Routing in Multihop Packet Switching Networks: Gb/s Challenge

The authors survey networking solutions that have been proposed for high-speed packet-switched applications. Using these solutions as examples, they identify the specific problems resulting from very high transmission rates and explain how these problems influence the design of high-speed networks and protocols. They conclude that the solutions based on deflection routing are the most promising ones and suggest a number of directions for their evolution.

Cesur Baransel, Wlodek Dobosiewicz, and Pawel Gburzynski

Not so long ago, computer networks with high transmission rates (e.g., several Mb/s) were naturally confined to local domains. Although such (and higher) transmission rates were available in telephony on long distances, they were used on a point-to-point basis. Concepts of highly-connected fast networks spanning geographical areas larger than the acreage typically covered by a single institution are relatively new and, besides the emerging ATM technology, there are no standard commercially available solutions that can be recommended for such projects.

Most of the legacy from local area networking is not easily adaptable to larger scale networks and/or networks with transmission rates substantially higher than several Mb/s. Consider FDDI as an example. The network operates at 100 Mb/s and is intended to provide campus-area service. The formula expressing the maximum effective throughput of FDDI can be roughly written as follows:

$$T = \frac{THT}{THT + L}$$

where THT is the maximum token-holding time during one rotation of the token, and L is the propagation time across the ring. One obvious property of T is that it doesn't grow as more stations are added to the network. Consequently, the more stations the network supports the smaller fraction of T is allocated to one station. In fact, FDDI assumes that at most one pair of stations can communicate at any given moment. Besides, due to the bandwidth allocation policy of FDDI, more stations means larger access delays. If the network is expanded geographically (even without increasing the number of stations), L will grow reducing the effective throughput in a linear fashion. The same phenomenon will occur, if somebody tries to extrapolate the FDDI concept onto transmission rates substantially higher than 100 Mb/s, e.g., into the Gb/s range. Thus, we have to conclude that the network doesn't scale up very well: its principle of operation has inherent limitations which restrict its applicability to a relatively narrow range of transmission rates, geographical areas, and populations of users.

All unidimensional networks, e.g., busses, rings, and stars, are bound to suffer from poor scalability to the increasing number of users. If all of a network's transmission resources (or a fixed large fraction of these resources) must be reserved for every single transfer, the network cannot cater to a large population of users. Most painfully, it cannot take advantage of the localized communities of interest which naturally occur in any large population.[1] Besides, the need to negotiate medium access across the entire network results in a throughput deterioration when the network diameter[2] is large. Networks that avoid this problem (e.g., Metaring [3]) suffer from poor fairness and/or starvation potential. Various protocol add-ons aimed at alleviating these problems are either partially successful (e.g., they exhibit slow responsiveness to dynamic unfairness patterns) or they tend to sacrifice throughput to achieve their objective.

The moral from the above observations is that the future of high-speed networking, at least beyond the local area scale, lies with meshed networks. A two-dimensional (planar) mesh network with N stations is potentially able to achieve a global throughput proportional to $\sqrt{N}$. By exploring other dimensions, this figure can be asymptotically brought as close to N as required. Mesh architectures also tend to be organized

CESUR BARANSEL is with the Department of Computing Science at the Turkish Military Academy, Ankara.

WLODEK DOBOSIEWICZ is with the Department of Computing Science at Monmouth University.

PAWEL GBURZYNSKI is with the Department of Computing Science at the University of Alberta.

[1] Attempts to accommodate communities of interest in some bus networks, e.g., DQDB [1], have been only partially successful.

[2] Expressed in the normalized way—as the number of bits separating two most distant stations in the network [2].

0890-8044/95/$04.00 1995 © IEEE

according to the geographical distribution of the interconnected nodes; thus, they may naturally take advantage of the *communities of interest* to further improve their performance.

The existence of multiple paths between nodes complicates the communication protocols by introducing *packet switching* — the very issue that the unidimensional networks were meant to avoid. General packet-switching and flow-control techniques traditionally employed in slow long-haul networks are usually inapplicable to gigabit networking. To take full advantage of the high transmission rate of its channels, a gigabit packet-switching network cannot spend too much time on making sophisticated routing decisions. Also, it should avoid buffering transient packets at intermediate nodes for extensive periods of time. These postulates stem not only from the natural need to avoid unnecessary delays, which are magnified by the high transmission rates, but also from the need to make these delays predictable. Modern networks are expected to handle traffic patterns of various kinds, and delay-sensitive or delay-variation-sensitive patterns will constitute a substantial part of their load.

In this article, we conduct a survey of packet-switching protocols applicable to meshed networks operating in the Gb/s range. By a packet-switching protocol we mean the network-specific portion of the third OSI layer (i.e., the network layer) of the protocol stack. The full set of packet-switching rules determines how the network organizes reliable packet delivery between a pair of communicating switches, possibly located in distant geographical regions of the communication subnet.[3] One part of a packet-switching protocol (according to our definition) is the routing scheme, i.e., the set of rules that assign incoming packets to output links. In general, we can talk about the following three components of the communication subnet which are relevant from our point of view:

- The routing protocol.
- The congestion-control mechanisms that can be effectively incorporated into the routing protocol.
- The network topology.

Clearly, these components are not independent, but they are closely related to each other and together offer a single functionality. Reflecting this relationship, this article is organized as follows: we first discuss routing protocols and congestion-control mechanisms employed in contemporary packet-switched networks, not necessarily in networks operating at very high transmission rates. Then, following some basic definitions related to the topology component, we investigate the challenges posed by the Gb/s transmission rates. A later section is devoted to case studies which cover a number of contemporary design proposals. Conclusions and suggestions for further research are presented in the last section.

Routing Protocols

In a multihop packet-switching network, the function of the routing algorithm is to guide the packets through the communication subnet to their proper destinations. Different taxonomies of routing algorithms are possible, e.g., *static* vs. *adaptive* or *centralized* vs. *distributed*. Here we prefer to put them into two groups, namely, *table-based routing* and *self-routing*.[4] The first class covers most of the traditional approaches that have been applied to many slow networks, including *shortest path routing* and *optimal routing*. Aside from other problems, as convergence delays proportional to the network diameter and susceptibility to oscillations, these algorithms are computationally expensive and require a substantial amount of bookkeeping and periodic transmission of status information among the nodes. In the case of self-routing, the routing decision is solely made based on information extracted from the packet's header, typically the destination address. Most of the multiprocessor interconnection networks use this scheme (e.g., shuffle-exchange networks, hypercubes, data manipulator networks, Benes networks, and Clos networks). Another well-known example that can be included in this category is *flooding*. This classification groups the routing algorithms according to the complexity of the routing decisions and, consequently, to the speed at which packet switching can be carried out. Another important factor affecting the switching speed is the organization of buffers for storing transient packets. When there are no buffers at all, purely photonic switching becomes feasible, and E/O and O/E (electronic-to-optic and optic-to-electronic) conversions can be avoided. However, the photonic switching technology is still in its infancy and for the time being is not a practical alternative to its electronic counterpart (see [4]).

The photonic switching technology is still in its infancy and for the time being is not a practical alternative to its electronic counterpart.

Table-Based Routing

In this category of routing schemes, upon a packet arrival at an intermediate node, the node consults a table to select the outgoing link on which the packet is to be forwarded. Although the location of the routing tables, the way they are maintained, and the information contained within them may differ from one implementation to another, some common characteristics are shared by all solutions that fit into this class:

- In most applications, the routing table contains an entry for every destination, indicating the output link appropriate for a packet addressed to that destination. Therefore, the table size increases with the network size and can be large for a network with many nodes.
- For practical reasons (e.g., to cope with congestion and to bypass faulty links or nodes), entries in the routing tables need to be updated. Therefore, some network capacity must be allocated to the extra traffic that disseminates status information, reducing the capacity available to the users.

The problem of large routing tables in networks consisting of a huge number of stations can be alleviated by introducing domains, each domain representing a cluster of closely-located stations which appear (almost) equally distant from a sufficiently remote location. From the point of view of such a location, it may be appropriate to route all packets addressed to the domain in the same way, effectively treating the domain

[3] *According to the OSI terminology.*

[4] *Also known as header routing.*

as if it were a single station. This approach requires a hierarchical structure of the destination address. Otherwise, lookup tables are needed to identify the domains, which may reduce or even nullify the potential savings.

Shortest-path Routing — The basic premise of a routing scheme from this class is to have at every switch a unique mapping of the destinations to the output links. Given a destination address extracted from the header of an incoming packet, the switch selects the output link that offers the "shortest path" to the destination. The notion of length may be static, i.e., it may reflect the propagation distance, the number of hops (intermediate switches), and the nominal link capacities, but it may also account for the perceived congestion level of the links. In the latter case, the shortest paths are determined dynamically and the routing algorithm can adapt itself to variable traffic conditions.

Regardless of the criteria determining the path length used in selecting the output link suitable for a given destination, all shortest-path schemes are essentially based on global topological knowledge of the network. This knowledge is represented by the list of all nodes and their interconnections (the network graph) with a cost assigned to every link [5]. One can see both centralized (as in TYMNET) or distributed (as in ARPANET) techniques of representing this knowledge; however, every node must have access to its locally and momentarily relevant portion to be able to make routing decisions. TYMNET uses virtual circuits and the shortest path calculations are performed by a special node called the *supervisor*. The supervisor also decides upon the path to be used by a virtual circuit. The intermediate nodes are notified about the path by a *needle* packet that travels from the source to the destination threading the virtual circuit along its way, with the data packets trailing behind. The shortest-path calculation algorithm used by the supervisor is a modified version of Floyd's algorithm [5].

In contrast, ARPANET employs a distributed approach in which every node maintains its own data base and carries out the shortest path calculations taking itself as the source. The original algorithm, based on the Bellman-Ford method, was implemented in 1969.[5] It has been modified twice since then, in 1979 and 1987, due to problems caused by oscillations. The latest modification was warranted by the increased traffic load, which once again lead to severe oscillations. The latest algorithm is still prone to oscillations, but not nearly as much as the first one [6]. The details of these algorithms can be found in [7].

The common drawback of all shortest-path algorithms is the use of only one path per source-destination pair and their poor adaptability to abrupt traffic shifts, which is further limited by their inherent susceptibility to oscillations [6].

Optimal Routing — Optimal routing is based on the theory of optimal multi-commodity flows. Assume that $Z_{ij}(r)$ is a function that gives the cost of transmitting data at rate r (which may be viewed as the percentage of the link utilization) through link ij. Now, the routing problem can be viewed as an optimization problem: the routing decisions should minimize the cost of resolving the offered load.[6] Most commonly used cost functions are related to link capacities and the amount of traffic carried by each link which is viewed as a *flow*.

The basic goal, i.e., optimal routing, is not always attainable solely by optimizing the average levels of link traffic. Theoretically, there exist more effective alternatives (e.g., ones that take queue lengths into consideration as well), but they are impractical due to the overhead and large delays involved in the exchange of the queue length information among the nodes [11].

For example, optimal routing in the CODEX network is based on the following cost function:

$$Z_{ij}(f_{ij}) = \frac{f_{ij}}{c_{ij} - f_{ij}} + d_{ij} f_{ij} \qquad (1)$$

where c_{ij} is the link capacity, f_{ij} is the data rate of the link and d_{ij} is the processing and propagation delay. CODEX uses virtual circuits for user traffic and datagrams for its own system messages. Every node monitors some parameters of its adjacent links and periodically broadcasts them to all other nodes. The above formula applies to the case when all links are of the same priority. For multiple priorities and other details see [11] and the references in that book.

Self Routing

In this class we put all routing techniques that either avoid routing tables completely or use static, possibly incomplete, routing tables which are seldom (or never) updated during the normal operation of the network. With self routing, a switch accepting an incoming packet is able to determine its fate locally without consulting the network's data base in its centralized or distributed form. The price paid for the simplicity of the routing algorithm is its suboptimal character. The gain is in the low cost of routing, the simplicity of the switch, and the absence of administrative traffic in the network.

Networks with Regular Topologies — If the network forms a regular grid with a simple repetitive structure, then every switch may be able to *de-facto* "know" the configuration of the entire network without resorting to a data structure describing the individual locations of all stations. Networks with highly regular topologies commonly occur as interconnection backplanes for multiprocessor systems.[7] Interconnection networks can be constructed from a single stage of switches or from multiple stages of switches. In a single-stage network, packets may have to pass through the switches several times before reaching their destinations. Therefore, single-stage networks are sometimes called recirculating networks and the subsequent passes of the same packet through the stage of switches are called recirculations. The number of recirculations depends on the connectivity. Generally, the higher the connectivity the smaller the number of recirculations. Typically, in a multi-stage network, one pass through the multiple stages of switches is sufficient to deliver a packet to its destination

[5] In that version, the nodes exchanged their estimated shortest distances to every destination every 625 ms.

[6] Provided that the cost function is sufficiently differentiable, it can be expanded as a Taylor series. If the first derivatives exist, then at local minima the Jacobian gradient vector has all elements zero. If the second derivatives exist, the Hessian is positive definite at the minimum. For convex functions, the local minima are also global. The gradient methods are based on the Taylor series expansion. Optimization methods which use only Jacobian gradient vector are termed first-order methods. If the optimization method utilizes second derivatives as well, it is called as a second-order method. The steepest descent method uses Jacobian gradient to determine a suitable direction of movement and is the fundamental first order method. All in all, the appropriate choice of the cost function greatly simplifies the optimization process. For details, see [8-10].

[7] Many internal designs for ATM switches are based on the interconnection paradigm, i.e., the switch is treated as a network of specialized processors.

[12]. A survey of switching techniques in high-speed interconnection networks can be found in [13, 14].

On a larger geographical scale, the regularity in the network topology is taken advantage of in the Manhattan-street network (MSN), which is a single-stage solution.[8] Although originally proposed to cover metropolitan areas, MSNs can also be used as interconnection networks.

Interconnection networks have been studied extensively in the literature, mostly owing to their applications in distributed computing. Several books are available on this subject, e.g., [15-18] as well as survey papers [12, 19-21]. MSNs were introduced by Maxemchuk [22] and since then they have been extensively studied by Maxemchuk [23-25] and other authors [26, 27]. Later, we discuss hypercubes, shuffle networks, and MSNs.

Flooding Networks — Another simplistic approach to routing is flooding which, in its purest sense, means that an incoming packet is forwarded on every outgoing link except the one it arrived on [11]. The outstanding qualities of flooding can be summarized as follows:

- The approach is highly robust, in case of link failures. As long as the network graph is not disconnected, packets always make it to their destinations. If the network is richly connected, flooding makes excellent use of alternative routes.
- Error recovery at the destination is simplified by the availability of extra copies of the same packet.
- No routing tables (or other data structures representing the network configuration) are required. Thus, network modifications can be made on a live network.
- The scheme is suitable for all topologies, possibly very irregular ones. Consequently, the network is easily expandable.
- Flooding automatically chooses the shortest path (since it chooses every possible path in parallel).
- It is simple to implement and introduces less processing overhead than any other routing scheme.

The most important weakness of flooding is that packets may loop and, as a result, unlimited numbers of copies of a single packet can crop up in the network. Therefore, some countermeasures to choke this process are necessary for the approach to be useful. In general, flooding is considered to be more useful in broadcasting rather than one-to-one communication. Even in networks based on sophisticated routing methods, flooding is occasionally used as a simple broadcasting technique, e.g., to disseminate various components of the network data base among individual stations. ARPANET uses flooding to broadcast periodic status information to the nodes. In a later section, we will discuss some flooding-based designs.

Congestion Control

Congestion is the network state where, because of mismanagement (e.g., improper access and routing), excessive requests, or faults, the demand for resources exceeds their availability [28]. When this happens, the queues at bottleneck nodes grow indefinitely and eventually exceed the available buffer space. Consequently, some packets will have to be discarded and later retransmitted, thereby wasting communication resources and feeding back the congestion. It is thus necessary to prevent excess traffic from entering the network.

A special (local) case of congestion is when due to a speed disparity and/or temporary unavailability of resources, one receiver cannot accept the incoming flow. Should this happen, the sender must be made aware of the situation as soon as possible and either adjust its speed or abstain from further transmissions which are bound to be rejected.

Congestion control is a dynamic problem and cannot be solved with static mechanisms alone. It is also a difficult problem to solve due to the following requirements that must be fulfilled by a good solution [29]:

- The scheme must have a low overhead and should not offer new traffic to the network during congestion.
- The scheme must be fair so that during the congestion the available resources are allocated fairly.[9]
- The method must be responsive. Due to the highly dynamic nature of the network, the resource availability profile changes very rapidly. The congestion-control procedure should be agile enough so that the demand curve can follow the capacity curve very closely.
- The procedure must be robust so that it can function effectively under unfavorable conditions (e.g., poor availability of network resources at the time of congestion).
- The scheme must be socially optimal. That is, it should optimize the performance of the entire network, as opposed to considering each user in isolation.

No single classification of congestion-control techniques will satisfy everybody, as the criteria that can be used for classification are often orthogonal and reflect the point of view of the researcher. One can naturally consider the following attributes of a congestion-control scheme:

- Preventive vs. reactive character of the method — Preventive schemes try to avoid congestion and reactive ones try do something about it, once it occurs.
- The OSI layer in which the scheme is implemented — for example, schemes operating in the transport layer involve the end-points of a data path and are global in nature, whereas data-link schemes take care of local congestion, e.g., resulting from incompatible transmission rates of two immediate neighbors.
- Feedback-based vs. feedback-free character of the scheme — generally, the attractiveness of feedback-based techniques decreases with the increasing transmission rate of the network and/or its geographical size. Feedback-free schemes are usually rate-based, i.e., they try to allocate some portion of the network's global rate to every data path and confine each sender to this portion.
- Guaranteed-delivery schemes vs. schemes that admit packet loss during congestion.

Even in networks based on sophisticated routing methods, flooding is occasionally used as a simple broadcasting technique.

[8] *In a congested Manhattan-street network, a packet may visit the same switch several times; however, multiple visits at the same switch are never necessary for a successful packet delivery.*

[9] *It should be pointed out that although several formal and precise definitions of fairness can be found in the literature, none in particular is widely inferred from the general term.*

Window-based flow-control schemes are very slow to adapt to changing load patterns and are only effective for congestion scenarios that last for several round-trip delays.

Window-Based Schemes

Consider the class of window-based schemes [29] which require the recipient to adjust the window size[10] of the sender by sending feedback signals. According to the above classification, window-based control techniques are reactive in nature and based on feedback. However, they can be implemented in any of the three relevant OSI layers,[11] be loss-less (if the amount of reserved buffer space accounts for the maximum feedback delay), or occasionally lose packets and force their retransmission. Window-based schemes have been used in a number of slow networks, ARPANET, TYMNET, SNA, and CODEX, to name a few. In networks that are not very fast, this approach is particularly popular and effective for preventing local congestion (in the sense mentioned above), i.e., for matching the source speed to the processing speed of the destination in a point-to-point scenario (e.g., in the data-link layer).

The applicability of window-based flow-control schemes to high-speed networks is addressed in a number of references [30, 31]. The problems associated with this approach mostly stem from the large normalized propagation delays across the network, which render the feedback information obsolete and useless. Window-based schemes are also very slow to adapt to changing load patterns and they are only effective for congestion scenarios that last for several round-trip delays.

Acceptance-Level Schemes

In high-speed networks, flow-control mechanisms have to be more preventive than reactive because, in order to react, the involved parties must exchange status information across the large normalized diameter of the network. Most of the contemporary designs or proposals[12] prevent congestion by exercising flow control at two levels. First, at the circuit-setup level (call-acceptance level, according to the ATM terminology) whether the new data stream can be accommodated within the network, considering its present load, is checked. According to our classification, such schemes operate in the transport layer and their primary role is to *prevent* excess traffic from entering the network, thus avoiding long-term congestion. The user is requested to submit indicators specifying the extent and quality of service demanded from the network. Typical examples are the declaration of the peak rate, the minimum throughput demanded in case of congestion, or some parameters describing the burstiness of the traffic (e.g., peak rate, average rate, and maximum burst size [32]). Regardless of the specific details of the individual designs, the bottom line is the necessity for the user to have a "contract" with the network before being able to proceed with the transmission. After the call/session/flow has been accepted by the network (using ATM terminology, we will say that the virtual path has been established), the responsibility of monitoring the user's adherence to the declared parameters is carried out at the packet (cell) level. Owing to the statistical and essentially unpredictable nature of the data flow, this task is anything but trivial. The bulk of all research in ATM networks is currently devoted to admission control and bandwidth allocation.

Grades of Service

In the face of the fact that the proportion of multimedia traffic in contemporary networks is already significant, and is going to increase substantially in the near future, it makes sense to consider the concept of "service grade" in the context of congestion-control policies. A multimedia application may be willing to accept a reduced bandwidth for its connection (and operate at a lower quality of service) rather than receive no service at all based on the original quality specification. For example, if there is no bandwidth at the moment to accept a videophone call, the customer may downgrade the request to a regular voice connection. But even within the domain of video traffic (e.g., tele-conferencing), one can think of several grades of service lower grades offering lower quality of the picture. The grade-of-service approach applies both at the connection-setup level and at the level of allocating bandwidth for individual packets (cells) relayed by a switch. A call can be admitted at a high quality of service and later downgraded, if the switch cannot deliver the high-quality service due to congestion. The user's contract with the network becomes now flexible, within the limits of user's willingness to put up with service deterioration. The algorithm for allocating bandwidth to multiple data streams handled by the switch at any given moment must take into account the possibility of reducing the effective bandwidth assigned to flexible connections. This complicates the optimization problem to be solved by the switch [33]. Note that a reduction in the quality of service implies a reduction in the cost of the connection both in terms of network resources and the charge to the user's account. Besides offering a reasonable service to its customers the network should also maximize its revenue; the optimization problem must be parameterized by both cost components. The issue of flexible bandwidth allocation along the lines suggested above was investigated to some extent in [34] and [35]. In [33], the problem is analyzed in depth and several allocation policies are proposed and compared.

Rate-Based Schemes

Current trends in preventive flow control seem to be towards rate-based mechanisms [29]. A well known rate-based control technique is the *leaky bucket* scheme [36]. It is a mechanism for policing the negotiated transmission rate, which is translated into the size of a virtual bucket allocated to the session. This bucket is filled by incoming packets, which may arrive spontaneously, and is emptied (leaks) at the negotiated constant rate.[13] Packets arriving when the bucket is full are discarded. Leaky bucket is basically a packet policing scheme that exercises control at an entry point to the network. Note that its virtue is not in guaranteeing a lossless connection at the negotiated rate, but rather in a simple means of enforcing the negotiated rate and indicating (and eliminating) the packets that violate a user's contract with the network.

[10] *The number of packets that can be outstanding in the network at a time.*

[11] *i.e., data-link, network, and transport.*

[12] *Including those aimed at ATM networks.*

According to our classification, leaky bucket is a preventive technique, operates in the transport layer (buckets are allocated on a per-connection basis), is not based on feedback, and admits packet loss. Various forms of the leaky bucket scheme have been proposed in the literature. One variation of this technique [31] deals with two categories of packets which are marked by the source as green or red. Green packets are transmitted at the rate negotiated during the call setup. Red packets represent the rate in excess of the contract and are handled differently at the intermediate nodes, according to the availability of resources. In particular, they can be dropped in case of congestion. The basic idea is to convey more important or loss-sensitive data using green packets.

Another rate-based control technique, dubbed *virtual clock*, was proposed as part of the *Flow Network* design [37]. The network extends guaranteed service to its users, but the users are expected to specify their bandwidth requirements. A bandwidth specification consists of two components: the average rate at which packets will be submitted to the network and the time interval over which the measured average should match the declared value. The technique attempts to model time-division multiplexing (TDM) of the network resources among the multiple data paths (dubbed *flows* in the design), but, unlike the traditional TDM schemes, it accounts for the variability of packet arrivals within each path. Therefore, the multiplexing is carried out based on *virtual time*, in which the arriving packets are assumed to be equally spaced. The scheme guarantees loss-less packet delivery as long as the observed average data rate of a flow measured over the declared interval does not exceed the value specified when the connection was set up. A flow violating its contract receives poor service and its packets are either placed at the end of the service queues or dropped. Functionally, the method resembles leaky-bucket policing (and it fits into the same category in our classification), but is more flexible. Besides the average rate, the user is able to specify the burstiness of the traffic.[14] The call is admitted or rejected based on these two specifications combined.

Local Schemes

Congestion-control schemes operating locally (in the data-link layer) tend to be simpler and less expensive in terms of negotiation delays than connection based techniques. In very high-speed networks, these delays are more pronounced and practically restrict the applicability of the solutions operating in the transport layer to connection-oriented sessions of a non-trivial duration. Datagram traffic is often handled differently. For example, in the *Flow Network*, a portion of network resources is set aside and reserved for datagrams. The network offers the best-effort service to datagram traffic, within the limitation of the pre-allocated resource pool.

Many simple switching techniques (e.g., self routing without intermediate buffering) accompanied by simple routing rules (e.g., as in Manhattan-street or shuffle-exchange networks), offer natural means of avoiding local congestion and guarantee reasonable packet delivery on the global scale. These methods belong to the routing protocols (congestion is avoided as a byproduct of the routing rules) and will be discussed in a later section where we present a number of case studies. To hint at some other solution operating in the data-link/network layer, let us mention the design introduced in [38]. With the proposed solution, every switch is equipped with a neural arbiter which learns to make optimal routing decisions by adjusting the parameters of a set of fuzzy rules that determine the suitability of an output link for an incoming packet. Although the scope of the exercise discussed in [38] seems rather insignificant (a 4 x 4 MSN), the solution suggests an interesting approach to handling congestion in a large-scale high-speed network. Notably, the neural arbiter bases its decisions on local information: the contents of the packet header and the status of outgoing links. Another solution with a similar flavor aimed at the integration of ATM call admission and link capacity control can be found in [39].

The issue of speed disparity between the communicating peers receives a special flavor in the context of internetworking. Connecting two networks with different transmission rates and different characterization of the traffic pattern at their gateways typically requires a non-trivial congestion resolution scheme. In [40], the bottleneck situation at a gateway that connects a lower-speed LAN to a high-speed MAN is discussed and a flow-control scheme applicable to such a scenario is proposed. A study of packet loss in high-speed networks interconnecting conventional local area networks can be found in [41].

Buffering Policies: Deflection vs. Store-and-Forward

It is possible for more than one incoming packet to opt for the same outgoing link at the same time. In such a case, the node has to decide what to do with the packet (or packets) that cannot be immediately relayed on their preferred links. Such a packet can be buffered until the link becomes available or it can be relayed on another (available) link along a sub-optimal path to the destination. In the latter case, we say that the packet has been *deflected*. Deflection routing is suitable for networks with limited or non-existent buffer space at the nodes. Generally, buffering transient packets at intermediate switches has a number of disadvantages which are amplified when the network operates at a very high transmission rate. The arguments in favor of eliminating buffers, or at least reducing their size to a minimum, can be stressed as follows:

• Networks with large (practically infinite) memory switches are as susceptible to congestion as networks with low-memory switches. In the former case, the queuing delays can get so long that by the time the packets come out of the switch, most of them may have been already retransmitted by the higher layers due to timeouts. In fact, too much memory is more harmful than too little memory, since the packets or their retransmissions have to be dropped after they have

Connecting two networks with different transmission rates and different characterization of the traffic pattern at their gateways typically requires a non-trivial congestion resolution scheme.

[13] *In some variants of this policy, a credit may be allowed and, as long as the credit is not exceeded, the bucket may be emptied at the arrival rate of the incoming packets. This way, the packets need not be delayed if they occasionally arrive a little too fast.*

[14] *If the measurement interval is long, the data stream can be suspected to exhibit a substantial degree of burstiness. Conversely, short intervals indicate good predictability and a steady rate of incoming traffic.*

The most painful disadvantage of deflection, as opposed to store-and-forward routing, seems to be the inherent unpredictability of delays of individual packets belonging to the same higher-level message.

consumed precious network resources [29].

• Due to the nature of real-time applications which are characterized by stringent delay requirements, long buffers should be avoided, at least for this particular group of users.

• Elimination of buffers can speed up switching significantly so that the process can follow the link speed as closely as possible. Particularly, all electronic components can be removed from the switch and replaced with their optic equivalents that can operate at the link speed by avoiding E/O and O/E conversions. Buffer-based contention resolution without resorting to E/O conversions seems to be very difficult at the current level of technology and requires cumbersome optical delay lines [4].

It is obvious that deflection causes some packets to traverse longer paths. Furthermore, unless some countermeasures are taken, it is possible for a packet to travel indefinitely. In general, it can be said that if the probability of deflection at every intermediate node is equal to 1/2 or greater, deflection routing is not a good alternative to buffer-based contention resolution schemes. The problem can be presented as the so-called *Gambler's Ruin* problem. Suppose that at a given instant a packet is at half-way distance to its destination and still has $d/2$ hops to cover. In other words, it has some money ($d/2$, the distance it has already covered) and it needs as much more to finish the game (to reach its destination) as opposed to going bankrupt (being pushed d hops away from its destination again, equivalent to going back to the position where it started). At every node it rolls a dice (competes with other packets for a particular link) and loses with a certain probability p_d. If it wins (and is not deflected), it gets closer to the destination by one hop. Otherwise, the remaining distance increases by the penalty of deflection. In general, the deflection penalty is at least 2. For example, in ShuffleNet, an undeflected packet can gain only one step in the right direction while losing k (the binary logarithm of the number of nodes) in case of deflection. The probability of deflection (p_d) can be decreased in the following ways:

- Keeping the offered load below the saturation threshold of the network at a level that guarantees few contentions.
- Choosing a network topology that offers multiple shortest paths or at least many paths that are only marginally worse than the shortest path.
- Giving priority to the packets that are closer to their destinations (reducing the relative impact of the deflection penalty).
- Giving priority to the packets that have been previously deflected. This approach attempts to balance the number of deflections suffered by a single packet in a congested network.
- Giving priority to the packets that have spent the longest time in the network. This solution is similar to the previous one, but it also accounts for the propagation distance traveled by a packet.
- Buffering the packets that are to be deflected for a short amount of time (e.g., until the next round) to give them another chance, as opposed to diverting them from their course immediately.
- Discarding the packets that have exceeded a certain hop count limit. Such packets stand a good chance of being obsolete and they unnecessarily compete with other packets, possibly causing them to deflect.

Undeniably, deflection routing can cause some deterioration in the performance of the network compared to its store-and-forward version with the same geometry. In [42] a comparative analysis is presented for ShuffleNet with $p = 2$.[15] The study shows that the maximum throughput achievable by the network operating under deflection routing can be substantially lower than that achieved by its store-and-forward counterpart, with the difference becoming bigger for a larger number of nodes. Nonetheless, even large deflection-based networks with several thousand nodes are able to achieve no less than 25 percent of the throughput attainable by their store-and-forward analogs with unlimited buffers.

The most painful disadvantage of deflection, as opposed to store-and-forward routing, seems to be the inherent unpredictability of delays of individual packets belonging to the same higher-level message. Although in many solutions based on the store-and-forward approach (including ATM networks), the delays of individual packets may also vary substantially, at least these solutions are capable of preserving the ordering of packets upon their arrival at the destination. Most people believe that the inherent inability of deflection-based routing schemes to deliver packets "in order" renders them unsuitable for "serious" applications in connection-oriented environments. We will return to this issue in the concluding section.

Topology

Topological properties of a network can be examined separately from the routing and congestion-control mechanisms and can provide clues regarding the suitability of different choices in the design of the routing scheme. They are also directly related to the maximum throughput achievable by the network and to its resistance to faults.

According to their topologies, networks can be grouped into two broad categories: point-to-point networks and broadcast networks.[16] A broadcast network employs a single channel accessed by all nodes. A node willing to transmit a packet must follow some rules to reserve the channel. Every single transmission propagates to all stations in the network and, in particular, it reaches the intended recipient. For the reasons mentioned in the introduction, broadcast networks are not good candidates for large high-speed networks, unless the vast majority of the traffic is indeed of a broadcast nature. But even then, the methods of accessing and reserving broadcast channels tend to incur overheads proportional to the propagation length of the channel. Consequently, the impact of the access overhead will grow with the increasing transmission rate of the network and/or its geographical diameter.

In a point-to-point network, a single channel connects one pair of nodes and is typically used for transfers in one direction.[17] Consequently,

[15] p *is the order of the network graph, i.e.,* p = *2 means that there are two incoming and two outgoing links per each node.*

[16] *For efficiency reasons, the topology of a very large network may be organized into several hierarchical layers, e.g., as in telephone networks. Consequently, hybrid topologies may form. We are not dealing here with such cases.*

there is no need to arbitrate channel access. Due to the high (quadratic) cost of providing a direct link from every node to all other nodes, in most networks of a non-trivial size a node is connected directly to a subset of nodes. Connections are made in such a way that between any given pair of nodes there is at least one path in each direction. By a *path* we mean a collection of node-to-node links connecting a given source to a given destination. In this structure, a packet needs to be relayed from node to node to reach its ultimate recipient. If a given node has more than one outgoing link, it must make routing decisions, i.e., for every incoming packet it must decide on which outgoing link the packet will be relayed.

The number of nodes that perform routing tasks along a given path is called the *hop count* of the path. Note that, according to our definition, the hop count of a path can be different from the number of nodes that the path passes through. This happens if the path includes repeaters, i.e., nodes with a fixed assignment of the input links to the output links. Generally, the situation may not be so simple as a node appearing as a repeater for one packet may appear as a routing switch for another.[18] We will say that a given node contributes to the hop count of a packet, if the node has actually made a routing decision determining the fate of the packet, i.e., the packet could have followed another path forking at the node in question.

For obvious reasons of efficiency, a routing node typically tries to relay packets along paths of minimum delay (i.e., the shortest paths) leading to their destinations. The determination of the shortest path is related to the costs assigned to the links. In most cases (see the section on congestion control), the cost of a link is determined by its nominal capacity, its length, and its current congestion level. If we assume that all links are of the same capacity and the same (or comparable) length, and if we ignore the congestion component of the link cost, then the shortest path is also the path with the minimum hop count.

In some cases, when the network topology is highly irregular, its properties can be fully described only by specifying the complete network graph together with the parameters of its edges. Generally, one can get some idea as to the expected behavior of a network by looking at the following global parameters:

The **network size** (denoted by N) defined as the number of nodes in the network.

The **network diameter** (denoted by $\mathcal{D}$) defined as follows:

$$\mathcal{D} = \max\{\pi_{ij}\} \quad 1 \le i, j \le N \tag{2}$$

where π_{ij} stands for the minimum number of hops separating nodes S_i and S_j. According to the above formula, $\mathcal{D}$ gives the maximum of all shortest path lengths over all pairs of nodes. One interesting characterization of this parameter is its dependence on the network size N (e.g., logarithmic vs. linear).

The **average hop count** ($\bar{h}$) the average length of a shortest path taken over all pairs of nodes. This parameter is calculated as follows:

$$\bar{h} = \frac{1}{N} \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \pi_{ij}}{N-1} \tag{3}$$

where $1 \le i; j \le N$ and $i \ne j$.

The **degree of connectivity** gives the number of incoming and outgoing links connected to one node. If all nodes have the same degree of connectivity we say that the network topology is regular. Such a network is also referred to as *p-connected* where p is equal to the in/out degree of a node. For some interconnection patterns the degree of connectivity has to be increased as the network size grows.

One more simple numerical parameter of a network, which isn't strictly topological in nature (although it depends primarily on the topology), is the *deflection penalty*. Defined with respect to the routing algorithm, this parameter gives the least upper bound on the number of hops that a single deflection adds to the packet's path on its way to the destination. Sometimes the deflection penalty can be defined without explicit reference to the routing scheme as the *girth* (the length of the shortest cycle, if any) of the network graph. This is the case when any packet can be potentially relayed on any outgoing link of a node (i.e., the routing algorithm potentially explores all paths in the network graph).

The topological properties of a network are directly related to its maximum achievable throughput (U). If all links are of the same capacity and length, meaning that a packet traverses one link within one unit of time, then $\bar{h}$ gives the sojourn time per packet, i.e., the mean amount of time a packet must spend in the network before reaching its destination. In other words, $\bar{h}$ gives the average cost per packet transmission in terms of time and/or the number of links that must be visited by the packet. If the network is symmetric, all links are of the same capacity, the amount of buffer space at a node is infinite (i.e., packets are never lost and they always travel along the shortest paths), the traffic is uniformly distributed, and the packet generation rate is the same for all nodes, then the relationship between h and U can be expressed as follows:

$$U = \frac{\text{Total Number of Links}}{\bar{h}} \tag{4}$$

Although some of the prerequisites to formula 4 are not very realistic, the formula can be used to estimate the maximum throughput of various realistic networks. Generally, the larger the network, the more the small irregularities in the network topology tend to cancel out statistically. One can argue that the assumption about the uniform distribution of traffic does not hold in a large network, due to the presence of local communities of interest. In such a case, some simple biased distributions can be considered [43, 44] which only slightly complicate the above formula.

Useful as it is, formula 4 has its limitations. Although it says that networks with lower values of $\bar{h}$ should achieve a better throughput than networks with longer shortest paths, it has been reported that some semi-random or random topologies with lower hop counts yield signifi-

According to their topologies, networks can be grouped into two broad categories: point-to-point networks and broadcast networks.

[17] *At least this is the way the channels appears to the data-link layer.*

[18] *For example, this may happen in a network in which a virtual topology is embedded into a physical topology. A single physical node may emulate components of several logical paths with different properties.*

Regular topologies with many alternative shortest paths between every pair of nodes may reduce the size of the routing problem by grouping large classes of solutions into clusters with the same rank.

cantly poorer throughput compared to regular networks with higher values of $\bar{h}$ [45, 46]. This is not surprising since the fairness and regularity of the topology, as well as the variance of $\bar{h}$ perceived by different nodes have their own merits which should be carefully considered prior to a meaningful evaluation.

Gb/s Networks and New Challenges

In this section, we will investigate the fundamental issues that render the Gb/s networks different from their slower counterparts and discuss their implications on the design of the network topology, routing schemes, and congestion-control mechanisms. We choose to group these issues into two categories:

Processing Bottleneck — In a packet switching network, the time required to make a routing decision cannot be longer than a single packet's transmission time. Otherwise, the system becomes unstable since the ratio $\rho = \lambda/\mu$ of the arrival rate λ to the service rate μ at a node becomes greater than 1 and queue lengths grow indefinitely. Assuming the 53-byte cell length[20] of ATM networks and the transmission rate of 1 Gb/s, a node has at most 424 ns to complete the following tasks for every packet arrival:

- Selecting the appropriate output link on which the packet should be relayed. This is accomplished either by using the routing information contained in the packet header or by consulting a table which is usually indexed by the destination address.
- Resolving a possible contention in such a way that the network performance (according to the routing criteria) is optimized.

A deflection-routing scheme can be synchronous, when multiple packets (slots) arrive at a switch at the same time and their fate is determined globally in a single compound routing decision [23, 47], or asynchronous, when packets are treated individually and relayed on whatever outgoing links are available at the time of their arrival [44]. With the synchronous approach, the routing decisions can be more intelligent because more options are available when they are being made. However, the incoming slots must be aligned prior to the decision which complicates the switch design and may require backpressure mechanisms if the network is large. A natural objective of a synchronous routing scheme is to fulfill the demands of as many packets as possible, e.g., by minimizing the overall deflection penalty. For a switch with non-trivial connectivity, the best decision can only be arrived at by solving a difficult optimization problem. Although, due to the discrete and limited set of parameters, there are always ways of solving this problem in a fast way (by resorting to lookup tables), the amount of storage required to represent the needed data as well as the access techniques required to retrieve those data on-line may pose serious implementation problems.

In a node with the connectivity degree of p, p incoming packets can be assigned to p outgoing links in $p!$ ways. With a store-and-forward approach the situation is even worse: multiple incoming packets can be directed to the same outgoing link, which gives p^p possible assignments. Table 1 illustrates the scope of the problem space for a few typical connectivity degrees.

p	Deflection	S & F
2	2	4
4	24	256
8	40,320	16,777,216
16	2.09×10^{13}	1.84×10^{19}

Table 1. *The complexity of the optimal routing problem in deflection and store-and-forward networks.*

Multiple priority levels may further contribute to the cost. Similar problems are encountered in telephone networks and multiprocessor interconnection networks. Most of the proposed solutions are heuristic in nature.

In this regard, structures with low connectivity degrees have better chances for realistic optimal implementations. Additionally, regular topologies with many alternative shortest paths between every pair of nodes may reduce the size of the routing problem by grouping large classes of solutions into clusters with the same rank. As pointed out in [44], small departures from the optimality of routing decisions may have no visible impact on network performance, yet they may significantly reduce the complexity of the routing problem. Moreover, by selecting one of equally-ranked alternatives in a nondeterministic way, the routing scheme reduces the likelihood of a livelock [25].

Alternatively, one can opt for asynchronous routing. With this approach, when a packet arrives at a switch, it is relayed on the most suitable from the currently available outgoing links. This idea works poorly in networks with low connectivity. To see the problem, imagine that a packet arrives at an idle 2-connected switch. Suppose that the packet prefers no specific output link, so the switch is free to select one of the two links at random. Now, while the packet is retransmitted, another packet arrives and prefers the busy link. Clearly, the new packet must be deflected, but it could have been relayed according to its preference, had the decision regarding the first packet been delayed until the second packet was available. One can easily imagine how this scenario could evolve into a situation in which packets arriving at the switch continuously on both incoming links are indefinitely deflected because they keep arriving "out of phase" and every new packet finds its preferred link busy. However, when the connectivity degree is high (e.g., 8 or 16), asynchronous routing may be a viable alternative to the synchronous approach. Note that besides enormously reducing the complexity of routing decisions, asynchronous routing schemes eliminate the need for slot alignment and can handle packets of variable length. In [44] the reader will find the performance study of a prototype gigabit network based on asynchronous deflection.

Dominating Propagation Delays — The propagation latency of channels resulting from the finite

[20] *Which is considered small for data applications.*

Network	Capacity (Mb/s)	Prop. delay (ms)	Ratio a
Local Net	10.00	5	0.05
WAN	0.05	20,000	1.00
Satellite	0.05	250,000	12.50
Fiber link	1000.00	15,000	15,000.00

■ Table 1. *The ratio of propagation delay to packet transmission time for typical communication channels.*

speed of electro-magnetic signals in the media is amplified by high transmission rates. In [30], this relationship is described by the parameter a which is defined as follows:

$$a = \frac{\text{Propagation Delay}}{\text{Packet Transmission Time}} \qquad (5)$$

Table 2 lists the values of a for a few typical channel types (the fiber link is assumed to connect two points across the continental United States), for the packet length of 1000 bits.

The value of a tells how many packets can be pumped into one end of the link before the first bit of the first packet appears at the other end. The large value of a for high-speed channels is the source of numerous problems encountered not only in routing and flow control, but also at the application level.

Impacts on Topological Design

Due to high installation and maintenance costs, it is reasonable to expect that real-life MANs and WANs will be deployed with some optimization criteria in mind. Taking into account that the propagation time on the links is the major contributor to packet delays, it is desirable to minimize the total cable length needed to connect the network. The relevant geometric problem of connecting n given points in the plane with a set of shortest possible lines is known as the *Euclidean minimum spanning tree problem* [48]. One way to solve this problem is to build a complete graph with $n(n-1)/2$ undirected edges, which are weighted according to the distance[21] between the vertices. Next, one can apply a minimum spanning tree (MST) construction algorithm to select the smallest subset of edges that connect the graph. All links in the network can then be laid along these edges.

Sometimes it is possible to improve upon the result produced by the above method by introducing additional (artificial) vertices to the graph, which do not correspond to any nodes in the network [50]. The extra vertices are referred to as the *Steiner points* and it can be proved that for any n points to be spanned there exists a minimum-length Steiner tree that contains no more than $(n-2)$ Steiner points. It is also conjectured that the cost of the minimum spanning tree (based on the original collection of nodes) is no more than $(2\sqrt{3})$ times higher than the cost of the minimum-length Euclidean Steiner tree [51]. However, the problem of building the minimum Steiner tree for a given collection of points is *NP*-complete [52]. It has been expressed in many forms and a catalog of its different formulations can be found in [53]. The most efficient known heuristic [54] which seems to produce good solutions in a reasonable time is given in [51].

The optimal physical configuration of a network in terms of its total cable span may be incompatible with the requirements of the routing algorithm, which may be based on the assumption that the network topology is highly regular (e.g., as in ShuffleNet). In order to retain the advantages of regular interconnection networks, embedding a virtual topology into a given physical topology can be considered. One way of achieving this is by resorting to *wavelength division multiplexing* (WDM), which carves the bandwidth offered by the optical fiber into a number of smaller chunks and assigns them on a wavelength basis to the nodes. A wavelength can be assigned to a transmitter-receiver pair (producing a virtual point-to-point link) or it can be shared by a group of transmitters and receivers. The latter approach is particularly useful for describing virtual topologies that can formally be defined only for certain "round" numbers of nodes (e.g., ShuffleNet or a hypercube). By dedicating a wavelength to a specific subnet (e.g., a plane in the hypercube), one can reserve the virtual links needed in the complete structure without connecting them to the actual nodes which may be absent. This way, incomplete configurations can be built and later expanded in an incremental way. The advocates of this approach claim that changes in the virtual topology can be carried out dynamically to follow the observed changes in the traffic pattern. Thus, the network can adapt its topology to the traffic conditions in a way that maximizes its performance. Details of WDM can be found in [49, 55-61]. These studies raise the following two problems:

- The embeddings of the hypercube and ShuffleNet are discussed in [55] and [49], respectively. However, the efficient bit-controlled routing algorithms developed for these networks are solely concerned with minimizing the number of hops and ignore the lengths of the links. In other words, the dominating propagation delays are excluded from the scope of the routing criteria. Consequently, the minimization of the hop count does not imply the minimization of the delay. A remedy is suggested in [55] which consists of assigning the wavelengths in a fashion that minimizes the physical lengths of the virtual connections. However, the authors point out that the related subproblems of network configuration, namely: 1) the mapping of the nodes in the physical topology to the nodes in the virtual topology, 2) the mapping of the edges in the physical topology to the edges in the virtual topology, 3) the allocation of the wavelengths to edges, are all expected to be *NP*-hard as the search space grows at least as fast as $N!$ Consequently, effective heuristics are in demand.
- The other problem is related to the cost of the routing procedure. In [57], the total network throughput of ShuffleNet is improved by modifying the topology such that the largest flow on any link is minimized. The throughput improvement ranges from 6.8 percent for the quasi-uniform traffic matrix to 24.3 percent for the ring-type

In order to retain the advantages of regular interconnection networks, embedding a virtual topology into a given physical topology can be considered.

[21] *The metric in which this distance is expressed is not necessarily Euclidean: the distance between two points (x1, y1) and (x2, y2) can be expressed either in Euclidean metric L2 as:* $(\sqrt{(x_1-x_x)^2+(y_1-y_2)^2})$ *or in rectilinear metric* l_1 *as* $(|x_1-x_2| + |y_1-y_2|)$. *Rectilinear metric has been suggested as a more realistic alternative for urban environments [49].*

Optimal routing is shown to be formally equivalent to optimal flow control. Consequently, the optimality criteria and algorithms developed for the latter are applicable to the former.

traffic. The employed flow deviation method requires an iterative dynamic recalculation of the shortest paths. However, it is not clear how a switch designed for ShuffleNet's connectivity pattern is supposed to function on the modified topology. It seems that the gain in flexibility is paid by re-introducing large routing tables into the network and increasing the cost of the routing algorithm.

The dynamic reconfiguration of virtual topologies based on the changes in the observed traffic pattern is too tempting to be rejected because of the difficulties mentioned above. It seems that the issue should be addressed in the context of the routing schemes that were originally designed for static topologies with physical, rather than virtual, regularities. Perhaps a good solution can be arrived at by designing special routing schemes that would inherently assume the dynamic nature of the network topology.

Impacts on Routing Protocol Design

Most routing protocols that were proposed for large and slow networks are based on variants of shortest-path or least-cost algorithms [11]. From a theoretical standpoint, the routing problem can be viewed as an optimization problem. Optimal routing is shown to be formally equivalent to optimal flow control [11]. Consequently, the optimality criteria and algorithms developed for the latter are applicable to the former. Given a specific cost function, optimal routing can be achieved at least in principle. One can always argue, however, whether the optimality criteria well reflect network goals, i.e., whether the optimal behavior of the network according to those criteria appears also optimal (in the informal sense) to the users.

The cost function typically associates each link with a certain value that is adjusted dynamically according to the varying load of the link. In the simplest case, the function returns one of two values indicating whether the link is available at the moment or used to relay a packet. In a global optimization scheme, the suitability of a link to relay a packet addressed to a given destination is affected by the status of the remaining components of the packet's path, i.e., by the congestion level at the nodes separating the decision-making switch from the destination. The coordination of the status information among the nodes is achieved using a rather complex collection of algorithms that work more or less independently and yet support each other by exchanging information or services. This approach presents serious drawbacks for gigabit networks. In [62], six performance measures are proposed that can be used to compare routing algorithms operating in a decentralized fashion, The first and the most important measure is the *speed of response*: "This measure is obviously extremely important in dynamic environments since the speed of response must be faster than the rate of change of network topology. Otherwise convergence will not occur and the routing algorithm will be useless."

Regarding flow based methods, in [11] it is stated that: "Implicit in flow models is the assumption that the statistics of the traffic entering the network do not change over time. This is a reasonable hypothesis when these statistics change very slowly relative to the average time required to empty the queues in the network and when link flows are measured experimentally using time averages."

In other words, any relevant changes that affect the traffic patterns in the network should be closely followed by the corresponding updates in the routing information available at individual nodes. The time interval between the changes must allow for the update information to propagate to the involved nodes, for the nodes to calculate a new optimum state, and for the network (specifically its users) to benefit from the reaction of the routing algorithm to the requirements of the new state.

In a gigabit network, owing to large a, the gap mentioned above may be one or two orders of magnitude larger than the actual duration of a typical transition from one traffic pattern to another. For example, to transmit a 1 MB file across a LAN (see Table 1) one can negotiate all the resources needed for this transfer in advance, which will take a tiny fraction of the time needed to transmit the first packet of the file. On the other hand, when the file is to be sent through a 1 Gb/s network spanning the area of continental United States, the amount of time required to announce the transfer to all the nodes that may be involved in it is almost twice as large as the amount of time needed to pump the file into the network by the source. Of course, to this we need to add the amount of time needed for the feedback information to propagate across the network, as well as the time required by the switches to recalculate their routing parameters.

In the light of the above observation, a reasonable set of optimality criteria applicable to a gigabit network must account for the relatively short duration of many transient traffic episodes. Such episodes should not force distant nodes to renegotiate their routing criteria. Instead, those nodes should base their decisions on local criteria, possibly employing a policy of verifying the criteria against long-term patterns observed over extended periods of time.

Impacts on Congestion Control

In [31] the unsuitability of the conventional mechanisms based on end-to-end or hop-by-hop windowing schemes for controlling congestion within high-speed networks is pointed out as follows:

- Window-based mechanisms typically rely on end-to-end exchange of control messages in order to regulate traffic flow. The control messages (sometimes with additional congestion information added by the intermediate nodes) are used as feedback by the source node to regulate its traffic. In high-speed networks, the propagation delays across the network typically dominate. Thus the feedback is usually outdated and any action the source takes is too late to resolve buffer overflows and avoid congestion. This argues for mechanisms that do not heavily rely on network feedback.
- It is also important that the congestion-control mechanisms operate at the speed of the communication link. For this reason, espe-

cially in the case of hop-by-hop window-based mechanisms, computationally-intensive control schemes are less desirable than simple schemes that can be easily implemented in high-speed hardware.

• The nature of the traffic also affects the design of the congestion control. While data traffic can usually be slowed down in order to cope with network congestion, it is likely that the real-time nature of the traffic will require some level of bandwidth guarantee. Real-time traffic (e.g., voice, video) has an intrinsic rate determined by the external factors that are outside the control of the network. Typically this rate can be estimated by the network prior to the establishment of the connection. The ability to slow down such sources is usually very limited. However, the packet arrival process is stochastic, implying that there is no guarantee that over short periods the resource will keep to the specified average rate. In addition, the initial estimate of the rate may be incorrect.

The subject is also discussed in [29], where the author draws attention to the basic principles of control theory. It is pointed out that no scheme can possibly solve a congestion problem that lasts shorter than the feedback delay needed to notify the offending parties and perceive their reaction. The author suggests a multi-level congestion control architecture for handling short-term and long-term congestion scenarios at different levels of the hierarchy.

Case Studies

In this section, we discuss a few proposals for a high-speed network design that have appeared in the literature. Their common characteristics are the self routing capability and the ability to function with a limited buffer space at individual nodes. The latter also implies the compatibility with routing schemes based on deflection.

Hypercube

A complete hypercube consists of $N = 2^n$ nodes which are numbered by n-bit binary numbers, from 0 to $2^n - 1$, interconnected in such a way that there is a link between two nodes if and only if the binary representations of their addresses differ in exactly one bit. Therefore, the nodal connectivity degree is n. The network diameter is also n, since this is the maximum number of binary positions on which two node addresses may differ. The distance between a pair of nodes is equal to the *Hamming distance* between their addresses expressed in binary format. The average hop distance is equal to half the diameter:

$$\bar{h} = \frac{(N \log_2 N)}{2(N-1)} \approx \frac{n}{2}$$

.

The penalty of deflection is two since all connections are bidirectional and a hop that doesn't reduce the Hamming distance to the destination by one, increases it by one. Other topological properties of the hypercube are discussed in [63].

A hypercube is a richly connected structure. Although it is non-optimal in terms of diameter, it can deliver optimal performance when the traffic is uniformly distributed, even with very simple routing mechanisms. Routing in a hypercube can be performed as follows. At every intermediate node XOR is performed on the address of the current node and the destination address. The locations of 1s in the bit pattern representing the result indicate the preferred routes. Assume that the output links of every node s are numbered in such a way that link number i leads to the node whose binary address differs from the address of s on position i. The basic routing algorithm for the hypercube is given in [64] as follows ($\oplus$ represents the XOR operation):

• Compute: *relativeaddr* = *currentaddr* $\oplus$ *destinationaddr*.
• Starting with the most significant bit of *relativeaddr*: let i be the bit number of first 1 in *relativeaddr*.
• Forward the packet on link i.

Numerous performance studies of hypercubes indicate that the structure is very suitable for deflection routing [44, 65-67]. Between two nodes, *node*(i) and *node*(j) with the Hamming distance of $H(i, j) < n$, there are $H(i, j)$ node-disjoint paths of length $H(i, j)$. Furthermore, n different node-disjoint paths whose lengths are less than or equal to $(H(i, j) + 2)$ are available [63]. Consequently, the average number of deflections suffered by a packet under uniform load is at most $O(\log n)$, regardless of the traffic intensity [66]. There are n unidirectional links in the network per node and a packet must traverse $n/2$ links on the average, limiting the maximum achievable throughput to two per node.[22] In [66], it is demonstrated that even with the added penalty of deflection, a maximum throughput very close to two can be sustained.

Since the number of nodes in a hypercube must be a power of two, there are large gaps in the sizes of the system that can be built with this architecture. One solution to this problem can be found within WDM-based structures as suggested previously. Another possibility is to use incomplete hypercubes which are defined for an arbitrary number of nodes [64]. In an incomplete hypercube, node connectivity rules remain the same as before (i.e., *link*(i) connects two nodes whose addresses differ at the i-th bit position and nowhere else), but some of the links are missing. The basic routing algorithm has to be modified as follows (the modification is indicated in boldface):

• Compute: *relativeaddr* = *currentaddr* $\oplus$ *destinationaddr*.
• Starting with the most significant bit of *relativeaddr*: let i be the bit number of first 1 in *relativeaddr*, **where link *i* exists for the current node**.
• Forward the packet on link i.

For an incomplete hypercube with N nodes, this algorithm offers the worst-case path length of $\lceil \log_2 N \rceil$.

The main disadvantage of hypercubes is that their nodal connectivity degree increases (logarithmically) with the network size [68].

A hypercube is a richly connected structure. Although it is non-optimal in terms of diameter, it can deliver optimal performance when the traffic is uniformly distributed.

[22] *I.e., two packets per one-packet time slot or two bits per one-bit slot.*

Consequently, hypercube networks cannot be scaled up without reorganizing the node structure.

Shuffle-Like (Minimum Diameter) Networks

As we pointed out previously, the network diameter D has a paramount impact on the maximum throughput achievable by the network. Thus, it is natural to consider designs that minimize D for a given number of nodes and degree of nodal connectivity. It is known that the minimum diameter D and the maximum connectivity degree p of a directed graph are related to each other in the following way:

$$N \le 1 + p + p^2 + \cdots + p^D = \frac{p^{(D+1)} - 1}{p - 1} \qquad (6)$$

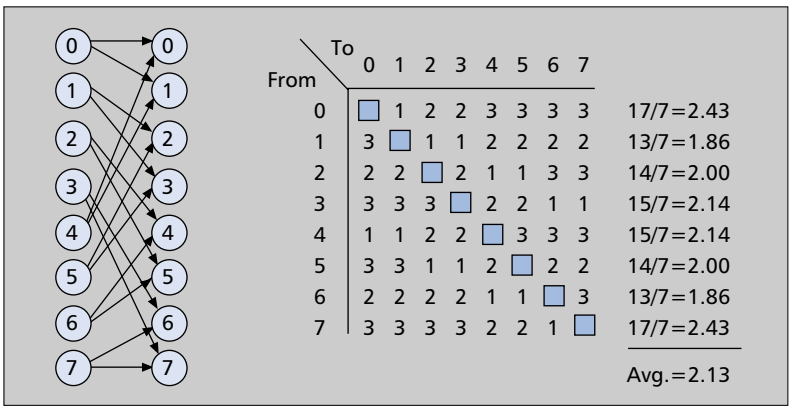
If the equality holds, then it is said that the *Moore bound* is achieved and the graph is called a *Moore graph*. Note that for a directed Moore graph the lower bound on diameter is:

$$D_{\min} = \left\lceil \log_p \left(N(p-1) + 1 \right) \right\rceil - 1 \qquad (7)$$

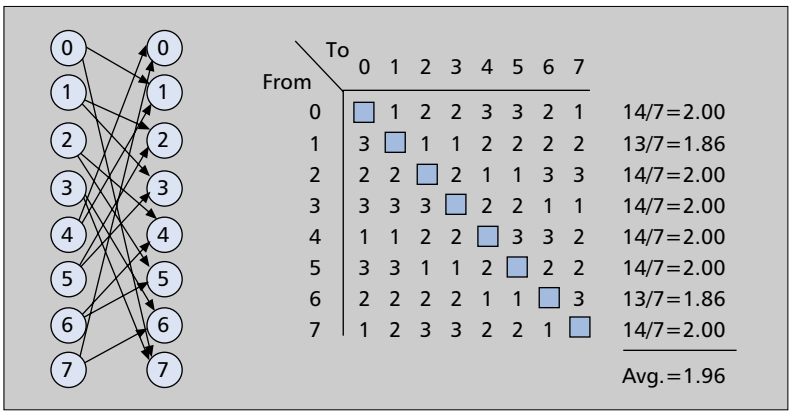
and the average hop distance $\bar{h}$ is bounded from below by [58]:

$$\bar{h}_{\min} = \frac{p - p^{D+1} + ND(p-1)^2 + D(p-1)}{(N-1)(p-1)^2} \qquad (8)$$

It is also known that there are no directed Moore graphs for nontrivial values of D and p [69].



■ Figure 1. *The topology of an eight-node de Bruijn network.*



■ Figure 2. *The topology of an eight-node shuffle network.*

One of the best known family of graphs which come close to the Moore bound are *de Bruijn graphs* [70]. A directed de Bruijn graph with connectivity p and diameter D has $N = D^p$ nodes. A general class of graphs has been proposed by Imase and Itoh [71, 72, 73] which contains de Bruijn graphs as a subclass. The design procedure outlined in [71] produces graphs with the upper bound

$$D \le \left\lceil \log_p N \right\rceil$$

for arbitrary values of N and p. Another approach to constructing networks with low mean internodal distances, based on simulated annealing, can be found in [46].

Shuffle-like networks provide mean internodal distances approaching the Moore limit [46]. However, these topologies are not defined for networks of arbitrary size. Figures 1-3 depict three shuffle-like networks with $p = 2$.

Shuffle-like networks based on de Bruijn graphs are defined when N is a power of the connectivity degree p. Figure 1 shows a special case of $p = 2$ and $N = 2^3$ in which $node(i)$ is connected to $node(2 * i \bmod N)$ and $node((2 * i + 1) \bmod N)$. In this network, even if the offered traffic is fully symmetric, the link loads are unbalanced [68]. This is due to the existence of self-loops[23] which carry no traffic. The maximum deflection penalty in a de Bruijn network is $\log_p N + 1$ since a packet can travel back to the point where it was deflected in at most $\log_p N$ steps. Communication networks based on de Bruijn graphs are discussed in [74-76].

The configuration shown in Fig. 2 is discussed in [24] and referred to as a shuffle-exchange network (SXN). The self-loops are eliminated by connecting nodes 0 and $N - 1$ to each other. Although this network has the lowest $\bar{h}$ among the three variations, no efficient routing algorithm that could explore these new connections is available and therefore they are only used in case of deflections or link failures.

Another variation on the shuffle theme was presented in [61] under the name of ShuffleNet and discussed further in [59, 60]. In ShuffleNet, $N = kp^k$ nodes are arranged in k columns of p^k nodes. Each node is addressed with its row and column coordinate pair (r, c). Columns are ordered left to right from 0 to $(k - 1)$ and rows are numbered top to bottom from 0 to $(p^k - 1)$. The row coordinates are represented in p-ary notation, $(r = r_{k-1} r_{k-2} \ldots r_0)$. Accordingly, the p nodes that any given node (r, c) is connected to are identified as follows:

$$\begin{aligned} &\left(\left(c+1 \right) \bmod k, r_{k-2} r_{k-3} \cdots r_0 0 \right) \\ &\left(\left(c+1 \right) \bmod k, r_{k-2} r_{k-3} \cdots r_0 1 \right) \\ &\vdots \\ &\left(\left(c+1 \right) \bmod k, r_{k-2} r_{k-3} \cdots r_0 \left(p-1 \right) \right) \end{aligned}$$

Figure 3a, redrawn in Fig. 3b, shows a special case of $p = k = 2$. Note that this connection pattern

[23] *Links $0 \to 0$ and $7 \to 7$ in the figure. In general, there are* p *of them in a* p*-connected network.*

has no self-loops. Furthermore, the variation in $\bar{h}_i$ across the nodes is zero since every node has the same number of nodes that lie within a given shortest path distance from it (i.e., there are two nodes reachable in one hop, three nodes reachable in two hops, and two nodes reachable in three hops from any node). The edge effects are eliminated by placing the nodes into two groups (shown as columns in Fig. 3b).

The diameter of the network is $\delta = 2k - 1$ with the deflection penalty of k. The average number of hops is given in [59] as:

$$\bar{h} = \frac{kp^k\left(p-1\right)\left(3k-1\right)-2k\left(p^k-1\right)}{2\left(p-1\right)\left(kp^k-1\right)} \qquad (9)$$

For the special case of $p = 2$, the above equation takes the following form [60]:

$$\bar{h} = \frac{1}{k2^k}\left[\sum_{j=1}^{k} j2^j + \sum_{j=1}^{k-1}\left(k+j\right)\left(2^k-2^j\right)\right] \qquad (10)$$

which simplifies to:

$$\bar{h} = \frac{1}{2^k}\left[3\left(k-1\right)2^{k-1}+2\right] \qquad (11)$$

giving the maximum throughput achievable under uniform load as:

$$U = \frac{k2^{2k+1}}{3(k-1)2^{k-1}+2} \qquad (12)$$

Consequently, the throughput available per user is:

$$U_i = \frac{2^{k+1}}{3(k-1)2^{k-1}+2} \approx \frac{4}{3}\frac{1}{k-1} \quad i=1,\cdots,k2^k \qquad (13)$$

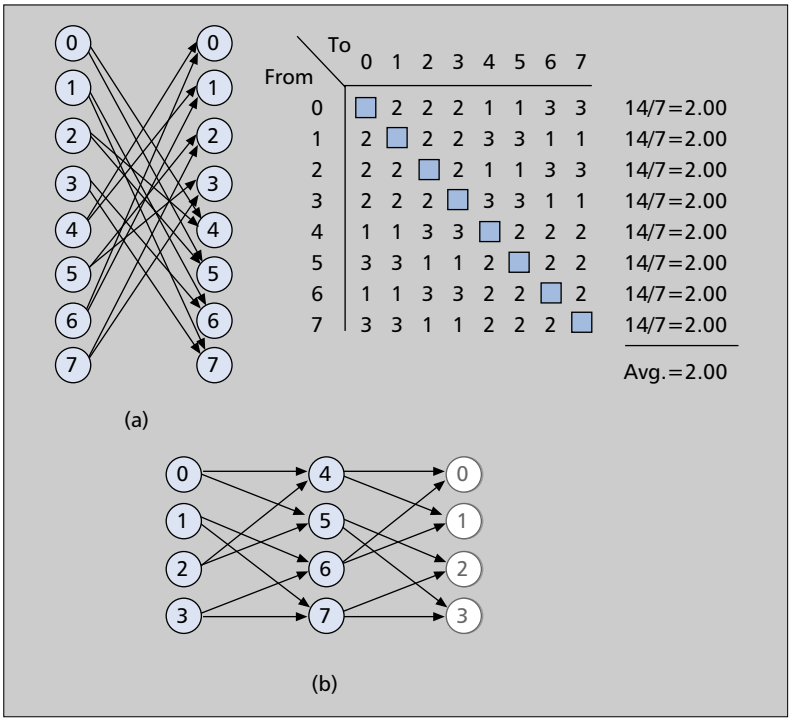
Note that the value of $\bar{h}$ increases with increasing k impeding the growth rate of the maximum achievable throughput. The 2-column configuration offers the highest throughput rate [59] and, for $k > 2$, no routing algorithm is known that yields a balanced use of links, even for a perfectly balanced load [77]. On the other hand, the increased number of columns increases the number of multiple shortest paths; the destination nodes that are k to $(2k - 1)$ hops away from a given source can be reached via more than one shortest path.[24] Table 3 [77] gives the average hop counts and throughput rates for different values of k.

Many routing algorithms are applicable to shuffle-like networks [12, 14, 18, 75]. Below we discuss two such algorithms proposed for ShuffleNet.

Under uniform traffic conditions, ShuffleNet performs very well with a simple routing algorithm which uses a single path for every source-destination pair. Given an intermediate node $(\hat{r}, \hat{c})$ and a transient packet addressed to destination (r^d, c^d), the packet is relayed along the output link leading to the following node [38]:

$$node((\hat{c}+1) \bmod k,\ \hat{r}_{k-2},\ldots,\ \hat{r}_0 r^d_{X-1}).$$

where X denotes the number of columns between the current node and the destination and is given by:



■ Figure 3. *The topology of an eight-node ShuffleNet.*

k	N	$\bar{h}$	U_i	U
2	8	2.0	1.0	8.0
3	24	3.25	0.617	14.8
4	64	4.625	0.433	27.7
5	160	6.06	0.33	52.8
6	384	7.53	0.265	101.9
7	896	9.09	0.222	198.8
8	2048	10.51	0.19	389.8
9	4608	12.001	0.167	767.7

■ Table 3. *Numerical properties of ShuffleNets.*

$$X = \begin{cases} \left(k+c^d-\hat{c}\right) \bmod k & \text{if } c^d \neq \hat{c} \\ k & \text{if } c^d = \hat{c} \end{cases}$$

In [77] it is shown that the allowable throughput per node with the fixed routing algorithm and realistic (nonuniform) traffic patterns is reduced by a factor between 0.3 and 0.5 with respect to that predicted for a uniform load. To reduce this throughput deterioration, an adaptive routing scheme has been proposed in [38]. Following the notation used in the description of the fixed routing algorithm, let D denote the number of columns between the source (r^s, c^s) and the destination (r^d, c^d):

$$D = \begin{cases} \left(k+c^d-c^s\right) \bmod k & \text{if } c^d \neq c^s \\ k & \text{if } c^d = c^s \end{cases}$$

If a packet cannot reach its destination in k hops, then the minimum-hop routing path length is $D + k$, regardless of what routing decisions are made in the first D hops. Therefore, if

[24] *This property of ShuffleNet is investigated in [78] with the help of signal flow graphs and the results are compared to those obtained for Manhattan-street networks.*

The torus that provides the topological surface of the network is a manifold, i.e., a finite two-dimensional space without a boundary. Consequently, the network has no boundary and it looks the same from every node.

the source and destination are more than k hops apart, the packet can be routed arbitrarily for the first D hops until it reaches column c^d of its destination. Then a single path of length k leads to (r^d, c^d). The routing algorithm requires marking each packet at the source according to the distance to its destination either as M-type (which stands for multiple minimum-hop paths) or as S-type (single minimum-hop path). Clearly, M-type packets require more than k hops to reach their destinations. At each intermediate node the remaining distance is calculated for the packet and its type field is updated if necessary. There are two ways for a packet's type to change. First, a type S packet can be deflected to a longer path increasing the remaining distance by k hops and changing its type to M. Second, a type M packet can reach the column of its destination and become type S. The test for type can be performed as follows:

TYPE S if $r^s_{k-1-D} = r^d_{k-1}$, $r^s_{k-2-D} = r^d_{k-2}$,..., and $r^s_0 = r^d_D$

TYPE M otherwise.

The routing scheme also employs buffers and deflects each packet once, at most. Type M packets are always placed in the shortest queue at a given node. Type S packets are normally routed according to the fixed algorithm. But if the buffer of the preferred outgoing link is full beyond a certain threshold, the packet is placed in the shortest queue with a special flag recorded in its header. A deflected packet is always placed in the appropriate buffer regardless of the threshold. If the buffer is full, then the packet is dropped. Throughput results for networks with sizes comparable to those discussed above are not available.

The throughput versus delay characteristics of de Bruijn networks are compared with those of ShuffleNets in [76]. For a given diameter, a de Bruijn network can support more nodes than its ShuffleNet counterpart, as shown in Table 4 (for $p = 2$)

The increase in the number of stations comes at the expense of some non-uniformity in edge loading under uniform traffic. To illustrate the differences between ShuffleNet and de Bruijn networks, we performed the following calculations on the sample networks shown in Figs. 1 and 3. First, we listed all shortest paths for every source-destination pair in the network, indicating also the intermediate nodes, e.g., the shortest path from *node*(0) to *node*(7) is (0 → 1 → 3 → 7) in the de Bruijn network, whereas ShuffleNet offers two possibilities: (0 → 4 → 1 → 7) and (0 → 5 → 3 → 7). Then we deleted the end nodes from the paths, i.e., nodes 0 and 7 in the example. Our observations can be stressed in the following two points:

- There are no multiple shortest paths in the de Bruijn network, whereas ShuffleNet offers two shortest paths from every node to two destinations out of seven.
- Fifty-six different shortest paths exist in the de Bruijn network. Due to the availability of multiple shortest paths, this number is 72 for ShuffleNet. For each node, the number of times that the node acted as a relay was determined. The results are listed in Table 5.

k	N	$\mathcal{D}$ (ShuffleNet)	$\mathcal{D}$ (de Bruijn)
2	$2^3 = 8$	3	3
4	$2^6 = 64$	7	6
6	$6 \times 2^6 = 384$	11	Not applicable
8	$211 = 2048$	15	11
16	$2^{20} = 1{,}048{,}576$	31	20

Table 4. *Diameters of de Bruijn networks and ShuffleNets.*

Node	de Bruijn	ShuffleNet
0	0	11
1	11	11
2	9	11
3	11	11
4	11	11
5	9	11
6	11	11
7	0	11

Table 5. *Node bias in de Bruijn networks.*

Note that in the de Bruijn network, *node*(0) and *node*(7) never act as relays; therefore, their output links are never used to convey any nonlocal traffic and are always available to them. The loads of other nodes are also unbalanced [76].

Two-Dimensional Toroidal Networks

A two-dimensional toroidal network is a rectangular mesh with orthogonal wrap-around connections (Fig. 4). The reasons why such a structure is an interesting topology for a communication network are stated in [79] as follows:

- Addressing and routing is straightforward.
- The topology is isotropic, i.e., every node has the same set of connections and perceives locally the same topology of the network. Consequently, all nodes can execute exactly the same protocol.
- The wrap-around connections decrease path lengths and eliminate the edge effects.
- For a metropolitan network, the topology easily covers a rectangular grid of streets and avenues, i.e., the topology makes sense geographically [22].

The torus that provides the topological surface of the network is a manifold, i.e., a finite two-dimensional space without a boundary. Consequently, the network has no boundary and it looks the same from every node. Not every network has this property; the edge effects in de Bruijn networks were discussed in the previous subsection.

Figure 4 shows two toroidal mesh networks with the nodal connectivity degree of two. The networks differ in the orientation of the links. Let us assume for simplicity that the network grid is a square with n rows and n columns (although in general it can be a rectangle). Thus, the total number of nodes N is equal to $n2$. Every

node can be identified by a pair of coordinates (r, c) representing its row and column numbers, respectively. Assume that the rows and columns are numbered from zero up, starting from the left top corner of the grid. The connection rules for the highway-transfer network (HTN) are given as follows:

$$(r,c) \text{ is connected to } \begin{cases} ((r+1) \bmod n, c) \\ (r, (c+1) \bmod n) \end{cases} \quad (14)$$

Similarly, in a unidirectional MSN a node (r, c) is connected to the following nodes:

$$\begin{array}{l} ((r+1) \bmod n, c) \text{ if } c \text{ is even} \\ ((r-1) \bmod n, c) \text{ if } c \text{ is odd} \\ (r, (c+1) \bmod n) \text{ if } r \text{ is even} \\ (r, (c-1) \bmod n) \text{ if } r \text{ is odd} \end{array} \quad (15)$$

Bidirectional MSNs (BMSN) are also considered; in such a network, every link goes in both directions and every node has four incoming and four outgoing links.

Below we list the interesting numerical properties of square highway-transfer networks and MSNs.

- $\mathcal{D}$ — for HTN it is $2(n-1)$. For MSN, it is n when $n/2$ is odd and $n+1$ when $n/2$ is even [27]. For BSMN it is $(n-1)$ when n is odd and n when n is even.
- $\bar{h}$ — for MSNs of different sizes different formulas are given in [27]. The most succinct form is obtained when n is divisible by 4:

$$\bar{h} = \frac{\frac{N}{2}(n+2)-4}{N-1} = \frac{n}{2}\frac{N}{N-1} + \frac{N-4}{N-1} \quad (16)$$

For BMSN we have ([26, 47]):

$$\bar{h} = \begin{cases} \frac{n^3}{2(N-1)} & \text{when } n \text{ is even} \\ \frac{n}{2} & \text{when } n \text{ is odd} \end{cases} \quad (17)$$

For HTN, the following formula can be derived:

$$\bar{h} = \frac{\sum_{i=1}^{n-1} i(i+1) + \sum_{i=1}^{n-1} i(2n-i-1)}{N-1} \quad (18)$$

which simplifies to $\bar{h} = N/(n+1) \approx n$.

- Penalty of deflection: for HTN, it is n; for MSN, it is 4 (which is constant and independent of the network size); and for BMSN, it is 2.
- Degree of connectivity: for MSN and HTN, it is 2; and for BMSN, it is 4.

The preference of outgoing links for a packet to be routed can be determined locally by comparing the destination coordinates to the coordinates of the routing node. In many cases, the packet gets closer to the destination by moving along the column as well as the row and the ordering of these steps is not important, e.g., as in the hypercube. Consequently, multiple shortest paths are available and there are many cases when a packet equally prefers two outgoing links. This property makes the networks highly suitable for deflection routing.

Manhattan-street Network — The MSN is a regular two connected network (Fig. 4) designed for the metropolitan area environment [22, 23]. A complete MSN grid must be rectangular and the number of columns and rows must be even. Therefore, the network is not defined for an arbitrary number of nodes. However, a fractional addressing scheme has been proposed that allows an arbitrary number of pairs of rows to be added at any position in the network, and includes a procedure to add one node at a time [23]. The latter feature affects the regularity of the network and makes it not always possible to relay packets along the shortest paths to their destinations, at least as long as the routing decisions are based on the local information available at the routing node. In this section, we consider regular networks with even numbers of rows and columns.

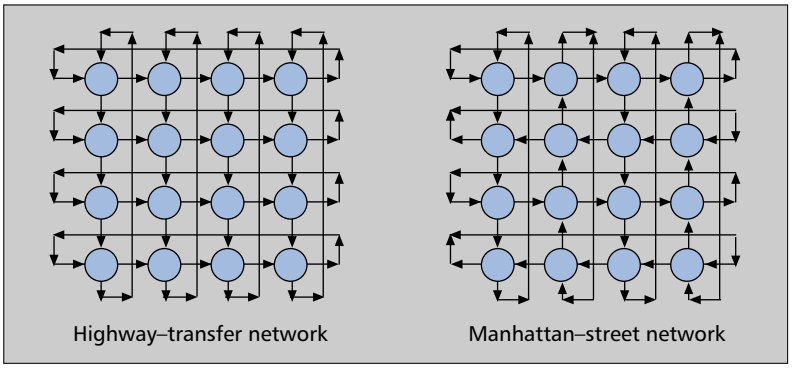


Figure 4. *Two toroidal meshes: the highway-transfer network HTN and the Manhattan-street network MSN.*

The virtue of local routing in a regular MSN is in ranking the outgoing links by comparing the destination address of an incoming packet (the row/column coordinates) with the address of the node making the routing decision. In [23] a precise set of local rules is given which guarantees that the preferred routes determined this way explore all shortest paths to the destination. The rules operate on transformed addresses which are derived from the actual addresses by assuming that the destination is located at the center of the network[25] and setting its row and column coordinates to zero. Then the relative address of the current node is calculated. Let m be the number of rows and n be the number of columns ($N = m \times n$). The transformed address (r, c) of a node with actual address (r_{fr}, c_{fr}) with respect to the destination node with actual address (r_{to}, c_{to}) is:

$$\begin{aligned} r &= m/2 - \{(m/2 - D_c(r_{fr} - r_{to})) \bmod m\} \\ c &= n/2 - \{(n/2 - D_r(c_{fr} - c_{to})) \bmod n\} \end{aligned}$$

where $\mathcal{D}_c$ and $\mathcal{D}_r$ are either -1 or $+1$, depending on the directions of the links. Since the destination lies at the center, the current node has to be located in one of the following sections of the transformed grid:

$$(r,c) \text{ is in } \begin{cases} Q_1 & \text{if } r > 0 \text{ and } c > 0 \\ Q_2 & \text{if } r > 0 \text{ and } c \le 0 \\ Q_3 & \text{if } r \le 0 \text{ and } c \le 0 \\ Q_4 & \text{if } r \le 0 \text{ and } c > 0 \end{cases}$$

It is shown in [23] that a fixed simple set of preferences can be associated with every section.

MSNs tend to perform best when the num-

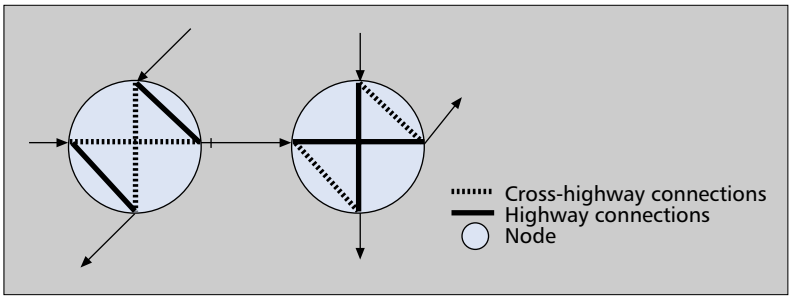
[25] *It cannot be located exactly at the center, but in one of the four central positions.*

ber of rows is the same as the number of columns [80]. When $N = n^2$, the throughput is limited to $2n^2/(n/2) = 4n$, since $\bar{h} = n/2$ and there are $2n^2$ links in the network. The performance of MSN is investigated in [24, 81] and compared to that of SXN. In [24], it is shown that the network performs very well under (synchronous) deflection routing. Even without buffering, 55 to 70 percent of the maximum theoretically possible throughput (achievable with infinite buffers) can be obtained under uniform load, while the addition of a small one-slot buffer per node increases this figure to 80 to 90 percent. A simple flow-control mechanism at the local source preventing the node from transmitting its own packet when both input links are busy is sufficient to guarantee that no packet loss will occur within the network.[26]

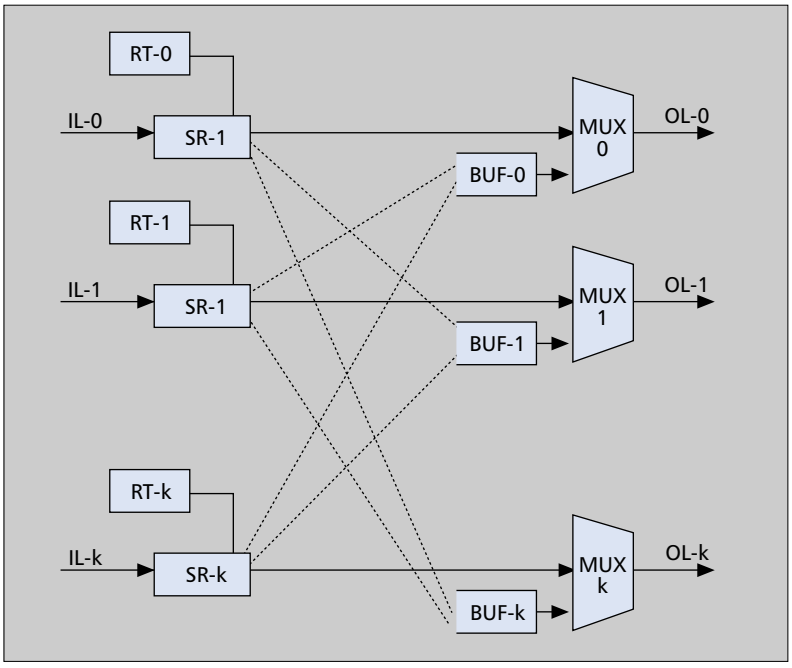
MSNs are often mentioned together with shuffle-exchange networks because both concepts are based on simple regular topologies and local routing rules. However, although there exist ways of relaxing the regularity requirement for MSN (and retaining most of the advantageous properties of the network), there are no simple means of doing the same with SXN. Furthermore, one should mention the following general differences in the behavior of MSN and SXN operating under deflection routing.

MSNs responds well to uniform saturation patterns. An oversaturated network (in which every node is constantly ready to transmit its own packet) is able to maintain essentially the same throughput as the maximum achieved below the saturation threshold.[27] In contrast, oversaturated SXNs tend to choke up with traffic, which results in a severe throughput deterioration. The effect becomes more pronounced as the number of nodes increases, because the deflection penalty in SXN grows with the network size. Consequently, flow-control mechanisms are needed for SXN, but not for MSN, at least as long as the load pattern is not excessively biased.

As a counterbalance to the above disadvantage, for the same number of nodes, the SXN topology offers smaller $\bar{h}$ than MSN. Besides, $\bar{h}$ expressed as a function of the number of nodes grows faster in MSN than in SXN. This implies that with perfect routing SXN will tend to achieve a higher maximum throughput per node than MSN. Indeed, this happens when buffers are present (i.e., routing decisions are close to optimal), but without buffers, due to a lower deflection penalty, MSN achieves a higher maximum throughput than SXN.



■ Figure 5. *Examples of highway and cross-highway connections in HTN.*



■ Figure 6. *Node configuration in HTN.*

Highway Transfer Network — The routing scheme of the Highway Transfer Network (HTN) is suitable for regular rectangular meshes (as shown in Fig. 4) as well as arbitrary (possibly irregular) topologies [82]. A highway is defined as a collection of adjacent links with each link belonging to only one highway. The flow of packets on a highway is unidirectional. A highway may be loop-shaped (i.e., look like a ring) or consist of an open-ended link (looking like a bus). The routing mechanism favors highway connections over cross-highway connections (Figs. 5 and 6) and can process packets going along the highways with less overhead.

The network operates in a slotted mode. A slot consists of a header followed by the payload part. The first bit of the header indicates whether the slot is full of empty. If the slot is full, the destination address is included in the header. Each node maintains a set of routing tables, one table associated with every incoming link. The routing tables are indexed by the destination addresses extracted from slot headers. Each entry contains an outgoing link identifier and a highway indicator. The highway indicator is a binary flag which tells whether the incoming link (on which the slot has arrived) and the outgoing link (on which the slot is to be relayed) belong to the same highway.

As illustrated in Fig. 6, each incoming link (IL) is connected to a shift register (SR) whose purpose is to buffer the header of an incoming slot before the slot is relayed on one of the outgoing links. The routing tables (RT) assign the incoming slots to the output links (OL). When a slot is to be forwarded on a highway, it is fed directly into the outgoing link (OL). On the other

[26] *Assuming no hardware malfunctions.*

[27] *The degradation does not exceed three percent.*

hand, if the slot is to be deflected from one highway to another, its contents are transferred to the FIFO buffer (BUF) associated with the outgoing link and the current slot to be relayed on the highway of the incoming link is marked as empty. With this approach, a slot traveling along a highway is relayed immediately.[28] The buffer associated with an outgoing link is examined whenever the slot to be relayed on the link turns out to be empty. In such a case, the first waiting slot is extracted from the buffer and relayed instead of the empty slot.

The basic assumption of the routing scheme is that packets traveling long distance to their destinations are relayed in the "highway mode" and experience shorter delays than packets trying to get *en route*. Therefore, it is obvious that the key to the success of this approach is in the proper placement of the highways. The designers stated that the technique was in its primitive stage and the optimal design rules were not clear at the time of writing. It can be argued that with the proper organization of the highways this scheme can produce better results than store-and-forward and cut-through switching methods, as observed on a square torus network.

Triangularly Arranged Network — Another routing strategy that we will briefly discuss here is defined on the so-called *Triangularly Arranged Connection Network* (TAC) and described in [80]. TAC is a three-connected toroidal mesh in which nodes are located on vertices of equilateral triangles and referenced by unique Cartesian pairs (x, y). The number of nodes needs to be a multiple of four in order for the links to be oriented properly; therefore, the network cannot be defined for an arbitrary number of nodes. Similarly to MSN, the next hop in the path to a given destination can be found solely by comparing the destination address with the address of the current node (Fig. 7).

Before going into the details of the routing algorithm, we should clarify one geometric property of the network. In Fig. 7, the triangle formed by nodes (3, 1), (5, 1), and (4, 2) is supposed to be equilateral. To interpret the network geometrically, it is assumed that the horizontal coordinates of the nodes represent correctly Cartesian distances along the horizontal axis, but the vertical coordinates are scaled properly to retain the Euclidean proportions of equilateral triangles. This means in particular that the difference between the y-coordinates of nodes (3, 1) and (4, 2) is not 1, but $\sqrt{3}$. Consequently, in the following algorithm, y-coordinates of the nodes are multiplied by $\sqrt{3}$ when they are used to calculate a distance or a vector magnitude. The distance between two nodes:

$$\left(d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \right)$$

and the magnitude of a vector

$$\left(m = \sqrt{x^2 + y^2} \right)$$

can both be evaluated and interpreted in the standard Euclidean sense.

Due to the orientation of the triangular links,

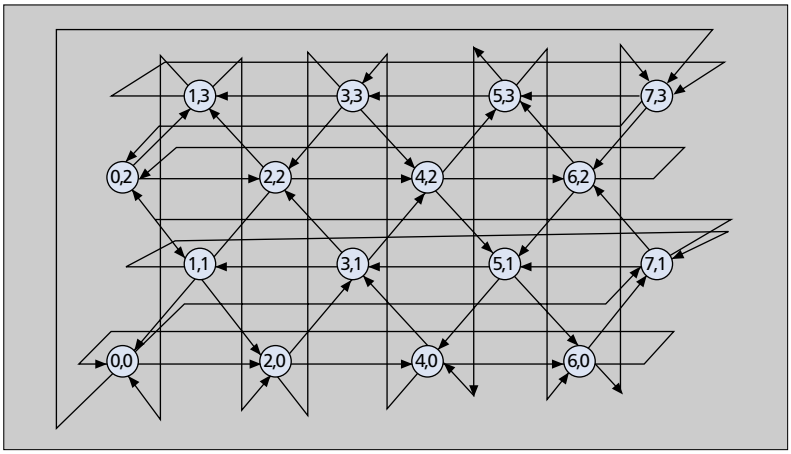


Figure 7. *A 4 x 4 TAC network.*

there appear to be eight different combinations of output link directions, which are denoted by three digits. The binary numbers are chosen in such a way that each bit represents a particular line, and a 1 in any position implies a rightward pointing arrow.[29] The first bit position denotes the main diagonal, the second the off diagonal, and the third the horizontal line. The routing algorithm is executed in three steps:

Step 1 — Suppose that a packet is to be relayed by node (5, 1) on its way to node (0, 2). This routing problem is illustrated in Fig. 8.[30] The relative coordinates of the destination are calculated as $(x_d - x_s, y_d - y_s) = (-5, 1)$. That reads: the x-coordinate of the destination is 5 units less than the x-coordinate of the current node (hence the minus sign) and the y-coordinate of the destination is 1 unit greater than that of the routing node. In other words, the minus sign of the x-coordinate of the destination shows that the destination lies 5 units to the *left* of the current node. By the same token, it is located 1 unit *up*.

Step 2 — Owing to the torus structure, the destination can be reached via four different paths. These alternatives and the distances involved are:

- Route 1: (–5, 1), i.e., 5 units left and 1 unit up from the current coordinate, has the distance of $\sqrt{28}$.[31]
- Route 2: (–5, –3), i.e., 5 units left and 3 units down from the current coordinate, has the distance of $\sqrt{52}$.
- Route 3: (3, 1), i.e., 3 units right and 1 unit up from the current coordinate, has the distance of $\sqrt{12}$.
- Route 4: (3, –3), i.e., 3 units right and 3 units down from the current coordinate, has the distance of $\sqrt{36}$.

Note that the distance calculations have nothing to do with the actual outgoing connections available at the routing node. The aim of this step is simply to determine the proper direction in which the packet should be forwarded. These routes and the distance involved in each case are depicted in Fig. 9. The minimum direct Euclidean distance is offered by route 3.

Step 3 — No physical connection exists at node (5, 1) that could take the packet 3 units

[28] *With the slight delay needed to examine the contents of its header.*

[29] *Note that there are no arrows that would point strictly up or down.*

[30] *To account for the torus structure, the destination node is redrawn at the right of the figure.*

[31] *All units in the vertical direction have been multiplied by* $\sqrt{3}$.

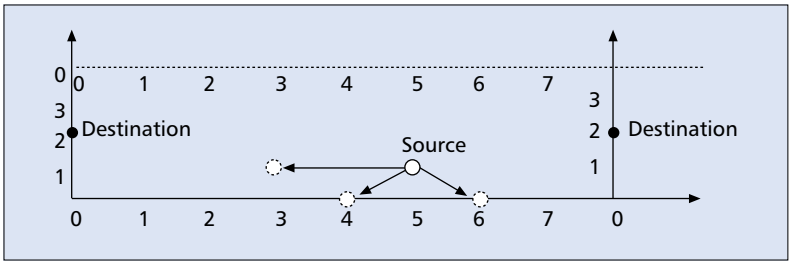


Figure 8. *TAC routing example: step 1.*

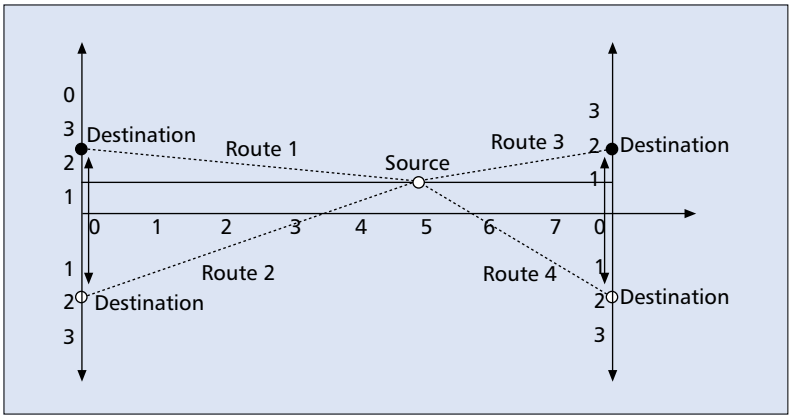


Figure 9. *TAC routing example: step 2.*

right and 1 unit up. Consequently, the algorithm tries to choose the link that can cover as much distance as possible in the indicated direction. The minimum distance vector calculated in the previous step enables the algorithm to choose between the three outgoing links available. The best choice is the direction along the main diagonal, leading the packet to node (6, 0) (Fig. 10). A similar operation will be performed at every intermediate node until the packet arrives at its destination.

In a simulation experiment in which packets that have not reached their destination after 7 hops on a 4 x 4 TAC network were dropped, 90 percent of all packets were able to make it to the destinations while queue lengths remained acceptable (less than 1/8 of the packets had to wait in queues).

Notably, the cost of implementing a TAC network is higher than that of an MSN with the same number of nodes: more links must be laid and the individual nodes are more complex and thus more expensive. The designers argue that the actual incremental cost may be small enough to be justified by the potential increase in performance. Table 6 compares the data path length in MSN and TAC networks with the same number of nodes.

These results are open to criticism. It seems to us that the better performance of TAC networks is related to the increased connectivity rather than the advantages of the triangular topology with respect to the MSN grid as claimed in [80]. For example, we find the maximum hop count to be 6 on an 8 x 8 MSN when we increase the connectivity of a node from 2 to 3. Moreover, the TAC routing algorithm introduces considerably more processing overhead when compared to that of MSN.

Bidirectional Toroidal Networks — The networks discussed in this section are based on rectangular toroidal grids (as in Fig. 4) with bidirectional connections. The total number of links is twice as much as in a unidirectional MSN with a given number of nodes. The maximum throughput of this topology (assuming a square grid with $n \times n$ nodes) is limited by $4n^2 = (n/2) = 8n$, under a uniform traffic pattern and with an unlimited buffer space available at the nodes.

A number of routing protocols have been suggested for this connection pattern. Differences between them lie in the way of resolving contentions. Although the number of outgoing links per node is increased only twofold with respect to the unidirectional network, the complexity of the routing problem (contention resolution) is multipled by a much larger factor. In a unidirectional network, two incoming packets can be assigned to two outgoing links in two ways, whereas the corresponding number for the bidirectional case is 4! = 24.

A bidirectional MSN with local deflection routing offers slightly shorter h and a smaller deflection penalty than its unidirectional counterpart. Consequently, the maximum throughput achieved by the bidirectional variant of the network is more than twice higher than for the unidirectional version.[32] Thus, one can say that it pays to multiply the number of links by two, as the gain is a more than twofold improvement in throughput. On the other hand, the routing operation becomes significantly more complicated.

HR^4-NET [26] employs a routing scheme that organizes the network into two different ring structures at two levels. Low-level rings (streets) are connected at each node by high-level rings (avenues). Each low-level ring (L-ring) is identified by its specific address which is used as routing information. Each packet travels on a high-level ring (H-ring) until it reaches a node that is located on the same L-ring as its destination. The routing decisions are precomputed and stored in a ROM which is indexed according to the preference of the incoming packets (3^4 possible cases: empty, H-ring, L-ring) and link availability (2^4 possibilities). The information retrieved from the ROM gives the allocation of the outgoing links to the incoming packet which maximizes the number of packets satisfied with their routes. In any case, the number of misdirected packets is not greater than two and its average[33] is 0.75 when four incoming packets are always present. The routing mechanism does not attempt to differentiate between two horizontal and vertical directions. Consequently, $\bar{h} = n^2/(n + 1)$ and the maximum throughput is bounded by $U = 4(n + 1)$, which is of the same order as in a unidirectional MSN. In [26], the loss of efficiency is justified with the simplicity of implementation. A shortest-path routing scheme for HR^4-NET can be found in [83]. It is a combination of the ideas explored in MSN (transformed addresses and grid sections with fixed preference rules) and the original HR^4-NET (10^4 precomputed switching decisions).

Another class of routing schemes applicable to bidirectional toroidal networks have found their place in Slotted Interconnected-Grid Net-

[32] *When the network operates without buffers, the improvement is of order 2.5 to 3, depending on the total number of nodes.*

[33] *Under uniform load.*

work (SIGNET) [84]. Routing in signet makes use of the concept referred to as the preference vector (PV), an ordered list which indicates the preferred links for a packet, given the quadrant that contains the packet's destination. Some routing algorithms look at the number of rows and columns separating the packet from the destination. Six different routing mechanisms are discussed in [84]. The first classification is made according to the setting of PV, namely in conformance with orthogonal (*O*) or diagonal (*D*) routing. With orthogonal routing, the packet moves in the row direction first, and then, when it hits the column of its destination, it moves along that column. Note that the same approach is used in HR^4-NET and HTN. The second algorithm attempts to route packets along a stepwise diagonal from source to destination. Thus, if the number of columns separating the packet from its destination is larger than the number of rows, the packet will be moved along the row and vice versa. A variation of diagonal routing is also suggested in [85] under the name of Z^2 (*Zig-Zag Routing*) policy.

After the preferences of all incoming packets have been determined, contention is resolved according to the priorities of the contending packets. These priorities can be assigned to the packets in three different ways. The *S* algorithm looks at the so-called *secondary counter* which gives the minimum of the number of rows and columns separating the packet from the destination. The smaller the value of the secondary counter the higher the packet's priority. The *D* algorithm determines the number of hops the packet must still travel, assuming that it will move along the shortest path to its destination. Packets that are closer to their destinations are privileged over those that have a longer remaining distance to travel. Finally, the *R* algorithm assigns priorities at random. Each of the two routing mechanisms can be combined with any of the three priority schemes, which results in six routing protocols denoted *OS, OR, OD, DS, DR, DD*. Simulation studies show that under uniform traffic, and for a variety of network sizes, the *OS* algorithm achieves the best overall performance in terms of the average delay and maximum throughput.

In [47], a locally-optimal routing algorithm is proposed for BMSN. Given a collection of packets arriving at a node within one cycle, the routing decision maximizes the reduction of the *weighted cumulative* distance of these packets to their destinations. With this approach, packets that are closer to their destinations have a higher priority in contention for their preferred outgoing links. The authors arrive at their routing criteria by noticing that when a packet must be deflected, it should be deflected towards the node from which it will prefer the maximum number of outgoing links. Consequently, deflections are forced towards the topological antipode of the destination because the antipode (if it exists) is characterized by the property that all its outgoing links are equally preferred by the packet. Generally, a node lying along the direction from the destination to its antipode offers the maximum number of preferred outgoing links from among all nodes to which the packet can be possibly deflected.

Network size	Number of links MSN	Number of links TAC	Average hops MSN	Average hops TAC	Maximum hops MSN	Maximum hops TAC
4 x 4	32	48	2.93	2.18	5	4
8 x 8	128	192	5.02	3.76	9	6
12 x 12	288	432	7.02	5.32	13	10
16 x 16	512	768	9.02	6.89	17	12

Table 6. *A comparison of MSN and TAC.*

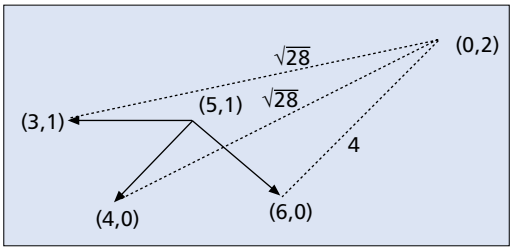


Figure 10. *TAC routing example: step 3.*

Flooding Networks

In this section we discuss a few networking solutions based on flooding. The first three concepts, Flooding Sink, Arbnet+, and Noahnet, have been designed for small-to-medium size local area networks and, for reasons that will become clear, are not suitable for metropolitan or wide area environments. They should be considered as alternatives to Ethernet or other networks in which all stations are "flooded" with every single transmission. The fourth solution presented in this section is less limited in scope, but it suffers from an unbalanced use of links, especially under heterogeneous traffic patterns.

Flooding Sink — As stated previously, the most severe problem with flooding is the presence of many unneeded replicas of the same packet. In Flooding Sink [86], this problem is avoided at the switch level. Every node remembers the *identifiers* of the last-forwarded 255 packets. When a packet arrives at the node and its identifier occurs in the list, the packet is ignored.

A Flooding Sink node is interfaced to the same number of input and output links. It receives packets from other nodes in the network and from its host. Each packet has a header containing the source address, the destination address, and a serial number. Within a certain time frame (dependent on the propagation diameter of the network), the pair <source address, serial number> uniquely identifies a packet and is referred to as the *packet identifier*. Any packet arriving on one of the input links is considered old if the node remembers its identifier. Such a packet is discarded and it doesn't propagate beyond the node. Otherwise, the packet is simply relayed on all outgoing links.

Technically, the above procedure involves buffering transient packets at the node. Every incoming link is connected to an input buffer. After a packet has been received completely, the node asserts its validity (examines its CRC code) and then passes the packet's identifier to

Arbnet+'s routing mechanism is based on a simple shortest-path tree search technique. With each transfer attempt, the protocol propagates the packet along a tree rooted at the source NIU.

the *eliminator* which tries to match it against one of the remembered identifiers. The storage for identifiers is associative and this part of the routing operation takes little time. If the packet is old, it is simply ignored. Otherwise, the pointers to the start and the end of the packet in the buffer are stored for processing. The destination address of the packet is compared to the address of the current node and the result of this comparison is stored together with the pointers. This way the node will know whether it is supposed to receive the packet (and pass it to the host) or relay it on the outgoing links.

Arbnet+ — Arbnet+ [87, 88] is built of *switches* connected by bidirectional point-to-point links. User devices (hosts) are connected to the network via Network Interface Units (NIUs) that can also be used as multiplexers to support multiple user devices attached to the same switch. The switches are responsible for routing packets and the *interswitch routing protocol* is topologically indifferent.

The access protocol of Arbnet+ is very similar to the IEEE 802.3 CSMA-CD medium access control layer. When an NIU has a frame to transmit, it listens to the link that interfaces it to the switch. If no activity is detected in the link, the NIU transmits the frame after an inter-frame delay and continues monitoring the link to detect a possible collision. If a collision occurs during transmission, the NIU stops the transmission immediately and emits a short jamming signal. Then it enters a random back-off mode. Otherwise the transmission will continue until the end of the frame has been reached. The NIU is then ready to transmit another frame or receive a frame from the network. The reception begins with the detection of an activity in the link. As soon as the destination address is recognized in the arriving frame, the NIU compares it to its own address. If no match occurs, the NIU emits a jamming signal into the link. Otherwise it passes the packet to the host.

The switch is in fact a simple transceiver. The arbitration of contention at the switch is based on the principle of first-come-first-served with blocking. The winning frame is repeated on all free output ports *on the fly*. In case of a collision, the switch stops the transmission and sends a *Clear Link Signal* (CLS) on the link on which the collision has occurred. All elements of this protocol are performed by hardware.

A collision in Arbnet+ can be of one of two types: *unintentional* or *intentional*. An unintentional collision occurs when two frames travel on the same link in the opposite directions. An intentional collision is forced by a switch that cannot accept a data frame for routing. In that case, the switch jams the incoming signal by sending a CLS.

A switch can be in one of three states: *idle*, *routing*, or *transmitting*. It enters the routing state upon detecting an incoming signal at one of its links. Subsequently, it repeats the incoming frame on all its free links with practically no delay. A frame that arrives at the switch when it is already in routing state, or when no free link is available, will not be repeated and a CLS will be inserted into the link of arrival.

Arbnet+'s routing mechanism is based on a simple shortest-path tree search technique. With each transfer attempt, the protocol propagates the packet along a tree rooted at the source NIU. A leaf collapses when a retransmission at that leaf is blocked, i.e., jammed by a CLS. When a switch detects that all its retransmission have been blocked, it sends a CLS up the tree (the *clear backward* operation) to collapse the branch connecting it to the uplink neighbor. Note that after some time, only the path connecting the source to the destination will remain active[34] and all the other branches of the network tree will have collapsed. When the source receives a CLS from all its links, it will conclude that the packet didn't make it. Such a packet will have to be retransmitted after a randomized back-off delay to avoid repetitive contention patterns (lockout).

To prevent looping of frames within the network and to properly recognize the success/failure of its transfer attempt, the root should not exhaust its transmission before a possible CLS indicating the collapse of the most distant leaf is given a chance to arrive at the switch. This implies a minimum frame size constraint for Arbnet+. The shortest frame should be longer than the longest possible round-trip delay between a pair of NIUs in the network. Shorter frames are reported to be eventually absorbed by the network due to the collisions at the expense of some degradation in performance. Arbnet+ offers lower delays and better throughput than Ethernet (with the same number of stations), because it tends to isolate the amount of network resources needed for a packet transmission to the shortest path in the network graph. However, this procedure is not instantaneous and its duration is proportional to the propagation diameter of the network. Consequently, the improvement is only visible when packets are long and the network is neither very large nor fast.

Noahnet — Noahnet is a LAN architecture implemented at the University of Delaware [89]. The network operates on a *randomly-connected* graph topology, uses a flooding protocol to route packets, and is intended for high-bandwidth media.

The network handles three types of packets: *data*, *status*, and *command*. Data packets carry the actual information exchanged among hosts. Status packets are used for two purposes: as acknowledgments, i.e., to indicate whether a *data* packet has arrived at a node in a good shape, and to indicate the flood status of a downstream node. The flood status can be: *forwarding*, *blocked*, or *got to destination* (GTD). At the time of writing, the only implemented command was *stop flooding*. All status messages are transmitted by a downstream node to its immediate upstream neighbor, whereas command messages are transmitted in the opposite direction.

A switch that receives a packet tries to relay it to all unoccupied adjacent nodes. The adjacent nodes repeat the same process until the packet either reaches its destination or it cannot be forwarded any further. A forwarding switch receives flood

[34] *Assuming that all the switches along it are available and the destination is willing to accept the packet.*

status packets from all its downstream nodes and, based on these packets, sends one status packet to its immediate upstream neighbor. As in Arbnet+, the path of the data packet forms a tree rooted at the source node. To speed up the operation of isolating the path to the destination and releasing the resources not needed for the transfer, the network carries out this procedure from both ends, i.e., from the leaves to the root as well as from the root to the leaves. A status packet that says "blocked" is interpreted as a collision in Arbnet+. Such a packet is sent by a collapsing leaf and it propagates upward to collapse the branch leading from the leaf to the root. A "stop flooding" command can be sent down the link to release switches downward from the top of various branches.

Controlled Flooding — Controlled Flooding, introduced in [90] and investigated further in [81], is a technique for reducing the total length of the path traveled by the multiple copies of every packet inserted into the network. The extent of flooding is limited by assigning a cost to every link traversal and allowing every packet a limited credit for traversing links. Links are treated as toll highways or bridges: every packet passing through a link must pay a toll. When a packet is launched at the source, it is assigned a numerical value which represents its credit or wealth. In order to traverse a link, the packet must possess a wealth which is at least equal to the cost of the link. The cost of a traversed link is deducted from the *wealth* of the packet. Therefore, at every intermediate node, the packet is repeated only on the links that it can afford. Finally, when its wealth is reduced below the cost of the cheapest link, the packet is discarded. Note that to route packets every node must only know the costs of its outgoing links. No routing tables are necessary.

The costs assigned to the links and the wealth assigned to the packets control the scope of flooding. Different patterns of allocating costs to links and wealth to packets result in different routing schemes. A heuristic algorithm for assigning costs to links is given in [90]. The objective of that algorithm is to minimize the number of nodes that unnecessarily receive packets addressed to some other regions of the network. A later study [91] claims that the proposed scheme yields an unbalanced resource usage and compares it to two other routing algorithms which choose routes along breadth-first search trees and shortest paths.

Conclusions

Gigabit transmission rates bring forward their specific issues influencing the design of the network hardware and software. The most drastic difference with respect to a slow network is the inflation of the network's apparent size. Even a geographically small network operating at a very high transmission rate looks large because the path between a pair of antipodal nodes can contain a substantial number of bits (and packets) at a time. This phenomenon alone is a strong indication that *locality* should be the predominant premise of a high-speed networking solution. Complicated routing schemes that with every decision try to comfort every remote node in the network are bound to fail because they cannot respond in a reasonable time to the feedback received across the large apparent network diameter. Similarly, "obsessive" congestion-control techniques that try to account for global changes in the traffic offered to the network will not work because they will tend to react to events that happened long ago and are no longer relevant.

Switched networks built of meshes of nodes interconnected via point-to-point channels scale better than busses, rings, or stars, but in contrast to those simple unidimensional topologies, they require non-trivial routing algorithms and, arguably, congestion control. Among the large number of routing schemes proposed for the gigabit range, the competition is between point-to-point store-and-forward techniques and deflection routing. Flooding-based solutions do not seem to offer a viable alternative as their performance is severely impaired by the multiple copies of every packet propagating out of their "legitimate" way. All techniques intended to contain flooding are ineffective when the network appears large and the length of a typical packet is much shorter than the network's diameter.

The store-and-forward approach in which packets addressed from a given source to a given destination are always routed along the same path has the advantage of delivering packets "in order." In contrast, deflection routing is tainted with the non-trivial reassembly problem: packets may arrive out of order and the destination must rearrange them upon arrival into the original message. Many people perceive this disadvantage as a disqualifying flaw in the whole concept of deflection routing. We believe that deflection routing with its simplicity and self-adaptation to varying loads is too attractive to be rejected that easily. One can argue that in many cases the need for the preservation of packet ordering at the destination is apparent and could be relaxed without affecting the integrity of data transfer. Consider, for example, a file transmission across the Internet. Most people would view this operation as a typical connection-oriented scenario in which the preservation of packet ordering is absolutely critical. Note, however, that the operating systems of the hosts involved in the data transfer perceive the file as a random collection of pages that just appear ordered because the user (or the transport layer) wants to view them that way. If the transfer protocol could be aware that the file consists of individual fragments scattered over a disk, it could transfer these fragments independently (together with their relative locations in the file) and the destination could store them on its disk as they arrive. Even better, the source could actually "optimize" the transfer by selecting the pages in the order that would minimize the total time needed to read them from the disk.

If we look carefully at those communication scenarios that appear to require the preservation of packet ordering, we will see that most of them fit into the following categories:

- Scenarios that in fact could be carried out with packets arriving in any order (like the file

Controlled Flooding is a technique for reducing the total length of the path traveled by the multiple copies of every packet inserted into the network.

Among the large number of routing schemes proposed for the gigabit range, the competition is between point-to-point store-and-forward techniques and deflection routing.

transfer case discussed above). They enforce packet ordering because higher protocol layers view them (unnecessarily) as stream-oriented sequential scenarios.

- Scenarios involving relatively short transfers (e.g., a piece of text to appear on a screen). Messages of this sort can be safely reassembled in a small buffer space at the destination.
- Long sustained continuous transfers that actually require packets to arrive in order (e.g., video, voice). Such scenarios typically admit a certain packet loss rate. Consequently, one can implement them with a limited reassembly buffer, dropping packets that arrive out of sequence while the buffer is full. Note that store-and-forward techniques generally do not guarantee packet delivery in this scenario either due to the jitter (even though the packets arrive in order, some of them may be too late to be useful).

One can identify a number of avenues for future research aimed at making deflection-routing schemes more attractive. In particular, the impact of deflections (and the packets arriving out of order) on the network performance under synchronous traffic scenarios should be investigated carefully and classified depending on the topology, the routing scheme, and the amount of buffer space available at a node. The average performance of many deflection schemes tends to improve drastically when small buffers are present at nodes. Perhaps in this way one can arrive at a compromise between store-and-forward and deflection routing which will retain the simplicity of the latter and some desirable properties of the former.

The present authors are aware of deflection schemes that limit the number of hops traveled by a packet on its way to the destination without losing packets or negotiating resources across the network.[35] With such schemes, it may be possible to limit the size of a reassembly buffer at the destination and accommodate continuous stream-type traffic of any duration without losing a single packet.

Another important issue is the mapping of virtual topologies onto real topologies. The backbones of future networks spanning metropolitan and larger areas will often consist of virtual links built on top of some generally available networking services, e.g., ATM. The right way of embedding a virtual network into a collection of such links should account for some specific properties of these links, e.g., the variability of delays and available bandwidth, the grade of bandwidth allocation, and the cost of tearing down a link during idle periods and setting it up again.

Synchronous deflection schemes suffer from the problem of slot alignment: all slots expected to arrive at a switch during one routing cycle must appear to have arrived simultaneously. In a network consisting of a moderate number of nodes, this problem can be solved by using alignment buffers that absorb slight irregularities in the arrival rate of slots from different links. When the number of stations is large, this simple approach may be ineffective. Then one can consider using node-to-node backpressure mechanisms indicating those irregularities to the neighbors. Alternatively, one can think of deflection schemes that operate correctly (without losing slots) despite occasional alignment problems. Ultimately, it is possible to switch to an asynchronous routing scheme which may provide a reasonable service when the connectivity degree of the network is not too small.

It is not clear how much is gained by insisting on the regularity of the network topology. With a regular topology, routing seems simpler because it can be performed locally without resorting to a routing table. But it seems unlikely that realistic networks are going to be perfectly regular. Optimal routing algorithms based on local table-free rules tend to be tricky and complex. Moreover, when the topology is even slightly irregular they either break down or become more complex and suboptimal. Perhaps the cost of using fixed routing tables exactly describing the network topology (which can be highly irregular) will be paid off by the real simplicity of the routing rules and the flexibility of the network structure.

[35] *Ongoing research.*

References

[1] A. R. Pach, S. Palazzo, and D. Panno, "Slot Pre-Using in IEEE 802.6 Metropolitan Area Networks," *IEEE JSAC*, vol. 8, no. 11, Oct. 1993, pp. 1249-1258.
[2] L. Kleinrock, Queueing Systems, (John Wiley & Sons, Inc., 1975).
[3] I. Cidon, and Y. Ofek, "Metaring A Full-Duplex Ring with Fairness and Spatial Reuse," *IEEE Trans. on Commun.*, vol. 41, no. 1, Jan. 1993, vol. 110-120.
[4] A. Jajszczyk and H. T. Mouftah, "Photonic Packet Switching," *IEEE Commun. Mag.*, vol. 31, no. 2, Feb. 1993, pp. 58-65.
[5] M. Schwartz, Telecommunication Networks, Protocols, Modeling and Analysis, (Addison-Wesley, 1987).
[6] D. Bertsekas, R. Gallager, Data Networks, second edition, (Prentice-Hall, 1992).
[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to Algorithms, (McGraw-Hill, 1990).
[8] P. R. Adby and M. A. H. Dempster, Introduction to Optimization Methods, (Halsted Press, 1974).
[9] H. A. Taha, Operation Research, An Introduction, (Collier-Macmillan, 1982).
[10] A. L. Peressini, F. E. Sullivan, and J. J. Uhl, Jr., The Mathematics of Non-linear Programming, (Springer-Verlag, 1988).
[11] D. Bertsekas and R. Gallager, Data Networks, (Prentice-Hall, 1987).
[12] H. J. Siegel and W. T. Hsu, "Interconnection Networks," chapter 6 in Computer Architectures, Concepts and Systems, V. M. Milutinovic, ed., (Elsevier Science Publishing, 1988).
[13] Y. Oie *et al.*, "Survey of Switching Techniques in High-Speed Networks and Their Performance," INFOCOM '90, pp. 1242-1251.
[14] J. Y. Hui, Switching and Traffic Theory for Integrated Broadband Networks, (Kluwer Academic Publishers, 1990).
[15] J-L. Baer, Computer Systems Architecture, (Computer Science Press, 1980).
[16] K. Hwang and F. A. Briggs, Computer Architecture and Parallel Processing, (McGraw-Hill, 1984).
[17] H. S. Stone, High Performance Computer Architecture, (Addison-Wesley, 1987).
[18] H. J. Siegel, Interconnection Networks for Large-Scale Parallel Processing, (McGraw-Hill, 1990).
[19] T. Feng, "A Survey of Interconnection Networks," *IEEE Computer*, Dec. 1981, pp. 12-27.
[20] D. A. Reed and D. G. Grunwald, "The Performance of Multicomputer Interconnection Networks," *IEEE Computer*, June 1987, pp. 63-73.
[21] S. Yalamanchili and J. K. Aggarwal, "A Characterization and Analysis of Parallel Processor Interconnection Networks," *IEEE Trans. on Computers*, vol. 36, no. 6, June 1987, pp. 680-691.
[22] N. F. Maxemchuk, "Regular Mesh Topologies in Local and Metropolitan Area Networks," *AT&T Technical J.*, vol. 64, no. 7, Sep. 1985, pp. 1659-1685.
[23] N. F. Maxemchuk, "Routing in the Manhattan Street Network," *IEEE Trans. on Commun.*, vol. 35, no. 5, May 1987, pp. 503-512.
[24] N. F. Maxemchuk, "Comparison of Deflection and Store-and-Forward Techniques in the Manhattan Street and Shuffle-Exchange Networks," INFOCOM '89, pp. 800-809.
[25] N. F. Maxemchuk, "Problems Arising from Deflection Routing: Live-Lock, Congestion and Message Reassembly," *Proc. of NATO Workshop on Architecture and Performance Issues of High Capacity Local and Metropolitan Area Networks*, 1990, pp. 209-233.
[26] F. Borgonovo, E. Cadorin, "HR4Net: A Hierarchical Random-Routing, Reliable and Reconfigurable Network for Metropolitan Area," INFOCOM '87, pp. 320-326.
[27] T. Y. Chung, D. P. Agrawal, "On the Network Characterization of and Optimal Broadcasting in the Manhattan Street Network," INFOCOM '90, pp. 465-472.
[28] B. Khasnabish, M. Ahmadi, M. Shridhar, "Congestion Avoidance in Large Supra-High-Speed Packet Switching Networks Using Neural Arbiters," GLOBECOM '91, pp. 140-144.
[29] R. Jain, "Congestion Control in Computer Networks: Issues and Trends," *IEEE Network*, vol. 4, no. 3, May 1990, pp. 24-30.
[30] L. Kleinrock, "The Latency/Bandwidth Tradeoff in Gigabit Networks;

Gigabit Networks Are Really Different!," *IEEE Commun. Mag.*, vol. 30, no. 4, April 1992, pp. 36-40.
[31] K. Bala, I. Cidon, and K. Sohraby, "Congestion Control for High Speed Packet Switched Networks," INFOCOM '90, pp. 520-526.
[32] J. S. Turner, "Managing Bandwidth in ATM Networks with Bursty Traffic," *IEEE Network*, vol. 6, no. 5, Sep. 1992, pp. 50-58.
[33] S. Deng, "Flexible Access Control in Broadband Communication Networks," Ph.D. Thesis, University of Alberta, Edmonton, Alberta, Canada, 1992.
[34] B. Kreimeche and M. Schwartz, "A Channel Access Structure for Wideband ISDN," *IEEE JSAC*, vol. 5, no. 8, Aug. 1987, pp. 1327-1335.
[35] G. Ramamurthy and R. S. Dighe, "A Network Access Control for Integrated Broadband Packet Networks," INFOCOM '90, pp. 896-907.
[36] J. S. Turner, "New Directions in Communications," *IEEE Commun. Mag.*, vol. 24, no. 10, Oct. 1986, pp. 8-15.
[37] L. Zhang, "The Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks," *ACM Trans. on Computer Systems*, vol. 9, no. 2, 1991, pp. 101-124.
[38] M. J. Karol and S. Z. Shaikh, "A Simple Adaptive Routing Scheme for Congestion Control in ShuffleNet Multihop Lightwave Networks," *IEEE JSAC*, vol. 9, no. 7 Sep. 1991, pp. 1040-1050.
[39] A. Hiramatsu, "Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Networks," *IEEE JSAC*, Sep. 1991, pp. 1131-1138.
[40] L. Wong and M. Schwartz, "Flow Control in Metropolitan Area Networks," INFOCOM '89, pp. 826-833.
[41] M. Gumbold, P. Martini, and R. Wittenberg, "Temporary Overload in High Speed Backbone Networks," INFOCOM '92, 1992, pp. 2280-2289.
[42] A. S. Acampora, S. I. A. Shah, "Multihop Lightwave Networks: A Comparison of Store-and-Forward and Hot-Potato Routing," *IEEE Trans. on Commun.*, vol. 40, no. 6, June 1992, pp. 1082-1090.
[43] S. P. Dandamudi, "Hierarchical Hypercube Multicomputer Interconnection Networks," (Ellis Horwood, 1991).
[44] P. Gburzynski and J. Maitan, "Deflection Routing in Regular MNA Topologies," *J. of High Speed Networks*, vol. 2, no. 2, 1993, pp. 99-131.
[45] C. Rose, "Mean Internodal Distance in Regular and Random Multihop Networks," *IEEE Trans. on Commun.*, vol. 40, no. 8 Aug. 1992, pp. 1310-1318.
[46] C. Rose, "Low Mean Internodal Distance Network Topologies and Simulated Annealing," *IEEE Trans. on Commun.*, vol. 40, no. 8 Aug. 1992, pp. 1319-1326.
[47] F. Borgonovo, E. Cadorin, "Locally-Optimal Routing in the Bidirectional Manhattan Network," INFOCOM '90, pp. 458-464.
[48] J. A. Bannister, L. Fratta, M. Gerla, "Topological Design of the Wavelength-Division Optical Network," INFOCOM '90, pp. 1005-1013.
[49] R. Sedgewick, Algorithms, second edition, [Addison-Wesley, 1988].
[50] E. Lawler, Combinatorial Optimization: Networks and Matroids, Holt, (Rinehart and Winston, 1976).
[51] J. M. Smith, D. T. Lee, and J. D. Liebman, "An O(n logn) Heuristic for Steiner Tree Problems on the Euclidean Metric," *Networks*, vol. 11, 1981, pp. 23-29.
[52] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to NP-Completeness," (W. H. Freeman and Co., 1979).
[53] M. X. Goemans, Y. Myung, "A Catalog of Steiner Tree Formulations," *Networks*, vol. 23, 1993, pp. 19-28.
[54] P. Winter, "Steiner Problem in Networks: A Survey," *Networks*, 17, 1987, pp. 126-167.
[55] R. J. Vetter, K. A. Williams, and D. H. C. Du, "Topological Design of Optically Switched WDM Networks," IEEE 742, pp. 114-127.
[56] J. A. S. Monteiro and M. Gerla, "Topological Reconfiguration of ATM Networks," INFOCOM '90, pp. 207-214.
[57] J. P. Labourdette and A. S. Acampora, "Logically Rearrangeable Multihop Lightwave Networks," *IEEE Trans. on Commun.*, vol. 39, no. 8, Aug. 1991, pp. 1223-1230.
[58] Z. Zhang and A. S. Acampora, "Analysis of Multihop Lightwave Networks," GLOBECOM '90, pp. 1873-1879.
[59] M. G. Hluchyj and M. J. Karol, "ShuffleNet: An Application of Generalized Perfect Shuffles to Multihop Lightwave Networks," INFOCOM '88, pp. 379-390.
[60] A. S. Acampora, "A Multichannel Multihop Local Lightwave Network," GLOBECOM '87, 1459-1467.
[61] A. S. Acampora, M. Karol, M. G. Hluchyj, "Terabit Lightwave Networks: The Multihop Approach," *AT&T Technical J.*, vol. 66, no. 6, Nov./Dec. 1987, 21-34.
[62] M. Schwartz, "Routing and Flow Control in Data Networks," IBM Research Report 36329, 1980.
[63] Y. Saad and M. H. Schultz, "Topological Properties of Hypercubes," *IEEE Trans. on Computers*, vol. 37, no. 7, July 1988, pp. 867-872.
[64] H. P. Katseff, "Incomplete Hypercubes," *IEEE Trans. on Computers*, vol. 37, no. 5, May 1988, pp. 604-608.
[65] T. Szymanski, "An Analysis of Hot-Potato Routing in a Fiber Optic Packet Switched Hypercube," INFOCOM '90, pp. 918-925.
[66] A. G. Greenberg, B. Hajek, "Deflection Routing in Hypercube Networks," *IEEE Trans. on Commun.*, vol. 40, no. 6, June 1992, pp. 1070-1081.
[67] B. Hajek, "Bounds on Evacuation Time for Deflection Routing," *Distributed Computing*, vol. 5, 1991, pp. 1-6.
[68] B. Mukherjee, "WDM-Based Local Lightwave Networks, Part II: Multihop Systems," *IEEE Network*, vol. 6, no. 4, July 1992, pp. 20-32.
[69] W. G. Bridges and S. Toueg, "On the Impossibility of Directed Moore Graphs," *J. of Combinatorial Theory*, Series B, vol. 29 1980, pp. 339-341.
[70] N. G. de Bruijn, "A Combinatorial Problem," *Koninklijke Netherlands: Academe Van Wetenschappen*, Proc, vol. 49, no. 20, 1946, pp. 758-764.
[71] M. Imase and M. Itoh, "A Design for Directed Graphs with Minimum Diameter," *IEEE Trans. on Computers*, vol. 32, no. 8, Aug. 1983, pp. 782-784.
[72] M. Imase, T. Soneoka, and K. Okada, "Connectivity of Regular Directed Graphs with Small Diameters," *IEEE Trans. on Computers*, vol. 34, no. 3, March 1985, pp. 267-273.
[73] N. Homobono and C. Peyrat, "Connectivity of Imase and Itoh Digraphs," *IEEE Trans. on Computers*, vol. 37, no. 11 Nov. 1988, pp. 1459-1461.
[74] M. R. Samatham and D. J. Pradhan, "The De Bruijn Multiprocessor Network: A Versatile Parallel Processing and Sorting Network for VLSI," *IEEE Trans. on Computers*, vol. 38, no. 4, April 1989, pp. 567-581.
[75] A. Esfahanian and S. L. Hakimi, "Fault-Tolerant Routing in De Bruijn Communication Networks," *IEEE Trans. on. Computers*, vol. 34, no. 9 Sep. 1985, pp. 777-788.
[76] K. Sivarajan and R. Ramaswami, "Multihop Lightwave Networks Based on De Bruijn Graphs," INFOCOM '91, pp. 1001-1011.
[77] M. Eisenberg and N. Mehravari, "Performance of the Multichannel Multihop Lightwave Network Under Nonuniform Traffic," *IEEE JSAC*, vol. 6, no, 7, Aug. 1988, pp. 1063-1078.
[78] E. Ayanoglu, "Signal-Flow Graphs for Path Enumeration and Deflection Routing Analysis in Multihop Networks," GLOBECOM '89, pp. 1022-1029.
[79] T. G. Robertazzi, "Toroidal Networks," *IEEE Commun. Mag.*, vol. 26, no. 4, June 1988, pp. 45-50.
[80] G. E. Myers and M. E. Zarki, "Routing in TAC a Triangularly Arranged Network," INFOCOM '90, pp. 481-486.
[81] A. G. Greenberg and J. Goodman, "Sharp Approximate Models of Adaptive Routing in Mesh Networks," Teletraffic Analysis and Computer Performance Evaluation, (Elsevier 1986), pp. 255-270.
[82] T. Kubo and K. Yoguchi, "Highway Transfer: A New Forwarding Technique for Real-Time Applications," INFOCOM '90, pp. 403-408.
[83] J. S. K. Wong and Y. Kang, "Distributed and Fail-Safe Routing Algorithms in Toroidal-Based Metropolitan Area Networks," *Computer Networks and ISDN Systems*, vol. 18, 1989/90, pp. 379-391.
[84] T. D. Todd and A. M. Bignell, "Performance Modelling of SIGnet MAN Backbone," INFOCOM '90, pp. 192-199.
[85] H. G. Badr and S. Podar, "An Optimal Shortest-Path Routing Policy for Network Computers with Regular Mesh-Connected Topologies," *IEEE Trans. on Computers*, vol. 38, no. 10, Oct. 1989, pp. 1362-1371.
[86] N. Hutchinson, T. Patten, and B. Unger, "The Flooding Sink: A New Approach to Local Area Networking," *Computer Networks and ISDN Systems*, vol. 11, 1986, pp. 1-14.
[87] H. K. Pung *et al.*, "Arbnet+: An Experimental Mesh-like Local Area Network," SICON '89, Singapore, pp. 301-306.
[88] H. K. Pung *et al.*, "Performance of Arbnet from the Logical Link Control Point of View," Singapore ICCS '90, pp. 1133-1137.
[89] D. J. Farber and G. M. Parulkar, "A Closer Look at Noahnet," SIGCOMM '86, pp. 205-213.
[90] O. Lesser and R. Rom, "Routing by Controlled Flooding in Communication Networks," INFOCOM '90, pp. 910-917.
[91] Y. Azar, J. Naor, and R. Rom, "Routing Strategies for Fast Networks," INFOCOM '92, 170-179.
[92] P. Tran-Gia and R. Dittmann, "Performance Analysis of the CRMA-Protocol in High-Speed Networks," Univ. of Wurzburg, Institute of Computer Science Research Report Series, Report No. 23, December 1990.

Biographies

CESUR BARANSEL received B.Sc. and M.Sc. degrees in computer engineering from Hacettepe University, Ankara, Turkey in 1985 and 1988, respectively, and a Ph.D. degree in computer sceince from the University of Alberta, Canada in 1994. His research interests are in commuications networks, parallel and distributed computing, and machine translation.

PAWEL GBURZYNSKI received M.Sc. and Ph.D. degrees in computer science from the University of Warsaw, Poland, in 1976 and 1982, respectively. Before coming to Canada in 1984, he was a research associate, systems programmer, and consultant in the Department of Mathematics, Informatics, and Mechanics at the University of Warsaw. Since 1985 he has been with the Department of Computing Science, University of Alberta, where he is an associate professor. His research interests are in communications networks, operating systems, simulation, and performance evaluation. He authored LANSF and SMURF — software packages for modeling communication protocols.

Wlodek Dobosiewicz received M.Sc. and Ph.D. degrees in computer science from the University of Warsaw, Poland. He has been an associate professor at the University of Alberta from 1985 to 1994, when he moved to Monmouth University, West Long Branch, New Jersey. He is currently working on network architecture and MAC protocols for high bandwidth computer networks. He has also published on sorting.

Perhaps the cost of using fixed routing tables describing exactly the network topology will be paid off by the real simplicity of the routing rules and the flexibility of the network structure.