantage is gained by doing this, since all these packets will get blocked if one of them does, while, on the other hand, all these lanes remain unavailable for packets traveling to other (less congested) destinations. In ATM, lane usage is governed by destination (flow group): identically-destined cells are not allowed to consume more than one lane, regardless of where they came from and which packet they belong to.

### *3.3 Hardware Complexity*

The hardware complexities of an ATM switch with multi-lane credit flow control and a multi-lane wormhole switch, with the same number of lanes, same buffer size, and flit size equal to cell size, are comparable. The control memories are larger in the ATM case, but their number and nature is about the same in the two architectures. We will make a comparison of these complexities for the case of one cell/flit per lane ($b = 1$, $L = B$), which is the simplest to implement because no per-lane cell/flit order information has to be maintained. First, both architectures need a routing table (a RAM) with a similar number of entries for a same network; in the ATM case, this RAM also serves as a VP/VC translation table, so each entry must be wider than the corresponding wormhole entry. Second, both architectures need a credit table (RAM) per output link; for $b = 1$, this is a 1-bit-wide memory in both cases, but the number of entries is $L$ in wormhole and $D$ in ATM ($D$ is usually quite larger than $L$). In addition, ATM needs a pool credit counter per output, while wormhole needs a lane-status (assigned/free) vector per output. Third, both architectures need appropriate structures to keep track of the status of each waiting cell or flit, and to search them in desired ways. One way to implement them is to use a content-addressable memory (CAM) with one entry per cell/flit in the buffer memory. In ATM, each CAM entry contains the cell's credit status, the output link, and the destination ID (part of the output VP/VC); on credit arrival, we search for an entry without credit and with the proper link and destination ID; on cell departure, we search for an entry with credit and with the proper link ID. In wormhole, each CAM entry contains again the cell's credit status and its output link, plus the outgoing lane ID (this field is narrower than the destination ID in ATM); the searches are similar to the ATM case, plus a search for flits waiting for lane assignment when a lane gets freed.

In conclusion, the nature of the control memories is similar in the two architectures, and their size a somehow larger in ATM. However, as will be seen in section 5, ATM needs much less buffer memory than wormhole to achieve comparable performance, while in some cases (hot-spot traffic) ATM performs better regardless of how much more buffer memory wormhole may have. Since the (data) buffer memory is larger (wider) than the control memories, it follows that ATM switches cost less than wormhole switches for a given performance level under uniform traffic, while ATM switches cost less *and* perform better than wormhole switches under hot-spot traffic.

### *3.4 Adaptive Routing*

One aspect of wormhole which is not studied in this paper is dynamic (adaptive) routing: if the network topology includes multiple paths from a given source to a given destination, then different packets may follow such different paths (at the risk of out-of-order delivery). We could not study this property of wormhole because ATM does not allow it, since ATM guarantees in-order delivery. Adaptive routing can improve the performance of wormhole networks in some cases. However, the inefficient use of lanes by wormhole that was discussed above applies to the case of adaptive routing as well: lanes are still dedicated to single packets at a time, and multiple lanes, *in multiple paths* now, can be occupied by slow-moving hot-spot packets, thus becoming unavailable for packets going to uncongested destinations.

## 4. Simulation Models

Two models were developed for the ATM and wormhole switches described in section 2.2, in order to compare them through simulation. The models are quite realistic and detailed, operating at the granularity of individual clock cycles. The ATM model is based on the way in which the implemented Telegraphos II switch chip operates [KVES95], with the appropriate extensions for the more general lane sharing protocol ($D \geq L$ rather than $D = L$); it also takes into consideration our current ATLAS I switch chip development, although there are differences between the chip and the way the model is formulated. Then, the wormhole switch model was developed based on the ATM model, introducing only the necessary changes. With this approach, all non-essential characteristics (e.g. data buffer cell/flit staggering, credit exchange, etc) are made the same in the two models, and the two network architectures are evaluated under comparable circumstances.

The detailed models, describing all actions in the route of a cell/flit through a switch, are given in an Appendix. Below, we summarize their important features that are needed in order to understand the results of the simulations.

- *Time Units.* All results are reported in units of *cell time.* This is the time taken to inject an ATM cell on a link. Time in the simulation flows in finer units: *clock cycles.* In our simulations, the cell time was defined as *fourteen (14)* clock cycles, reflecting an assumed processing of cells at the rate of 4 bytes per cycle ($53/4 = 13.25$).
- *Flit Size.* The payload of a wormhole flit is assumed to be the same as the payload of an ATM cell; so, a packet is fragmented into the same number of cells or flits. The header of wormhole flits is smaller than the ATM cell header (5 bytes), especially for non-head flits, since it only contains a lane ID. To account for this difference, our simulations assume that the time taken to inject a flit on a link is *thirteen (13)* clock cycles, i.e. 1 cycle less than the cell time. This also accounts for the wormhole credits being shorter than the ATM credits.
- *Link Capacity.* All network links, including source and destination, have a capacity of 1 cell per cell time in ATM, or 14/13 (=1.077) flits per cell time in wormhole. These figures include the transmission of credits for the reverse-going link, i.e. the simulations assume that credits are transmitted at zero cost (they do not interfere with normal cell/flit transmission).
- *Link Latency.* All links of the simulated networks, including source and destination links, have a propagation delay of 0.07 cell times (1 clock cycle). This choice was made in order to simulate a multiprocessor interconnection network where all processors are in close physical proximity.
- *Cut-Through Delay.* The switches modeled feature cut-through capability (as in [KVES95] [KaVE95]), in order to reduce the latency when the network load is low. The cut-through delay of the switch is 0.3 cell times (4 clock cycles) for ATM and 0.36 cell times (5 clock cycles) for wormhole, as detailed in the appendix; in other words, the head of a cell or flit arriving at an idle switch will depart after exactly 4 or 5 clock cycles, respectively.
- *Round-Trip Time (RTT).* The cell/flit-credit RTT is 0.57 cell times (8 clock cycles) in ATM and 0.64 cell times (9 clock cycles) in wormhole.
- *Network Topology.* We simulated banyan (butterfly) networks, similar to [Dally90]. Figure 4 shows such a banyan topology for a small size network.
- *Simulation Length.* Each simulation run for the 64×64 network lasts for a number of cycles such that 13,000 packets (i.e. a few hundred thousand cells/flits) traverse the network. We base our measurements on the last 10,000 of these packets, considering that the network reaches a stable state after the first 3,000 packets. Using confidence interval techniques, we have verified in typical cases that our result accuracy is 0.01 with 0.95 confidence.

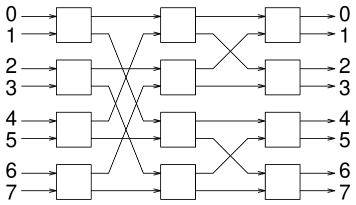


*Figure 4: A 3-stage 8×8 banyan network*

- *Traffic Sources.* There is one producer of cells/flits for every incoming link to the network. Producers produce fixed sized packets, each composed of a fixed number of cells/flits, with exponential interarrival time. Thus, the input is Poisson at the packet level, but bursty at the cell/flit level, capturing the important characteristics of realistic network traffic. The Poisson model is used as a standard reference traffic pattern, because our objective is a qualitative comparison of the architectures; the results show that the difference in performance is so large that the precise traffic model becomes less important. The choice of fixed size packets was made so that we can compare our results to those of [Dally90]. Each generated packet is targeted to a destination that is chosen at random. In all but the hot-spot experiments, packet destinations are chosen uniformly.
- *Input Load.* This is the average rate of packet generation in each traffic source, measured in cells or flits per cell time (i.e. same units as the link capacity).
- *Source Queue Organization.* Each producer has one separate queue per destination. In the ATM network, cells (destinations) that acquire credits depart from these queues in chronological order of generation. In the wormhole network, packets (of any destination, i.e. from any queue) acquire available lanes in chronological order of generation, with the added restriction that packets from a given queue (going to a given destination) are not allowed to acquire more than one lane at a time. This latter restriction is favorable to the wormhole architecture but is not a standard part of it; it was added because, in the hot-spot simulations, without it, the entire network was flooded by hot-spot packets, leading to catastrophic performance.
- *Traffic Sinks.* There is one consumer of cells/flits for each output link of the network. Sinks always consume cells/flits as soon as these arrive.
- *Shared Buffering.* The simulated switches use the shared buffering organization, like our Telegraphos II and ATLAS I chips do [KaVE95]. As shown in section 5.1, this performs better than the input queueing (single-lane) or input buffering (multi-lane) organizations that were simulated in [Dally90], while implementation cost is similar for the two organizations [KaVE95]. Note that the shared buffering organization that we simulate concerns the elimination of head-of-line blocking inside each lane; from the point of view of space allocation, static partitioning per incoming link is used (otherwise, credit management becomes too complex).

- *B: Buffer Memory Size.* As illustrated in figure 3, this parameter is the size of the buffer pool in each switch that is dedicated to each incoming link; it is measured in cells or in flits.
- *L: Number of Lanes.* This parameter was defined in section 2.2. For ATM, this parameter is the ratio of the buffer pool size $B$ to the number of credits per flow group $b$ (figure 3).
- *Saturation Throughput.* One measure of performance that we report, for uniformly destined traffic, is the average outgoing link throughput when the input load exceeds 1. We measured this under input load equal to 2.0; see section A.3 for a more detailed discussion. We measure throughput in cells or flits per cell time (same units as input load and link capacity).
- *Delay.* Delay is our other measure of performance. We report total cell/flit delay, i.e. the sum of the queueing delay in the traffic source plus the network delay, for individual cells or flits −not for entire packets. The source queueing delay is counted from the clock cycle when the packet is generated (all cells/flits of a packet are generated simultaneously) to the cycle when the head of the first cell/flit of the packet leaves the source. The network delay is counted from the cycle when the head of a cell/flit enters the network to the cycle when that (entire) cell/flit exits from the network, averaged over all cells/flits that traverse the network. We also measured the source and the network delays separately, as well as entire packet delays, but we consider them less interesting, so we do not report them.

## 5. Measurements and Interpretation

The models have been coded to obtain a simulator. We used CSIM [CSIM17], with C as the implementation language of the simulator. The runs were executed on SPARC and ALPHA workstations. We study saturation throughput and delay as functions of the number of lanes for various buffer sizes (and for various input loads, in the case of delay). We begin with uniformly destined traffic and fixed packet size. Then, we study the effect that the packet (burst) size has. After that, we add hot-spot traffic to the network, and we examine how that affects the non-hot-spot traffic. All previous experiments are for a 6-stage (banyan) network made of 2×2 switches, i.e. a 64×64 network. Finally, we study how performance depends on network and switch size.

### *5.1 Uniformly-Destined Fixed-Size Packets*

In these experiments on the 64×64 network, packets have uniformly distributed destinations, and the packet size is fixed at 20 cells or flits. Figure 5 plots saturation throughput. Clearly, ATM performs much better than wormhole for a given buffer size and number of lanes, for the reasons explained in section 3.2. The ATM throughput approaches the link capacity with buffer sizes as small as 4 or 8 cells; by contrast, wormhole requires 16-flit buffers or larger in order to provide similar throughput. Another observation is that the ATM performance is good almost independent of the number of lanes, while wormhole needs multiple lanes in order to achieve good saturation throughput. As we will see later, the number of lanes makes a difference in ATM when the delay under bursty or hot-spot traffic is considered.

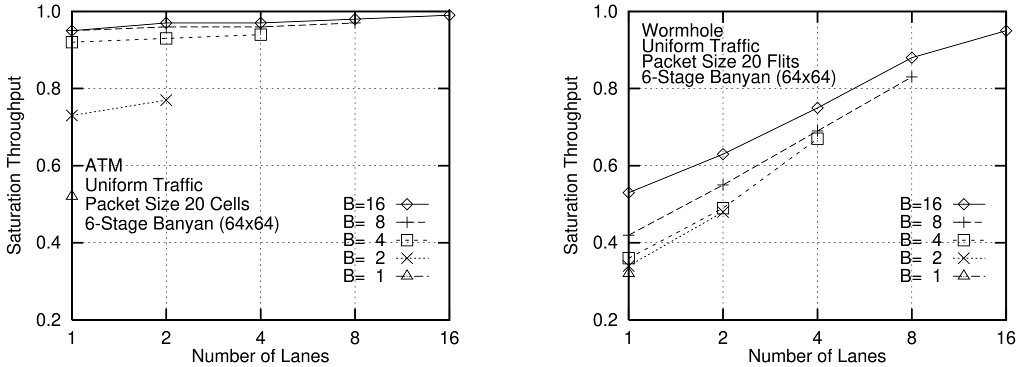


*Figure 5: Saturation throughput versus lanes and buffer size: ATM (left), wormhole (right)*

The saturation throughput curve of figure 5 for wormhole routing and B=16 corresponds to the similar curve measured in [Dally90] figure 8, n=6. For the comparison to be valid, our throughput values must be multiplied by 13/14 to account for our link capacity being 14/13 for wormhole (section 4); after this scaling, we observe the following. For single-lane wormhole, [Dally90] simulated input queueing –a blocked packet tail in a buffer also blocks the head of the next packet which is behind it in the same buffer– while we simulated shared buffering –no such blocking occurs; the result is that [Dally90] measured a saturation throughput of 0.34, while our (scaled) value is 0.49, i.e. quite better. For multi-lane wormhole, [Dally90] simulated input buffering (his figure 6(B)), again with head-of-line blocking inside each lane; the results are ([Dally90] versus our (scaled) value): 0.49 versus 0.58 (L=2), 0.64 versus 0.70 (L=4), 0.78 versus 0.81 (L=8), and 0.85 versus 0.87 (L=16). This verifies the known property that shared buffering, which suffers no head-of-line blocking, performs better than input queueing/buffering, while, as we have shown in [KaVE95], their implementation cost is similar.

Figure 6 plots delay for two "low" input loads, 0.2 and 0.4, and for one "high" load value, 0.8; note that three of the vertical scales are quite different from the other three. As seen, when the input load is much lower than the saturation throughput, the delay is quite low – in the range of 10 to 20 cell times[3]. Delay grows very quickly when the load approaches the saturation point. Since ATM features higher saturation points than wormhole, ATM gives lower delays, in general.

---

[3] an interesting second-order effect can be observed for low load, visible especially when comparing the ATM plots for loads 0.4 and 0.8: under low load (0.4), delay *increases* with increasing buffer space, for fixed number of lanes; this is opposite from the expected behavior. The expected behavior is seen for high load (0.8): delay *decreases* with increasing buffer space, probably because we then get further away from saturation. The reverse behavior under low load should probably be attributed to the way we measure delay: source delay is only for the first cell of a packet, while network delay is the average of all cells. When buffer space increases with a fixed number of lanes, network delay probably increases because more of the packet's cells quickly enter the network (the lane that they follow is now wider); then, they have to wait inside the network (where their delay is counted), rather than outside the network (where their delay was not counted).

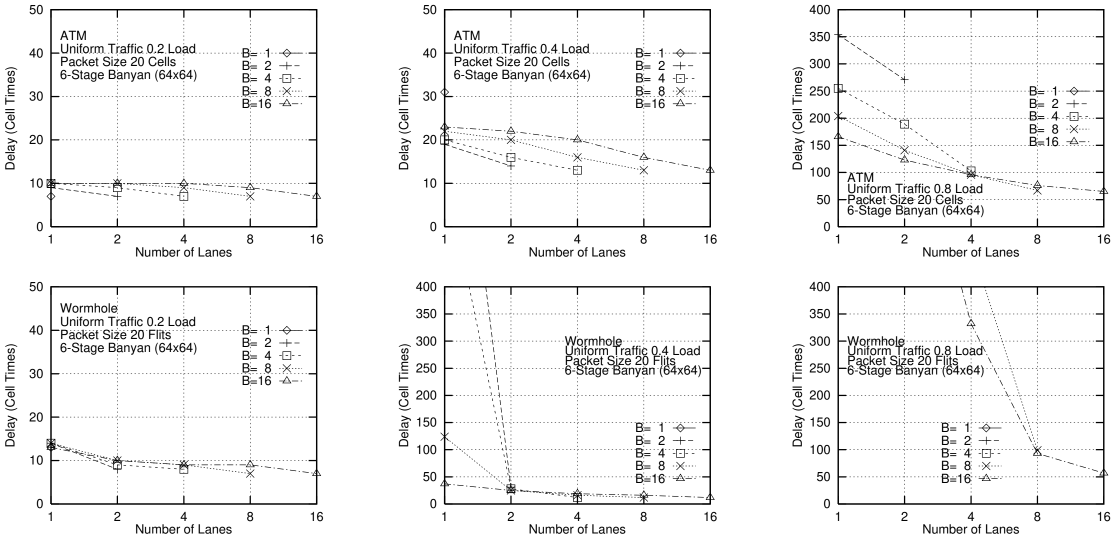


*Figure 6: Delay versus lanes and buffer size, for various input loads: input load 0.2 (left), 0.4 (middle), 0.8 (right); ATM (top), wormhole (bottom)*

### *5.2 Burst (Packet) Size Dependence of Performance*

Burst size is an important parameter in network performance. Multiprocessor, multimedia, and other gigabit LAN sources often generate bursty traffic. Consider, for example, file servers or remote-memory pagers shipping Kilobytes or Megabytes of data, all at once. Consider also the sizes of the messages exchanged in the parallel scientific applications studied in [CHKM93]: 8 Kilobytes or larger in half of the applications, 512 Kilobytes in one application.

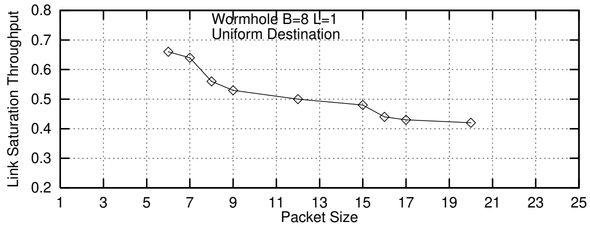


*Figure 7: Saturation throughput versus packet (burst) size; wormhole, B=8, L=1*

In wormhole networks, long packets (bursts), spanning (occupying) several switches in a network route, block many other packets. To verify this hypothesis, we measured the saturation throughput of our wormhole network for packet sizes that span a range from a little shorter than one buffer size to a little longer than two buffer sizes. Figure 7 shows the result: as the packet size increases, performance drops quite more sharply when this size crosses boundaries of integer multiples of the per-switch, per-lane buffer size (8 flits).

In the general case now, figure 8 plots saturation throughput for a few burst sizes in the range 0.5 to 8 Kilobytes (10, 20, 40, 160 cells or flits). As seen, throughput stays always high and

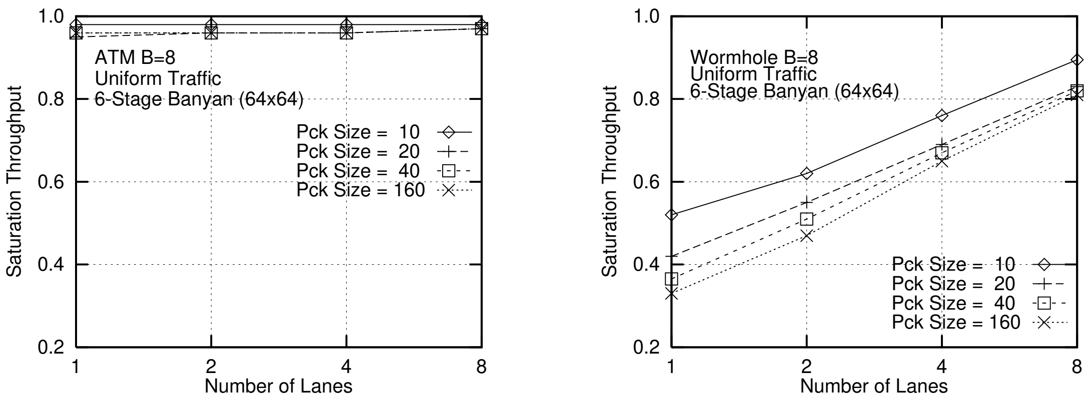


*Figure 8: Saturation throughput versus lanes for various packet (burst) sizes: ATM (left), wormhole (right); buffer size is 8 cells or flits*

is minimally affected in ATM, while in wormhole throughput decreases considerably with increasing burst size, as predicted above. As in section 5.1, the number of lanes is not important for throughput in ATM, while it is in wormhole.

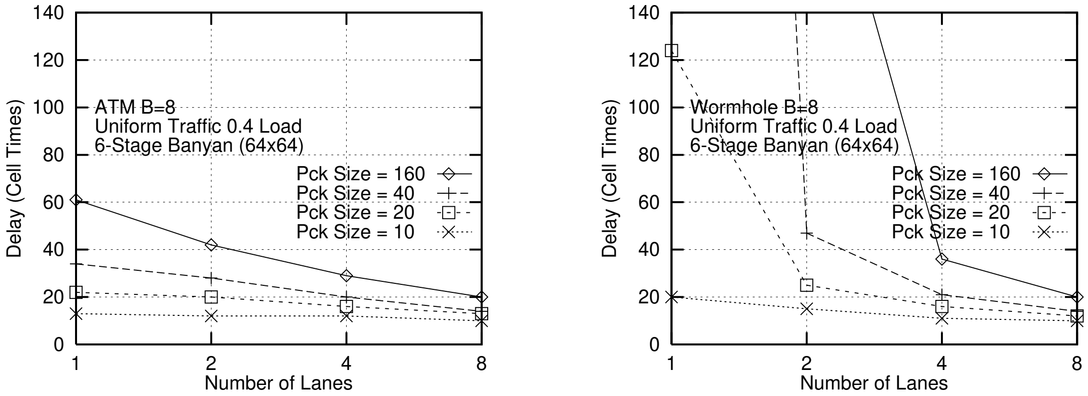


*Figure 9: Delay versus lanes for various packet (burst) sizes: ATM (left), wormhole (right); input load is 0.4 and buffer size is 8 cells or flits*

Figure 9 plots delay for the same burst sizes, when the input load is 0.4. Contrary to the case of throughput, ATM networks are sensitive to burst size in delay characteristics. Furthermore, the number of lanes makes a difference now: for bursty traffic, multi-lane ATM provides significantly shorter delay than single-lane ATM (by a factor of three for 8 Kilobyte bursts in this case). When the number of lanes is large, delay is low for all burst sizes, in ATM. For wormhole, burst size and number of lanes affect delay dramatically, as expected, given that these factors affect considerably the saturation throughput, and delay is very sensitive to how close the network is to saturation. In all cases, ATM networks offer better performance than wormhole ones.

Another study that we are conducting concerns *bimodal* traffic – a mixture of small and large packets, i.e. a network where short messages co-exist with large, bursty transfers. Under these circumstances, it is desirable that the short messages be minimally affected by the presence of the large bursts (more precisely, for a given total load, the "short" portion of the traffic should be minimally affected when the burstiness of the remaining traffic increases). In our experiments, half of the input load is due to packets of size 2 cells or flits, and the other half load is created by long packets; the size of the latter varies from 10 to 160 cells or flits (thus, short packets are 5 to 80 times more numerous than long packets). We measured delay under a (total) input load of 0.4 (0.2 short and 0.2 long packets), separately for the cells of short packets and for those of long packets. The buffer size, $B$, is fixed at 8 cells or flits per link.

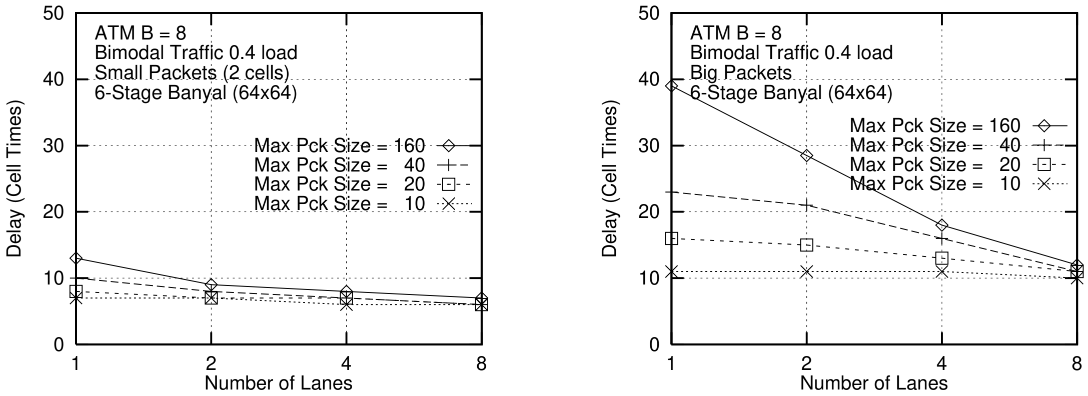


*Figure 10: Delay of cells in short (left) and long (right) packets in ATM when the size of the long packets varies; total input load is 0.4 and buffer size is 8 cells or flits*

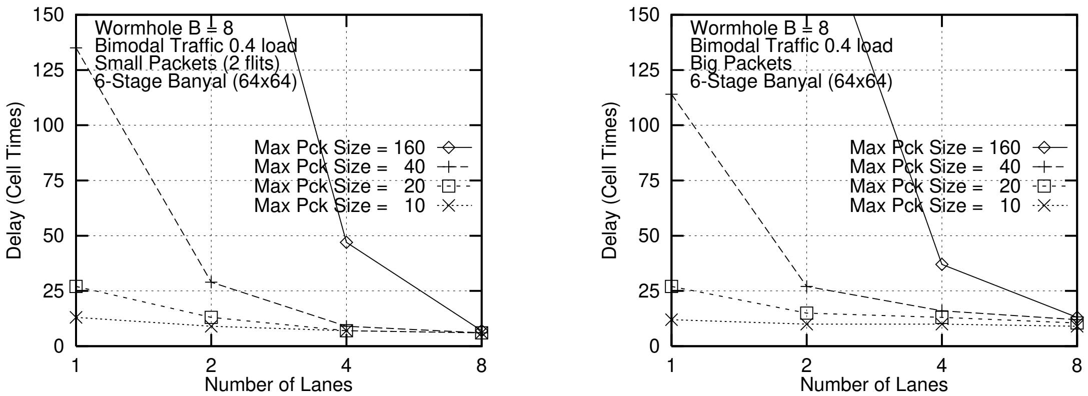


*Figure 11: Delay of cells in short (left) and long (right) packets in wormhole when the size of the long packets varies; total input load is 0.4 and buffer size is 8 cells or flits*

Figure 10 plots the delays for the case of ATM. The delay of the cells in the long packets (right) behaves in an analogous way to the case where the entire traffic consists of long packets (figure 9 left). Here, the delays are shorter, because long packets constitute only half of the traffic, but the dependences and the shape of the curves remains the same. On the other hand, the delay of the cells in short packets (left) is *always short*, close to the value when all packets are short in figure 9, and this delay is affected very little by the length, i.e. the burstiness, of the other half of the traffic. This is precisely the desirable behavior for the network, as discussed above.

Figure 11 shows the corresponding plots for the case of wormhole routing. The two plots – short-packet cell delay (left) and long-packet cell delay (right) – resemble each other, and they are both similar to figure 9 (right) except that the values here are somehow lower than there, since the average packet length is lower here. As seen in figure 11 (left), the delay of the cells in the short messages is greatly affected by the length, i.e. the burstiness, of the other traffic in the network, which is an undesirable property of wormhole routing, contrary to ATM.

### *5.3 Delay Increase due to Hot-Spot Traffic*

Network performance under *non-uniform destination* traffic is very important in modern LANs. The traffic in gigabit LANs, especially when running multiprocessor and multimedia applications, is quite different from traditional network traffic that results from statistical multiplexing of many independent sources [Kung92]. Among others, in this new environment the number of sources or destinations may be small and sources may be dependent –for example, server computers or synchronization points in parallel applications easily create hot spots in the network.

In these experiments, we superimpose traffic going to one, two, or four particular hot-spot destinations over a background traffic of uniformly-destined 0.2 input load; the hot-spot traffic is equivalent to a load of 1.0 for each hot-spot. Thus, hot-spot traffic is saturated, while non-hot-spot traffic is relatively ''light''. In other words, for the 64×64 banyan network that we simulated, the traffic load from source $i$ to destination $j$ is (0.2)/64 when $j$ is not a hot-spot destination, and (1.0)/64 when $j$ is a hot-spot. When multiple hot-spots exist, we located them as ''far'' from each other as possible: $j = 0$ and $j = 32$ for two hot-spots, and 0, 16, 32, 48 for four hot-spots. Packet (burst) size is fixed at 20 cells or flits.

In an environment with hot-spots, the interesting measurement is the delay of network traffic directed towards non-hot-spot destinations. Traffic targeted to hot-spot destinations is bound to have high delay, due to output contention. In an ideal network, non-hot-spot traffic should remain unaffected by the possible existence of hot-spots. Figure 12 shows the average non-hot-spot delay, as a function of the number of lanes, when zero, one, two, or four hot-spot destinations exist. Then, figure 13 further analyzes the non-hot-spot delay for the case of one hot-spot destination, in order to see how this delay varies for the different destinations, as a function of the *distance* from the hot spot.

For ATM networks, figure 12 shows the following very clear behavior. The delay of non-hot-spot traffic is the same as in the no-hot-spot case (zero hot spots) when $L \geq 2$ in the one-hot-

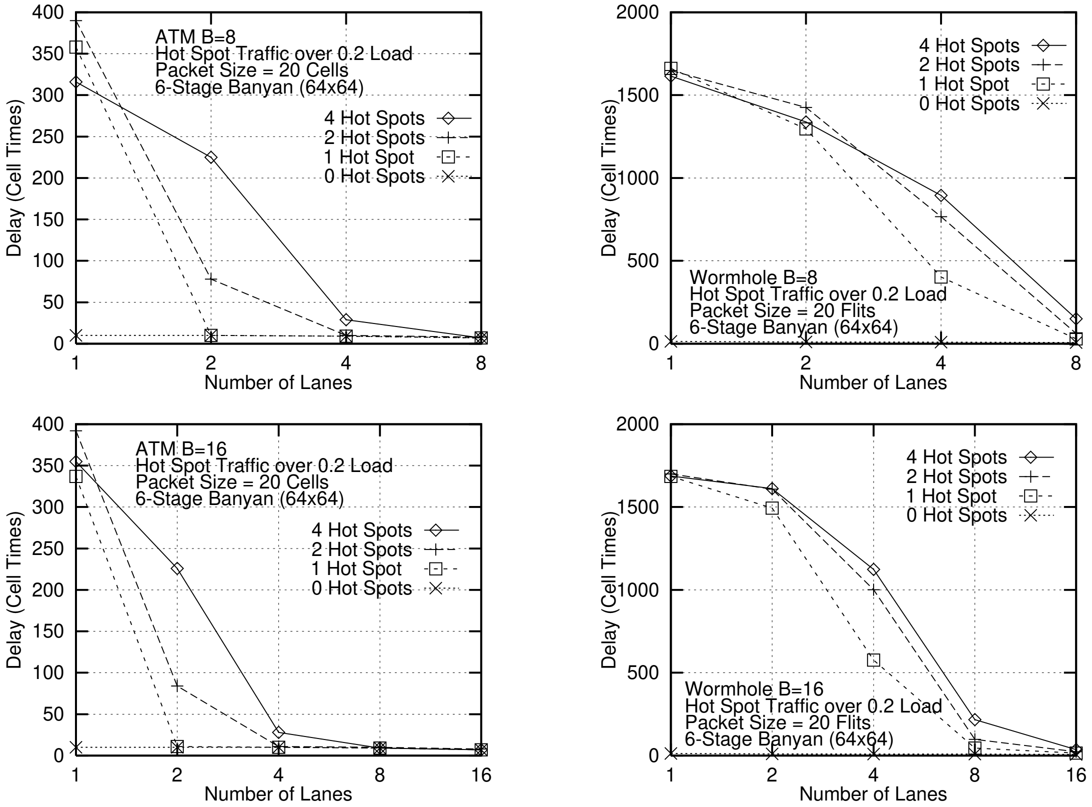


*Figure 12: Delay of non-hot-spot traffic in the presence of 0, 1, 2, or 4 hot-spot destinations; ATM (left), wormhole (right); non-hot-spot load is 0.2, buffer space is 8 (top), 16 (bottom) cells or flits*

spot case, when $L \geq 4$ in the two-hot-spot case, and when $L \geq 8$ in the four-hot-spot case. In other words, ATM exhibits the *ideal* behavior when *the number of hot spots is less than the number of lanes.* This good behavior is a direct consequence of the way lanes are used in ATM (section 3.2): all cells going to one destination (e.g. hot-spot) are allowed to occupy at most one lane of buffer space, regardless of where they came from and which packet they belong to. Thus, as long as there are more lanes than hot spots, there will always remain at least one lane that is not blocked by hot-spot traffic; non-hot-spot cells can use that lane to overtake the congested traffic, suffering *no additional delay* due to this traffic.

Contrary to ATM, non-hot-spot traffic in wormhole networks suffers long delays even when the number of lanes is considerably higher than the number of hot spots. This is seen in figure 12 (notice that the vertical scale is 5 times larger than for ATM)[4], and further in figure 13. Clearly, it is due to the fact that wormhole networks allow multiple packets with the same destination to

[4] interestingly, in wormhole networks, delay increases as the number of buffers per lane increases. This effect is due to the way we measure delay, as explained in the footnote of section 5.1: as the buffers increase, the number of flits entering the network increases; thus, the time a flit spends in the network increases, although the flit enters the network earlier.

occupy multiple lanes at the same time; thus, in our case, all lanes are easily blocked by (congested) hot-spot packets, dramatically delaying all other traffic as well.

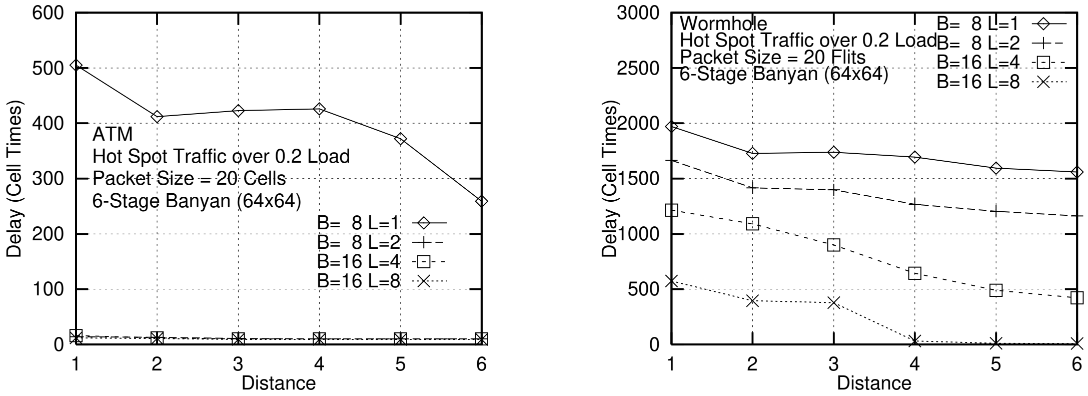


*Figure 13: Delay versus distance from the (single) hot spot: ATM (left), wormhole (right)*

It is quite interesting to understand the effect of hot-spot traffic on cells directed to *nearby* destinations. Intuitively, it is expected that destinations close to the hot-spot are affected more than destinations far from it. This is illustrated in Figure 13, plotting average cell delay to a destination $D$ as a function of $D$'s distance from the (single) hot spot $H$. Distance is defined as follows: given any network source $S$, consider the paths from $S$ to $D$ and from $S$ to $H$; distance is the number of network stages where these two paths are separate. As seen from the figure, for networks that have few lanes, where the presence of a hot spot significantly affects the remaining traffic, cell delay is indeed affected by the above distance, decreasing as the destination gets further away from the hot spot.

However, figure 13 reveals one more phenomenon. In ATM networks with sufficient lanes for the given number of hot spots ($L \geq 2$), the non-hot-spot delay is uniformly low, *independent* of distance from the hot-spot. On the contrary, in wormhole networks, even when the lanes are many more than the hot spots (8 lanes and 1 hot spot), and thus the *average* non-hot-spot delay is low (45 cell times), the (few) destinations that are close to the hot spot suffer very high delay (400 to 600 cell times for the 7 destinations within a distance of 3 from the hot spot), while the rest of the destinations, which are far from the hot spot enjoy reasonable or low delays (32 cell times at distance 4, 11 at distance 5, and 9 cell times for the 32 destinations at distance 6). This demonstrates unfairness in the wormhole network, since the performance characteristics experienced by many destinations are affected by their location rather than by their traffic pattern.

### *5.4 Network and Switch Size Dependence*

All measurements above were for a 64×64 banyan network, made of 2×2 switches. Figure 14 studies how performance varies when the network size changes. Saturation throughput and delay are plotted for networks with 4 stages (16×16), 6 stages (64×64), and 8 stages (256×256) made of 2×2 switches. To make the plots more interesting (to spread the values further apart from each other), buffer size varies with the number of lanes, increasing from 1 cell or flit per link for single-lane networks to 16 cells or flits per link for 16-lane networks (the buffer space per link is equal to the number of lanes).

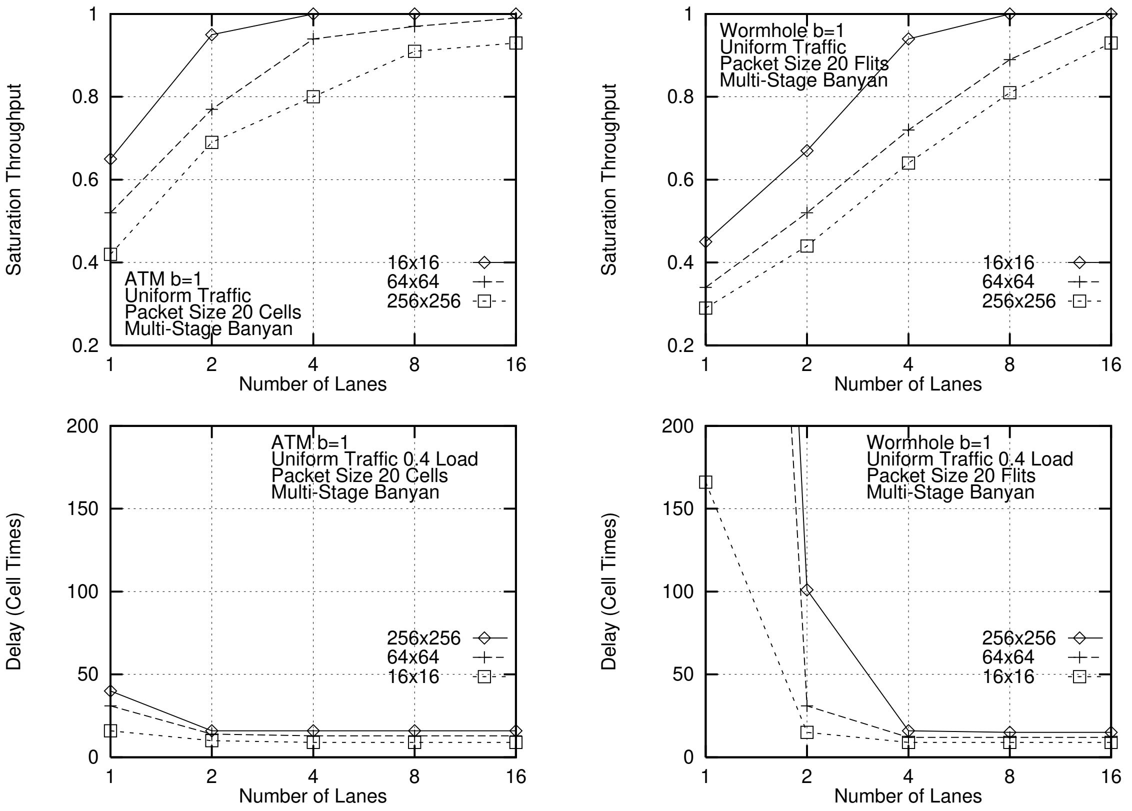


*Figure 14: Saturation throughput (top) and delay (bottom) versus lanes for various network sizes: ATM (left), wormhole (right); buffer size is 1 per lane, input load (bottom) is 0.4*

As we see, performance degrades with increasing network size, or, to put it differently, larger networks also require larger buffers per switch and more lanes in order for the same level of performance to be maintained.

The last experiment that we conducted was to vary the fan-in and fan-out of the switches that are used to construct the network. Figure 15 shows performance measurements for a 64×64 Banyan network made of 6 stages of 2×2 switches, or 3 stages of 4×4 switches, or 2 stages of 8×8 switches. As we see, performance improves with increasing switch fan-in and fan-out, or, to put it differently, performance improves when the network has less stages.

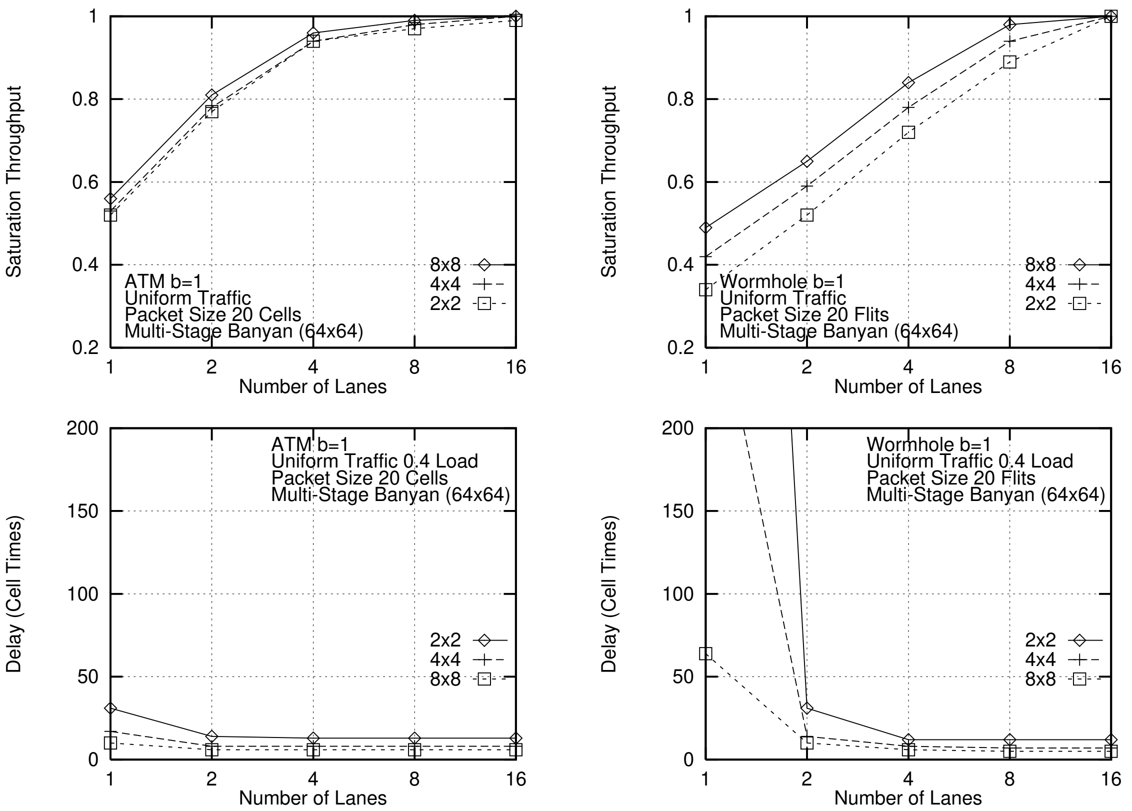


*Figure 15: Saturation throughput (top) and delay (bottom) for a 64×64 Banyan made of varying size switches: ATM (left), wormhole (right); buffer size is 1 per lane, input load (bottom) is 0.4*

## Conclusions

ATM extended with credit-based flow control has notable similarities to wormhole routing, a popular architecture for multiprocessor interconnection networks. They both route packets after segmenting them into fixed-size units, called cells in ATM or flits in wormhole. Their implementation complexity is comparable. And they both base flow control on credits and lanes, although they use lanes differently, which is the reason for their significant performance differences. Wormhole dedicates each lane to a single packet until the end of that packet's transmission; if the packet gets blocked, so does the lane. Also, wormhole allows packets with the same destination to simultaneously occupy multiple lanes on the same link; while no advantage is gained relative to using a single lane, the remaining traffic is deprived of the additional lanes.

We compared the performance of these two network architectures, using simulation. Our simulation models operate at the clock cycle granularity, and are quite realistic, being based on two real switches: we have implemented an ATM-like switch chip (i.e. fixed-size cells) with credit flow control and one dedicated lane per VC, and we are currently designing a gigabit-link ATM switch chip with optional credit FC and multiple lanes shared among the VCs.

Our simulation results show that, for a same buffer size and number of lanes, ATM performs consistently better than wormhole –by a large margin in several cases. For ATM, saturation throughput approaches link capacity with buffer sizes as small as 4 or 8 cells per link (for 64-port networks), and stays high regardless of the number of lanes. On the other hand, multi-lane ATM switches provide much lower delay than single-lane ones when bursty or hot-spot traffic is present in the network. When the number of lanes is higher than the number of hot-spot destinations, non-hot-spot traffic remains virtually unaffected by the presence of hot spots. For wormhole routing, much larger buffer sizes and many lanes are needed in order to achieve high saturation throughput. When saturation throughput is low, delay characteristics suffer as well, because any given operating point of the network is closer to saturation. Bursty and hot-spot traffic negatively affects wormhole performance to a much larger degree than it does for ATM.

We conclude that ATM, which was originally conceived for wide area networking and was subsequently also used in the local area, is an appropriate technology for multiprocessor interconnection networks as well, provided it is extended with credit-based flow control; perhaps, then, it deserves the title of universal network architecture.

## Acknowledgements

Most of this work was funded by the Commission of the European Union, through the "ASIC-COM" project of the ACTS Programme. George Stamoulis, Costas Courcoubetis, and Apostolos Traganitis gave us useful comments on an early draft of the paper. We wish to thank a lot all of the above.

## Appendix: Detailed Simulation Models

The models are described below for 2×2 switches, but the actual switch fan-in and fan-out is a parameter. The switches can be interconnected in various topologies to form large networks.

### *A.1 ATM Switch Model*

Figure 16 shows the model for the 2×2 ATM switch. There is an upper path modeling the data flow, and a lower path modeling the control flow. The model contains 5 servers and a number of queues that are described in detail below. Servers 1 and 5 are responsible for cell reception and transmission respectively, while the remaining 3 servers model the routing and flow control decisions in the switch. When an incoming cell arrives, server 1 separates the body of the cell from its header, and as the header follows the control path, the body is being stored in the shared data buffer of the switch. When a cell is being transmitted, server 5 obtains its header from a queue; then, it recovers the cell body associated with this header from the data buffer and "injects" the whole cell in the outgoing link. So, servers 1 and 5 can be considered to operate on the complete incoming cells, while servers 2, 3 and 4 operate only on cell headers. In more detail, the route of a cell through the switch is as follows.

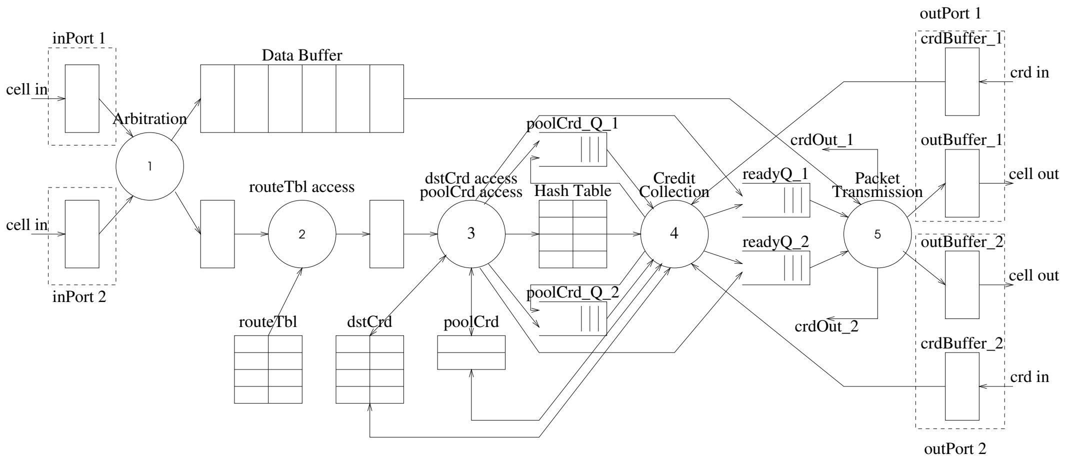


*Figure 16: ATM switch model*

- *Server 1:* the arriving cell is staggered appropriately, so that it can access the cell memory (data buffer) of the switch without conflicting with other incoming or outgoing cells; as in the Telegraphos II switch chip, only one cell read/write operation can be initiated per clock cycle [KaVE95]. An empty cell position is found in the data buffer, and its address is provided, so that the cell can be stored in that position. The data buffer accesses proceed in parallel with the remaining control functions, i.e. flow control and routing decisions, described below.
- *Server 2:* while the data buffer is written with the cell data, the header of the packet is examined to determine the outgoing link on which the packet will be transmitted.
- *Servers 3 and 4:* after the routing decision, the flow control decision is made to determine if the cell can be queued immediately for transmission over the outgoing link. The cell header checks the Destination Credit Table and the Pool Credit Counter of the corresponding outgoing link, and finds if the necessary credits are available so that the cell can be considered ready for transmission (the Pool Credit Counter indicates the number of free cell buffers in the next switch of the cell's path, and the Destination Credit Table contains the number of available credits to the same destination). Depending on the existence of credits, one of the following three choices will be made:
  *(i)* if both credits exist, then the cell is ready for transmission. So, a descriptor for the cell is

appended to the Ready Queue of the appropriate outgoing link. Since there will be a credit sent to the previous upstream switch of the cell after its transmission, the information contained in the descriptor includes the incoming link and the destination of the packet;
*(ii)* if no destination credit exists, the packet descriptor is placed in a Hash Table. If the size of the Hash Table is $M$, the hashing function used is $Destination\_Address \,\mathbf{mod}\, M$. FIFO order of cells mapping to the same Hash Table entry is maintained. When the appropriate destination credit(s) arrives, the cell will be taken from the table and will proceed to check for the appropriate pool credit;
*(iii)* if a destination credit exists but there is no pool credit, the descriptor is appended to a queue of packets waiting for a pool credit. One such queue exists for each outgoing link.

- *Server 5:* when a cell transmission is almost complete, the next cell descriptor in the head of the Ready Queue is dequeued, and the data buffer is accessed to draw the cell body. Also, the appropriate credit counts are reduced, and a credit is sent to the upstream neighbor, indicating the empty space created.

Each server in the ATM switch model operates within one clock cycle. Given that server 4 is bypassed when both credits already exist, the minimum delay of a cell through an idle switch should be 4 clock cycles. As the switch data buffer provides cut-through capability, this minimal delay is indeed achieved. Each link delay is one clock cycle (short links). The cell-credit RTT is 8 clock cycles.

All the above actions are related to the route of a cell in the switch. In addition to regular ATM cells, there will also be credits arriving at the switch input links. Each credit includes a destination identifier indicating that the next downstream switch has buffer space available to accept a cell directed to that destination. When such a credit arrives at a switch, not only does it increase the corresponding count for the indicated destination, but the pool credit count as well. The switch actions related to a credit arrival are implemented in two modules:

- Hash Table manager: when a credit arrives, there are two possibilities. If a cell is waiting on a credit for the associated destination in the Hash Table, then the cell will be forwarded to the pool credit manager. If there is no cell waiting in the Hash Table, the destination credit count will be incremented. In the meantime, the appropriate pool credit count is incremented.
- As the pool credit count increases, if cells are waiting in a queue for the pool credit, then the cell in the head of the queue will be forwarded to the Ready Queue, while the pool credit count is decremented.

### *A.2 Wormhole Switch Model*

A similar model was developed for the wormhole switch, as shown in figure 17. This model was derived from the ATM case, in order to obtain the least number of differences possible, so that the simulation comparisons are as "fair" as possible. Taking into account the functions of a wormhole network, the route of a flit through the wormhole routing switch is as follows.

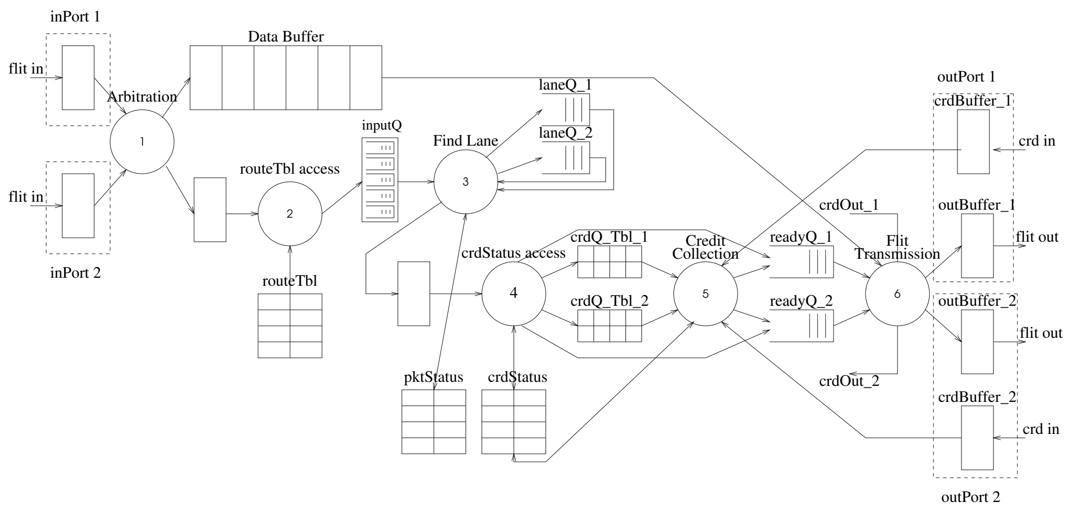


*Figure 17: Wormhole switch model*

- *Server 1:* the arriving flit is staggered appropriately, so that it can access the data buffer of the switch without conflicting with other incoming or outgoing flits. An empty flit position is found in the data buffer, and its address is provided, so that the flit can be stored in that position. The data buffer accesses proceed in parallel with the remaining control functions, i.e. flow control and routing decisions, described below.
- *Server 2: Routing Table:* packet-head flits are examined, to determine their outgoing link.
- *Server 3: Lane Assignment:* packet-head flits look for an available lane on the desired outgoing link. In case none exists, they get blocked, and so do all flits subsequently arriving on the same input lane.
- *Servers 4 and 5:* after having established the outgoing link and lane, the descriptor proceeds to the module that makes the flow control decision, in order to determine if the flit can be queued for transmission over the outgoing link. Only one kind of credits are needed here; these are equivalent to the destination credits in ATM. In wormhole, the equivalent process to securing a pool credit is finding a free lane. There is one credit count for every lane, indicating the available buffer space in the next downstream switch for that lane. Depending on the existence of a credit, one of the following two choices is made:
  *(i)* if a credit exists, the flit is ready for transmission, and its descriptor is appended to the Ready Queue corresponding to the appropriate outgoing link (there is one Ready Queue per outgoing link);
  *(ii)* if no credit exists, the flit descriptor is queued on the appropriate queue, until the appropriate credit arrives. The queue, in which the flit waits, is associated with the outgoing link and with the lane it is going to use ---there is 1 table of queues related to each link;
- *Server 6:* when a flit transmission is almost complete, the next flit descriptor in the head of the Ready Queue is dequeued, and the data buffer is accessed to draw the flit body. Also, the

appropriate credit count is reduced, and a credit is sent to the previous upstream switch of the cell indicating the empty space created.

As seen, wormhole has one more server than ATM; correspondingly, the cut-through delay is 5 clock cycles. Each link delay is again one clock cycle (short links). The cell-credit RTT is 9 clock cycles here.

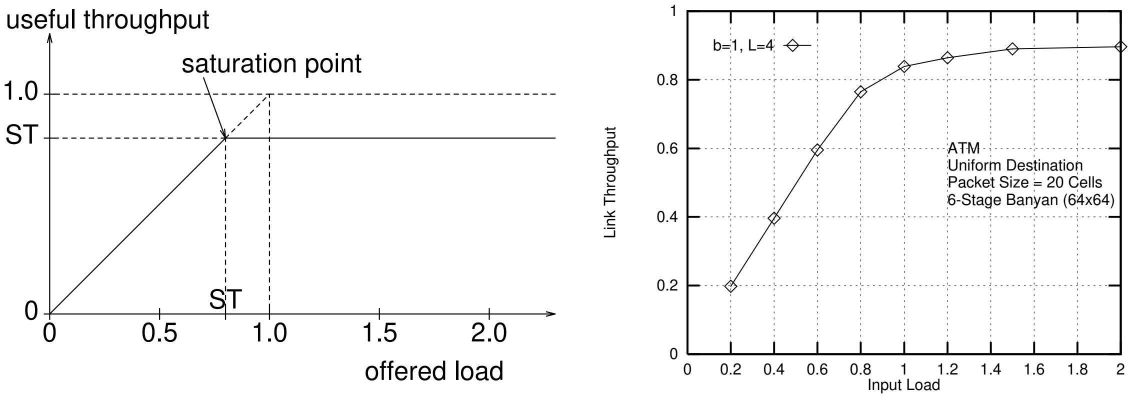


*Figure 18: Throughput versus input load: real (left) and simulated (right)*

### *A.3 Saturation Throughput*

When the uniformly-destined load offered to a symmetric, credit-flow-controlled network by its sources increases, the network throughput varies as shown in figure 18. A single number, the *saturation throughput (ST),* suffices to describe the throughput-versus-load function. In this paper, we only report this number when studying throughput as a measure of performance. For a large enough number of lanes, the saturation throughput is 1.0, while for fewer lanes it has lower values. The right part of figure 18 shows a typical throughput-versus-load measurement in our simulation environment. For input loads near 1.0, the throughput result of the simulator is lower than the expected real throughput, because of the variance in the distribution of the random destinations, and because the duration of our simulation is not long enough to overcome the effects of that variance. There are significant differences among the numbers of packets arriving at different destinations; overloaded destinations cannot absorb more than their saturation throughput, while underloaded destinations contribute reduced values of throughput, so the average is lower than the real expected value.

## References

**[ACPN95]** T. Anderson, D. Culler, D. Patterson, and the NOW team: "A Case for NOW (Networks of Workstations)", *IEEE Micro Magazine,* vol. 15, no. 1, February 1995, pp. 54-64.

**[Bode95]** N. Boden, D. Cohen, R. Felderman, A. Kulawik, C. Seitz, J. Seizovic, W-K. Su: "Myrinet: A Gigabit-per-Second Local Area Network", *IEEE Micro Magazine,* vol. 15, no. 1, February 1995, pp. 29-36.

**[Bork90]** S. Borkar e.a.: "Supporting Systolic and Memory Communication in iWarp", *Proc. of the 17th Int. Symp. on Computer Architecture,* ACM SIGARCH vol. 18, no. 2, June 1990, pp. 70-81.

**[CHKM93]** R. Cypher, A. Ho, S. Konstantinidou, P. Messina: "Architectural Requirements of Parallel Scientific Applications with Explicit Communication", *Proc. of the 20th Int. Symp. on Computer Architecture,* ACM SIGARCH vol. 21, no. 2, May 1993, pp. 2-13.

**[CSIM17]** Mesquite Software, Inc.: "CSIM17 User's Guide" (see also: H. Schwetman: "CSIM User's Guide", Microelectronics and Computer Technology Corporation, March 1990).

**[DaSe87]** W. Dally, C. Seitz: "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks", *IEEE Trans. on Computers,* vol. C-36, no. 5, May 1987, pp. 547-553.

**[Dally90]** W. Dally: "Virtual-Channel Flow Control", *Proc. of the 17th Int. Symp. on Computer Architecture,* ACM SIGARCH vol. 18, no. 2, May 1990, pp. 60-68.

**[Dec94]** "GIGAswitch/ATM Networking Switch", *Digital Equipment Co.,* Nov. 1994, URL: file: //ftp.digital.com/pub/Digital/info/infosheet/ EC-F3924-42.ps

**[GeKl80]** M. Gerla, L. Kleinrock: "Flow Control: A Comparative Survey", *IEEE Trans. on Communications,* vol. 28, no. 4, April 1980, pp. 553-574.

**[Gies78]** A. Giessler, e.a.: "Free Buffer Allocation – An Investigation by Simulation", *Comput. Networks,* vol. 2, 1978, pp. 191-208.

**[Inmo93]** "IMS C104 Packet Routing Switch – Preliminary Datasheet", *INMOS Ltd,* 1000 Aztec West, Bristol, BS12 4SQ. UK, June 1993.

**[KaHM87]** M. Karol, M. Hluchyj, S. Morgan: "Input versus Output Queueing on a Space-Division Packet Switch", *IEEE Trans. on Communications,* vol. COM-35, no. 12, December 1987, pp. 1347-1356.

**[KVES95]** M. Katevenis, P. Vatsolaki, A. Efthymiou, M. Stratakis: "VC-level Flow Control and Shared Buffering in the Telegraphos Switch", *Proceedings of the Hot Interconnects III Symposium,* Stanford Univ., CA, USA, Aug. 1995. URL: file://ftp.ics.forth.gr/tech-reports/1995/ 1995.HOTI.VCflowCtrlTeleSwitch.ps.gz

**[KaVE95]** M. Katevenis, P. Vatsolaki, A. Efthymiou: "Pipelined Memory Shared Buffer for VLSI Switches", *Proceedings of the ACM SIGCOMM '95 Conference,* Cambridge, MA USA, 30 August - 1 Sep. 1995, pp. 39-48. URL: file://ftp.ics.forth.gr/tech-reports/1995/ 1995.SIGCOMM95.PipeMemoryShBuf.ps.gz

**[KaSV96]** M. Katevenis, D. Serpanos, P. Vatsolaki: "ATLAS I: A General-Purpose, Single-Chip ATM Switch with Credit-Based Flow Control", *Hot Interconnects IV Symposium,* Stanford Univ., CA, USA, Aug. 1996. URL: file://ftp.ics.forth.gr/tech-reports/1996/ 1996.HOTI.ATLAS_I_ATMswitchChip.ps.gz

**[KuAg93]** J. Kubiatowicz, A. Agarwal: "Anatomy of a Message in the Alewife Multiprocessor", *Proceedings of the 7th ACM Int. Conference on Supercomputing,* Tokyo, Japan, July 1993.

**[Kung92]** H. T. Kung: "Gigabit Local Area Networks: A Systems Perspective", *IEEE Communications Magazine,* vol. 30, no. 4, April 1992, pp. 79-89.

**[KuBC94]** H.T. Kung, T. Blackwell, A. Chapman: "Credit-Based Flow Control for ATM Networks: Credit Update Protocol, Adaptive Credit Allocation, and Statistical Multiplexing", *Proceedings of the ACM SIGCOMM '94 Conference,* London, UK, 31 Aug. - 2 Sep. 1994, ACM Comp. Comm. Review, vol. 24, no. 4, pp. 101-114.

**[LeBo92]** J. LeBoudec: "The Asynchronous Transfer Mode: A Tutorial", *Computer Networks and ISDN Systems,* vol. 24, no. 4, May 1992.

**[Nuge88]** S. Nugent: "The iPSC/2 Direct-Connect Communications Technology", *Concurrent Supercomputing –the Second Generation,* Intel Corporation, 1988, pp. 59-68.

**[Ohsa95]** H. Ohsaki, e.a.: "Rate-Based Congestion Control for ATM Networks", *Computer Communication Review,* ACM SIGCOMM, vol. 25, no. 2, April 1995, pp. 60-72.

**[OzSV95]** C. Ozveren, R. Simcoe, G. Varghese: "Reliable and Efficient Hop-by-Hop Flow Control", *IEEE Journal on Sel. Areas in Communications,* vol. 13, no. 4, May 1995, pp. 642-650.

**[RaNe95]** K. Ramakrishnan, P. Newman: "ATM Flow Control: Inside the Great Debate", *Data Communications,* June 1995.

**[Rind77]** J. Rinde: "Routing and Control in a Centrally Directed Network", *Proc. Nat. Comput. Conf.,* Dallas, Tx, USA, June 1977.

**[SiDa79]** J. Simon, A. Danet: "Controle des Ressources et Principes du Routage dans le reseau TRANSPAC", *Proc. Int. Symp. Comput. Networks, Versailles, FR, February 1979.*