Deflection routing in regular MNA topologies

(As appeared in Journal of High Speed Networks)

Paweł Gburzyński*
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada T6G 2H1

Jacek Maitan†
Lockheed Research Lab
O/9150 B/251 3251 Hanover Street
Palo Alto, CA 94304 USA

ABSTRACT: We introduce MNA (*Multigrid Network Architecture*)—a networking concept aimed at applications calling for very high transmission rates (of the order of gigabits per second). An MNA network is a collection of photonic switches linked by optical channels. A switch is responsible for relaying incoming packets along the best available routes to their destinations. Packets that cannot be relayed optimally are *deflected*, i.e., relayed along suboptimal paths. The technology for building MNA switches has been developed in Lockheed Palo Alto Research Lab [20, 21, 22]. In this paper, we discuss the logical aspects of MNA, present a model for investigating the performance of MNA in regular topologies, and apply this model to a number of network configurations.

*Supported in part by NSERC Grant No. OGP9183.
email: pawel@cs.ualberta.ca.
†Supported in part by Lockheed Internal Research Funding and NASA contract NAS2-13223.
email: jmaitan@isi.edu.

1 Network architecture

MNA is based on a fast photonic switching device whose logical structure is presented in Figure 1. The switching device has a number of *network input ports*, the same number of *network output ports*, a number of *host input ports*, and the same number of *host output ports*. The number of *network port pairs* is denoted by k_n and the number of *host port pairs* is denoted by k_h ($0 \leq k_h \leq k_n$). Typically, k_n is 8 or 16. The number of host port pairs can be 0 in which case no host can be connected to the switch.

Figure 1 should be put here.

A switch operates as a fast routing device. Based on the destination address encoded in the header of a packet arriving on an input port, the switch determines via which output port the packet is to be relayed. Packets are not buffered before they are relayed, i.e., the *wormhole* technique (cf. [6, 7]) is used. The switch is capable of relaying k_n packets in parallel. Thus, if k_n packets arrive simultaneously on all the input ports, all these packets will be relayed to the output ports.

Multiple MNA switches can be configured into networks (e.g., see Figure 2). Not all $2k_n$ network ports of a switch must be used. However, if some of those ports are left disconnected, the number of connected output ports must not be less than the number of connected input ports. As this rule is enforced for all switches in the network, it is equivalent to say that these numbers must be equal. It makes sense to connect an output port of a switch to one of its own input ports.

Figure 2 should be put here.

If $k_h = 0$ for a switch S_i, i.e., no hosts are connected to the switch, the switch does not originate any traffic: it just relays packets arriving from the network. The packets are relayed based on the routing function $\mathcal{R}_i$ programmed into the switch S_i. Given the destination address d extracted from the packet header, $\mathcal{R}_i(d)$ partitions the output ports (numbers from 1 to k_n) into an ordered list $\mathcal{K}_1, \ldots, \mathcal{K}_p$ of disjoint subsets.[1] This list determines the suitability of each of the output ports to relay the packet on its way to S_d. The ports included in $\mathcal{K}_1$ are most suitable and the ports in $\mathcal{K}_p$ are least suitable. All ports within each $\mathcal{K}_j$ are deemed equally suitable/unsuitable.

[1] The routing function is implemented using high-speed lookup techniques.

At any moment, some of the output ports can be busy (i.e., relaying some packets). A packet arriving on an input port is relayed via the most suitable of the idle output ports. If multiple idle output ports have the same suitability, one of these ports is chosen at random. Thus, classes are ordered, but all ports within a given class have the same suitability rank.

Note that, according to the connectivity rules, if a packet arrives at a switch on one of its input ports, at least one output port must be idle. Thus, there is always a way to relay the packet, although it may not always be the best possible way. No packet is ever blocked at a switch. This routing mechanism resembles the so-called *deflection routing* introduced in [23, 24] for Manhattan-Street and Shuffle-Exchange networks. In our case, the operation of the network is completely asynchronous and no packets are ever buffered at the switches.

If $k_h > 0$, i.e., there are some hosts connected to the switch, the switch is expected to recognize packets addressed to these hosts. In such case, the image of the routing function $\mathcal{K}_s$ is extended by the set of the host output ports. A packet addressed to one of the hosts connected to the switch will be relayed via the port connecting the switch to the host. Note that if this port happens to be busy, the packet can be relayed to one of the network output ports. This operation can be viewed as "buffering" the packet in the network. It is also possible to have multiple input/output connections to the same host.

If a packet arrives on one of the host input ports, the switch should invoke its routing function $\mathcal{K}_s$, in the same way as for a packet arriving on a network input port. However, before the switch decides to use an output port to relay the host packet, it must know that the output port won't be needed to relay a network packet before the host packet has been entirely transmitted. Note that the switch must be able to relay all packets arriving on the network input ports, even if all k_n network packets arrive simultaneously. Therefore, the switch must be able to "predict the future" of the network input ports within the interval of one packet transmission.

This is accomplished by inserting a delay line before each network input port. The delay line has a sensing tap at a distance slightly more than one packet before the port. If the tap has been sensed idle for the time interval corresponding to the packet transmission, the *slot* of this input port can be used to insert a host packet into the network. Thus, for a host packet transmission, the host output port is temporarily connected to one of the input ports satisfying the above condition. If no such port exists at the moment, the host is notified that the transmission will have to be

postponed. Otherwise, the packet is handled as if it has arrived from the network. A switch that is not connected to a host need not be equipped with delay lines.[2]

2 Topologies

MNA switches can be interconnected into arbitrary graphs. The only requirement needed to guarantee that packets are never lost[3] is that for each switch, the number of used input network ports must not be greater than the number of used output ports. This way the network, besides passing packets to their destinations, plays the role of a distributed buffer for the packets that have been delayed due to congestion.

In fact, we are interested in topologies with somewhat stricter connectivity rules. Namely, we postulate that each connection be bidirectional, i.e., if there is a link from switch S_1 to S_2, there must be a link from S_2 to S_1. This way, if a switch becomes inoperable, its neighbors can avoid losing packets, provided that they are informed about the failure.

In this section we describe a few regular topologies for MNA. The common advantage of these topologies is that no switch is privileged or discriminated against. Thus, they may serve as starting points for implementing actual network configurations. Moreover, the regularity of the network topology results in a simplicity and regularity of the routing function $\mathcal{K}_s$. This is not a serious advantage from the viewpoint of implementing an actual MNA network,[4] but it simplifies the model and provides some methodological ways of investigating it.

We assume that the switches in a network are addressed by integer numbers from 0 to $N-1$, where N is the total number of switches. In the following discussion, a switch will be denoted by the capital letter S with an index representing the switch address. Thus S_i stands for the switch number i.

The configuration of hosts is irrelevant from the viewpoint of topology description. In this section, we are interested in the *communication subnet*[5] of the network. One can assume that every switch can be equipped with up to k_n host port pairs and potentially connected to that many hosts.

[2] In fact, some short delay lines are needed for the calculation of the routing function.

[3] Assuming there are no component failures.

[4] In reality, the routing function is described by lookup tables and any (possibly irregular) function can be described this way.

[5] According to the OSI terminology.

2.1 Hypercube

Hypercubes [4, 27] are regular networks typically used as backplanes of parallel computers.

Let us assume for simplicity that the number of switches N is a power of two. By the *Hamming distance* between two nonnegative numbers i and j, denoted $\mathcal{H}(i,j)$, we understand the number of positions on which the binary representations of i and j are different Thus $\mathcal{H}(i,j)$ is equal to the number of *ones* in the binary representation of the *exclusive OR* of i and j. We say that a collection of N switches are connected into a *binary hypercube* if each switch is connected with its immediate Hamming neighbors, i.e., two switches S_i and S_j are connected by a direct link if and only if $\mathcal{H}(i,j) = 1$. A hypercube network of 8 switches is shown in Figure 3.

Figure 3 should be put here.

If N happens to be a power of two, each switch has exactly $log_2(N)$ Hamming neighbors. If N is not a power of two, some switches are connected to fewer neighbors than others and the hypercube is incomplete. By definition, each connection in a hypercube is bidirectional, i.e., if $\mathcal{H}(i,j) = 1$, then there exists a link from S_i to S_j and another link from S_j to S_i. This property results from the symmetry of the Hamming distance which is direction-independent.

The hypercube topology is not very flexible because the number of switches to be connected (N) implies the (minimum) number of network ports per switch which is $\lceil log_2(N) \rceil$. Conversely, for a given number of network ports per switch k_n, the maximum number of nodes in a hypercube network is 2^{k_n}.

One can increase the flexibility of the hypercube topology in several ways (e.g. see [4, 6]). We consider one variant of the binary hypercube which is interesting from the viewpoint of MNA. In this variant, the list of neighbors of a switch S includes the switch whose address is the bitwise negation of the S's address. Such a connection corresponds to a diagonal in the hypercube.

Figure 4 should be put here.

For example, the "best" (and the maximum) number of 8×8 switches to form a regular hypercube (without diagonal connections) is 256. However, 128 switches can be fully connected into a symmetric diagonal hypercube by using one diagonal connection per switch. By breaking some of

the diagonals, one can reclaim any even number of spare ports between 2 and 128. These ports can be used to connect arrays of 128 switches into *meta*-hypercubes, rings, tori (see below), etc. A fully connected diagonal hypercube with 8 switches is presented in Figure 4.

2.2 Torus

Another symmetric topology for interconnecting switched networks is *torus* [28]. In a classical two-dimensional torus, switches are arranged as shown in Figure 5. The number of switches N needed to form a complete torus structure must be a square.

Figure 5 should be put here.

Let $D = \sqrt{N}$. Assuming that the switches are numbered form 0 to $N-1$ by rows (e.g., the switches in the first row are numbered $0, \ldots, D-1$, in the second row $D, \ldots, 2D-1$, etc.), the switch number i can be viewed as an encoded pair of planar coordinates:

$$c(i) = i \; mod \; D, \quad r(i) = \lfloor \frac{i}{D} \rfloor;$$

where c and r indicate the column and row position of the switch in the planar array. Each torus switch S_i is connected with four neighbors whose coordinates are:

$$[c(i), r(i)+1],\ [c(i), r(i)-1],\ [c(i)+1, r(i)],\ [c(i)-1, r(i)];$$

were the operations + and − are taken modulo D.

Each connection is assumed to be bidirectional. Therefore, each switch in the network presented in Figure 5 has four input ports and four output ports. This number is somewhat small for a commercial version of the MNA switch. Eight-by-eight switches can be connected into two "parallel" tori or, more interestingly, they can explore diagonal connections (Figure 6). With this approach, each switch S_i is connected with eight neighbors, the coordinates of the additional four neighbors being:

$$[c(i)+1, r(i)+1],\ [c(i)+1, r(i)-1],\ [c(i)-1, r(i)+1],\ [c(i)-1, r(i)-1];$$

Figure 6 should be put here.

It is possible to build multidimensional tori. The number of switches in a full m-dimensional torus must be an m-th power of an integer number D which gives the length of one dimension. The number of port pairs for a switch in an m-dimensional torus is $2m$ or $4m$, depending on whether diagonal connections are used.

2.3 Ring

Yet another generic topology for interconnecting MNA switches considered in this paper is *chordal ring* [2, 12]. Assume that k_n is even. A switch S_i is connected with $k_n/2$ "forward" neighbors $S_{i+1 \bmod N}$, $S_{i+2 \bmod N}, \ldots, S_{i+k_n/2 \bmod N}$ and $k_n/2$ "backward" neighbors $S_{i-1 \bmod N}$, $S_{i-2 \bmod N}, \ldots, S_{i-k_n/2 \bmod N}$.[6] This is illustrated in Figure 7. By definition, each connection is symmetric and bidirectional.

Figure 7 should be put here.

For some applications, due to geometric constrains, the ring topology may be more feasible than, say, a hypercube or torus. A chordal ring can be laid using a single cable with a number of wires independent of the number of switches in the network. One can also imagine an MNA-based network of satellites circling the globe. Each satellite can "see" a number of its "forward" and "backward" neighbors which are connected to it via microwave or laser channels.

3 Routing functions

Given an arbitrary network of MNA switches, the routing function $\mathcal{R}_i$ for a switch S_i can be precomputed and stored in a lookup table. Assuming that the function is static, i.e., its value is determined exclusively by the recipient's address, it should minimize the number of hops to the destination. The rank of an output port p should be determined by the length (in terms of the number of hops) of the shortest path from the neighbor connected to S_i via p to the destination d. If two ports offer the same number of hops, they can be ranked by the number of ways in which the shortest path can be achieved. If these ranks are also equal, the ports can be put into the same class $\mathcal{K}_j$ or some further criteria can be used (e.g., the length of the second shortest path and the number of ways in which this length can be met).

[6] If $a < b$, then $a - b \bmod N = a + N - b \bmod N$.

According to [25], the routing function in a deflection network should be somewhat nondeterministic—to avoid *live-locks*. An algorithm for building the routing function for any graph of MNA switches can be obtained by a straightforward adaptation of the well-known *shortest path* algorithm [13].

Given a network topology (not necessarily a regular one), we can define the notion of the *average length of the shortest path* between two switches. This measure (denoted by $ALSP$ for brevity) is given by the following formula:

$$ALSP = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \frac{D_{min}(S_i, S_j)}{N(N-1)} \quad ,$$

where N is the number of switches in the network and $D_{min}(S_i, S_j)$ stands for the minimum number of hops between the switches S_i and S_j. [7]

$ALSP$ can be viewed as a rough performance measure of the network. Clearly, one should expect lower delays when $ALSP$ is low and *vice versa*. Note that $ALSP$ is independent of the routing function; however, a "decent" routing function should be consistent with $ALSP$ in the sense that in the absence of contention, each packet should be relayed along its shortest path to the destination. This property of the routing function will be called the *shortest path property*.

For the regular topologies presented in the previous section, the routing functions are also regular and can be defined in a systematic way.

For example, let us consider the hypercube topology. For a straightforward hypercube with $N = 2^D$ switches, the output ports of a switch S_i receiving a packet addressed to S_d, $d \neq i$, can be divided into two classes $\mathcal{K}_1$ and $\mathcal{K}_2$. The first class consists of the ports that move the packet *closer* to the destination (i.e., the Hamming distance of the packet's location from the destination is reduced) and the second class includes the ports that increase the packet's distance from the destination. All ports within their classes are assumed to have equal ranks. An attempt is first made to relay the packet to any port in class $\mathcal{K}_1$. If all ports in $\mathcal{K}_1$ are busy, an idle port from $\mathcal{K}_2$ is chosen. Note that when $\mathcal{H}(i, d) = D$, $\mathcal{K}_2$ is empty. This corresponds to routing the packet from the corner of the hypercube opposite to the destination switch.

To calculate $ALSP$ for a straightforward hypercube, it is sufficient to consider paths to/from a single switch, e.g. S_0. This holds for any regular topology which, by definition, is symmetric with respect to any switch.

[7] We assume that $D_{min}(S_i, S_i) = 0$.

Given a switch S_j, $0 < j < N$, its shortest distance from S_0 is equal to the number of ones in the binary representation of j. The number of switches whose shortest distance from S_0 is $d \leq D$ is equal to the number of ways in which one can put exactly d ones on D binary positions. Thus, for a straightforward hypercube:

$$ALSP_{hyp} = \frac{\sum_{i=1}^{D} \left(\begin{array}{c} D \\ i \end{array} \right) i}{N-1} \tag{1}$$

which approaches $D/2$ for large N.

The routing function for a diagonal hypercube is slightly more complicated. Let $\mathcal{H}(i,j)$ denote the Hamming distance between switches S_i and S_j. The actual shortest distance (in terms of hops) between these switches is equal to $min(\mathcal{H}(i,j), 1 + D - \mathcal{H}(i,j))$. Thus, when a packet is relayed via one of the output ports, its distance from the destination may:

- decrease by one hop,
- increase by one hop,
- remain the same.

The third scenario is only possible when D is even and $\mathcal{H}(i,d) = D/2$, where S_d is the destination switch and S_i is the switch making the routing decision. Note that then any port that doesn't decrease the packet's distance from the destination doesn't increase it either. Thus, the routing function can still produce two port classes. Another peculiarity occurs when D is odd and $\mathcal{H}(i,d) = (D+1)/2$. Irrespective of how the packet is routed, its distance from the destination is then reduced.

For a diagonal hypercube we have:

$$ALSP_{dhyp} = \frac{\sum_{i=1}^{D} \left(\begin{array}{c} D \\ i \end{array} \right) min(i, D-i+1)}{N-1} \tag{2}$$

which approaches $D/4$ for big N. Thus, the shortest path between two switches in a diagonal hypercube is about two times shorter than in a regular one. Note however, that each switch in a diagonal hypercube has one port more than a switch in a regular hypercube with the same number of switches.

Similar calculations can be performed for the other regular topologies. Due to the limited size of this paper, they have been skipped.

4 Performance model

The results of queuing-based models [7, 8, 9] for regular switched network topologies are not applicable to MNA, as the network doesn't operate in a store-and-forward manner. Unlike in some other models [4, 6], packets in MNA are never blocked. A packet that cannot be relayed along the shortest route to the destination is buffered within the network and, in a negative feed-back fashion, reduces its capability to handle new traffic.

In this section we present a theoretical model for investigating the performance of regular MNA networks. More specifically, the model will help us determine the average number of hops made by a packet on its way to the destination and the maximum throughput achievable by the network under uniform load. The only assumption of our model is the operational regularity of the network understood as the "equal status" of all links. More formally, this assumption can be stated as follows: a uniform offered load that neither favors nor discriminates against any switch is spread uniformly over all links in the network. Note that this property cannot be fulfilled by the network topology alone: it is a joint property of the topology and the routing function.

Many authors investigating the performance of regular switched network topologies (e.g., hypercubes) consider hierarchical distribution of traffic ([8] contains an excellent survey of methods and literature). With this approach, the likelihood of a packet being addressed to a given destination switch decreases with the distance of that destination from the source. This is a reasonable assumption if the network is used to interconnect multiple processors of a distributed computer. Then, the hierarchical nature of the traffic results from the way in which cooperating processes are assigned to their processors. On the other hand, MNA-based architectures fit better into a local network category where uniform traffic models are more appropriate.

Assume that every switch in the network, besides relaying packets that arrive on its input ports, is capable of generating packets of its own. In other words, each switch has a host connected to it. The traffic generated by the hosts is uniform and described by a Poisson process. The model conveniently abstracts from two technical attributes of the network: the packet length and the length of links, yet it turns out to be surprisingly accurate. The arrival rate, denoted by λ, tells the average number of bits arriving to the network in a one-bit time slot. As is customary in a homogeneous networking environment, we use bits as the common units of information length, time and distance.

The total number of links in a network with N switches is $N \times k_n$, where k_n is the number of input (output) network ports per switch. Assume that the offered load represented by λ is below the network saturation point. Let ϱ denote the average number of hops (links) traveled by a packet on its way from the source to the destination. Then:

$$\mathcal{P} = \frac{\varrho\lambda}{Nk_n} \tag{3}$$

gives the probability that a given link is found busy. Of course, ϱ is in fact a function of λ. Thus the above formula describes a relationship between $\mathcal{P}$, ϱ, and λ: it is not a prescription for calculating $\mathcal{P}$ for a given network load.

Assume, however, that $\mathcal{P}$ is known. If the network is regular (according to our definition), then $\mathcal{P}$ is the same for all links. The operation of routing a packet at a switch can be viewed as an element in a Markovian chain. Namely, the routing decision does not depend on the routing history of the packet. Our goal is to derive a method for calculating ϱ as a function of $\mathcal{P}$. This will give us a system of equations. By solving these equations, we will derive ϱ as a function of λ.

Due to the fact that the network is regular, we may consider a fixed destination switch and all paths leading to that switch. Let S_0 be our destination. The question we ask now is: "Given a switch S_i, $i \neq 0$ making a routing decision for a packet addressed to S_0, what is the expected number E_i of hops to be made by this packet on its way to S_0?"

The packet will be routed to one of the neighbors of S_i. It will make one hop to the neighbor, then its fate will be determined by that neighbor. This gives us the following formula for E_i:

$$E_i = 1 + \sum_{j=1}^{k_n} P_j E_{s_j} \tag{4}$$

where P_j is the probability that the packet will be relayed via the output port number j, and s_j is the index of the switch connected to S_i via that port. Clearly, $E_0 = 0$.

The probabilities P_j are determined by the routing function $\mathcal{R}_i(0)$. Assume that $\mathcal{K}^i_1, \ldots, \mathcal{K}^i_{m_i}$ are the port classes generated by $\mathcal{R}_i(0)$. The ordering of these classes reflects the order in which they are used to obtain a free port ($\mathcal{K}^i_1$ is tried first and $\mathcal{K}^i_{m_i}$ is used last). Let $\overline{\mathcal{K}^i_p}$ denote the number of elements (ports) in $\mathcal{K}^i_p$. The event consisting in selecting a port from a class $\mathcal{K}^i_p$, $1 \leq p \leq m_i$ occurs if the following two conditions hold simultaneously:

- No port in one of the preceding classes $\mathcal{K}_1^i, \ldots, \mathcal{K}_{p-1}^i$ is free, i.e., all these ports are busy.
- At least one port in class $\mathcal{K}_p^i$ is free.

If there are multiple free ports in $\mathcal{K}_p^i$, one such port is selected at random. Thus, the ordering of ports within a class is irrelevant.

At the moment when we are making a routing decision, one output port must be free unconditionally. This is guaranteed by the fundamental property of the MNA switch which says that the number of (connected) input and output ports must be the same.

Any output port can be unconditionally free with probability $1/k_n$. A port that is not unconditionally free is busy with probability $\mathcal{P}$. Thus, the probability that a port from class $\mathcal{K}_p^i$ will be used to relay the packet is:

$$\Psi_p^i = \left(1 - \frac{\sum_{j=1}^{p-1} \overline{\mathcal{K}_j^i}}{k_n}\right) \mathcal{P}^{\sum_{j=1}^{p-1} \overline{\mathcal{K}_j^i}} \left(1 - \left(1 - \frac{\overline{\mathcal{K}_p^i}}{k_n - \sum_{j=1}^{p-1} \overline{\mathcal{K}_j^i}}\right) \mathcal{P}^{\overline{\mathcal{K}_p^i}}\right) \quad (5)$$

We assume that the result of an empty summation, i.e., when the starting index is greater than the boundary index, is zero. Formula 5 can be explained as follows. The first of the three factors gives the probability that none of the ports in the classes preceding $\mathcal{K}_p^i$ is unconditionally free. The second factor gives the probability that all ports in the classes preceding $\mathcal{K}_p^i$ are busy (assuming that none of them is unconditionally free). Finally, the third factor gives the probability that a free port is found in $\mathcal{K}_p^i$ (under assumption that it hasn't been found in one of the preceding classes).

The probability P_j of relaying the packet via the port number j is given by:

$$P_j = \Psi_p^i / \overline{\mathcal{K}_p^i}$$

where $\mathcal{K}_p^i$ is the class containing the port.

Given the values $E_1, \ldots, E_{N-1}$, the average number of hops made by a packet in the network can be calculated as:

$$\varrho = \frac{\sum_{i=1}^{N-1} E_i}{N-1} \quad (6)$$

Equations 3, 4 (which is actually a set of linear equations) and 6 constitute a system of $N+1$ equations (we ignore the trivial equation $E_0 = 0$) with $N+1$ unknowns. One simple way to solve this system numerically is to notice that (for a given λ) $\mathcal{P}$ is an increasing function of ϱ. Therefore, the

following iterative procedure can be used to determine ϱ for a given value of λ:

1. Set $\mathcal{P} = 0$.
2. Solve the linear set of equations[8] 4 to calculate $E_1, \ldots, E_N$.
3. Using Formula 6, calculate ϱ.
4. Calculate the new value of $\mathcal{P}$ from Formula 3 and go to 2.

The above algorithm generates at step 3 a sequence $\varrho_0, \varrho_1, \ldots$ of approximations for ϱ. If this sequence converges, it means that λ is below saturation and then:

$$\varrho = \lim_{n \to \infty} \varrho_n$$

is the sought value of the average packet hop count. In practice, the algorithm can be stopped as soon as ϱ reaches some numerical accuracy, e.g., the precision of the machine arithmetic. To determine the saturation load (the maximum throughput) of the network, one has to find λ_{sat}—the maximum value of λ for which the sequence ϱ_n converges.

Note that the initial value of ϱ (ϱ_0) corresponds to no load in the network. Thus it coincides with the *ALSP* parameter for the network. If *ALSP* is known, it can be used as the first approximation of ϱ and the first step of the algorithm can be skipped.

5 Performance

In this section we present some performance results obtained from the model introduced in Section 4 and from simulation. Simulation models of the investigated networks were built in SMUPRH [14, 15]. These models carefully reflected all the relevant technical details of a realistic implementation. In particular, they accounted for the details ignored by the theoretical model, i.e., a definite packet size and channel length.

5.1 Average shortest path length

One simple performance measure of an MNA network is the average shortest path length (*ALSP*) introduced in Section 3. Generally, one would expect

[8] This set is sparse and can be solved efficiently, even if the number of switches is large, e.g., by the methods described in [19].

that a network with smaller $ALSP$ should perform better than a network for which $ALSP$ is higher.

To compare networks with different numbers of ports per switch, we propose the following weighted variant of $ALSP$, which relates the measure to the network complexity:

$$WALSP = ALSP \times k_n$$

Figure 8 should be put here.

Figure 8 shows the dependence of $WALSP$ on the network size (the number of switches) for five representative regular topologies of MNA. These topologies are: the regular hypercube (hyp), the hypercube with diagonals (d-hyp), the four-dimensional torus (tor), the two-dimensional torus with diagonals (d-tor), and the chordal ring (ring). For the two hypercube topologies, the number of network ports per switch (k_n) depends on the number of switches. For the remaining topologies, k_n was fixed at eight. In the case of the tori, k_n is determined by the torus dimension and is not flexible. In the chordal ring topology, any even number of ports is legal (eight was chosen as a reasonable port number for a commercial MNA switch).

The two hypercubes and the four-dimensional torus are very close. The two-dimensional torus is slightly worse and the chordal ring is much worse than the other topologies. There is a trade-off between the flexibility of an MNA topology and its performance. Note that the ring topology is most flexible as it imposes absolutely no restriction on the number of switches and practically no restriction on k_n. On that scale, the two-dimensional torus with diagonals is more flexible than the four-dimensional torus. In the former, the number of switches must be a square; in the latter, it must be a fourth power.

5.2 Throughput versus number of hops

All MNA networks exhibit very similar performance characteristics for uniform traffic. This performance does not visibly depend on the packet length and the length of channels. Note that our model introduced in Section 4 was built without assuming any specific packet length or channel length.

The actual observed packet delay depends on three factors: the packet access time (i.e., the amount of time spent by the transmitter awaiting an idle input port), the combined length of the links traveled by the packet on its way to the destination, and the routing delay at the relaying switches.

The last two factors can be expressed together as the number of "hops" made by the packet. The packet delay expressed in this manner is independent of the link length. Simulation experiments indicate that the actual length of links has little impact on the observed average number of hops. Moreover, the packet access time turns out to be an insignificant fraction of the total delay, at least for a load below saturation.

Figure 9 should be put here.

Figure 9 shows the performance of a regular MNA hypercube with 256 switches. The solid line represents results obtained from our analytical model. The discrete points were obtained experimentally in four different environments. In the basic reference network used in our experiments, the total packet length was 1192 bits[9] and all channels were of the same length equal to 100 bits.[10] This network is denoted by "hyp1" in the figure. In the network denoted by "hyp2," the channel length was increased to 10000 bits, but it was still the same for all channels. In "hyp3," the channel length was generated as an exponentially distributed random number with the mean of 10000 bits. Finally, "hyp4" denotes the reference network with three times shorter packets (the payload was reduced from 1152 bits to 384 bits[11]).

Figure 10 should be put here.

Figure 10 shows the performance of three networks: a diagonal hypercube with 128 switches, a two-dimensional diagonal torus with 121 switches, and a four-dimensional torus with 81 switches. In all cases, the discrete points represent the results of simulation experiments performed on the corresponding reference network.

The average packet delay (measured in terms of hops from the source to the destination), grows rather slowly with increasing load. The saturation point is very sharp: the network chokes up very rapidly as soon as the saturation load is exceeded. This phenomenon, which is rather common to MNA networks, is discussed in Section 5.4.

Our analytical model seems inadequate to describe the performance of a ring configuration of MNA. Note that the ring variant of MNA does not

[9] This corresponds to 40 bits of frame and 1152 bits of payload. The latter number is equal to three times the payload size of an ATM packet.

[10] This corresponds to 20 meters at $1Gb/s$.

[11] Which coincides with the ATM packet length.

fulfill the "operational regularity" postulate formulated at the beginning of Section 4. Namely, in a network consisting od sufficiently many switches, the output ports connecting a switch to more distant neighbors are used more heavily than other ports. This is illustrated in Table 1 which lists the observed relative occupancy rate for different ports in the chordal ring with 128 switches. The number of output ports per switch was 8. In Table 1, these ports have been grouped into pairs, each pair including ports connecting the switch to two symmetric neighbors. P1 represents the ports leading to the closest neighbors and P4 represents the most distant connections. The numbers in each row add to 1. Each number can be viewed as a relative frequency of finding a port of the given pair busy.

Table 1 should be put here.

The two ports connecting the switch to its most distant neighbors are busy much more often than any other ports. This is very pronounced for light loads and tends to disappear when the load becomes heavier. Note that the load represented by the last row is above the saturation point.

Figure 11 should be put here.

Our model, which assumes that the probability of finding a port busy is the same for all ports, will tend to produce optimistic delay results for a ring network. This is due to the fact that the ports that offer shorter paths to distant destinations are busy more often in reality than in the model. Figure 11 confirms these expectations.

5.3 Maximum throughput

The results presented in the previous section show that our model well predicts the behavior of operationally regular MNA networks. It somewhat fails for the ring topology which is not operationally regular; however, even in that case, it very accurately predicts the maximum throughput achieved by the network. Indeed, the numbers in Table 1 tend to equalize when the load becomes heavier. At the exact point of saturation they should be exactly equal! Unfortunately, this point is extremely unstable and practically impossible to catch by simulation.

Figure 12 should be put here.

This observation allows us to claim that our model accurately predicts the maximum throughput of any MNA network with a regular topology, even one that is considerably irregular in the operational sense.

Figure 12 shows the maximum throughput achieved by a ring network with $k_n = 8$ versus the number of switches. The simulation results for the reference network agree spectacularly well with the analytical predictions.

Figure 13 should be put here.

Figure 13 compares the maximum throughput achieved by different MNA topologies. As some of these topologies are based on different values of k_n, the throughput in Figure 13 has been weighted, i.e., divided by k_n. Notably, Figure 13 is consistent with Figure 8.

5.4 Oversaturated networks

All MNA networks exhibit very similar behavior when the offered load exceeds the saturation threshold. The observed throughput drops rapidly and, irrespective of the offered load (provided that this load is above the saturation level), sits at one point, at $70 - 95\%$ of the maximum throughput achieved for the undersaturated network. For example, for the saturated regular hypercube from Figure 9, the observed throughput was 110 and the average hop count was between 13 and 14.

A saturated MNA network stabilizes at one point in consequence of two counteracting feedback mechanisms. As soon as the offered load crosses the saturation threshold, deflected packets push the actual load even further until the network breaks down. Theoretically, the network should really break down and settle at a point at which routing is performed entirely at random. This point can be easily calculated from our model by setting $\mathcal{P}$ (Equation 3) to 1. However, this estimate is too pessimistic by more than one order of magnitude. For example, for the hypercube network with 256 switches, the saturation throughput calculated this way is 6.8 at the average of 302 hops per packet. This is completely out of line with the observed behavior of the hypercube. As it turns out, busy links make it more difficult for the hosts to insert new packets into the network (Section 1). Thus, a saturated network automatically restrains the hosts from chocking it up completely and is able to maintain a reasonable throughput, even under extreme load conditions. In consequence of this phenomenon, the performance curve of an MNA network consists of two topologically disjoint parts: a continuous line

describing the network behavior below the saturation load and an isolated point representing the saturated state of the network.

The above observation suggests that our system of equations introduced in Section 4 (Equations 3, 4, 6) should offer two sets of solutions (for $\mathcal{P}$ and ϱ) for a given value of λ. To obtain the second set, we should interpret λ as the observed throughput of the network rather that the offered load (below saturation the two interpretations are the same). The reasoning is as follows. Assume that we are able to insert traffic into the network "magically," i.e., without having to acquire a free input port for the pertinent time interval. Then, a given value of observed throughput (achievable by the network) can be arrived at in two ways. One way is to offer the load that coincides with the throughput. Under such conditions, the network operates below its saturation level. The other way is to offer a load that is higher than the saturation throughput. By balancing this load properly, we can sustain a given observed throughput. The values of $\mathcal{P}$ and ϱ in this case will be (much) higher than in the previous scenario.

Of course, the second scenario is extremely unstable and, more importantly, unrealistic. Once the saturation point has been crossed, new incoming packets are queued at the hosts and (regardless of the offered load) each host becomes permanently ready to transmit. But new packets can only be inserted into the network at a specific rate determined by the availability of idle input ports. This way, from amongst the apparent continuum of states, the feedback mechanism mentioned above selects one point of equilibrium.

The way we solve our model in Section 4 naturally selects the solution describing the equilibrium state below saturation. In this section we discuss a method of finding the equilibrium state of an oversaturated network.

Let L denote the packet length (it seems that this time we cannot get away without it). As all packets are equal in size, L can be viewed as the length of a transmission slot. Each host has a continuous supply of packets to transmit. It can transmit a new packet as soon as it finds an input port which is going to be silent for at least L bits (Section 1). Given an input port, we now ask the following question: "What is the expected waiting time for a period of silence of length L to appear on the port?" The answer to this question will let us calculate the maximum rate at which new packets can be inserted into the port. Clearly, it depends on $\mathcal{P}$ (the probability of finding the port busy—see Equation 3) and on the distribution of silence periods in the ports. Assume (somewhat optimistically) that this distribution is known. Let $G(x)$ be its density function. Activities perceived on an input port p monitored by a transmitter T consist of packets interleaved with periods

of silence. The time interval separating the beginnings of two consecutive packets perceived by T on p will be called a *frame*. Thus, a frame consists of a packet followed by a period of silence. The probability that the period of silence at the end of a frame is shorter than s is given by the following formula:

$$P(x < s) = \int_0^s G(x)dx$$

First, let us assume that the moment at which T starts awaiting a period of silence on p coincides with the beginning of a frame.

$$P(x < L) = \int_0^L G(x)dx \tag{7}$$

is the probability that the period of silence in this frame is shorter than L. With probability $1 - P(x < L)$, the transmitter will get the awaited silence period in the current frame. Otherwise, it will have to wait for the beginning of the next frame and repeat the waiting. Thus, the mean waiting time D_f is described by the following formula:

$$D_f = L \times (1 - P(x < L)) + P(x < L) \times (L + S^- + D_f)$$

where

$$S^- = \frac{\int_0^L xG(x)dx}{\int_0^L G(x)dx} \tag{8}$$

is the mean length of the silence period, under condition that it is shorter than L. Solving the above equation, we get:

$$D_f = \frac{P(x < L) \times S^- + L}{1 - P(x < L)} \tag{9}$$

Now we will consider the general case.

There is a continuous supply of packets; therefore, we may assume that T starts looking for a period of silence on port p immediately after finishing a previous transmission. At this moment, T perceives on p a period of silence (possibly very short) belonging to the current frame. The transmission that has been just completed could be the first transmission of T within the frame, the second transmission, the third transmission, etc. The probability that a transmission completed by T is its j'th transmission within the same frame is given by the following formula:

$$Q_j = \sum_{i=j}^{\infty} \frac{1}{i} \times \tilde{P}_i \tag{10}$$

where

$$\tilde{P}_i = \frac{\int_{iL}^{(i+1)L} G(x)dx}{\int_{L}^{\infty} G(x)dx} \tag{11}$$

is the probability that $iL \leq s < (i+1)L$, where s is the total length of the silence period in the current frame. Formula 10 can be explained as follows. If the length of the silence period is between iL and $(i+1)L$, $i \geq j$, then T can use this period to transmit i packets. Thus, the probability that a packet transmitted by T within such a silence period is the j'th packet transmitted by T in the current frame[12] is equal $1/i$. Note that we know that T has just completed a transmission, hence the condition $s \geq L$ (the denominator in Formula 11).

Now assume that the transmission completed by T was the j'th transmission within the frame. If the leftover of the silence period is longer than L, the waiting time is 0. Otherwise, T will skip the remaining portion of the silence period and get to the already considered case of starting at the beginning of a new frame. Thus, the waiting time is described by the following expression:

$$D_j = \hat{P}_j(\hat{S}_j - iL + D_f) \tag{12}$$

where

$$\hat{P}_j = \frac{\int_{jL}^{(j+1)L} G(x)dx}{\int_{jL}^{\infty} G(x)dx}$$

is the probability that $s < (j+1)L$, under condition that $s \geq jL$ and

$$\hat{S}_j = \frac{\int_{jL}^{(j+1)L} xG(x)dx}{\int_{jL}^{(j+1)L} G(x)dx}$$

is the total expected length of the silence period, under assumption that it is between jL and $(j+1)L$.

To obtain the general formula for the expected waiting time, one should combine all D_j's with their respective probabilities Q_j. This gives us:

$$D_e = \sum_{j=1}^{\infty} Q_j D_j \tag{13}$$

[12]Note that T has a continuous supply of packets to transmit.

as the average waiting time for the silence period of length at least L.

The mean length of a silence period perceived on an input port is given by the following formula:

$$\mu = \frac{L(1-\mathcal{P})}{\mathcal{P}} \tag{14}$$

where $\mathcal{P}$ is determined by Equation 3. The distribution G, which is not known, can be approximated by the exponential distribution[13] with $G(x) = e^{-x/\mu}$.

Formula 13 gives implicitly the maximum rate at which bits can be inserted by a host into one input port. This rate is:

$$\gamma = \frac{L}{L + D_e} \tag{15}$$

Notably, the way this rate has been calculated makes it dependent on $\mathcal{P}$, but independent of L. Indeed, D_e has been obtained[14] from a distribution whose mean value is given by Formula 14. Clearly, D is proportional to μ which, in turn, is proportional to L. This L is cancelled in Formula 15. Thus, we can get away without L after all!

To find the equilibrium state in an oversaturated network, we propose the following simple iterative method:

1. Set $\mathcal{P}$ to an arbitrary value between 0 and 1. A value somewhat higher than the one obtained for a slightly undersaturated network may be used as the starting point.
2. Solve the linear set (Equations 4) and calculate ϱ (the mean hop count).
3. Calculate the "observed" load of the network as

   $$\lambda_o = \frac{N \times \mathcal{P} \times k_n}{\varrho}$$

   (see Equation 3) and the "acceptable" load as

   $$\lambda_a = N \times k_n \times \gamma$$

   If $\lambda_a \approx \lambda_o$, stop. Otherwise, proceed at 4.

[13] This approximation turns out to be much better than it looks at first sight. In an oversaturated network of a non-trivial size, the average number of packet hops is of the order of tens. Thus, the correlation between packets inserted by the same transmitter into the same input port dissipates rather fast. Note that different packets are usually relayed via different output ports.

[14] Note that instead of D, we could have calculated D/L using essentially the same (although somewhat obscured) reasoning.

4. If $\lambda_a > \lambda_o$ (the network can accept more load), increase $\mathcal{P}$; otherwise decrease $\mathcal{P}$ and continue at 2. Bisection can be used at this step to adjust $\mathcal{P}$ in a systematic way.

The above algorithm can be stopped as soon as a reasonable accuracy is reached at point 3. The value of λ_o (or λ_a) produced by the algorithm describes the effective throughput of the oversaturated network. At the same time, ϱ gives the average number of hops made by a packet on its way to the destination.

Table 2 should be put here.

Table 2 compares the saturation points of the reference networks (see the previous section) obtained by simulation and computed using our model. Even for the ring network (which is not operationally regular), the predicted value is in a very good agreement with observation.

Figure 14 should be put here.

Figure 14 shows how the saturation throughput of our networks relates to their maximum throughput achieved under "normal" conditions. The vertical axis tells the percentage of the maximum throughput achieved in an oversaturated network. This percentage tends to drop with the increasing number of switches in the network.

The analytical results and the observed network behavior presented in Table 2 and Figure 14 have been obtained under the assumption that each switch has k_n independent host connections (i.e., $k_h = k_n$). Thus, as soon as any of the network input ports perceives a period of silence of length at least L, a host packet can be inserted into the network via this port. Experiments carried out for networks with $k_n = 8$ show that when $k_h < k_n$, the saturation throughput of the network tends to be slightly higher than in the case when $k_h = k_n$. For $k_h = 1$, the observed increase in the saturation throughput was of the order of $2 - 5\%$.

5.5 Packet access time

Formally, we define the *packet access time* as the amount of time elapsing since the moment a packet becomes ready for transmission until its transmission is started. To relate this measure to the packet length, we can express it as:

$$A_p = \frac{D_a}{D_a + L} \tag{16}$$

where D_a is the actual packet delay as defined above[15] and L is the packet length. This way, A_p describes the access time as the fraction of the total time spent on processing the packet at the transmitting switch.

As long as the network load is below the saturation threshold, the packet access time is an insignificant fraction of the total delay. One can try to derive this delay formally, as a function of offered load, using the results of the previous section.

Assume that an input port p is monitored by a transmitter T waiting for a period of silence of length at least L. Now, we cannot postulate that the host has a continuous supply of packets: it is more reasonable to assume that the transmitter gets a packet to transmit at a random moment. The mean waiting time of the transmitter can be estimated using the following reasoning.

The activities perceived by T on p are *frames* (see the previous section) consisting of packets interleaved with periods of silence. If the frames constituting the history of activities in p are drawn at random (e.g., from an urn), then the probability that a frame contains a period of silence shorter than L is given by Formula 7. From the viewpoint of the transmitter monitoring port p and shooting the frames at random, the probability of hitting a frame with a longer silence period is higher than the probability of hitting a frame with a shorter silence period, as the former frame lasts longer than the latter. Taking this into account, we get the following formula for the probability that at a random moment the transmitter perceives a frame containing a period of silence shorter than L:

$$P_s = P(x < L) \times \frac{S^- + L}{P(x < L) \times (S^- + L) + (1 - P(x < L)) \times (S^+ + L)} \quad (17)$$

where S^- is given by Formula 8 and

$$S^+ = \frac{\int_L^\infty xG(x)dx}{\int_L^\infty G(x)dx}$$

is the mean length of the silence period under condition that it is longer than L.

Thus, with probability P_s the transmitter has hit a frame with a period of silence shorter than L. Then, the expected waiting time is equal to

$$D_s = \frac{L + S^-}{2} + D_f$$

[15] Expressed in bits.

where D_f is determined by Formula 9.

With probability $1 - P_s$ the current frame contains the period of silence longer than L. Three scenarios are now possible:

1. With probability $L/(L + S^+)$ the transmitter has hit the packet part of the frame. Then the expected waiting time is $L/2$.

2. With probability $(S^+ - L)/(L + S^+)$ the transmitter has hit an initial period of the silence part for which the leftover is still not shorter than L. The expected waiting time in such case is 0.

3. With probability $L/(L + S^+)$ the transmitter has hit the silence part at the moment when the leftover is shorter than L. The expected waiting time in this scenario is $(L/2) + D_f$.

Putting all these cases together we get

$$D_a = P_s \left(\frac{L + S^-}{2} + D_f \right) + (1 - P_s) \frac{L}{L + S^+} (L + D_f) \tag{18}$$

Formula 18 allows us to calculate the expected waiting time for a period of silence of length L in an undersaturated network. Similarly as in the previous section, we can approximate the length distribution of the silence periods with the exponential distribution. The mean value of this distribution is a function of $\mathcal{P}$ (Formula 14) and, consequently, of λ (Formula 3). Note that the relative access time A_p (Formula 16) is independent of L.

Formula 18 has been derived under assumption that there is only one input port monitored by the transmitter. In reality, a transmitter connected to a switch monitors all k_n input ports at the same time. The expected waiting time for k_n ports monitored simultaneously, depends on the actual distribution of the waiting time for one port. Unfortunately, this distribution seems rather difficult to derive formally. A rough estimate of this waiting time can be obtained by assuming that the expected waiting time for k_n ports is $1/k_n$ of the expected waiting time for one port. This would be the case, if the distribution of the waiting time were exponential.

Figure 15 should be put here.

Figure 15 shows the estimated weighted packet access time (Formula 16) as a function of the probability of finding a port busy. For all reference

networks, except the chordal ring, the highest observed value of P below saturation was 0.5. For the chordal ring, it was 0.7, but even then the average number of hops (16) completely overshadows the impact of the packet access time on the overall delay. In fact, our estimate turns out to be rather pessimistic. Figure 16 shows the observed weighted packet access time in four reference networks (assuming $k_h = k_n$). The maximum observed weighted access time for the chordal ring was 0.69.

Figure 16 should be put here.

Figure 16 suggests that the distribution of the waiting time for a single input port is more advantageous than the exponential distribution.

Even in an oversaturated network, the packet access time is not a dominant component of the packet delay. Assume that there is one transmitter (host) per switch. Let λ_a denote the saturation throughput of the network. Each transmitter is able to insert one bit every N/λ_a bits of time. Thus, the weighted transmitter access delay is given by the following formula:

$$A_p^{sat} = \frac{N}{N + \lambda_a}$$

For example, in the torus network with 121 switches, the saturation throughput is 86 at the mean packet hop count of 7.87. The weighted access delay is 0.58, i.e., the access overhead to transmit a 1152-bit packet is 2743 bits. Although this is more than twice the packet transmission time, the "hop" delay is still much more significant. The ring network with 121 switches achieves the saturation throughput of 42 at the mean packet hop count of 18. The weighted transmitter access delay is 0.74 and the overhead for the transmission of a 1152-bit packet is 4430 bits.

5.6 Standard deviation of packet hop count

The observed variability of the packet delay (measured in terms of the number of hops) is reasonably low for practically all regular MNA networks. Figure 17 shows the standard deviation of hop count versus throughput in four reference networks.

Figure 17 should be put here.

The standard deviation of the packet hop count in all networks remains practically at the same level through the entire range of traffic conditions

below the saturation point. The highest variability of the packet hop count in the ring network results from the biggest differences in the length of the shortest paths between pairs of switches. The observed standard deviation of the hop count in saturated networks was between 8 and 10, depending on the topology.

5.7 Comments on biased traffic

Our analytical model is based on the assumption of uniform load which is spread evenly over all channels. We have seen, however, that in the case of the ring network (which is not operationally regular) the maximum throughput is predicted correctly by the model. One can speculate that moderate departures from the uniformity of the traffic pattern will have small influence on the asymptotic behavior of a reasonably large MNA network. This is due to the equalizing nature of deflection routing in MNA which, under heavy traffic conditions, tends to divert excess traffic via whatever channels are found to be underutilized.

To investigate what happens when the traffic is biased, we have carried out a simulation experiment involving two reference networks: the planar torus with 121 switches (a representative of the "small *WALSP*" family) and the chordal ring with 128 switches (representing networks with a poorer connectivity). In both cases the traffic was biased: the probability of selecting a given switch as a recipient of a message was inversely proportional to the number of hops separating that switch from the source. Such a traffic pattern can be viewed as an approximation of real traffic conditions in a tightly coupled distributed computer [8] or in a wide or metropolitan area network [26].

The results for the torus network were very similar to those presented in Figure 10. The maximum throughput achieved by the network was 8.5% higher than for the uniform traffic and the mean hop count was slightly lower (about 20% under light and 15% under heavy load). These differences can be easily explained by the shorter average distance traveled by a packet under biased traffic conditions. They become more pronounced in a network with a poorer connectivity, i.e., higher *WALSP*. In the ring network, the gain in maximum throughput was almost 40% and the mean hop count was reduced by 35–25%. This suggests that low-connectivity solutions may be more cost-effective in wide-area applications, not only due to the geographic constraints, but also from the viewpoint of traffic characterization.

6 Relation to previous work and suggestions for further research

Numerous issues related to deflection networks have been discussed in the literature [1, 3, 5, 10, 11, 16, 17, 18, 23, 24, 25]. Most of the models considered there are based on two simple and regular network architectures proposed by Maxemchuk in [23, 24], namely the Manhattan Street and Shuffle Exchange networks [5, 10, 11, 16, 18, 25]. Notably, Greenberg and Hajek [17] investigate the performance of synchronous hypercubes based on deflection routing. Bannister and Borgonovo [3] consider general topologies and nonuniform traffic, but their (approximate) model is restricted to two-connected networks, i.e., $k_n = 2$.

The Manhattan Street network is reminiscent of our planar torus architecture, especially in its bidirectional variant with 4 incoming and 4 outgoing ports per switch [5, 10, 11]. Both Manhattan Street and Shuffle Exchange networks operate in a slotted fashion [23, 24, 25]. In the original version, each switch has two input and two output ports interfacing it to the network, and two ports connecting it to the host. Within one-slot cycle, it is determined whether packets arrive on the network input ports and, if so, the routing decisions determining the fate of the packets are made by a centralized *decision box*. The host is allowed to insert its packet, if a slot buffer (associated with an output port) is available in the current cycle. The differences between this approach and our architecture can be stressed in the following points:

- Our networks operate in an asynchronous and unslotted manner. An internal switch, which is not connected to a host, requires no intermediate buffers for the packets being relayed. Routing decisions are made "on the fly" and the *decision box* is distributed. In a switch connected to a host, buffers (delay lines) are associated with the input rather than output ports.

- The minimum connectivity of a commercial MNA switch (k_n) is 8, whereas both Manhattan Street and Shuffle Exchange networks use switches with $k_n = 2$ or 4. Higher connectivity of a switch (8 or 16 as opposed to 2 or 4), coupled with the natural randomization of the routing function in MNA,[16] makes the problems mentioned in

[16] Note that the randomization is easier to carry out and more "natural," if the switch connectivity is higher.

[25] (live-lock, lockout) less likely to occur—perhaps to the extent of rendering them practically negligible.

- We are not concerned with the complexity of the routing function. The rationale behind simple and regular topologies with low-connectivity switches is in the simplicity and locality of the routing function which can be computed by a switch based on local information, e.g., by comparing the packet's destination address with the switch address. In our approach, the size of the lookup memory at a switch depends on the network size. The cost analysis based on our prototype indicates that even for large networks consisting of thousands of MNA switches, the cost of the lookup memory is a reasonable fraction of the switch price. With flexible routing functions we can easily implement arbitrary network configurations.

Despite these differences, some qualitative results obtained for Manhattan Street networks and synchronous hypercubes apply to MNA networks. Not surprisingly, mean hop count in undersaturated deflection networks exhibits similar characteristics across different architectures and topologies. In particular, one performance graph in [17] is similar in shape to the graphs in Figures 9 and 10. On the other hand, the behavior of a saturated MNA network and the characteristics of the packet access delay in MNA seem to be specific to this architecture.

In [25], Maxemchuk gives an overview of some problems that may occur in any deflection network, irrespective of its architecture and topology. These problems are: *live-lock* (packets circulating in the network without ever reaching their destinations), *lockout* (starvation of some switches by greedy or overloaded neighbors), and *congestion* (the network breaks down under saturating load). According to [25], live-locks can be eliminated by a proper randomization of the routing function. Note that routing functions in MNA naturally tend to be randomized. Such a natural randomization may not always be the "proper" one, i.e., it may not always absolutely eliminate live-locks, but, with $k_n = 8$ or 16, it may reduce their likelihood to an acceptable level. For example, in commercial Ethernet it is theoretically possible that two or more packets will collide for, say, two hours; however, nobody considers this possibility as a serious threat to the network reliability. Our experiments indicate that the performance of an MNA network doesn't suffer much if the routing function is simplified by throwing less eligible output ports into the same class and ignoring small differences in

their eligibility. This approach has two advantages: it reduces the size of the lookup tables and decreases (or even nullifies) the likelihood of a live-lock.

The probability of a lockout clearly depends on the switch connectivity and decreases much more than linearly with increasing k_n. Assuming that it is impossible for a single host to keep its switch constantly busy, lockout scenarios for MNA (with $k_n = 8$) are difficult to conceive. We are not partisans of feedback-based solutions suggested in [25]. Such solutions should be avoided in high-speed networks, at least at the *medium access control* level, as their quality degrades with increasing transmission rate. One should rather accept a reasonable non-zero probability of a lockout than negotiate medium access across the entire network.

As far as congestion is concerned, we have demonstrated that MNA handles saturation gracefully, at least as long as the traffic is approximately uniform. It would be interesting to investigate this issue for more realistic congestion scenarios involving proper subsets of the hosts. Intuitively, it seems that the equalizing nature of deflection in a highly connected network will help to disperse such congestions via its underutilized regions.

More research is needed to put the above speculations on a formal ground. Another avenue for further investigation is the eligibility of the MNA backbone for implementing connection-oriented and synchronous/isochronous services. We plan to carry out additional simulation experiments to determine the buffer requirements for reassembling stream messages at their destinations. At the same time, we would like to build a small MNA network to experiment with a simplified implementation of TCP/IP.

7 Conclusions

We have discussed a completely distributed packet-routing architecture called Multigrid Network Architecture, for building low-cost high-speed networks.

To prove the feasibility of such networks, we have prototyped a resource allocator which has an estimated performance of 8×10^6 packets/s for an 8×8 packet switch. Currently, we are completing a hardware prototype of an 8×8 pizza-box-sized packet switch capable of handling 1 Gbps traffic at each port.

The results presented in this paper indicate that despite the statistical nature of deflection routing used in MNA, the networks exhibit very good

performance characteristics. Within the "normal" range of traffic conditions (i.e., below the saturation point), the packet delay depends very little on the network load and the variability of this delay is also quite low. Moreover, an oversaturated network remains stable: it just switches into a different mode of operation. Thus, although MNA configurations are inherently packet switched networks, it is conceivable to use them as backbones for implementing circuit switched protocols.

References

[1] G. Albertengo, R. Lo Cigno, and G. Panizzardi. The deflection network: A reliable high speed packet network for computer communication. In *Proceedings, Advanced Computer Technology, Reliable Systems and Applications, COMPUEURO 91*, pages 84–88, Bologna, May 1991.

[2] B. Arden and H. Lee. Analysis of chordal ring network. *IEEE Transactions on Computers*, 30(4):291–295, Apr. 1981.

[3] J. Bannister and F. Borgonovo. A procedure to evaluate the mean transport time in multibuffer deflection-routing networks with nonuniform traffic. In *Proceedings of IEEE INFOCOM'92*, pages 1069–1078, Florence, Italy, May 1992.

[4] L. Bhuyan and D. Agrawal. Generalized hypercube and hyperbus structures for a computer network. *IEEE Transactions on Computers*, 33(4):323–333, Apr. 1984.

[5] F. Borgonovo and E. Cadorin. Locally-optimal deflection routing in the bidirectional Manhattan network. In *Proceedings of IEEE INFOCOM'90*, pages 458–464, 1990.

[6] W. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 39(6):775–785, June 1990.

[7] W. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, 36(5):547–553, May 1987.

[8] S. Dandamudi. *Hierarchical Hypercube Multicomputer Interconnection Networks*. Ellis Horwood, 1991.

[9] S. Dandamudi and D. Eager. Hierarchical interconnection networks for multicomputer systems. *IEEE Transactions on Computers*, 39(6):786–797, June 1990.

[10] M. Decina, V. Trecordi, and G. Zanolini. Throughput and packet loss in deflection routing multichannel-metropolitan area networks. In *Proceedings of GLOBECOM'91*, pages 1200–1208, 1991.

[11] M. Decina, V. Trecordi, and G. Zanolini. Performance analysis of deflection routing multichannel-metropolitan area networks. In *Proceedings of IEEE INFOCOM'92*, pages 2435–2443, Florence, Italy, May 1992.

[12] K. Doty. New designs for dense processor interconnection networks. *IEEE Transactions on Computers*, 33(5):447–450, May 1984.

[13] R. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, June 1962.

[14] P. Gburzyński and P. Rudnicki. Object-oriented simulation is SMURPH: A case study of DQDB protocol. In *Proceedings of 1991 Western Multi Conference on Object-Oriented Simulation*, pages 12–21, Anaheim, California, Feb. 1991.

[15] P. Gburzyński and P. Rudnicki. *The SMURPH Protocol Modelling Environment*. University of Alberta, Department of Computing Science, Edmonton, Alberta, Canada T6G 2H1, 1991.

[16] A. Greenberg and J. Goodman. Sharp approximate models of adaptive routing in mesh networks. In O. Boxma, J. Cohen, and H. Tijms, editors, *Teletraffic Analysis and Computer Performance Evaluation*, pages 255–270. Elsevier Science Publishers B.V. (North-Holland), 1986.

[17] A. Greenberg and B. Hajek. Deflection routing in hypercube networks. *IEEE Transactions on Communications*, 40(6):1070–1081, June 1992.

[18] A. Krishna and B. Hajek. Performance of shuffle-like switching networks with deflection. In *Proceedings of IEEE INFOCOM'90*, pages 473–480, San Francisco, CA, June 1990.

[19] K. Kundert. Sparse matrix techniques. In A. Ruehli, editor, *Circuit Analysis, Simulation and Design*. North-Holland, 1986.

[20] J. Maitan and A. Harwit. A multidisciplinary approach to the development of low-cost high-performance lightwave network. In *Second NASA Space Communication Technology Conference*, Nov. 1991.

[21] J. Maitan and Z. Ras. Reconfigurable network architecture for distributed problem solving. In M. Fedrizzi, editor, *Interactive Optimization and Mathematical Programming*. Springer Verlag, 1991.

[22] J. Maitan, L. Walichiewicz, and B. Wealand. Integrated communication and information fabric for space applications. In *AIAA/NASA Second International Symposium on Space Information Systems*, pages 1175–1184, Sept. 1990.

[23] N. Maxemchuk. The Manhattan street network. In *Proceedings of GLOBECOM'85*, pages 255–261, 1985.

[24] N. Maxemchuk. Comparison of deflection and store-and-forward techniques in Manhattan-street network and shuffle-exchange networks. In *Proceedings of IEEE INFOCOM'89*, pages 800–809, 1989.

[25] N. Maxemchuk. Problems arising from deflection routing. In Pugolle, editor, *High Capacity Local and Metropolitan Networks*, pages 209–233. Springer Verlag, 1991.

[26] A. Pach, S. Palazzo, and D. Panno. Improving DQDB throughput by a slot preuse technique. In *ICC'92*, Chicago, USA, June 1992.

[27] C. Seitz. The cosmic cube. *Communications of the ACM*, 28(1):22–33, Jan. 1985.

[28] L. Wittie. Communication structures for large networks of microcomputers. *IEEE Transactions on Computers*, 30(4):254–273, Apr. 1981.

Figures

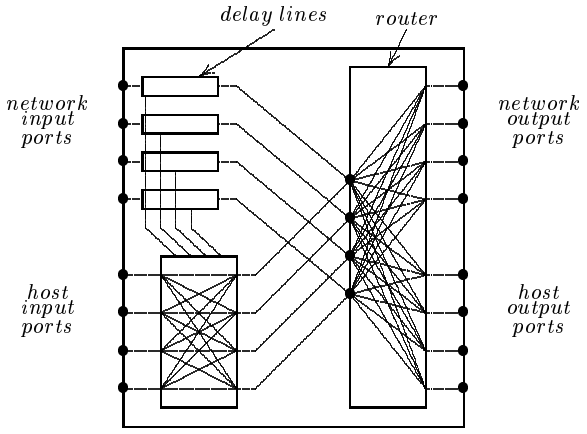

Figure 1: The structure of an MNA switch ($k_n = k_h = 4$).

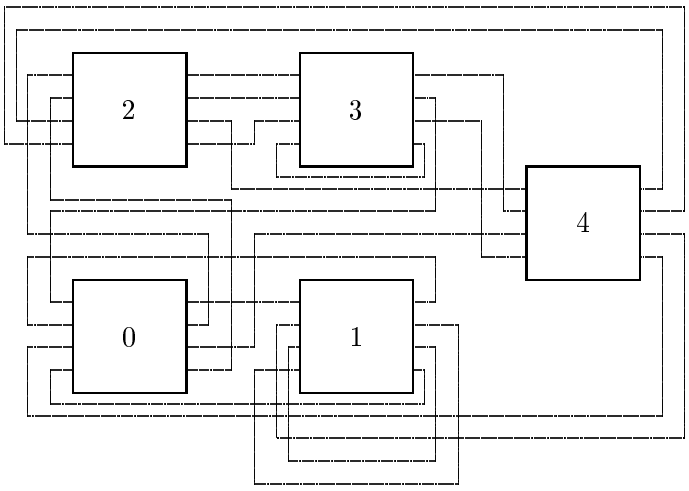

Figure 2: A sample MNA network.

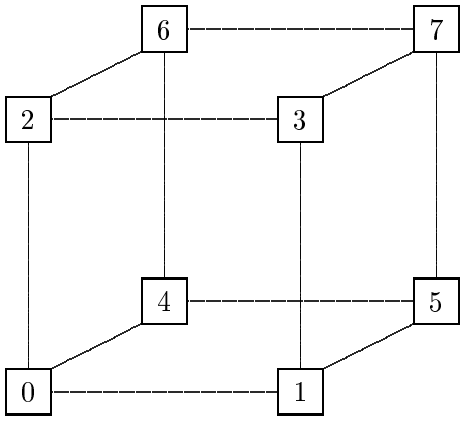

Figure 3: A hypercube network with 8 switches.

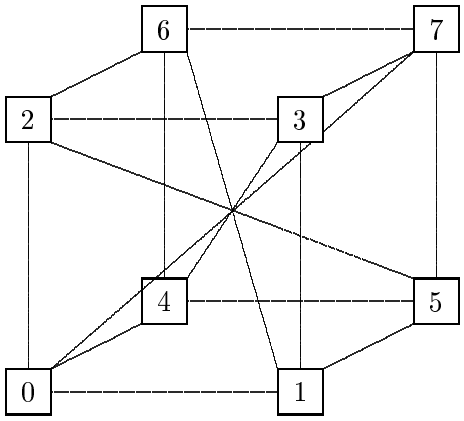

Figure 4: A diagonal hypercube with 8 switches.

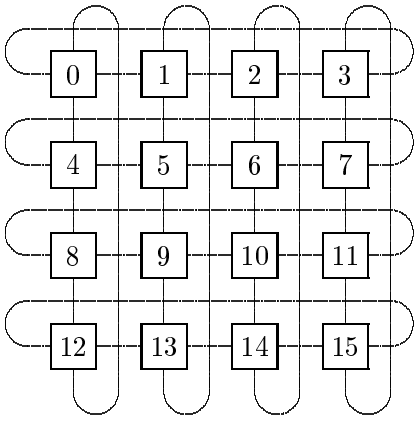

Figure 5: A two-dimensional torus with four connections per switch.

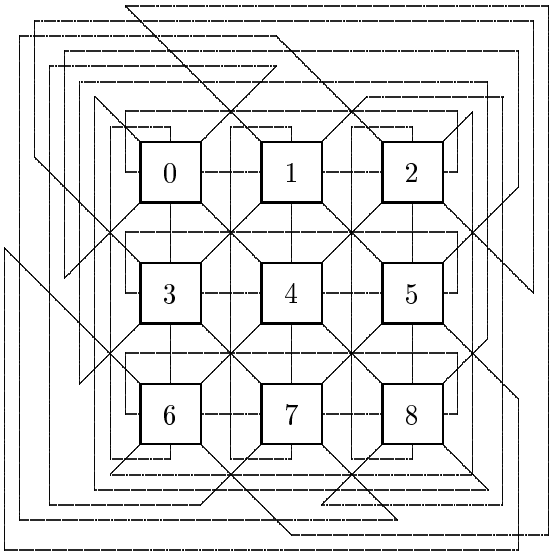

Figure 6: A two-dimensional diagonal torus with eight connections per switch.

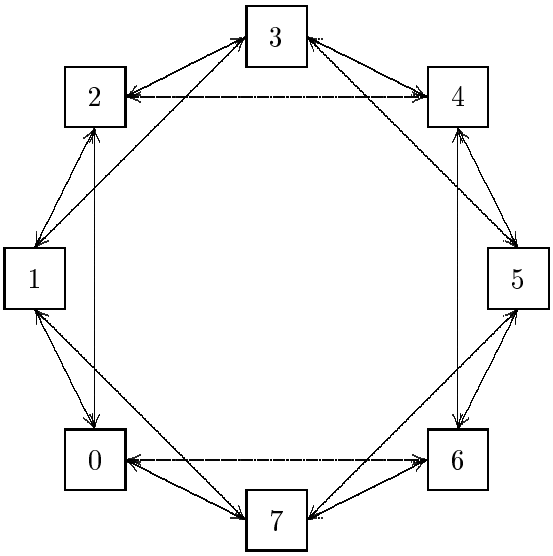

Figure 7: A chordal ring with 8 switches ($k_n = 4$). Each double arrow represents two separate links.

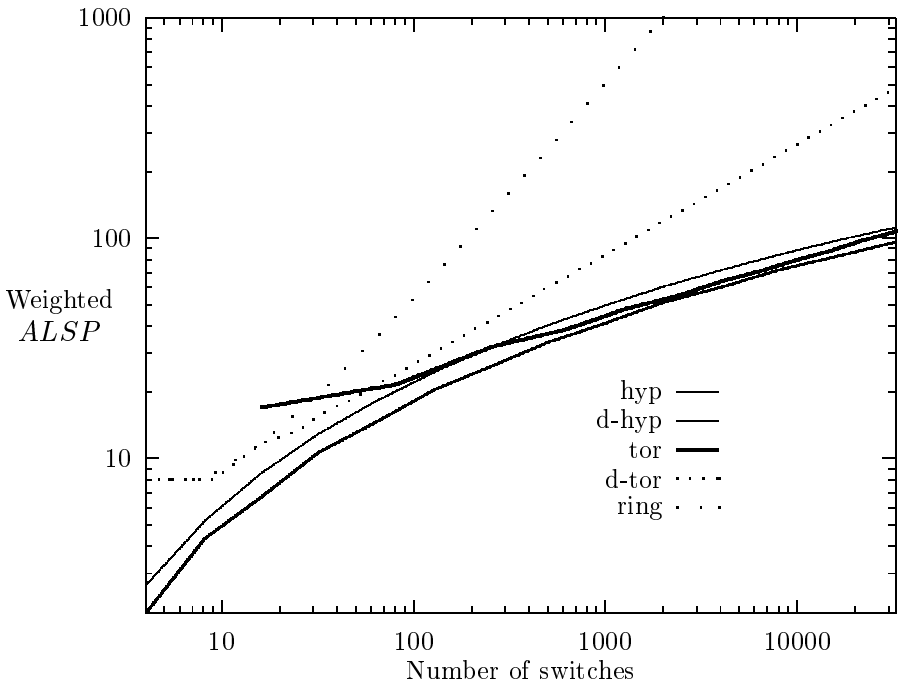

Figure 8: Weighted $ALSP$ versus network size.

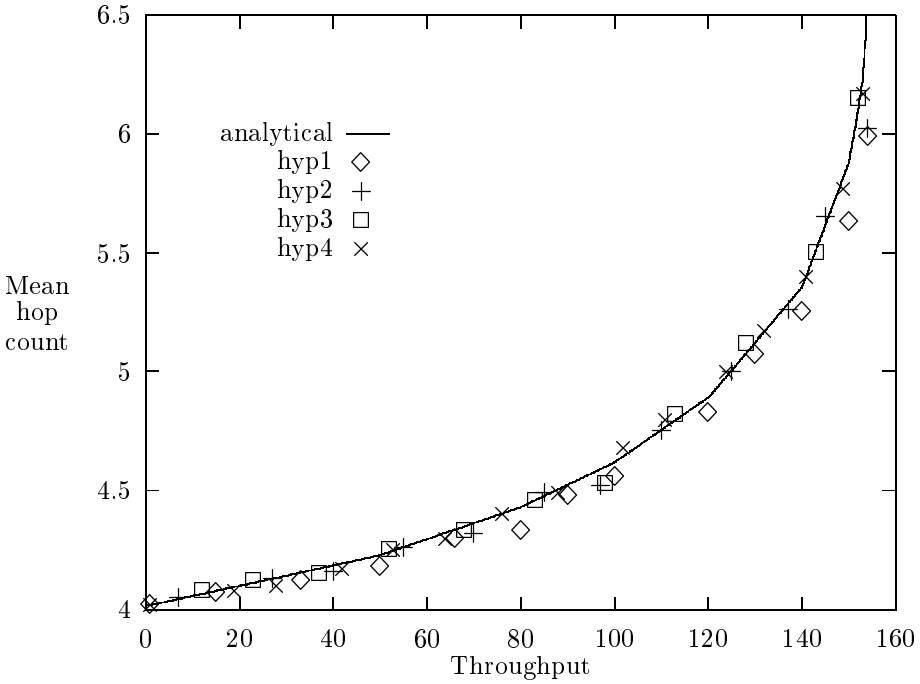

Figure 9: Performance of regular hypercube with 256 switches.

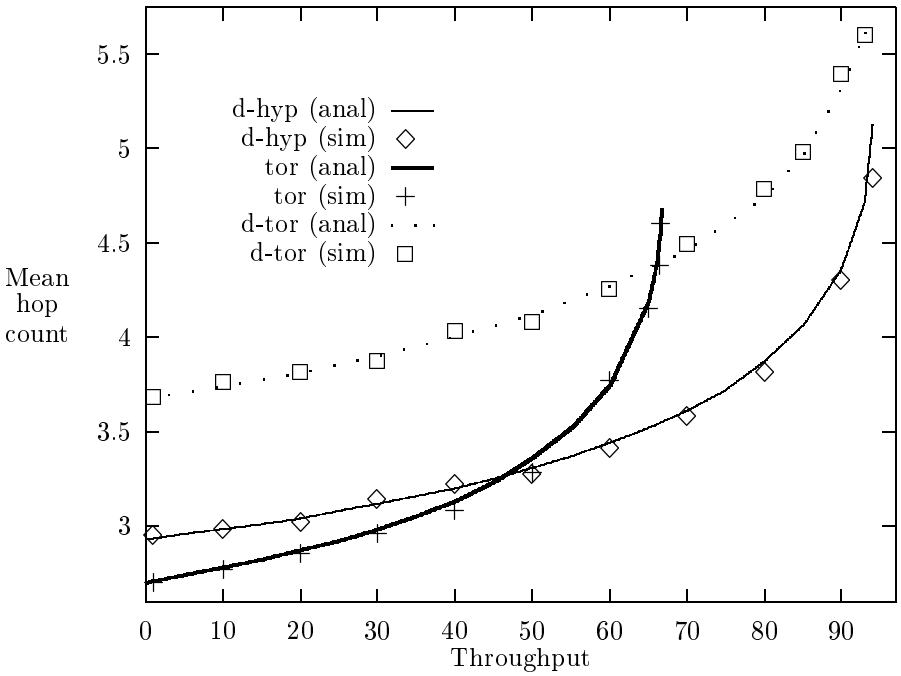

Figure 10: Performance of three reference networks: diagonal hypercube with 128 switches (d-hyp), 4-D torus with 81 switches (tor), and 2-D torus (with diagonals) with 121 switches (d-tor).

Load	P1	P2	P3	P4
1.0	0.0059	0.0208	0.0743	0.8990
10.0	0.0394	0.0670	0.1536	0.7400
20.0	0.0802	0.1188	0.2230	0.5780
30.0	0.1235	0.1645	0.2570	0.4550
40.0	0.1742	0.2038	0.2660	0.3560
45.0	0.2150	0.2300	0.2580	0.2970
50.0	0.2310	0.2390	0.2540	0.2760

Table 1: Relative port occupancy rate for the chordal ring with 128 switches.

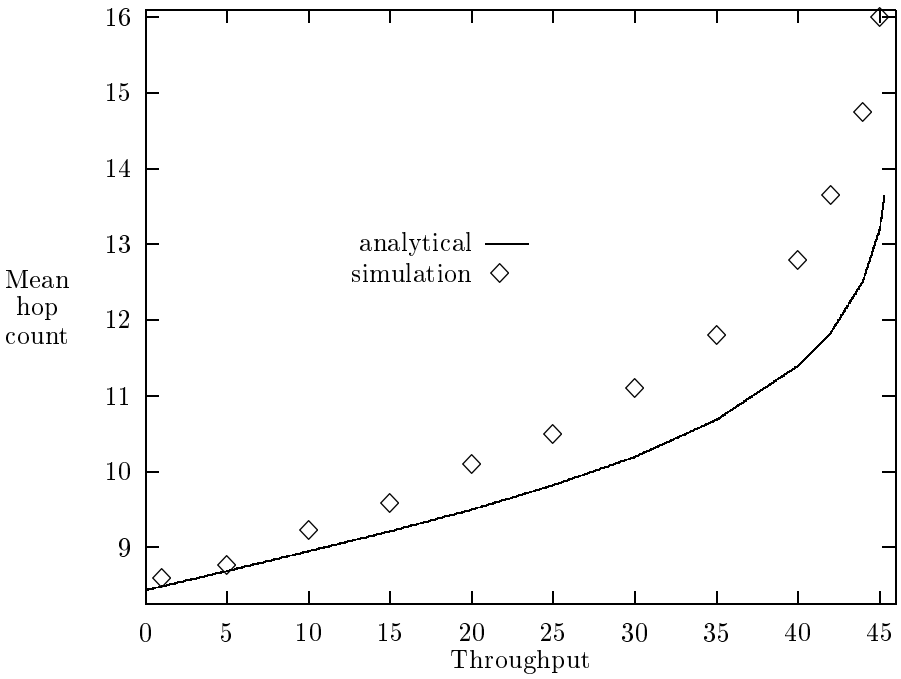

Figure 11: Performance of chordal ring with 128 switches.

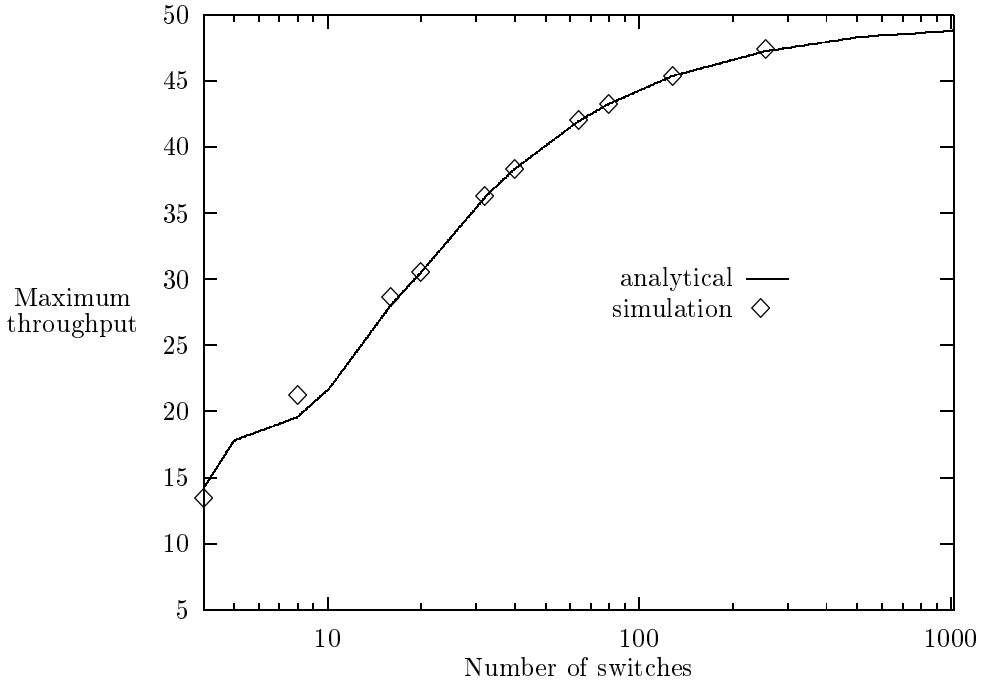

Figure 12: Maximum throughput versus network size for chordal ring.

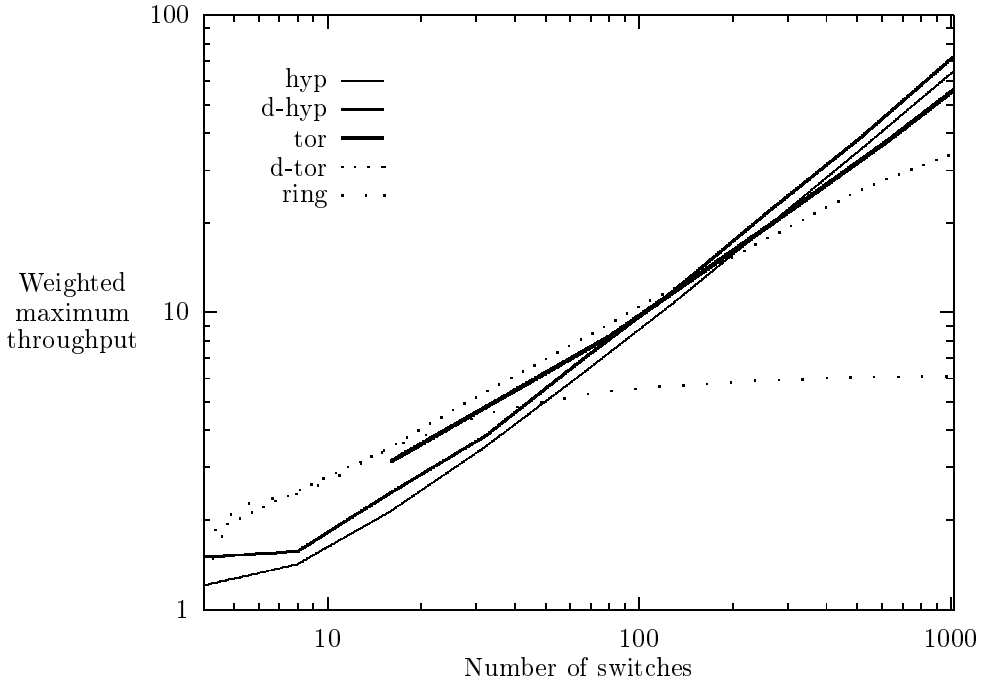

Figure 13: Weighted maximum throughput versus network size.

Network	Predicted			Observed		
	λ	ϱ	$\mathcal{P}$	λ	ϱ	$\mathcal{P}$
Regular hypercube (256 switches)	111	13.4	0.73	110	13.6	0.73
Diagonal hypercube (128 switches)	68	10.7	0.71	68	10.7	0.71
Planar torus (121 switches)	87	7.8	0.70	88	7.5	0.68
4-D torus (81 switches)	55	8.3	0.70	55	8.1	0.69
Chordal ring (128 switches)	43	18	0.76	42	18	0.74

Table 2: Saturation points of the reference networks.

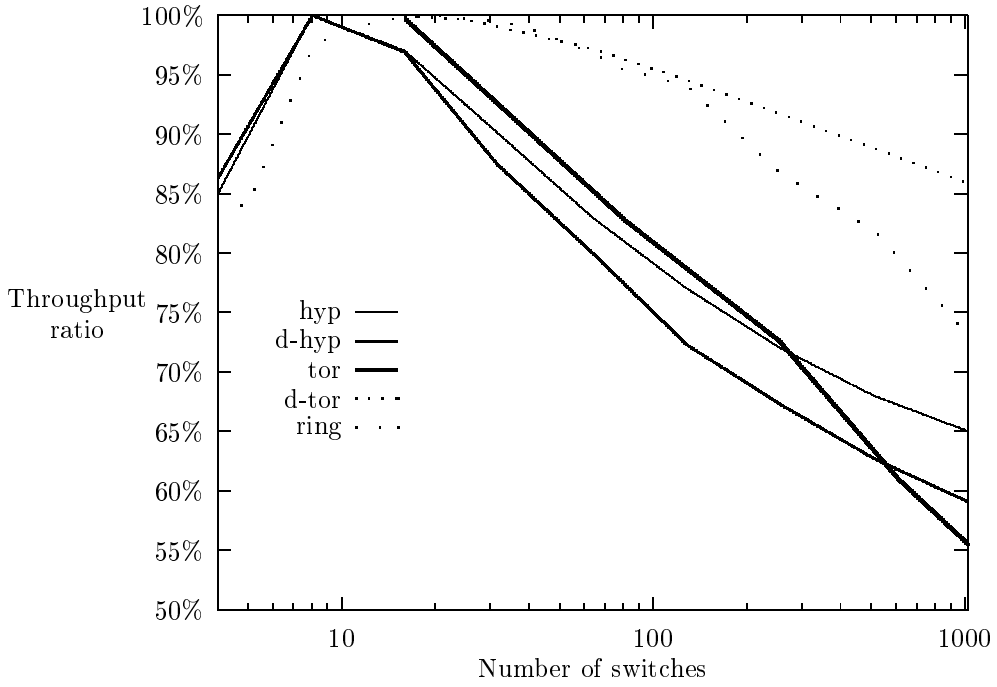

Figure 14: Saturation throughput as the percentage of maximum throughput versus network size.

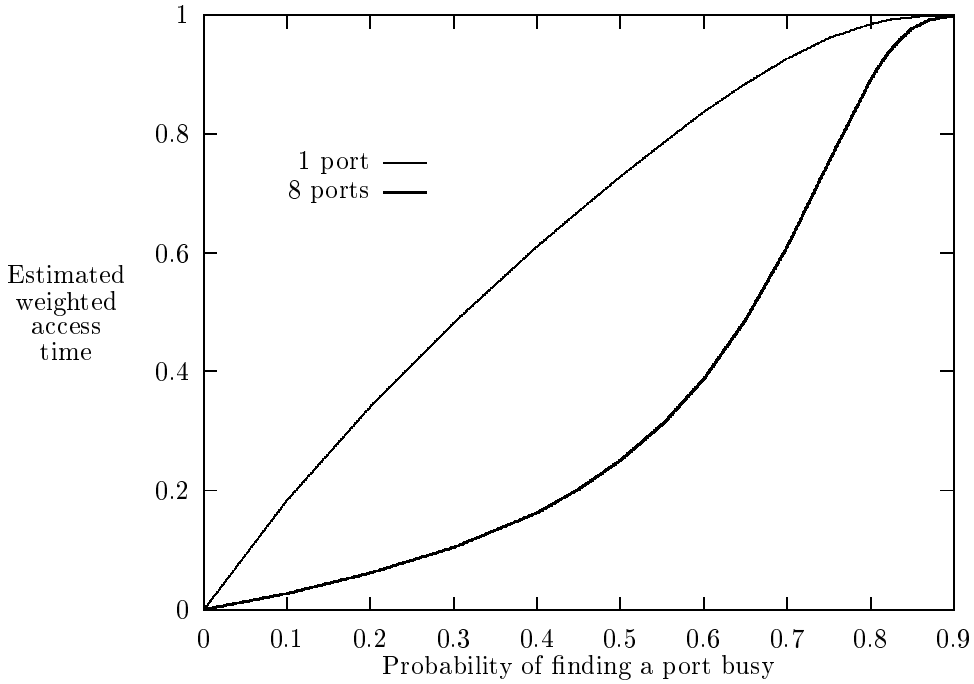

Figure 15: Estimated weighted packet access time as a function of P.

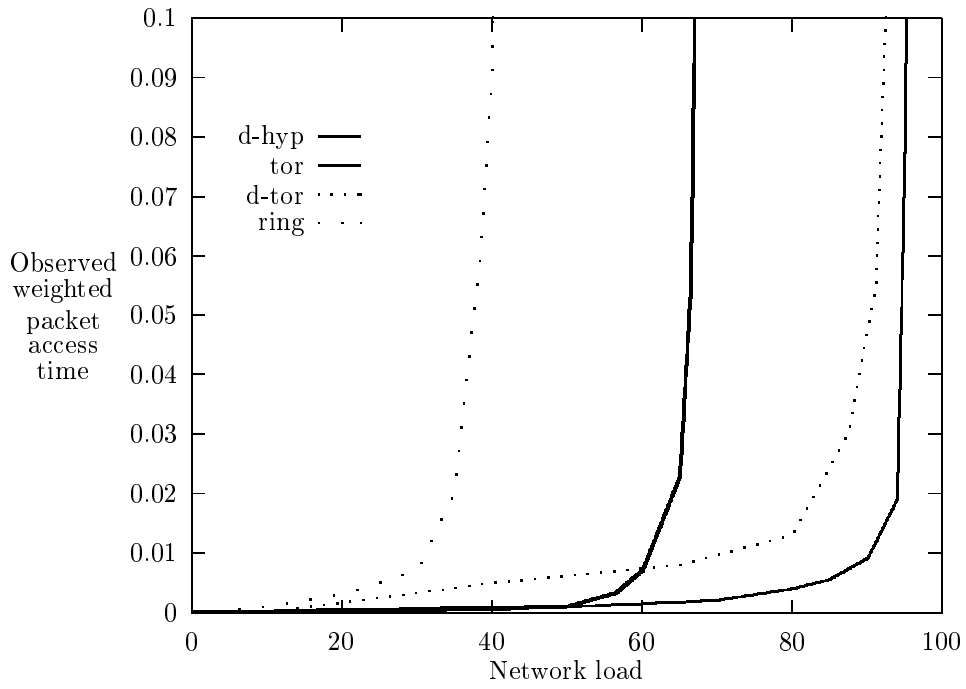

Figure 16: Observed weighted access time.

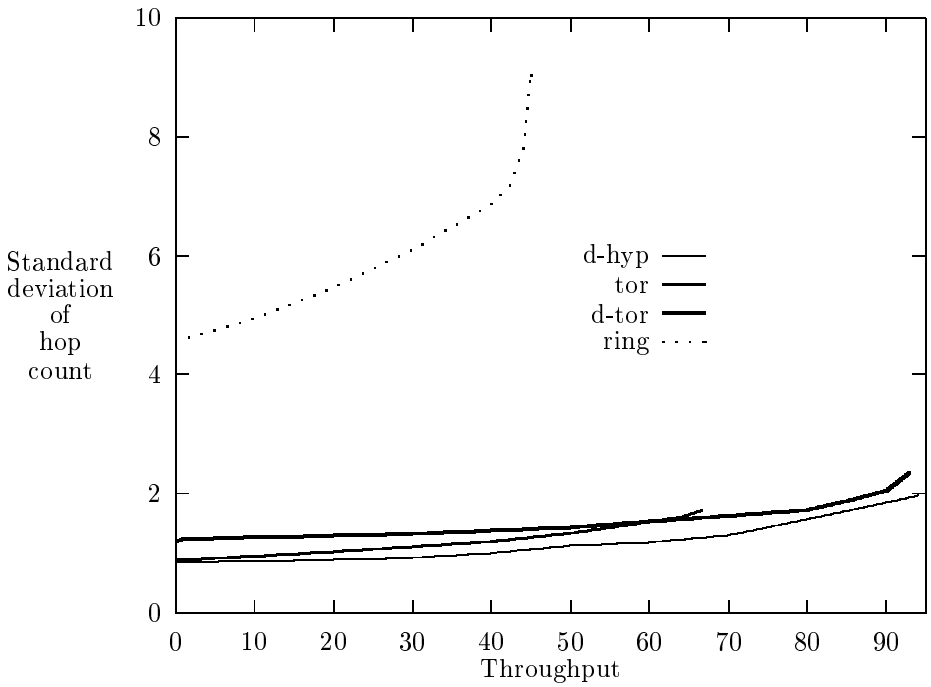

Figure 17: Standard deviation of packet hop count for diagonal hypercube and chordal ring (128 switches).