Universal Continuous Routing Strategies

Christian Scheideler and Berthold Vöcking*
Department of Mathematics and Computer Science
and Heinz Nixdorf Institute, University of Paderborn
33095 Paderborn, Germany

November 28, 1999

Abstract

We analyze universal routing protocols, that is, protocols that can be used for any communication pattern in any network, under a stochastic model of continuous message generation. In particular, we present two universal protocols, a store-and-forward and a wormhole routing protocol, and characterize their performance by the following three parameters: the maximum message generation rate for which the protocol is stable, the expected delay of a message from generation to service, and the time the protocol needs to recover from worst case scenarios. Both protocols yield significant performance improvements over all previously known continuous routing protocols. In addition, we present adaptations of our protocols to continuous routing in node-symmetric networks, butterflies, and meshes.

1 Introduction

A fundamental problem in any parallel or distributed system is the efficient communication of data between processors. While it is possible to design a specific routing algorithm for each possible interconnection network, a much more general approach is to create a single *universal* routing algorithm that can be used in any network [LMRR94,GO93,CMSV96]. In addition to providing a unified approach to routing in standard networks, universal routing algorithms are ideally suited to routing in irregular networks that are used in wide-area networks and that arise when standard networks develop faults. Furthermore, universal routing places no restrictions on the pattern of communication that is being implemented (such as requiring that it forms a permutation).

Most parallel and distributed systems utilize either *store-and-forward* or *wormhole* routing. In store-and-forward routing, each message can cross a single link in unit time. The store-and-forward model is a standard model which has been widely used to study routing and other problems in parallel computers (see, e.g. [L92]). In wormhole routing, messages are sent as *worms*, each consisting of a sequence of fixed size units called *flits*. Wormhole routing is an extremely popular strategy for data movement in parallel computers and is used in a variety of machines including the Intel Paragon, CRAY T3D, MIT J-Machine, and Stanford DASH.

In this paper we will present and analyze universal on-line algorithms for both store-and-forward and wormhole routing under a stochastic model of continuous message generation.

* email: {chrsch,voecking} @uni-paderborn.de, fax: +49-5251-606482. Supported in part by DFG-Sonderforschungsbereich 376 "Massive Parallelität: Algorithmen, Entwurfsmethoden, Anwendungen", by EU ESPRIT Long Term Research Project 20244 (ALCOM-IT), and by DFG Leibniz Grant Me872/6-1.

1.1 The Routing Models

The interconnection network will be modeled as an undirected graph $G = (V, E)$ where each edge in E consists of two *links*, one in each direction. Each generated message is moved into the *injection buffer* of its *source node*, i.e., the node where the message was generated. Each message is assigned a *routing path* along which it has to be moved forward to its *destination node*, where it is stored in a *delivery buffer*. We require the routing paths to be *simple*, that is, no link appears more than once in any path. Our routing protocols are *oblivious* which means that the routing paths are specified at generation time and cannot be modified during the routing. We assume that the routing paths are described within the messages. Usually, the nodes of the network are able to decide which link to use for a given destination. In this case it is sufficient to store the destination instead of the whole path in the message.

When the store-and-forward routing model is used, it will be assumed that each link has unlimited *link buffers* and can forward one message (which we call *packet*) per time step. Each packet must be stored in the buffer associated with a link prior to crossing the link. At most one packet can cross a link in unit time.

When the wormhole routing model is used, it will be assumed that every message is routed as a worm consisting of fixed size units called *flits*. The *length* of a worm is defined as the number of flits it contains. (In order to simplify our analyses, we always assume in the following that all worms have the same length, denoted by L. This, however, can be relaxed as we will point out in Section 3.3.) The first flit is called the *head* and the remaining flits are called the *body* of the worm. During the routing, a worm occupies a contiguous sequence of links along its path, one flit per link. We assume that at most one flit from each of at most B different worms can cross a link in unit time. While many papers on wormhole routing assume that the parameter B (which we call the *link bandwidth*) equals one, most parallel machines that support wormhole routing multiplex several worms over each link in order to prevent deadlocks and/or improve performance [DS87, D90]. Furthermore, the use of optical links with extremely high bandwidths (see [SM93]) and their ability to support ATM-based B-ISDN networks with multiple virtual channels per link indicate that link bandwidths greater than one will become increasingly common in the near future. None of our wormhole routing algorithms ever delays worms that encounter congestion. As a result, our algorithms can operate without any buffers for delayed flits (which is a significant advantage when optical links are used, as the flits do not have to be converted to and from electronic form for buffering).

1.2 The Continuous Message Generation Model

The messages are generated by a set of N *generators*, each of them mapped to one of the nodes in the network. We allow any relationship between the number of generators and the number of nodes of the network. Furthermore, we place no restrictions on the distribution of the generators among the nodes. That is, one node could have several generators, whereas another node may have none. So a generator may represent a thread or process, whereas a node may represent a processor. In each time step, each generator g placed at a node v creates a message with some probability p_g. This probability is called the *injection rate* of g. For each message, the generator randomly selects a destination and a routing path from v to this destination according to an arbitrary probability distribution. The random generator process has to match only the following two conditions.

- Each generator is operating independently from other generators.
- The generation of a message and its routing path is independent from previous time steps.

Note that we do not demand that the destinations are chosen uniformly from the set of all nodes, or that packets with same source and destination node follow the same routing path.

The *injection rate of a node* is defined to be the sum over all injection rates of generators placed on the node. Note that we allow a node to have an injection rate greater than 1. This is useful, since we allow each node to send out a packet (or B flits) along each outgoing link in one time step.

Define λ_e to be the expected number of messages generated in a time step that contain the link e in their routing paths. Of course, λ_e is determined by the injection rates at the nodes and probability distributions for selecting the routing paths. We define the *link load* λ to be the maximum over all λ_e for all links e in the network.

A protocol is called *stable* for a given link load λ if the number of messages in the injection (and link) buffers does not grow unboundedly with time. We are interested in protocols that are stable for high link loads. Of course, since a link can transport at most one packet per step, we require $\lambda \leq 1$ in case of packet routing; and since a link can transport no more than B flits per step, we require $\lambda \leq B/L$ in case of wormhole routing. Our main interest is to guarantee short delays for the messages, i.e., we aim to minimize the time from generation to service for each message.

1.3 Previous Results

To our knowledge, all previous results about continuous routing assume that the path a message has to take in order to reach its destination is specified at generation time and not allowed to be changed during the routing. Within this class of routing schemes, usually the following two scenarios are studied.

The first is that there exist no buffers apart from the injection and delivery buffers. Hence, if too many messages want to use the same link at the same time, some have to be eliminated and therefore rerouted. Protocols using this strategy are called *backoff protocols*. In order to guarantee a small contention, they heavily rely on sending messages with random startup times. The second scenario is that every link has a buffer in which messages can be stored. Within this scenario, random startup times are not needed, but usually only a *contention resolution rule* deciding which message to forward if more than one want to use the same edge at the same time.

Backoff protocols have been usually studied for the situation that some set of clients is connected via a bus to one server. The most popular protocol that is used for contention resolution on an Ethernet is the *Binary Exponential Backoff Protocol* of Metcalfe and Boggs [MB76]. In this protocol, a counter is associated to each request which keeps track of the number of times the request has failed to reach the server. In case that a request has already failed b times, it is retransmitted after t steps, where t is chosen uniformly at random from the set $\{1, \ldots, 2^b\}$ (in practice, t is chosen uniformly at random from $\{1, \ldots, 2^{\min\{10,b\}}\}$). In [GGM88], Goodman *et al* modify the protocol in a way that if a request has already failed b times to reach the server, it is retransmitted on each successive step with probability 2^{-b}. They show that the modified protocol is stable as long as the sum of the request rates over all clients (denoted by λ) is sufficiently small. Håstad, Leighton and Rogoff [HLR87] prove that if $\lambda > 1/2$, the modified protocol is unstable. However, they show that any (modified) superlinear polynomial backoff protocol (in which a client retransmits with probability $(b+1)^{-\alpha}$, $\alpha > 1$) is stable as long as $\lambda < 1$. In [GM96], Goldberg and MacKenzie generalize the result in [HLR87] by showing that any superlinear polynomial backoff protocol is also stable for contention resolution with multiple servers, that is, each server handles independently from the other servers contention in the same way as an Ethernet channel. Raghavan and Upfal also consider the problem of contention resolution with multiple servers [RU95]. They describe a contention-resolution protocol that is stable as long as the sum of the request rates associated with any client or server is bounded from above by a constant $\lambda' < 1$. The expected waiting time of a message in their protocol is $O(\log N)$. This is much smaller than the expected waiting time of messages in any previously studied backoff protocol, which they

show to be $\Omega(N)$. In [PS95], Paterson and Srinivasan gave a protocol for the same problem with $O(1)$ expected waiting time. Both protocols in [RU95] and [PS95] have the drawback that, like the modified backoff protocols, the protocols require random number generations on each time step.

There has recently been a lot of progress also for continuous routing in fixed connection networks. First approaches in this field concentrated on analyzing the *throughput* of a strategy, that is, the expected number of messages that reach their destination in any time step if every node generates a new message with a random destination at every time step and messages that cannot be forwarded or stored because of limited link bandwidth or buffer size are eliminated and not rerouted. Most previous work in this area focused on butterfly networks. Suppose each of the n input nodes on the highest level of the butterfly decides to send a message to a random output on the lowest level. If more than B of the n messages attempt to traverse the same link, then B of them are arbitrarily selected to move forward and the others are eliminated. Early work on estimating the throughput for this situation was done by Patel [P81], who derived a recursive formula for it. Koch [K88] gave a closed formula on the througput. For constant bandwidth B, he showed that the throughput in the butterfly is $\Theta(n/(\log n)^{1/B})$. A similar result is proved by Rehrmann, Monien, Lüling, and Diekmann for buffered bufferflies with bandwidth 1 [RMLD96]. They showed that the throughput is $\Theta(n/(\log n)^{1/2})$, if each link has a buffer of size 1. Recently, Datta and Sitaraman [DS97] generalized their result to link buffers of arbitrary size $q \geq 1$. They derive precise bounds for the expected throughput, the packet loss rate, and the expected delay of a packet. A survey of some of these and similar results is given in [AM95].

The first rigorous continuous routing analysis is due to Leighton. In [L90], he studies the average case behavior of greedy routing strategies on $n \times n$-meshes. He considers the case where each injected packet has a random destination and packets are injected at each node according to a Bernoulli distribution with rate $\gamma < 4/n$. For this situation, he can show that for the "farthest-to-go" contention resolution rule the mesh is stable and with high probability the packets encounter a delay of at most $O(\log n)$. These results are strengthened by Kahale and Leighton in [KL95], where they show that the same results also hold for any contention resolution rule. Mitzenmacher [M94] is able to provide nearly matching upper and lower bounds for the average delay of packets in a mesh, using the FIFO rule. In particular, as the network load reaches capacity, his upper and lower bounds differ by a factor of at most 8/3. Broder and Upfal investigate continuous packet routing on the $n \times n$-torus without link buffers [BU96]. They show that the expected delay of each packet is $O(n)$ for an injection rate of γ/n at every node with γ being a small constant. Of course, this delay is optimal, but a generalization of their analysis to higher dimensional tori or other networks seems to be difficult.

Performance analyses for continuous routing on butterfly (or banyan) networks are given in [YLL90] and [TRH91]. Both papers give complex recursive formulas (but no closed formula) on the average delay for continuous store-and-forward routing with arbitrary buffer size. Their analyses are based on Markov Chain theory. In [ST91], Stamoulis and Tsitsiklis show that the average delay in the n-input butterfly with unbounded buffers is $O(\log n)$ if the injection rate at the inputs is smaller than 2. Similar results are shown for the hypercube. Their anlaysis is based on queueing theory. In particular, they derive results for the butterfly and hypercube by studying layered *Markovian networks*. (A network is called Markovian if the next link to be traversed is a random function of the last link traversed and is independent of the packet identity and the previous route.) Harchol-Balter and Wolfe [HW95] show that the results, Stamoulis and Tsitsiklis obtain for layered Markovian networks hold for any Markovian network. They also show that the technique introduced in [ST91] cannot be extended to non-Markovian networks by providing a counter example.

Whereas all results above are based on a stochastic model of continuous message generation, recently a new model called *adversarial queueing theory* emerged. This approach was introduced by Borodin *et al* in [BKR96]. Their model permits an adversary to demand network bandwidth up to a prescribed *injection rate*. For $\epsilon > 0$, they say that an adversary *injects at rate* $1-\epsilon$ if it generates requests subject to

the following condition: during any t consecutive steps, the adversary can make at most $t(1-\epsilon)$ requests. A request is defined as an arbitrary set of edge-disjoint paths. Along each of these paths a packet has to be sent. In [BKR96], Borodin *et al* show several stability results for greedy protocols on DAGs and directed cycles. Andrews *et al* [AAF96] extend their results by showing, for instance, that there exist simple greedy protocols that are stable against an adversary for all networks. The advantage of the model of adversarial queueing theory over a stochastic message generation model is that the results are more robust in that they do not hinge upon particular probabilistic assumptions. Its disadvantage (at least for deterministic adversaries) is that the bounds on the delays of the packets are bounds for worst case scenarios. Therefore it can only be used to compare contention resolution rules under the worst case, which might rarely occur in real systems. So studying deterministic adversaries is useful if the stability of a system has to be maintained under any circumstance, whereas stochastic message generation models are useful for benchmarking systems. Borodin *et al* also defined stochastic adversaries that fill the gap between deterministic adversaries and stochastic message generation models. However, both papers in the field of adversarial queueing theory do not exploit this notion.

1.4 New Results

We introduce universal routing protocols for continuous store-and-forward and wormhole routing. Both protocols are very simple and intuitive. Our main results are bounds on the routing time for the messages under specific injection rates (which implies the stability of the system under these rates).

In Section 2, we describe the store-and-forward protocol for arbitrary networks with unbounded link buffers. Our protocal is able to handle constant link load $\lambda < 1/\mathrm{e}$.* We show that, if all routing paths are shortest paths, the protocol delivers each packet p_0 that has to travel a distance D_0 in time $O(D_0)$, with high probabililty†. Obviously, this result is optimal. Further, we give applications of the protocol to standard networks:

- For the butterfly with n inputs and n outputs, we show for constant injection rates $< 2/\mathrm{e}$ at the inputs that each packet is delivered in optimal time $O(\log n)$, w.h.p.
- For node-symmetric networks with diameter D, we show for injection rates $< 1/(\mathrm{e} \cdot D)$ at every node that each packet is delivered in time $O(D)$, w.h.p. This result is optimal for bounded-degree networks, since the above injection rates yield constant link load on these networks.

In Section 3, we introduce the wormhole protocol. For $\lambda = \Theta(\frac{B}{L \cdot D^{1/B}})$, we show that the expected routing time for a message is $O(D + L)$, where D is the maximum length of all routing paths, B the bandwidth and L the length of the worms. Apart from the injection and delivery buffers at the source and destination nodes, the protocol needs no buffers. Of course, the above routing time is optimal, and the link load is optimal for $B \geq \log D$. For several standard networks, the protocol improves all known results:

- For the butterfly with n inputs and n outputs, we show for injection rates of $\Theta(\frac{B}{L \cdot (\log n)^{1/B}})$ at the inputs that each message is delivered in expected time $O(L + \log n)$. This result generalizes Koch's lower bound on the throughput [K88] to non-constant bandwidth.
- For the d-dimensional mesh with side length m, we show for injection rates of $\Theta(\frac{B}{L \cdot m \cdot d^{1/B}})$ at each node that each message is delivered in expected time $O(d \cdot m + L)$. For $L = 1$ and $B = 1$, our

*Throughout the paper, e denotes the Eulerian number.

†Throughout the paper, the terms "*time T, with high probability*" or "*time T, w.h.p.*" mean "with time $T + t$, with probability at least $1 - 2^{-\Omega(t)}$".

strategy is very similar to the one of Broder and Upfal for 2-dimensional tori [BU96]. Our result generalizes their result to higher dimensions.

- For node-symmetric networks with diameter D, we show for injection rates of $\Theta(\frac{B}{L \cdot D^{1+1/B}})$ at every node that each message is delivered in expected time $O(D+L)$.

The above routing times are optimal. Moreover, the injection rates are optimal for $B \geq \log\log n$ on the butterfly, $B \geq \log d$ on the mesh, and $B \geq \log D$ on node-symmetric networks with bounded degree.

Note that bandwidth B can be simulated with slowdown B by networks with bandwidth 1 and buffers of size B for worm length $L = 1$. This yields an efficient protocol for continuous store-and-forward routing with bounded buffers.

Since we consider infinite processes it might happen that at some time step the number of messages in the buffers deviates significantly from the expected value which causes a much higher delay for messages generated shortly afterwards then stated above. We consider such events as *worst case scenarios*. In Section 4, we describe how to modify the store-and-forward and wormhole protocol such that they can recover quickly from any worst case scenario concerning the generation of messages.

An important parameter to measure the performance of protocols after worst case scenarios is called *recovery time*. We say that a protocol *recovers* within t time steps if, for any message generated at least t time steps after any worst case scenario, the expected routing time deviates by at most a constant factor from its expected routing time in the steady state.

In case of store-and-forward routing we develop a protocol that recovers within $O(D + \log(N + |E|))$ time steps; and in case of wormhole routing we develop a protocol that recovers within $O((D + L)\log(N + D + L))$ time steps. Both results are the first for store-and-forward and wormhole routing in arbitrary networks.

2 The Store-And-Forward Protocol

In this section we introduce a routing protocol for store-and-forward routing with unbounded buffers.

Suppose we are given a network of arbitrary size and degree, and suppose we have N packet generators in this network. Each generator has a unique ident-number from $[N]$[‡]. We denote the *birth date* of a packet p which is the time step in which it was generated by $\mathrm{birth}(p)$, and the ident number of its generator by $\mathrm{id}(p)$. We assume that each packet is sent along a shortest path to its destination. Further, suppose $K \geq 1$ and $m \geq 2$ are integers.

2.1 Description of the Protocol

During the routing all packets that wait for moving forward over a link e are stored in a link buffer $\mathcal{Q}_e$. In each step, one packet from each non-empty link buffer is forwarded over the respective link. The other packets are *delayed* for one step.

The decision which of the contending packets of a link buffer $\mathcal{Q}_e$ to forward over e is made with the help of *ranks* depending on the birth dates of the packets. Define the initial rank of a packet p by $\mathrm{rank}(p) := \mathrm{birth}(p) \cdot K + k(p)$ with $k(p)$ randomly chosen from $[K]$. Whenever a packet traverses a link, its rank is increased by $m \cdot K$. If two or more packets in a buffer $\mathcal{Q}_e$ are contending to move forward over the link e, then one of those with minimum rank is chosen. Thus, for each outgoing link e, a step looks like this:

(1) choose a packet $p \in \mathcal{Q}_e$ with minimum rank,

[‡] Throughout the paper, $[N]$ denotes $\{0, \ldots, N-1\}$.

(2) increase the rank of p by $m \cdot K$, and

(3) move p forward over the link e.

In order to break ties, if there are several packets with the same minimum rank, then the one from the generator with smallest ident number is chosen.

In the following, we denote the rank of p while waiting for moving forward over link e by $\text{rank}^e(p)$. Further, we define the *ident rank* of p at e as $\text{rank}^e(p) + \frac{\text{id}(p)}{N}$. This ident rank is denoted by $\text{id-rank}^e(p)$. Note that, at each edge, the ident ranks of all packets are distinct. The protocol ensures that whenever a packet p delays a packet p' at a link e it is $\text{id-rank}^e(p) < \text{id-rank}^e(p')$.

2.2 Analysis of the Store-And-Forward Protocol

Our analysis is similar to the one for static store-and-forward routing in [MV95]. It is based on a delay sequence argument.

Definition 2.1 (delay sequence)
A delay sequence of length s consists of

- *s collision links $e_1, \ldots, e_s$,*
- *s delay packets $p_1, \ldots, p_s$,*
- *s integers $r_1, \ldots, r_s \geq 0$, and*
- *s integers $\ell_1, \ldots, \ell_s \geq 0$.*

We say a packet p_0 is delayed by a delay sequence if the following conditions are satisfied:

(1) packet p_i delays packet p_{i-1} at link e_i, for $1 \leq i \leq s$;

(2) the number of links on the routing path of p_0 from e_1 to the destination of p_0 is ℓ_1 (inclusive e_1), and for $2 \leq i \leq s$, the number of links on the routing path of p_{i-1} from e_i to e_{i-1} is ℓ_i (inclusive e_i and exclusive e_{i-1}), and

(3) $r_i = \text{rank}^{e_i}(p_{i-1}) - \text{rank}^{e_i}(p_i)$, for $1 \leq i \leq s$.

Lemma 2.2 *Let p_0 be a packet that has to travel a distance D_0, and suppose p_0 takes $m \cdot D_0 + t$ (or more) steps to reach its destination for $t \geq 1$. Then p_0 is delayed by a delay sequence of length $s \geq \frac{m-1}{m} \cdot t$ with $\sum_{i=1}^{s} r_i \leq \frac{m}{m-1} \cdot K \cdot s$ and $\sum_{i=1}^{s} \ell_i \leq \frac{1}{m-1} \cdot s$.*

Proof. The delay sequence can be constructed as follows. We follow p_0's routing path backwards to the last link at which it was delayed. This link we call e_1. Let p_1 be the packet that caused the delay, since it was preferred against p_0. We now follow the path of p_1 backwards until we reach a link e_2 at which p_1 was forced to wait because of a packet p_2. We change the packet again and follow the path of p_2 backwards. This construction can be continued until we reach a packet p_s that was not delayed in an earlier step. The path from the source of p_s to the destination of p_0 recorded by this process is called *delay path*. It consists of contiguous parts of the delay packets' routing paths. We define the ℓ_i's to be the lengths of these parts as described in Condition (2) of Definition 2.1, and we define the r_i's to be the differences between the respective ranks as described in Condition (3) of this definition.

It remains to prove the lower bound on s and the upper bounds on the sums of the r_i's and ℓ_i's. Define t' to be the number of time steps covered by the above construction. Then

$$t' \geq \text{birth}(p_0) + m \cdot D_0 + t - \text{birth}(p_s) \ , \tag{1}$$

because t' is the time from the birth of packet p_s to the arrival of packet p_0 which happens at step $\text{birth}(p_0) + m \cdot D_0 + t$ or later. Let R_0 denote the rank of p_0 at its destination, and R_s the rank of p_s at its source. Then $\lfloor R_0/K \rfloor = \text{birth}(p_0) + m \cdot D_0$ and $\lfloor R_s/K \rfloor = \text{birth}(p_s)$. The protocol ensures $R_s \leq R_0$. Applying these equations to Equation (1) yields

$$t' \geq \left\lfloor \frac{R_0}{K} \right\rfloor + t - \left\lfloor \frac{R_s}{K} \right\rfloor \tag{2}$$

$$\geq t \ . \tag{3}$$

Define r to be the range of the ranks in the sequence. Then

$$r = R_0 - R_s \overset{(2)}{\leq} K \cdot (t' - t) + (K - 1) \overset{t \geq 1}{\leq} K \cdot t' \ . \tag{4}$$

Define ℓ to be the length of the delay path from the source of p_s to the destination of p_0. Since the ranks in our sequence are increased by $m \cdot K$ on every link on the delay path, we have

$$\ell \leq \frac{r}{m \cdot K} \overset{(4)}{\leq} \frac{t'}{m} \ . \tag{5}$$

Further, we have

$$t' = s + \ell \ , \tag{6}$$

because each time step in our sequence is represented either by a move or a delay. Now, we can calculate the lower bound on the length of our sequence, i.e.,

$$s \overset{(6)}{=} t' - \ell \overset{(5)}{\geq} \frac{m-1}{m} \cdot t' \overset{(3)}{\geq} \frac{m-1}{m} \cdot t \ , \tag{7}$$

the upper bound on the range of the ranks, i.e.,

$$\sum_{i=1}^{s} r_i \leq r \overset{(4)}{\leq} K \cdot t' \overset{(7)}{\leq} \frac{m}{m-1} \cdot K \cdot s \ ,$$

and finally, the upper bound on the length of the delay path, i.e.,

$$\sum_{i=1}^{s} \ell_i \leq \ell \overset{(5)}{\leq} \frac{t'}{m} \overset{(7)}{\leq} \frac{1}{m-1} \cdot s \ .$$

■

Lemma 2.3 *If all routing paths are shortest paths, then the delay packets in a delay sequence delaying p_0 are pairwise distinct, i.e., $p_i \neq p_j$, for $0 \leq i < j \leq s$.*

Proof. Suppose, in contrast to our claim, that there is some packet p appearing twice in the delay sequence. Then there exist i and j with $0 \leq i < j \leq s$ and $p = p_i = p_j$. Thus, the collision links e_j where p delays p_{j-1} and e_{i+1} where p was delayed by p_{i+1} can be found on the routing path of p in this order.

Let ℓ denote the number of links on the routing path between e_j and e_{i+1} (inclusive e_j and exclusive e_{i+1}). Then the rank of p is increased ℓ times by $m \cdot K$ while moving from e_j to e_{i+1}, and therefore,

$$\text{id-rank}^{e_{i+1}}(p) = \text{id-rank}^{e_j}(p) + m \cdot K \cdot \ell \ . \tag{8}$$

On the other hand, each packet p_k delays the packet p_{k+1} at link e_{k+1}. As a consequence, $\text{id-rank}^{e_{k+1}}(p_{k+1}) > \text{id-rank}^{e_{k+1}}(p_k)$, for $i \leq k \leq j-1$. Further, on every link on the delay path from e_j to e_{i+1} the ident-ranks are increased by $m \cdot K$. Thus, we have

$$\begin{aligned} \text{id-rank}^{e_{i+1}}(p) &> \text{id-rank}^{e_j}(p) + m \cdot K \cdot \sum_{k=i+1}^{j} \ell_k \\ &\geq \text{id-rank}^{e_j}(p) + m \cdot K \cdot \ell \ . \end{aligned} \tag{9}$$

Note that $\sum_{k=i+1}^{j} \ell_k \geq \ell$, because the routing path of p is a shortest path.

Clearly, (9) contradicts (8). Consequently, there is no packet that appears twice in the delay sequence. ∎

Theorem 2.4 *Suppose $\lambda < 1/\mathrm{e}$, and suppose all routing paths are shortest paths. Let p_0 be a packet that has to travel a distance D_0. Then p_0 is routed by the above protocol (with suitably chosen K and m) in time $O(D_0)$, w.h.p.*

Proof. Let $DS(m, K, s)$ denote the expected number of delay sequences of length s with $\sum_{i=1}^{s} r_i \leq \frac{m}{m-1} \cdot K \cdot s$ and $\sum_{i=1}^{s} \ell_i \leq \frac{1}{m-1} \cdot s$ with respect to the random packet generation and the random rank selection. $DS(m, K, s)$ can be bounded as follows. There are at most

$$\binom{s + \frac{m}{m-1} \cdot K \cdot s}{s}$$

possibilities to select the r_i's and

$$\binom{s + \frac{1}{m-1} \cdot s}{\frac{1}{m-1} \cdot s}$$

possibilities to select the ℓ_i's.

We now count inductively the expected number of choices for the delay packets and collision links under the assumption that the r_i's and ℓ_i's are fixed. Suppose e_{i-1} and p_{i-1} are determined for $1 \leq i \leq s$. (In order to have a starting point for our induction we define e_0 to be the last link on the routing path of p_0.) Then e_i is determined as well, since we can follow backwards the routing path of p_{i-1} from e_{i-1} until we reach e_i at distance ℓ_i. Thus, e_i and the rank of p_i at this edge are fixed, since the r_i's and ℓ_i's determine the rank of p_i at edge e_i.

What is the expected number of canditates for p_i under the assumption that e_i and the rank of p_i at this edge are specified? – It is at most λ/K. This is because the rank of a packet includes the birth date of p_i and a random number chosen from $[K]$, and the expected number of packets which are generated in a fixed step with a fixed rank is bounded by λ/K. Note that this bound holds even if we

assume that $p_0, \ldots, p_{i-1}$ are already specified, because we know from Lemma 2.3 that p_i is distinct from these packets.

Consequently,

$$DS(m,K,s) \leq \binom{s + \frac{m}{m-1} \cdot K \cdot s}{s} \cdot \binom{s + \frac{1}{m-1} \cdot s}{\frac{1}{m-1} \cdot s} \cdot \left(\frac{\lambda}{K}\right)^s .$$

Applying the equation $\binom{a}{b} \leq \left(\frac{\mathrm{e} \cdot a}{b}\right)^b$ gives

$$DS(m,K,s) \leq \left(\mathrm{e} \cdot \left(\frac{1}{K} + \frac{m}{m-1}\right) \cdot (\mathrm{e} \cdot m)^{\frac{1}{m-1}} \cdot \lambda\right)^s .$$

Now we choose m and K such that

$$\left(\frac{1}{K} + \frac{m}{m-1}\right) \cdot (\mathrm{e} \cdot m)^{\frac{1}{m-1}} < \frac{1}{\mathrm{e} \cdot \lambda} \ .$$

This can be done because $\lambda < \frac{1}{\mathrm{e}}$ and the terms $\frac{1}{K} + \frac{m}{m-1}$ and $(\mathrm{e} \cdot m)^{\frac{1}{m-1}}$ approach 1 as m and K increase. Define

$$\epsilon := \mathrm{e} \cdot \left(\frac{1}{K} + \frac{m}{m-1}\right) \cdot (\mathrm{e} \cdot m)^{\frac{1}{m-1}} \cdot \lambda \ .$$

Then $DS(m,K,s) < \epsilon^s$, for constant $\epsilon < 1$.

From Lemma 2.2 we can conclude that the probability that the routing of packet p_0 takes $m \cdot D_0 + t$ (or more) steps is bounded by

$$\sum_{s=\lceil \frac{m-1}{m} \cdot t \rceil}^{\infty} DS(m,K,s) \leq \sum_{s=\lceil \frac{m-1}{m} \cdot t \rceil}^{\infty} \epsilon^s \leq 2^{-a \cdot t + b}$$

for suitable constants $a > 0$ and b. ∎

2.3 Applications

The store-and-forward protocol can be applied to routing in the butterfly as, for instance, defined in [L92]. This gives the following result.

Theorem 2.5 *Suppose we are given a butterfly network with n input nodes and n output nodes. Let the injection rate at every input node be $\gamma < 2/\mathrm{e}$. Further, suppose each packet is sent along the unique shortest path to a random destination chosen independently and uniformly from the set of outputs. Then each packet is delivered in time $O(\log n)$, w.h.p.*

Proof. For symmetry reasons, the link load λ_e is the same for all edges e on the same level. Since the expected number of packets that are generated in one step is $\gamma \cdot n$ and the number of edges on a level is $2n$, we have $\lambda = \gamma \cdot n/(2n) < 1/\mathrm{e}$. Hence, applying Theorem 2.4 yields the above theorem. ∎

A network $G = (V, E)$ is *node-symmetric* if for any pair u, v of nodes in G there exists an automorphism $\varphi : V \to V$ mapping u to v such that for the graph $G_\varphi = (V, E_\varphi)$ with $E_\varphi = \{\{\varphi(x), \varphi(y)\} \mid \{x, y\} \in E\}$ it holds that $G_\varphi = G$. Applying the store-and-forward protocol to node-symmetric networks gives the following result.

Theorem 2.6 *Suppose we are given a node-symmetric network with diameter D, and suppose the injection rate at every node is $\gamma < 1/(\mathrm{e} \cdot D)$. Let $\mathcal{P}(u,v)$ denote the set of all shortest paths from u to v, for all nodes u and v. Further, suppose that each packet is sent to a random destination which is chosen uniformly from the set of all nodes, and suppose that the routing path of a packet with source u and destination v is chosen from $\mathcal{P}(u,v)$ by some random process which is independent of the node labels. Then each packet is delivered in time $O(D)$, w.h.p.*

Proof. For a node v, define $\lambda_v := \sum \lambda_e$ over all outgoing links e. Because of symmetry reasons λ_v is the same for all nodes v in the network, namely at most $n \cdot D \cdot \gamma / n = D \cdot \gamma$. Consequently, $\lambda \leq D \cdot \gamma < \frac{1}{\mathrm{e}}$. Therfore, applying Theorem 2.4 yields the above theorem. ∎

Instead of determining the routing paths at the source nodes beforehand, the routing paths can be selected *on-line*, i.e., each node chooses a suitable link for transmitting a packet just when it arrives there. For instance, this link can be chosen randomly, uniformly, and independently from the set of outgoing links which belong to a shortest path to the destination of the packet.

3 The Wormhole Protocol

In this section we introduce a routing protocol for wormhole routing. The protocol uses no buffers except for the injection and delivery buffers.

Suppose we are given a network of arbitrary size and degree. In this network we have N message generators. At each time step, each generator generates a message of length L with some probability that is independent of the other generators and the generation of messages at previous time steps. These messages have to be routed as worms of length L along their routing paths. It is not allowed to divide worms into subworms of smaller length. We assume that the network has bandwidth B. That means that up to B flits of B different worms can cross a link in one time step.

In the following, let $\mathrm{birth}(\omega)$ denote the birth date of worm ω, and D be an upper bound on the lengths of the routing paths.

3.1 Description of the Wormhole Protocol

The main idea of our protocol is that each worm repeatedly tries to route to its destination until it is successful. We say a flit is *blocked* if it cannot cross a link, since B flits from B other worms occupy this link. If the kth flit of a worm is blocked at a link, then the flits from k to L are discarded when reaching this link. The flits 1 to $k-1$ do not recognize that the kth flit is blocked. If a worm completely reaches its destination then an acknowledgement is sent back along the routing path to its source.

In a *trial* for a worm ω, the source node of ω tries to send a copy of the worm to its destination. The trial is called *successful* if neither one of the flits nor the acknowledgement flit is blocked on the routing path. A worm is stored in the injection buffer at its source node until it has a successful trial. Note that the protocol never delays a worm on its routing path. Hence, no link buffers are needed, and a trial takes at most $R := 2D + L - 1$ time steps. The source node starts a trial for ω every R steps. Hence, at most one copy of ω is active at the same time. The first trial starts at time $\mathrm{birth}(\omega)$. In the following we identify a worm with its active copy, and we say *a worm is blocked* if one of its flits or its acknowledgement is blocked.

It remains to specify which of the worms are allowed to move forward, if more than B worms are contending for using a link. For each worm ω, define $\mathrm{rank}(\omega) := \mathrm{birth}(\omega) + r(\omega)$ to be the *rank* of ω, where $r(\omega)$ is randomly chosen from $[R]$. Then in case of contention we prefer worms with smaller

ranks. If there are worms with equal ranks then we prefer the B worms whose generators have the smallest ident numbers.

3.2 Analysis of the Wormhole Protocol

Our analysis is similar to the one for static wormhole routing in [CMSV96]. It is based on a delay sequence argument.

Definition 3.1 ((s, B, R)-delay sequence)
An (s,B,R)-delay sequence consists of

- *s collision links $e_0, \ldots, e_{s-1}$, and*
- *$s \cdot B$ delay worms $\omega_1, \ldots, \omega_{s \cdot B}$.*

We say a worm ω_0 is delayed by this sequence, if the following conditions are met

(1) worm $\omega_{i \cdot B}$ is blocked at link e_i by the B worms $\omega_{i \cdot B+1}, \ldots, \omega_{(i+1) \cdot B}$ for $0 \leq i \leq s-1$;

(2) $\text{rank}(\omega_{s \cdot B}) \leq \ldots \leq \text{rank}(\omega_1) \leq \text{rank}(\omega_0)$; and

(3) $\text{rank}(\omega_0) - \text{rank}(\omega_{s \cdot B}) \leq 2sR$.

We call s the length of the sequence.

Lemma 3.2 *Suppose ω_0 is a worm that had at least t unsuccessful trials. Then ω_0 is delayed by an (s,B,R)-delay sequence with length $s \geq t/3$.*

Proof. The delay sequence can be constructed as follows. Let e_0 be the link where ω_0 had its last collision. Define $\omega_1, \ldots, \omega_B$ to be the worms that blocked ω_0 in this link with $\text{rank}(\omega_0) \geq \text{rank}(\omega_1) \geq \ldots \geq \text{rank}(\omega_B)$. Suppose ω_B is not in its first trial when blocking ω_0. Then there must be a link e_1 where ω_B has been blocked in the preceding trial. Define $\omega_{B+1}, \ldots, \omega_{2B}$ to be the worms that caused the blocking with $\text{rank}(\omega_B) \geq \text{rank}(\omega_{B+1}) \geq \ldots \geq \text{rank}(\omega_{2B})$. Of course, we can continue this construction until we reach a worm $\omega_{s \cdot B}$ which is in its first trial. Clearly, our construction satisfies Condition (1) and (2) of Definition 3.1.

Let t_1 denote the time step of the collision in e_0 and t_s denote the time step of the collision in e_{s-1}. Since the time between the collision in link e_{i-1} and the collision in link e_i is at most $2R-1$ (for $1 \leq i \leq s-1$), we get $t_1 - t_s \leq (2R-1)(s-1) \leq 2R(s-1)$. As a consequence,

$$\begin{aligned} \text{rank}(\omega_0) - \text{rank}(\omega_{s \cdot B}) &\leq \text{birth}(\omega_0) + R - \text{birth}(\omega_{s \cdot B}) \\ &\leq (t_1 - (t-1)R) + R - (t_s - R) \\ &\leq t_1 - t_s + 2R \;\leq\; 2Rs \;. \end{aligned}$$

Hence, Condition (3) is satisfied as well.

Finally, we have to show that our delay sequence has the required length. Of course, if $1 \leq t \leq 3$ then $s \geq 1 \geq t/3$. Applying the above equation $t_1 - t_s \leq 2R(s-1)$ yields

$$\begin{aligned} s - 1 &\geq \frac{t_1 - t_s}{2R} \\ &\geq \frac{(\text{birth}(\omega_0) + (t-1)R) - (\text{birth}(\omega_{s \cdot B}) + R)}{2R} \\ &\geq \frac{(\text{rank}(\omega_0) - R + (t-1)R) - (\text{rank}(\omega_{s \cdot B}) + R)}{2R} \\ &\geq \frac{t-3}{2} \;. \end{aligned}$$

Hence, we have $s \geq \frac{t-3}{2} + 1 \geq \frac{t}{3}$ for $t \geq 3$. ∎

Theorem 3.3 *Let $\lambda \leq \frac{B}{12eL(2D)^{1/B}}$ and $B \leq R$. Suppose ω_0 is a worm generated at an arbitrary time step. Then the expected number of unsuccessful trials for ω_0 is at most $\frac{3}{2^B-1}$. Further, the probability that ω_0 needs more than t trials is at most $2^{-Bt/3+1}$.*

Proof. First we estimate the expected number of (s, B, R)-delay sequences. Of course, there are $\binom{2Rs+sB}{sB} \leq \left(\frac{3eR}{B}\right)^{sB}$ ways to choose the ranks of the delay worms.

Now, suppose $\omega_{i \cdot B}$ is fixed for some $0 \leq 1 \leq s-1$. Then there are $2D$ possibilties to choose e_i, since it is a link on the routing path of this worm. We know that at least one flit of each of the worms $\omega_{i \cdot B+1}, \ldots \omega_{(i+1) \cdot B}$ has to collide with the fixed worm $\omega_{i \cdot B}$. Thus, there is a time interval of $2L-1$ steps in which the head of each of these worms has to travel over link e_i. As a consequence, the expected number of B-tuples with distinct candidates for $\omega_{i \cdot B+1}, \ldots \omega_{(i+1) \cdot B}$ is at most $((2L-1) \cdot \lambda/R)^B$. This is because λ/R is an upper bound on the expected number of worms with fixed rank passing through a fixed link in a fixed time step.

Thus, the expected number of (s, B, R)-delay sequences is at most

$$\left(\frac{3eR}{B}\right)^{sB} (2D)^s \left(\frac{(2L-1)\lambda}{R}\right)^{sB} \leq \left(\frac{3e \cdot (2D)^{1/B} \cdot (2L-1)\lambda}{B}\right)^{sB} \leq 2^{-Bs} \tag{10}$$

for $\lambda \leq \frac{B}{12eL(2D)^{1/B}}$. Let S be the maximum length over all delay sequences delaying ω_0. Then the expected value for S is at most

$$\sum_{s=1}^{\infty} s \left(2^{-Bs} - 2^{-B(s+1)}\right) = \sum_{s=1}^{\infty} 2^{-Bs} = \frac{2^{-B}}{1-2^{-B}} = \frac{1}{2^B - 1} .$$

Let $E[S]$ denote the expected value of S. From Lemma 3.2 we conclude that the expected number of unsuccessful trials for ω_0 is at most $3 \cdot E[S]$. Thus, it is at most $\frac{3}{2^B-1}$.

Finally, we have to bound the probability that ω_0 needs at least t trials. Applying Lemma 3.2 and Equation (10) yields that this probability is at most $\sum_{s=\lceil t/3 \rceil}^{\infty} 2^{-Bs} \leq 2^{-Bt/3+1}$. ∎

3.3 Applications

In this subsection we give some applications of the wormhole routing protocol to standard networks. The proofs for the following two theorems are analogous to those in Section 2.3.

Theorem 3.4 *Suppose, we are given a butterfly network with n input nodes and n output nodes. Let the injection rate at every input node be at most $\frac{B}{12eL(2\log n)^{1/B}}$. Further, suppose each worm is sent to a random destination chosen independently and uniformly from the set of outputs along the unique shortest path. Then each worm is delivered in expected time $O(L + \log n)$.*

Theorem 3.5 *Suppose, we are given a node-symmetric network (as defined in Section 2.3) with diameter D. Let the injection rate at every node be at most $\frac{B}{12eL(2D)^{1/B}}$. Further, suppose that the routing paths are selected as described in Theorem 2.6. Then each worm is delivered in expected time $O(D+L)$.*

For meshes, we can further improve the maximum injection rate.

Theorem 3.6 *Suppose, we are given a d-dimensional mesh with side length m. Let the injection rate at every node be at most $\frac{B}{12eL \cdot m(2d)^{1/B}}$. Assume each worm is sent to a random destination chosen independently and uniformly from the set of nodes, and assume that each worm is first routed according to dimension 1, then according to dimension 2, and so on. Then each worm is delivered in expected time $O(m \cdot d + L)$.*

Proof. In the following, A denotes an arbitrary linear array of length m in one dimension of the mesh. Let $E[A]$ be the expected number of messages generated at one time step that want to use links of A during the routing. Furthermore, let γ denote the injection rate at each node. Because of symmetry reasons, for uniformly and independently chosen destinations, $E[A]$ is the same for each linear array A. (Note that this argument is not true if we considered links instead of linear arrays.) Since the total number of linear arrays used by messages generated at one time step is at most $n \cdot \gamma \cdot d$ and a d-dimensional mesh of side length m consists of $d \cdot m^{d-1}$ linear arrays, $E[A]$ is at most

$$\frac{n \cdot \gamma \cdot d}{d \cdot m^{d-1}} = \gamma \cdot m$$

for any linear array A.

Regarding linear arrays instead of links in the proof of Theorem 3.3 yields that the expected delivery time in the above theorem holds for a linear array load of $\lambda \leq \frac{B}{12eL(2d)^{1/B}}$. The theorem then follows from the equality $\gamma \cdot m = \lambda$. ∎

This result shows that for the d-dimensional mesh an optimal injection rate can already be reached for $B = \log d$.

Note that all results in this section also hold if the length of the worms is not fixed any longer. Instead, we only need to know an upper bound L for the *expected* length of a worm to be sent out by any node in one time step.

4 Handling Worst Case Scenarios

The drawback of the protocols described above is that their time bounds only hold in the steady state, that is, the number of messages stored in the injection (and link) buffers is close to the expected value. But since we consider infinite processes it might happen that at some time step the number of messages in the buffers deviates significantly from the expected value which causes a much higher delay for messages generated shortly afterwards than stated in Theorem 2.4 and Theorem 3.3. We consider such events as *worst case scenarios*. In this section we describe and analyze one possible solution to handle such situations for store-and-forward and wormhole routing. As before, $G = (V, E)$ denotes the network, N the number of generators, λ the link load, and D the maximum length of the routing paths.

4.1 Store-and-Forward Routing

We slightly modify the protocol of Section 2. Each link now has an additional buffer called *overflow stack*. This stack stores packets that are older than T time steps, i.e., at the end of each time step t, each packet p with $birth(p) \leq t - T$ is removed from the link buffer where it is currently stored and pushed onto an overflow stack. After some time the packets in the overflow stacks are *reinjected* into the respective link buffers, i.e., in each time step, a link reactivates the packet at the top of the stack with probability $\frac{1}{\gamma \cdot |E|}$. This packet is suspended for further $m \cdot T$ time steps before it is moved back to its link buffer with the actual time step as new birth date. T and γ must be chosen sufficiently large.

The exact values are given in the following analysis. As before, the rank of a packet consists initially of the packet's birth date and a random number chosen from $[K]$, and it is increased by $m \cdot K$ whenever the packet moves forward.

Now suppose that an arbitrary scenario happened before time step T_0. Then the following theorem implies that the recovery time of the modified store-and-forward protocol is bounded by $m \cdot T$ (and therefore the protocol is stable).

Theorem 4.1 *Let α, γ, m, and K be sufficiently large constants. Suppose $\lambda < 1/\mathrm{e}$, $T \geq \alpha(D + \log(N + |E|))$, and suppose all routing paths are shortest paths. Let p_0 be a packet which is generated at time $\geq T_0 + m \cdot T$ and has to travel a distance D_0. Then the expected routing time for p_0 is $O(D_0)$. (Under the assumption that p_0 is not moved to an overflow stack, the routing time is $O(D_0)$, w.h.p.)*

Proof. In each step, each of the at most $|E|$ overflow stacks reinjects a packet to a link buffer with probability $1/(\gamma \cdot |E|)$. Define the *effective link load* λ_{eff} to be the maximum expected number of packets generated or reinjected in a time step that want to traverse a link. Then, for $\gamma \geq 2/(1/\mathrm{e} - \lambda)$, it is

$$\lambda_{\text{eff}} < \lambda + |E| \cdot \frac{1}{\gamma \cdot |E|} \leq \frac{1}{\mathrm{e}} - \frac{\frac{1}{\mathrm{e}} - \lambda}{2} < \frac{1}{\mathrm{e}} \ .$$

Let a delay sequence of length s for p_0 be defined as in Definition 2.1. Then the following Lemma holds.

Lemma 4.2 *Let p_0 be a packet that has to travel a distance D_0, and suppose p_0 takes $m \cdot D_0 + t$ (or more) steps to reach its destination for $1 \leq t \leq T - m \cdot D$. Then p_0 is delayed by a delay sequence of length $s \geq \frac{m-1}{m} \cdot t$ with $\sum_{i=1}^{s} r_i \leq \frac{m}{m-2} \cdot K \cdot s$ and $\sum_{i=1}^{s} \ell_i \leq \frac{1}{m-2} \cdot s$.*

Proof. Analogously to the construction in the proof of Lemma 2.2, we follow the routing path of p_0 backwards until we reach a link at which this packet was delayed. This link we call e_1, and the packet that caused the delay we call p_1. Then we follow the path of p_1 backwards until we reach a link e_2 at which p_1 was forced to wait because packet p_2 was prefered against it. Now we change the packet again and follow the path of p_2 backwards. This construction can be continued until we reach either a packet p_s which did not start before T_0 and was not delayed before delaying p_{s-1} (case 1) or was delayed by a packet $p_{s'}$ with birth date smaller than T_0 (case 2). As before, we define the ℓ_i's and r_i's as described in Condition (2) and (3) of the delay sequence definition. Thus, we have found a delay sequence of length s in which none of the packets was generated before T_0.

In case 1, our construction is exactly the same as in the proof of Lemma 2.2, and consequently $s \geq \frac{m-1}{m} \cdot t$, $\sum_{i=1}^{s} r_i \leq \frac{m}{m-2} \cdot K \cdot s$, and $\sum_{i=1}^{s} \ell_i \leq \frac{1}{m-2} \cdot s$.

In case 2, we can bound these terms as follows. Define t' to be the number of time steps covered by the above construction. Then

$$\begin{aligned} t' &\geq \text{birth}(p_0) - \text{birth}(p'_s) \\ &\geq (T_0 + m \cdot T) - T_0 \\ &\geq m \cdot T \qquad (11) \\ &\geq m \cdot t \ . \qquad (12) \end{aligned}$$

Define r to be the range of the ranks in the sequence. The rank of p_0 at its destination is at most $\text{birth}(p_0) \cdot K + K - 1 + m \cdot K \cdot D_0$, and the rank of $p_{s'}$ at its source is at least $\text{birth}(p_{s'}) \cdot K$. Thus,

$$\begin{aligned} r &\leq \text{birth}(p_0) \cdot K + K - 1 + m \cdot K \cdot D_0 - \text{birth}(p_{s'}) \cdot K \\ &\leq K \cdot (\text{birth}(p_0) + 1 + m \cdot D_0 - \text{birth}(p_{s'})) \ . \end{aligned}$$

Define t_0 to be the step in which p_0 reaches its destination. Then $t_0 \geq \text{birth}(p_0) + m \cdot D_0 + 1$. This gives

$$r \leq K \cdot (t_0 - \text{birth}(p_{s'})) = K \cdot t' \ . \tag{13}$$

Define ℓ to be the length of the delay path from the source of $p_{s'}$ to the destination of p_0. Since the ranks in our sequence are increased by $m \cdot K$ on every link on the delay path, we have

$$\ell \leq \frac{r}{m \cdot K} \overset{(13)}{\leq} \frac{t'}{m} \ . \tag{14}$$

Since each time step in our sequence is represented either by one of the $s+1$ delays of the delay packets $p_0, \ldots, p_s$, by one of the at most $T-1$ delays of $p_{s'}$, or by one of the ℓ moves on the delay path, we have

$$t' \leq s + \ell + T \overset{(14)+(11)}{\leq} s + \frac{2t'}{m} \ . \tag{15}$$

Now, we can calculate the lower bound on the length of our sequence, i.e.,

$$s \overset{(15)}{\geq} \frac{m-2}{m} \cdot t' \overset{(12)}{\geq} (m-2) \cdot t \geq \frac{m-1}{m} \cdot t \ ,$$

the upper bound on the range of the ranks, i.e.,

$$\sum_{i=1}^{s} r_i \leq r \overset{(13)}{\leq} K \cdot t' \overset{(15)}{\leq} \frac{m}{m-2} \cdot K \cdot s \ ,$$

and finally, the upper bound on the length of the delay path, i.e.,

$$\sum_{i=1}^{s} \ell_i \leq \ell \overset{(14)}{\leq} \frac{t'}{m} \overset{(15)}{\leq} \frac{1}{m-2} \cdot s \ .$$

■

We devide the routing time of p_0 into contiguous subintervals $\Delta_0, \Delta_1, \ldots$ such that Δ_0 begins with the generation of p_0, Δ_i, for $i \geq 1$, begins when p_0 is moved for the ith time onto an overflow stack, and the last of these subintervals ends with the delivery of p_0.

We first calculate the expected length of Δ_0. The delay sequence in the above lemma fulfills nearly the same conditions as the one in Lemma 2.2 for the protocol in Section 2. In fact, the analysis for the routing time of p_0, under the assumption that p_0 is not moved to an overflow stack, can be completely adapted from this section. Thus, if p_0 is not moved to an overflow stack then it arrives at its destination in at most $m \cdot D_0 + t$ steps with probability $1 - 2^{-a \cdot t + b}$, for suitable chosen constants m, K, $a > 0$, and b. Thus the expected routing time for p_0 is at most

$$\begin{aligned} m \cdot D_0 + \sum_{t=1}^{T - m \cdot D} 2^{-a \cdot t + b} + T_{\text{overflow}} &\leq m \cdot D_0 + \frac{2^b}{2^{-a} - 1} + T_{\text{overflow}} \\ &= O(D_0) + T_{\text{overflow}} \end{aligned}$$

where T_{overflow} is defined to be the expected time from the beginning of subinterval Δ_1 until the delivery of p_0. Thus, it remains only to show that T_{overflow} is in $O(D_0)$. In the following, we will do even more: We will show that $T_{\text{overflow}} < 1$, for suitablly chosen T in $O(D + \log N + \log |E|)$.

Suppose p_0 is pushed onto an overflow stack for the ith time at time $\text{birth}(p_0)+T$, for $i \geq 1$. (Note that $\text{birth}(p_0)$ is the time when p_0 was reinjected for the $(i-1)$th time, for $i \geq 2$.) Define $E[|\Delta_i|]$ to be expected length of time interval Δ_i. In order to calculate an upper bound on this value, we estimate the number of packets that are pushed above p_0 onto the stack. None of the packets with birth date smaller than $\text{birth}(p_0)$ can be pushed onto p_0. The expected number of packets p with $\text{birth}(p_0) \leq \text{birth}(p) < \text{birth}(p_0) + (m+1) \cdot T$ is at most $(m+1) \cdot T \cdot (N+1)$. This is because each generator creates at most one packet in a time step, and the expected number of reinjected packets in a step is at most $1/\gamma < 1$. The fact that p_0 was pushed to an overflow stack can indicate a "bad situation". But, according to Lemma 4.2, the expected number of packets pushed above p_0 onto the overflow stack in each step after $\text{birth}(p_0) + (m+1) \cdot T$ is at most $\lambda_{\text{eff}} \cdot 2^{-a \cdot (T - m \cdot D) + b}$. Further, the expected number of packets which are removed from the stack in a step is $1/(\gamma \cdot |E|)$. Hence, k steps after the insertion of p_0 into the stack, the expected number of packets on the stack above p_0 is at most

$$(m+1) \cdot T \cdot (N+1) + k \cdot \lambda \cdot 2^{-a \cdot (T - m \cdot D) + b} - \frac{k}{\gamma \cdot |E|}$$

which is at most

$$(m+1) \cdot T \cdot (N+1) - \frac{k}{2 \cdot \gamma \cdot |E|} \ ,$$

for some suitable T in $O(D + \log |E|)$. The above term becomes 0 for

$$k = \kappa := 2 \cdot (m+1) \cdot T \cdot (N+1) \cdot |E| \cdot \gamma \ .$$

Consequently, p_0 is expected to be removed from the stack after at most κ steps after the insertion. Then it is suspended for $m \cdot T$ steps until it is reinjected, and after the reinjection it is routed for at most T further steps through the network until it reaches its destination or until is moved again to an overflow stack which means that the next interval, Δ_{i+1}, begins. Thus,

$$E[|\Delta_i|] \leq \kappa + m \cdot T + T \leq 4 \cdot m \cdot T \cdot N \cdot |E| \cdot \gamma \ .$$

Now we estimate the probability P_i that p_0 is moved at least i times onto an overflow stack. A reinjected packet is pushed again onto an overflow stack if it does not arrive at its destination in T steps. The suspension time of $m \cdot T$ steps ensures that the system has recovered when the packet is reinjected. That means the probability that the packet takes more than T steps to arrive at its destination is at most $2^{-a \cdot (T - m \cdot D_0) + b}$. Hence,

$$P_i \leq 2^{(-a \cdot (T - m \cdot D_0) + b) \cdot i} \ .$$

Note that $P_i \cdot E[|\Delta_i|]$ is at most $\frac{1}{2^i}$ for some suitable T in $O(D_0 + \log N + \log |E|)$. As a consequence,

$$T_{\text{overflow}} = \sum_{i=1}^{\infty} P_i \cdot E[|\Delta_i|] \leq \sum_{i=1}^{\infty} \frac{1}{2^i} < 1$$

which completes the proof of Theorem 4.1. ∎

4.2 Wormhole Routing

We slightly modify the protocol in Section 3. As before, let $R := 2D + L - 1$ be the time distance between two trials for a worm. Each generator now has an injection buffer storing only worms that

were generated during the last $T \cdot R$ time steps (T will be determined later), and an overflow stack. Worms that did not successfully reach their destination within T trials are moved from the injection buffer to the overflow stack. In case that the overflow stack is not empty, each time step a generator reactivates the worm at the top of the stack with probability $\frac{\lambda}{6N}$, suspends it for further $T \cdot R$ time steps and then moves it back to the injection buffer with the actual time step as new birth date and a new, randomly chosen rank as described in Section 3.1.

Suppose now that a worst case scenario happened until time step T_0. Then the following theorem implies that the recovery time of the modified wormhole protocol is bounded by $O((D+L)\log(N+D+L))$ (and therefore the protocol is stable).

Theorem 4.3 *Let α be a sufficiently large constant, and suppose that $\lambda \leq \frac{B}{28eL(2D)^{1/B}}$, $B \leq R$, and $T \geq \alpha \log(N+D+L)$. Let ω_0 be a worm generated at an arbitrary time step $\geq T_0 + 2R \cdot T$. Then ω_0 is delivered in expected time $R(1+\frac{4}{2^B-1})$.*

Proof. Before we start proving the theorem, let us introduce the following type of delay sequence.

Definition 4.4 ((s,B,R,T)-delay sequence)
An (s,B,R,T)-delay sequence consists of

- *s collision links $e_0, \ldots, e_{s-1}$, and*
- *$s \cdot B$ delay worms $\omega_1, \ldots, \omega_{s \cdot B}$.*

We say a worm ω_0 is delayed by this sequence, if the following conditions are met

(1) worm $\omega_{i \cdot B}$ is blocked at link e_i by the B worms $\omega_{i \cdot B+1}, \ldots, \omega_{(i+1) \cdot B}$ for $0 \leq i \leq s-1$;

(2) $\mathrm{rank}(\omega_{s \cdot B}) \leq \ldots \leq \mathrm{rank}(\omega_1) \leq \mathrm{rank}(\omega_0)$; and

(3) $\mathrm{rank}(\omega_0) - \mathrm{rank}(\omega_{s \cdot B}) \leq (2s+T)R$.

Lemma 4.5 *Let $T \geq 7$. Suppose ω_0 is a worm generated at a time step $\geq T_0 + 2R \cdot T$ that had at least t trials. Then ω_0 is delayed by a $(T/3, B, R, T)$-delay sequence or by an (s, B, R)-delay sequence with $t/3 \leq s < T/3$, neither of them having worms with birth dates $\leq T_0 + R$.*

Proof. Suppose first that there exists a delay sequence of length $s = \frac{T}{3}$ for ω_0. Let t_1 denote the time step where ω_0 had its last collision and t_s denote the time step where $\omega_{(s-1)B}$ has been blocked by $\omega_{(s-1)B+1}, \ldots, \omega_{s \cdot B}$. Since a worm can have an age of at most $T \cdot R$ time steps it holds:

$$\begin{aligned}
\mathrm{rank}(\omega_0) - \mathrm{rank}(\omega_{T/3 \cdot B}) &\leq \mathrm{birth}(\omega_0) + R - \mathrm{birth}(\omega_{s \cdot B}) \\
&\leq (t_1 - (t-1) \cdot R) + R - (t_s - T \cdot R) \\
&\leq t_1 - t_s + R + T \cdot R \\
&\leq (2R-1) \cdot (s-1) + R + T \cdot R \\
&\leq (2s+T) \cdot R \ .
\end{aligned}$$

Thus this sequence fulfills the requirements of an (s, B, R, T)-delay sequence.

Now suppose that no delay sequence of length $\frac{T}{3}$ can be constructed for ω_0. Then, according to Lemma 3.2, there must exist an (s, B, R)-delay sequence with length $\frac{t}{3} \leq s < \frac{T}{3}$. Since we only consider delay sequences of length $s \leq \frac{T}{3}$ it furthermore holds for $T \geq 7$ that

$$\begin{aligned}
\mathrm{birth}(\omega_{s \cdot B}) &\geq \mathrm{birth}(\omega_0) - [s(2R-1) + T \cdot R + R] \\
&> (T_0 + 2T \cdot R) - (2T \cdot R - R) = T_0 + R \ .
\end{aligned}$$

This completes the proof. ∎

The property that all worms within a delay sequence are born after time step $T_0 + R$ is important for the further probabilistic analysis. The reason for this is that worms generated at time step T_0 or earlier can only be prevented from reaching their destinations by worms generated at time step $T_0 + R$ or earlier. Therefore, the assumption that there has been a worst case scenario up to time step T_0 does not affect the generation rates of worms after time step $T_0 + R$. Since each of the N overflow stacks moves a worm to the injection buffer with probability at most $\frac{\lambda}{6N}$, if follows that the effective link load caused by worms generated at time steps $> T_0 + R$ is bounded by $\lambda + N \cdot \frac{\lambda}{6N} = \frac{7}{6}\lambda$. Therefore, the expected number of (s, B, R)-delay sequences for ω_0 with $s < \frac{T}{3}$ is at most

$$\binom{sB + 2Rs}{sB} (2D)^s \left(\frac{(2L-1)7\lambda}{6R}\right)^{sB} \leq 2^{-Bs}$$

with λ chosen as in Theorem 4.3. By using a similar counting argument as for the (s, B, R)-delay sequences, the expected number of $(T/3, B, R, T)$-delay sequences is at most

$$\binom{\frac{T}{3}B + (2 \cdot \frac{T}{3} + T)R}{\frac{T}{3} \cdot B} \cdot (2D)^{\frac{T}{3}} \left(\frac{(2L-1)7\lambda}{6R}\right)^{\frac{T}{3} \cdot B} \leq 2^{-\frac{T}{3} \cdot B}$$

with λ chosen as in Theorem 4.3. Thus, under the assumption that ω_0 is not moved to the overflow stack, the expected number of unsuccessful trials of ω_0 is at most $\frac{3}{2^B - 1} + T \cdot 2^{-T/3 \cdot B}$. It remains to bound the expected time a worm stays in the overflow stack of its generator.

Suppose, worm ω_0 is pushed onto the overflow stack. Then we can again use Lemma 4.5 to prove that, for any worm ω_1 generated at least $2T \cdot R$ time steps after ω_0 was generated, the probability that ω_1 enters the overflow stack of ω_0 is bounded by $2^{-T/3 \cdot B}$. For all worms generated between time step $\text{birth}(\omega_0)$ and $\text{birth}(\omega_0) + 2T \cdot R$, however, we cannot use this bound. But since a generator can generate at most one worm in each time step, at most $2T \cdot R$ worms can be generated by the generator of ω_0 during these time steps. Hence, k steps after ω_0 entered the overflow stack, the expected number of worms above ω_0 on the stack is at most

$$2T \cdot R + k\left(2^{-T/3 \cdot B} - \frac{\lambda}{6N}\right) \leq 2T \cdot R - \frac{k \cdot \lambda}{12N}$$

for $T \geq \frac{3(\log(N/\lambda)+4)}{B}$. Consequently, ω_0 is removed from the stack in expected time $\frac{24T \cdot R \cdot N}{\lambda}$.

According to our protocol it holds that, after ω_0 is removed from the stack, it is suspended for further $T \cdot R$ steps. Therefore, it takes at least $2T \cdot R$ steps after $\text{birth}(\omega_0)$ before ω_0 is moved back to its injection buffer. In this case we can use Lemma 4.5 to show that the probability that ω_0 is pushed at least i times to the overflow stack is at most $(2^{-T/3 \cdot B})^i$. Thus the expected number of steps ω_0 spends on the overflow stack is at most

$$\sum_{i=1}^{\infty} \left(2^{-T/3 \cdot B}\right)^i \left(\frac{24T \cdot R \cdot N}{\lambda} + 2T \cdot R\right)$$

which is below $\frac{R}{2^{B+1}}$ for suitably chosen $T = \Theta(\log(N + D + L))$. Therefore, the expected number of steps to route ω_0 is at most

$$R + R\left(\frac{3}{2^B - 1} + T \cdot 2^{-\frac{T}{3} \cdot B}\right) + \frac{R}{2^{B+1}} \leq R\left(1 + \frac{4}{2^B - 1}\right) \quad .$$

This completes the proof of Theorem 4.3. ∎

5 Summary and Open Problems

In this paper, we presented universal routing protocols for store-and-forward and wormhole routing, and analyzed them under a stochastic model of contiuous message generation. In general, our upper bounds on the link loads for which the protocols are stable are asymptotically optimal. For the store-and-forward routing protocol this is easy to see, since it is stable up to link loads of 1/e. In case of wormhole routing, the optimality of the $\Theta(\frac{B}{L \cdot D^{1/B}})$ bound can be derived for $L \geq D$ from a lower bound given by Cole *et al* [CMS96]. So the questions that are still open within the stochastic message generation model are:

- Is there a universal store-and-forward routing protocol that is stable for any constant link load $\lambda < 1$ and ensures a delay of $O(D)$, w.h.p.?
- Can the results for store-and-forward routing be extended to arbitrary simple paths and/or systems with bounded buffers?
- What is the maximum link load, for which a protocol is stable within the wormhole routing model, if we want to keep that any worm can still be routed in expected time $O(D + L)$?
- Can the time for recovery be reduced without increasing the bounds on the expected routing times?

Note that it has been shown by Andrews *et al* [AAF96] that there exists a universal store-and-forward protocol that is stable for any constant link load $\lambda < 1$. The upper bounds they give for the delays of the packets, however, are exponential in D.

As mentioned earlier, besides the notion of a deterministic adversary, Borodin *et al* [BKR96] also defined the notion of a stochastic adversary. It might be interesting to see, by how much the delays of the packets increase under a growing adaptibility of the stochastic adversary.

References

[A87] D. Aldous. Ultimate instability of exponential back-off protocol for acknowledgement based transformation control of random access communication channels. *IEEE Transactions on Information Theory* 33, pp. 219-223, 1987.

[AAF96] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, T. Leighton, Z. Liu. Universal stability for greedy contention-resolution protocols. In *Proc. of the 35th IEEE Symposium on Foundations of Computer Science*, pp. 380-389, 1996.

[AM95] R.Y. Awdeh, H.T. Mouftah. Survey of ATM switch architectures. *Computer Networks and ISDN Systems* 27, pp. 1567-1613, 1995.

[BKR96] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, D.P. Williamson. Adversarial queueing theory. In *Proc. of the 28th ACM Symposium on Theory of Computing*, pp. 376-385, 1996.

[BU96] A. Broder, E. Upfal. Dynamic deflection routing on arrays. In *Proc. of the 28th ACM Symposium on Theory of Computing*, pp. 348-355, 1996.

[CMS96] R. Cole, B. Maggs, R. Sitaraman. On the benefit of supporting virtual channels in wormhole routers. In *Proc. of the 8th ACM Symposium on Parallel Algorithms and Architectures*, pp. 131-141, 1996.

[CMSV96] R. Cypher, F. Meyer auf der Heide, C. Scheideler, B. Vöcking. Universal algorithms for store-and-forward and wormhole routing. In *Proc. of the 28th ACM Symposium on Theory of Computing*, pp. 356-365, 1996.

[D90] W. Dally. Virtual-channel flow control. In *Proc. of the 17th International Symposium on Computer Architecture*, pp. 60-68, 1990.

[DS87] W. Dally, C. Seitz. Deadlock-free message routing in multicomputer networks using virtual channels. *IEEE Transactions on Computers* 36, pp. 547-553, 1987.

[DS97] S. Datta, R. Sitaraman. The performance of simple routing algorithms that drop packets. To appear in *Proc. of the 9th ACM Symposium on Parallel Algorithms and Architectures*, 1997.

[GM96] L.A. Goldberg, P.D. MacKenzie. Analysis of practical backoff protocols for contention resolution with multiple servers. In *Proc. of the 7th ACM-SIAM Symposium on Discrete Algorithms*, pp. 554-563, 1996.

[GGM88] J. Goodman, A.G. Greenberg, N. Madras, P. March. Stability of binary exponential backoff. *Journal of the ACM* 35(3), pp. 579-602, 1988.

[GO93] R. Greenberg, H.C. Oh. *Universal Wormhole Routing.* Technical Report, University of Maryland, 1993.

[HW95] M. Harchol-Balter, D. Wolfe. Bounding delays in packet-routing networks. In *Proc. of the 27th ACM Symposium on Theory of Computing*, pp. 248-257, 1995.

[HLR87] J. Håstad, T. Leighton, B. Rogoff. Analysis of backoff protocols for multiple access channels. In *Proc. of the 19th ACM Symposium on Theory of Computing*, pp. 241-253, 1987.

[KL95] N. Kahale, T. Leighton. Greedy dynamic routing on arrays. In *Proc. of the 6th ACM-SIAM Symposium on Discrete Algorithms*, 1995.

[K85] F.P. Kelly. Stochastic models of computer communication systems. *J. Royal Statistic Soc.* 47, pp. 379-395, 1985.

[K88] R. Koch. Increasing the size of a network by a constant factor can increase the performance by more than a constant factor. In *Proc. of the 27th IEEE Symposium on Foundations of Computer Science*, pp. 221-230, 1988.

[L90] T. Leighton. Average case analysis of greedy routing algorithms on arrays. In *Proc. of the 2nd ACM Symposium on Parallel Algorithms and Architectures*, 1990.

[L92] T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann, San Mateo, CA, 1992.

[LMRR94] T. Leighton, B. Maggs, A. Ranade, S. Rao. Randomized routing and sorting on fixed-connection networks. *Journal of Algorithms* 17, pp. 157-205, 1994.

[MB76] R. Metcalfe, D. Boggs. Distributed packet switching for local computer networks. *Communications of the ACM* 19, pp. 395-404, 1976.

[MV95] F. Meyer auf der Heide, B. Vöcking. A packet routing protocol for arbitrary networks. In *Proc. of the 12th Symposium on Theoretical Aspects of Computer Science*, pp. 291-302, 1995.

[M94] M. Mitzenmacher. Bounds on the greedy routing algorithm for array networks. In *Proc. of the 6th ACM Symposium on Parallel Algorithms and Architectures*, pp. 346-353, 1994.

[P81] J.H. Patel. Performance of processor-memory interconnections for multiprocessors. *IEEE Transactions on Computers* C-30(10), pp. 771-780, 1981.

[PS95] M. Paterson, A. Srinivasan. Contention resolution with bounded delay. In *Proc. of the 34th IEEE Symposium on Foundations of Computer Science*, 1995.

[RMLD96] R. Rehrmann, B. Monien, R. Lüling, R. Diekmann. On the communication throughput of buffered multistage interconnection networks. In *Proc. of the 8th ACM Symposium on Parallel Algorithms and Architectures*, pp. 152-161, 1996.

[RU95] P. Raghavan, E. Upfal. Stochastic contention resolution with short delays. In *Proc. of the 27th ACM Symposium on Theory of Computing*, pp. 229-237, 1995.

[SM93] M. Settembre, F. Matera. All optical implementations of high capacity TDMA networks. *Fiber and Integrated Optics* 12, pp. 173-186, 1993.

[ST91] G.D. Stamoulis, J.N. Tsitsiklis. The efficiency of greedy routing in hypercubes and butterflies. In *Proc. of the 3rd ACM Symposium on Parallel Algorithms and Architectures*, pp. 248-259, 1991.

[TRH91] H.H. Theimer, E.P. Rathgeb, M.N. Huber. Performance analysis of buffered banyan networks. *IEEE Transactions on Communications* 39(2), pp. 269- 277, 1991.

[YLL90] H. Yoon, K. Lee, M. T. Liu. Performance analysis of multibuffered packet-switching networks in multiprocessor systems. *IEEE Transactions on Computers* 39(3), pp. 319-327, 1990.