A General Approach to Dynamic Packet Routing with Bounded Buffers

Andrei Z. Broder* Alan M. Frieze† Eli Upfal‡

Abstract

We prove a sufficient condition for the stability of dynamic packet routing algorithms. Our approach reduces the problem of steady state analysis to the easier and better understood question of static routing. We show that certain high probability and worst case bounds on the quasi-static (finite past) performance of a routing algorithm imply bounds on the performance of the dynamic version of that algorithm. Our technique is particularly useful in analyzing routing on networks with bounded buffers where complicated dependencies make standard queuing techniques inapplicable.

We present several applications of our approach. In all cases we start from a known static algorithm, and modify it to fit our framework. In particular we give the first dynamic algorithms for routing on a butterfly or two-dimensional mesh with bounded buffers. Both the injection rate for which the algorithm is stable, and the expected time a packet spends in the system are optimal up to constant factors. Our approach is also applicable to the recently introduced adversarial input model.

*Digital Systems Research Center, 130 Lytton Avenue, Palo Alto, CA 94301, USA. E-mail: broder@pa.dec.com

†Department of Mathematical Sciences, Carnegie Mellon University. Research supported in part by NSF Grants CCR-9925008, CCR-9530974. E-mail: af1p@andrew.cmu.edu

‡Computer Science Department, Brown University. Research supported in part by NSF Grant CCR-9731477. Part of this work was done at the IBM Almaden Reserach Center, San Jose, Calofornia. E-mail: eli@cs.brown.edu

1. Introduction

The rigorous analysis of the dynamic performance of routing algorithms is one of the most challenging current goals in the study of communication networks. So far, most theoretical work on this area has focused on static routing: A set of packets is injected into the system at time 0, and the routing algorithm is measured by the time it takes to deliver all the packets to their destinations, assuming that no new packets are injected in the meantime (see Leighton [8] for an extensive survey). In practice however, networks are rarely used in this "batch" mode. Most real-life networks operate in a *dynamic* mode whereby new packets are continuously injected into the system. Each processor usually controls only the rate at which it injects its own packets and has only a limited knowledge of the global state.

This situation is better modeled by a stochastic paradigm whereby the packets are continuously injected according to some inter-arrival distribution, and the routing algorithm is evaluated according to its long term behavior. In particular, quantities of interest are the maximum arrival rate for which the system is stable (that is, the arrival rate that ensures that the expected number of packets waiting in queues does not grow with time), and the expected time a packet spends in the system in the steady state. The performance of a dynamic algorithm is a function of the inter-arrival distribution. The goal is to develop algorithms that perform close to optimal for any inter-arrival distribution.

Several recent articles have addressed the dynamic routing problem, in the context of packet routing on arrays [7, 10, 5, 2], on the hypercube and the butterfly [13] and general networks [12]. Except for [2], the analyses in these works assume a Poisson arrival distribution and require unbounded queues in the routing switches (though some works give a high probability bound on the size of the queue used [7, 5]). Unbounded queues allow the application of some tools from queuing theory (see [3, 4]) and help reduce the correlation between events in the system, thus simplifying the analysis at the cost of a less realistic model.

Here we focus on analyzing dynamic packet routing in networks with bounded buffers at the switching nodes, a setting that most accurately models real networks. Our goal is to build on the vast amount of work that has been done for static routing in order to obtain results for the dynamic situation. Rather than produce a new analysis for each routing network and algorithm we develop a general technique that "reduces" the problem of dynamic routing to the better understood problem of static routing.

In Section 2 we prove a general theorem that shows that any communication scheme (a routing algorithm and a network) that satisfies a given set of conditions, defined only with respect to a *finite history* is stable up to a certain inter-arrival rate. Furthermore we bound the expected routing time. At first glance these conditions seems very restrictive and hard to satisfy, but in fact, as we show later, many of the previous results on static routing can be easily modified to fit into our framework. The theorem applies to any inter-arrival distribution: the stability results and the expected routing time of a packet inside the network depend only on the expectation of the inter-arrival distribution. The relationship between the inter-arrival distribution and the waiting time in the input queues is more complicated and is formulated in the theorem.

In Sections 3,4 and 5 we present three applications of our general theorem to packet routing on the butterfly network. In Section 6 we present an application to packet routing on a mesh. We assume that packets arrive according to an arbitrary inter-arrival distribution and have random destinations. In Section 7 we present similar results for an alternative input model, *the adversarial model* [1], whereby probabilistic assumptions are replaced by a deterministic condition on edge congestion.

Section 3 presents the first dynamic packet routing algorithm for a butterfly network with bounded buffers under constant injection rate. Our algorithm is stable for any inter-arrival distribution with expectation greater than some absolute constant. The expected routing time in an n-input butterfly is $O(\log n)$ and in the case of geometric inter-arrival times the expected time a packet spends in the input queue is also $O(\log n)$. Thus, the performance of the algorithm is within constant factors from optimal in all parameters. Our dynamic algorithm is based on the static routing results of Ranade [11] and Maggs and Sitaraman [9].

The above algorithm is not a "pure" queueing protocol (in such a protocol packets always move forward unless progress is impeded by an already-full queue) because as in the algorithms devised in [11, 9] it generates and uses extra messages and mechanisms to coordinate the routing. Maggs and Sitaraman studied the question of a "pure" queuing protocol routing with bounded buffers. They gave an algorithm that routes n packets on an n input butterfly with bounded buffers in $O(\log n)$ steps. Based on their technique we develop in Section 4 a simple greedy algorithm for dynamic routing. It is stable for any inter-arrival distribution with expectation $\Omega(\log n)$, the routing time is $O(\log n)$, and in the case of a geometric inter-arrival distribution the expected wait in the queues is also $O(\log n)$.

In Section 5 we apply our approach to a dynamic version of the simple oblivious routing algorithm on the butterfly described in [14, 8]. This algorithm routes $n \log n$ packets (all logarithms in this paper are base 2) on an $n \log n$ butterfly in expected $O(\log n)$ steps, and with high probability no buffer has more than $O(\log n)$ packets. Our dynamic version of this algorithm uses a butterfly with buffers of size $O(\log n)$ and is stable for any inter-arrival distribution with expectation greater than some absolute constant. The expected routing time is $O(\log n)$ and the expected time a packet waits in a queue in the case of geometric inter-arrival distribution is also $O(\log n)$. Note that for dynamic routing, which is an infinite process, it does not suffice to have a high probability bound on the size of the buffer memory needed at a given time: we must prove that the algorithm is stable for some *fixed* buffer size.

The result of Section 2 is couched in terms of a general network, but the above examples are all concerned with the butterfly network. In Section 6 we exhibit the generality of the result by applying it to an $n \times n$ mesh. Our dynamic algorithm is based on the Greedy Algorithm of Section 1.7 of Leighton [8]. Our algorithm is stable for any inter-arrival distribution with expectation at least Cn for some fixed constant C. The expected time a packet spends in the network is $O(n)$. In the case of Poisson arrival (geometric inter-arrival distribution) the expected time the packet spends in the queue is also $O(n)$. This is optimal up to constant factors. Leighton [7] studied this problem and obtained similar results provided that the buffers in the routing switches are *unbounded*. More precisely, Leighton's algorithm ensures that at any *fixed* time, with high probability, no routing queue has more than 4 packets. However, for any sufficiently long execution the maximum size of any queue exceeds any given bound. Our results build on Leighton's analysis, by augmenting his algorithm with a simple flow control mechanism, which ensures that every routing queue is bounded at all times, and thus only finite buffers are needed.

In an attempt to avoid probabilistic assumptions on the input, Borodin *et al.* [1] defined the *adversarial* input model. Instead of probabilistic assumptions, for any time interval there is an absolute bound on the number of generated packets that must traverse any particular edge. Surprisingly, our general technique can be applied here as well. In Section 7 we briefly sketch how the results of Sections 3–6 can be extended to this model.

These examples demonstrate several ways of applying our scheme. The analysis required is similar to the analysis used in the proof of the corresponding static case with several small modifications. Most notably, as often done in practice, we sometimes augment the original static algorithm with a simple "flow control" mechanism, such as acknowledgments. Our general theorem can be applied to other topologies and algorithms provided that an appropriate static case analysis can be constructed.

2. The stability criterion

Our model is as follows: we are given a routing algorithm $\mathcal{A}$ acting on a network $\Gamma(n)$ with n inputs and n outputs. Each input receives new packets with a inter-arrival distribution $\mathcal{F}$. We distinguish between *usual* and *unusual* distributions. We first describe the situation for usual distributions. By this we mean that the probability that the number of arrivals in any time period significantly exceeds its expectation falls off exponentially. A more precise definition is left until later. In the usual case the packets are placed into an unbounded FIFO queue at the input node. Packets have an output destination chosen independently and uniformly at random. When a packet reaches the front of its queue, it is called *active*. At some point after becoming active, the packet is removed from its queue and eventually routed to its destination. For convenience we assume that a packet chooses its random destination upon becoming active.

In an arbitrary input distribution we modify our routing scheme as follows. We maintain at each input node v two queues, Q_1 and Q_2. On arrival, packets are placed in Q_1; the front packet in Q_1 leaves it to Q_2 according to a geometric service time at a rate greater than the arrival rate of $\mathcal{F}$; then Q_2 feeds the network as above. The precise details are discussed in Theorem 2.1 below.

We are interested in determining under what conditions the queuing system is *ergodic* (or stable), that is, under which conditions the expected length of the input queues is bounded as $t \to \infty$. To this purpose we have to study the *inter-departure* time, which is the interval from when a packet becomes active until it leaves the queue, and the packet next in line (if any) becomes active. Besides stability, we are also interested in the expected time a packet spends in the queue, and the expected time it spends in the network.

Since the inter-arrival times are independent, if the inter-departure times are also independent, then each queue can simply be viewed as a G/G/1 system and the stability condition would trivially be that the inter-departure rate exceeds the inter-arrival rate. However the usual situation is that there are complex interactions among packets during routing and thus the inter-departure times are highly dependent and hard to analyze.

The goal of this section is to define a set of relatively simple sufficient conditions such that if the routing algorithm satisfies them, then the system is stable up to a certain inter-arrival rate and we can bound the expected time a packet spends in the queue and in the network. This is captured in the following. We assume that the system is empty of packets at time $t = 0$. We use $\mathcal{H}_t$ to denote the history of the process up to time t i.e. the outcome of the random choices made at times 1 through t.

Let μ denote the expected inter-arrival time and then $p = 1/\mu$ is the inter-arrival rate.

Theorem 2.1 *Assume that the randomized routing algorithm $\mathcal{A}$ acting on the network $\Gamma(n)$ is characterized by four parameters a, b, m, and T, where a and b are positive constants, and m and T are positive integers that might depend on n and satisfy $1/n^a < m/T < 1$ and $T < n^b$. Assume that the algorithm satisfies the following conditions:*

1. *Every packet is delivered at most n^a steps after becoming active.*
2. *For every time $\tau \geq 0$ there exists an event $\mathcal{E}_\tau$ with the following properties:*
   - (a) *$\neg\mathcal{E}_\tau$ implies that any packet that at time τ was among the first m packets in its queue, is delivered before time $\tau + T$.*
   - (b) *For any fixed time τ,*
   $$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{H}_{\tau - n^b}) \leq B_{\mathcal{E}} = \frac{(m/T)^7 p}{n^{2a+2b+3}}.$$
   - (c) *$\mathcal{E}_\tau$ is determined by $\mathcal{H}_{\tau+n^b}$.*
   Thus for any $k \geq 1$,
   $$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{E}_{\tau - 2in^b}, i = 1, 2, \ldots, k) \leq B_{\mathcal{E}}. \tag{1}$$

If there exists a positive constant ϵ such that the inter-arrival distribution $\mathcal{F}$ has an inter-arrival rate smaller than $(1 - \epsilon)m/T$, then

1. *The system is stable.*
2. *The expected elapsed time between when a packet becomes active and it is delivered is at most $2T + O(1)$.*
3. *The expected time a packet spends in the input queue is bounded by $O(T) + f(T/m)$, where f is a function that depends only on $\mathcal{F}$ and not on the routing process. (For "usual" distributions such as geometric $f(T/m) = O(T/m)$).*

Proof: Assume first that the inter-arrival time is geometric, that is, at each step, each input receives a new packet with some fixed probability $p < (1-\epsilon)m/T$. (We will show later how to extend the proof to a general inter-arrival distribution).

Fix an input v and let $Q(t)$ denote the length of the queue at node v at time t. Let

$$\pi(t, L) = \mathbf{Pr}(Q(t) \geq L).$$

We show that the system is stable by proving a uniform bound, independent of t, on $\pi(t, L)$. Let

$$\beta = \frac{\epsilon}{4}\frac{m}{T} \quad \text{and} \quad U = \left(\frac{T}{m}\right)^3 n^{a+b+1}.$$

(Hence $U > m$.) We will establish the bound using the following two inequalities:

- For $L \geq U$

$$\pi(t, L) \leq \pi\left(t - \lceil \tfrac{L}{2} \rceil, (1+\beta)L\right) + \delta e^{-\gamma L}. \tag{2}$$

- For $m \leq L < U$

$$\pi(t, L) \leq \pi\left(\left\lceil \left(t - \tfrac{2U}{\epsilon p}\right)^+ \right\rceil, U\right) + \frac{2U}{\epsilon p} B_{\mathcal{E}} + e^{-\phi p L}. \tag{3}$$

(Define as usual x^+ to be $\max\{0, x\}$.)

where $\gamma = \Omega((m/T)^3/n^{a+b})$, $\delta = O(n^b)$, and ϕ is a positive constant. Since $\pi(t, L) = 0$ for $t < L$, these inequalities imply that for $L \geq U$,

$$\begin{aligned}
\pi(t, L) &\leq \delta e^{-\gamma L} + \delta e^{-(1+\beta)\gamma L} + \delta e^{-(1+\beta)^2 \gamma L} + \cdots \\
&\leq \delta e^{-\gamma L}(1 + e^{-\beta\gamma L} + e^{-2\beta\gamma L} + \cdots) \\
&= \frac{\delta e^{-\gamma L}}{1 - e^{-\beta\gamma L}}.
\end{aligned}$$

and that for $m \leq L < U$,

$$\pi(t,L) \leq \frac{\delta e^{-\gamma U}}{1-e^{-\beta\gamma U}} + \frac{2U}{\epsilon p} B_{\mathcal{E}} + e^{-\phi p L}.$$

Combining the two bounds we get

$$\begin{aligned}
\mathbf{E}(Q(t)) &= \sum_{L\geq 1} \pi(t,L) \\
&\leq m + \frac{U\delta e^{-\gamma U}}{1-e^{-\beta\gamma U}} + \frac{2U^2}{\epsilon p} B_{\mathcal{E}} + 2e^{-\phi p m} + \frac{2\delta e^{-\gamma U}}{1-e^{-\beta\gamma U}} \\
&= O(m)
\end{aligned}$$

Since this holds for any inter-arrival rate bounded by $(1-\epsilon)m/T$, by Little's Theorem the expected time a packet spends in the queue is $O(T)$.

We now turn to proving the recurrence (2). Since the inequality is trivially true for $t < L$, assume that $t \geq L$. Let $t_0 = t - \lceil \frac{L}{2} \rceil$. Let I denote the number of packets arriving at input v between t_0 and t, and let J denote the number of packets leaving the queue at v during this interval. Let s_i denote the inter-departure time of the i'th packet to become active at v after time t_0, that is, the interval from when this packet reaches the front of queue until it departs. (If there was an active packet at time t_0 then s_1 denotes how long it took that packet to depart.) Let

$$M = \lfloor (1-\beta) \tfrac{m}{T} \tfrac{L}{2} \rfloor.$$

We claim that if $Q(t) \geq L$, then at least one of the following three events holds:

$\mathcal{F}_a \equiv Q(t_0) \geq (1+\beta)L$. (Large initial queue.)

$\mathcal{F}_b \equiv I \geq (1+\beta)p\dfrac{L}{2} - 1$. (Excessive number of new arrivals.)

$\mathcal{F}_c \equiv s_1 + s_2 + \cdots + s_M > L/2$. (Slow processing.)

Indeed assume $\neg\mathcal{F}_a$, $\neg\mathcal{F}_b$, and $\neg\mathcal{F}_c$ and consider two cases:

Case 1: $Q(t_0) > L/2$. This means that at time t_0 the queue contained more than M packets, and $\neg\mathcal{F}_c$ implies that M packets left the queue by time

$t = t_0 + \lceil L/2 \rceil$. Thus $J \geq M$, and

$$\begin{aligned}
Q(t) &= Q(t_0) + I - J \\
&< (1+\beta)L + (1+\beta)p\frac{L}{2} - (1-\beta)\frac{m}{T}\frac{L}{2} \\
&= L + \left(2\beta + p + \beta p - \frac{m}{T} + \beta\frac{m}{T}\right)\frac{L}{2} \\
&\leq L + \left(4\beta + p - \frac{m}{T}\right)\frac{L}{2} \\
&\leq L.
\end{aligned}$$

Case 2: $Q(t_0) \leq L/2$. Then

$$\begin{aligned}
Q(t) \leq Q(t_0) + I &< \frac{L}{2} + (1+\beta)p\frac{L}{2} \\
&\leq \frac{L}{2} + \left(1 + \frac{\epsilon m}{4T}\right)(1-\epsilon)\frac{m}{T}\frac{L}{2} \\
&< L.
\end{aligned}$$

Thus, in order to prove the recurrence (2) it suffices to show that for $L \geq U$

$$\mathbf{Pr}(\mathcal{F}_b) \leq e^{-\gamma L} \tag{4}$$

and that

$$\mathbf{Pr}(\mathcal{F}_c \wedge (Q(t) \geq L)) \leq O(n^b)e^{-\gamma L} \tag{5}$$

Equation (4) follows immediately from standard bounds on the binomial distribution

$$\mathbf{Pr}(\mathcal{F}_b) = \mathbf{Pr}\left(I \geq (1+\beta)p\frac{L}{2} - 1\right) \leq e^{-\beta^2 pL/7}.$$

To prove equation (5) note that if at any time during $[t_0, t]$ the queue at v contains less than m packets, then $Q(t) \geq L$ only if $I \geq L - m$ and the probability of the latter can be bounded as above. So let's assume that for all $\tau \in [t_0, t]$, we have $Q(\tau) \geq m$.

Let now z denote the number of occurrences of $\mathcal{E}_\tau$ during $[t_0, t]$. By the hypothesis of the theorem $s_1 + s_2 + \cdots + s_M \leq Mn^a$. We partition the interval $[t_0, t_0 + Mn^a]$ into $2n^b$ sets, $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{2n^b}$ where $\mathcal{T}_i = \{t_0 + i - 1 + 2kn^b : 0 \leq k \leq \lfloor (Mn^a - i + 1)/(2n^b) \rfloor\}$. Let z_i denote the number of occurrences of $\mathcal{E}_\tau$ for $\tau \in \mathcal{T}_i$. Note that if packet i becomes active at time τ and if $\neg\mathcal{E}_\tau$ then we have $s_i + s_{i+1} + \cdots + s_{i+m} \leq T$; if $\mathcal{E}_\tau$ we can use the bound $s_i \leq n^a$. Thus we have the following series of implications:

$$\begin{aligned}
& s_1 + s_2 + \cdots + s_M > \frac{L}{2} \\
\Rightarrow \quad & (M-z)\frac{T}{m} + n^a z > \frac{L}{2} \\
\Rightarrow \quad & n^a z > \frac{L}{2} - M\frac{T}{m} \geq \frac{L}{2} - (1-\beta)\frac{L}{2} = \frac{\beta L}{2} \\
\Rightarrow \quad & z > \frac{\beta L}{2n^a} \\
\Rightarrow \quad & \exists i:\ z_i > \frac{\beta L}{4n^{a+b}}
\end{aligned}$$

It follows from (1) that

$$\mathbf{Pr}(z_i > u) \leq \binom{M/(2n^b)}{u}\left(\frac{(m/T)^7}{n^{2a+2b+3}}\right)^u \leq \left(\frac{Me}{2n^b u}\cdot\frac{1}{n^{2a+2b+3}}\right)^u$$

So

$$\mathbf{Pr}\left(\exists i:\ z_i \geq \frac{\beta L}{4n^{a+b}}\right) \leq 2n^b\left(\frac{Me}{2n^{2a+3b+3}}\cdot\frac{4n^{a+b}}{\beta L}\right)^{\beta L/(4n^{a+b})} \leq 2n^b e^{-\gamma L}.$$

This completes the proof of recurrence (2) and we turn to recurrence (3). If $t < L/(2p)$ then for some constant ϕ,

$$\mathbf{Pr}(Q(t) \geq L) \leq \mathbf{Pr}(L \text{ packets arrive in } [0, L/(2p)]) \leq e^{-\phi L}.$$

Hence assume that $t > L/(2p)$ and let $t_0 = \lceil (t - 2U/(\epsilon p))^+ \rceil$. Define the following three events:

$\mathcal{F}_a \equiv Q(t_0) \geq U$.

$\mathcal{F}_b \equiv$ The event $\mathcal{E}_\tau$ does not occur for any $\tau \in [t_0, t]$.

$\mathcal{F}_c \equiv$ The queue at v receives at most $(1-\frac{\epsilon}{2})\frac{m}{T}\theta$ new packets in any interval $[t-\theta, t]$ with $\theta \geq \frac{L}{2p}$.

We bound $\mathbf{Pr}(Q(t) \geq L)$ via the inequality

$$\begin{aligned}\mathbf{Pr}(Q(t) \geq L) \leq \mathbf{Pr}(\mathcal{F}_a) + \mathbf{Pr}(\neg\mathcal{F}_b) + \mathbf{Pr}(\neg\mathcal{F}_c) \\ + \mathbf{Pr}(Q(t) \geq L \mid \neg\mathcal{F}_a, \mathcal{F}_b, \mathcal{F}_c).\end{aligned} \tag{6}$$

By definition

$$\mathbf{Pr}(\mathcal{F}_a) = \pi(t_0, U).$$

Clearly

$$\mathbf{Pr}(\neg\mathcal{F}_b) \leq \left(\frac{2U}{\epsilon p} + 1\right) B_{\mathcal{E}},$$

and since $(1 - \frac{\epsilon}{2})\frac{m}{T}\theta \geq (1 + \frac{\epsilon}{2})p\theta$

$$\mathbf{Pr}(\neg\mathcal{F}_c) \leq \sum_{\theta \geq \frac{L}{2p}} e^{-\epsilon^2 p\theta/12} \leq \frac{1}{2}e^{-\phi L} \tag{7}$$

for a *constant* ϕ.

Now assume $\neg\mathcal{F}_a$, $\mathcal{F}_b$, and $\mathcal{F}_c$ and notice that if $\mathcal{F}_b$ holds, then as long as the queue is not empty it loses at least m packets in any interval of T steps. If $Q(t) \geq L$ we claim that these assumptions imply that there is a step in the interval $[t_0, t]$ in which the queue is empty; otherwise

$$\begin{aligned}Q(t) &\leq Q(t_0) + \left(1 - \frac{\epsilon}{2}\right)\frac{m}{T}(t - t_0) - m\left\lfloor\frac{t - t_0}{T}\right\rfloor \\ &< Q(t_0) + m - \frac{\epsilon}{2}\frac{m}{T}(t - t_0)\end{aligned}$$

which is less than m since if $t_0 = 0$ then $Q(t_0) = 0$, and otherwise $t - t_0 = \lfloor 2U/(\epsilon p)\rfloor$ and $Q(t_0) \leq U - 1$.

Thus, under the assumptions $\neg\mathcal{F}_a$, $\mathcal{F}_b$, and $\mathcal{F}_c$, if there are L packets in the queue at time t, then there is an interval $[t - \theta', t]$, such that
(i) the queue was empty at time $t - \theta' - 1$;
(ii) the queue was not empty in any step in the interval $[t - \theta', t]$
(iii) at least $L + m\lfloor\frac{\theta'}{T}\rfloor > L + \frac{m\theta'}{T} - m$ new packets arrived at the queue in that interval.
But if $L \geq m$ and $\theta' > L/(2p)$ then (iii) contradicts $\mathcal{F}_c$. So we only have to consider the probability that (iii) holds for an interval with $L \leq \theta' \leq L/(2p)$. This is bounded by

$$\max_{\substack{L \geq m \\ L \leq \theta' \leq L/(2p)}} \left\{\sum_{i \geq L + \frac{m\theta'}{T} - m} \binom{\theta'}{i} p^i (1-p)^{\theta' - i}\right\} \leq \max_{L \leq \theta' \leq L/(2p)} e^{-\epsilon^2 p\theta'/3} \leq \frac{1}{2}e^{-\phi pL}. \tag{8}$$

This completes the proof of equation (3).

Let us now see how to go from a geometric inter-arrival distribution to something more general. We observe that in the proof above the inter-arrival distribution is only required to satisfy (4), (7), and (8). Suppose that the inter-arrival time is a random variable X with distribution $\mathcal{F}$. Let $p = 1/\mathbf{E}(X) < 1$. We say that $\mathcal{F}$ is usual if there exist constants A_0 and A_1 such that in any interval of length t, the number N of arrivals satisfies

$$\mathbf{Pr}(N \geq (1+\epsilon)pt) \leq A_0 e^{-A_1 \epsilon^2 pt}$$

for any $0 \leq \epsilon \leq 1$. Clearly if $\mathcal{F}$ is usual, then our proof will go essentially unchanged provided that $p < (1-\epsilon)\frac{m}{T}$.

Assume finally that the arrival of packets to the queue is governed by some arbitrary inter-arrival distribution $\mathcal{F}$. Let Q_1 and Q_2 be the two queues in front of a generic node v, as described at the beginning of this section. We move packets from the front of Q_1 to the end of Q_2 with probability $p = (1-\epsilon)\frac{m}{T}$. Our analysis has shown that Q_2 is stable, and that the expected wait in Q_2 is $O(T)$. The queue Q_1 is a G/M/1 queue. Thus, if the expected inter-arrival time to Q_1 is smaller than p, then the queue is stable and the expected waiting time in Q_1 is determined (see [6] for details) by the distribution $\mathcal{F}$, as follows: Let x be the non-trivial (that is, $x \neq 1$) root of the equation (the Laplace transform)

$$x = \int_0^\infty e^{-pt(1-x)}\, d\mathcal{F}(t);$$

The expected wait in the queue is then $x/(p(1-x))$.

We now bound the expected time that a packet takes to reach its destination once it becomes active. Consider a long interval of time $[0, L]$. Suppose that x_a packets arrive during this interval and $\mathcal{E}_\tau$ occurs for x_b values of τ. Given this the average time that a packet takes to reach its destination once it becomes active is at most

$$\frac{(x_a - x_b)T + x_b n^a}{x_a} \leq T + \frac{x_b n^a}{x_a}.$$

Now

$$\begin{aligned}
\mathbf{E}\left(\frac{x_b n^a}{x_a}\right) &\leq \frac{\mathbf{E}(x_b) n^a}{\frac{1}{2} Lp \mathbf{Pr}(x_a > \frac{1}{2} Lp)} + n^a L \mathbf{Pr}(x_a \leq \tfrac{1}{2} Lp) \\
&\to \frac{2 n^a B_{\mathcal{E}}}{p} && \text{as } L \to \infty \\
&\to 0 && \text{as } n \to \infty.
\end{aligned}$$

Consequently, the expected time that a packet takes to reach its destination once it becomes active is at most $2T$ as claimed. $\square$

In the next four sections we deal with applications of Theorem 2.1 to the cases where the underlying topology is (i) a butterfly with $\log n$ levels (rows) of n nodes (switches) or (ii) an $n \times n$ mesh. In both cases there are buffers on edges and unbounded queues at input vertices. We show stability for several protocols under suitable assumptions about input rate and internal buffer size. We will explicitly consider geometric inter-arrival distributions. The general case is implicitly dealt with as in the proof of the main theorem.

3. Dynamic routing on a butterfly with constant injection rate and bounded buffers

For this section we assume that the buffer size q is a sufficiently large constant. We first fix $m = \Theta(\log n)$ and we will subsequently describe a protocol and define $\mathcal{E}$, T, a, and b to satisfy the conditions of Theorem 2.1.

Our approach is based on the second algorithm of Maggs and Sitaraman [9] and in places we follow their description very closely. This algorithm uses *tokens* whose main role is to define a *wave number* for each packet. We will assume that tokens occupy the same amount of space as a packet. Imagine that behind each input node queue there is an infinite sequence of tokens, packets and blanks. The odd positions are always taken by tokens and the even positions contain packets or blanks, where the packets occur randomly with probability p. The tokens are labeled $1, 2, \ldots$. The label of a token is referred to as its wave number. As opposed to [9] we actually use these labels within the algorithm, not only in its analysis.

At each time step we examine the front of the sequence. If it is blank then we simply delete this blank and go to the next time step. If there is a token or packet then we delete it from the sequence and place it in the back of the input queue. The front element (which could be a packet or a token) of the queue tries to enter the network only if it is *eligible* (we define this subsequently). An eligible packet enters the system if the buffer on the edge that it intends to use is, or becomes not full during the current time step. Upon entrance into the network a token splits into two tokens, one for each outgoing edge. Thus both buffers need to have space before an eligible token can enter.

The wave number $w(\Pi)$ of packet Π is the wave number of the token that immediately precedes it in entering the network. The rank of a packet is a pair (w, c) where w is the wave number and c is the column number of its input. The rank of a token is given by its wave number. Ranks are ordered lexicographically.

An important invariant of the algorithm is that packets go through a switch in increasing order of *rank*.

A switch labeled $(l, c = c_0, c_1, \ldots, c_{L-1})$ where l is the level and $L = \log n$, has a 0-edge entering it from switch $(l-1, c - c_l 2^{l-1})$ and a 1-edge entering it from switch $(l-1, c-(c_l-1)2^l)$. The buffer of the i-edge is called the i-buffer.

The behavior of each switch is governed by a simple set of rules. By *forwarding* a packet or token we mean sending it to the appropriate queue in the next level. If that queue is full, the switch tries again in consecutive time steps until it succeeds. A switch can either be in 0-mode or 1-mode and is initialized to be in 0-mode. In i-mode, a switch forwards packets in the i-buffer until a token is at the head of the i-buffer. At that time, if $i = 0$ then the switch simply changes to 1-mode; otherwise, if $i = 1$ then there will be tokens at the front of both queues and the switch waits until it can forward *both* tokens, each to one of its outgoing edges. (These tokens have the same wave number). It then switches back to 0-mode.

It will be important in the subsequent analysis to ensure that if Π and Π' are packets or tokens residing simultaneously in the network then $|w(\Pi) - w(\Pi')| \leq A \log n$ for some constant $A > 0$. This is achieved as follows: At every time step, every output node generates two *chips*. The $2n$ chips generated at time t will be referred to as *generation* t. Each generation travels back through the network one level at a time. The chips make their journey so that each chip occupies a different edge at each step. By the time a chip of generation t has reached a switch s, it has iteratively computed the lowest wave number of any packet/token which left the network at time t from an output node reachable from s. Thus when generation t reaches the input nodes, each input node knows the lowest wave number $w^*(t)$ of any packet/token that left the network at time t. This happens at time $t + \log n$. w^* is initialized at zero and if no packet leaves the network at time t then $w^*(t) = w^*(t-1)$. Note that if Π is a packet/token which is in the network at time t or later then $w(\Pi) \geq w^*(t)$ since packets go through network switches in increasing order of rank.

At time τ a packet/token Π will be *eligible* to enter the network, only if

$$w(\Pi) \leq w^*(\tau - \log n) + A \log n.$$

It follows that if Π is any packet/token already in the network at time τ, or eligible at time τ, then

$$w^*(\tau - \log n) \leq w(\Pi) \leq w^*(\tau - \log n) + A \log n. \tag{9}$$

We focus now on one of the first m packets of a queue at time τ. Denote it by Π. Assume for the time being that Π is eligible at time τ. Maggs and Sitaraman define a *delay sequence* of packets and tokens in a familiar way – an (r, f) delay sequence consists of (i) a path P from an output node to an input node, (ii) a sequence $s_1, s_2, \ldots, s_r$ of not necessarily distinct buffers, (iii) a sequence $\Pi_1, \Pi_2, \ldots, \Pi_r$ of *distinct* packets and tokens and (iv) a non-increasing sequence $w_1, w_2, \ldots, w_r$ of wave numbers. The wave numbers of the tokens are shown to decrease strictly as one moves along the delay path, in fact they decrease by one from one token to the next. The length $\lambda = \lambda(P)$ is equal to $2f - \log n$ where f is the number of *forward* edges of the path. It is a little confusing at first, but here forward edges go from level x to level $x + 1$, for some x, and are traced *backwards* by P, assuming it is directed from output to input. It is assumed that Π_i goes through buffer s_i and has wave number w_i. Maggs and Sitaraman show (Lemma 4.1) that if packet Π takes $\log n + d$ time to exit from the network, then there is a $(d + (q-2)f, f)$ delay sequence, with $\Pi_1 = \Pi$, for some $f \geq 0$.

We have to argue that the delay sequence does not contain many tokens. Let k denote the number of tokens in our delay sequence. We see that

$$k \leq A \log n,$$

since the wave numbers of tokens decrease by one along the delay path, equation (9) holds, and any packet/token on the delay path must be in the network at some time after τ, and thus has wave number at least $w^*(\tau - \log n)$.

If we assume (and we will subsequently remove this assumption)

A: The destinations of packets under consideration are random

then the expected number of delay sequences for Π can be bounded as follows.

Choose $\lambda = 2f - \log n$ for the length of a path P. Let $\mathcal{P}$ denote the set of possible delay paths and note that $|\mathcal{P}| \leq 4^\lambda$. Choose a delay $d \geq K \log n$ where K is a large constant. (We assume $q \gg K \gg A$.) Let $r = d + (q-2)f$.

We have to count the number of (r, f) delay sequences with delay path P. Choose positive integers $a_1, a_2, \ldots, a_k$ so that $a_1+a_2+\cdots+a_k \leq r$ and along our path there are tokens at positions $a_1, a_1+a_2, \ldots, a_1+a_2+\cdots+a_k$. There are $\binom{r}{k}$ choices for the a_i's.

Let $J = [r] \setminus \{a_1, a_1+a_2, \ldots, a_1+a_2+\cdots+a_k\}$. Now choose an edge buffer s_j for each $j \in J$. Observe that having chosen P our choices are now restricted. However for each edge in P we can choose the multiplicity of its buffer in the delay sequence. This can be done in at most $\binom{r+\lambda-1}{\lambda-1}$ ways.

Let d_j be the depth of the edge with buffer s_j. There are 2^{d_j} inputs which could send a packet along this edge. The probability that there is such a packet with a particular wave number (fixed by the preceding token) is at most $2^{d_j}(p2^{-d_j-1}) \leq 1/2$.

Thus the expected number of delay sequences is at most

$$\sum_{k\leq A\log n}\sum_{f\geq \log n}\sum_{P\in\mathcal{P}}\sum_{d>K\log n}\sum_{a_1,\ldots,a_k}\sum_{s_j:j\in J}(1/2)^{r-k}$$

$$\leq \sum_{k\leq A\log n}\sum_{d\geq K\log n}\sum_{f\geq\log n}4^{\lambda}\binom{r}{k}\binom{r+\lambda-1}{\lambda-1}2^{k-r}$$

$$\leq \frac{1}{n^2}\sum_{k\leq A\log n}\sum_{d\geq K\log n}\sum_{f\geq\log n}\left(4^{2f/r}\left(\frac{2e}{k/r}\right)^{k/r}\left(\frac{2e}{\lambda/r}\right)^{\lambda/r}\cdot\frac{1}{2}\right)^{r}$$

$$\leq \frac{1}{n^2}\sum_{k\leq A\log n}\sum_{d\geq K\log n}\sum_{f\geq\log n}\left(\frac{2}{3}\right)^{r}$$

(for sufficiently large q, K)

$$\leq \frac{1}{n^2}\sum_{k\leq A\log n}\left(\frac{2}{3}\right)^{K\log n}\cdot 3\cdot\frac{1}{1-(2/3)^{q-2}},$$

which can be made $O(n^{-B})$ for any positive constant B by choosing K sufficiently large.

Let us now deal with Assumption **A**. One cannot assert that the destinations of packets in the network at time τ are random. There is a tendency for "bad" configurations to "linger". However, one can assert that the destinations of packets with wave numbers in $[w, w+k-1]$ are random for any *fixed* w. What we have actually proved is that there is unlikely to be a delay sequence made up from random packets with wave numbers in $[w, w+k-1]$ where $w = w_r$. We know however that

$$w^*(\tau) \leq w_\tau \leq w^*(\tau) + A\log n,$$

and thus we can assume conservatively that if $\tau_0 = \tau - 2AnS\log n$ and $w_0 = w^*(\tau_0)$ then

$$w_0 + 2A\log n \le w_\tau \le w_0 + A(2nS+1)\log n.$$

Here S is a polynomial upper bound (proved below in Lemma 3.1) on the time taken for an active packet or token to get through the network. We use the facts:

(a) $w^*(t-1) \le w^*(t) \le w^*(t-1)+1$.

(b) $w^*(t+nS) \ge w^*(t)+1$.

Of course w_0 itself is a random variable. the conditional distribution of the destinations in wave w for $w > w_0 + A\log n$ are random because no packet in this wave could have been in the network at time τ_0. Let us therefore define

> $\mathcal{E}_\tau \equiv$ There exists $w \in [w_0 + 2A\log n, w_0 + A(2nS+1)\log n + (K+1)\log n]$ and a delay sequence of length exceeding $K\log n$ made from packets in waves $[w, w + A\log n]$.

The probability of $\mathcal{E}_\tau$ is $O(Sn\log n/n^B)$ and can be made suitably small. If $\mathcal{E}_\tau$ does not occur then all of the eligible packets among the first m in each queue at time τ will be serviced in time $(K+1)\log n$.

We have therefore dealt with eligible packets at time τ. Some of the first m packets in the queue can be ineligible. If $\mathcal{E}_\tau$ does not occur then they will be serviced in a further $(K+1)\log n$ time because they will be eligible at time $\tau + (K+1)\log n$ (all eligible packets in the network at time τ will be out) and the extra $(K+1)\log n$ in the definition of $\mathcal{E}_\tau$ means there are no delay paths for them either. We thus take $T = 2(K+1)\log n$.

We now have to give an estimate for S.

Lemma 3.1 *Under this protocol, no packet takes more than $S \le 4An\log^2 n$ steps to complete its service once it has become active.*

Proof: If we trace an input-output path then the tokens we meet have wave numbers which decrease by one each time. This is a basic property of the scheduling protocol. At each time step at least one packet or token of lowest wave number moves. Thus if X denotes the set of lowest wave tokens or packets at time t, then X will be through the network at time $t + 2n\log n$. The network can have no more than $A\log n$ distinct eligible wave numbers at any time and so we get an upper bound of $2An\log^2 n$ for eligible packets. An active but ineligible packet might then have to wait this long to become eligible. $\Box$

From our definition of $\mathcal{E}_\tau$ we see that it depends only on the destinations of packets that have wave numbers in the range $[w_0+2A\log n, w_0+A(2nS+1)\log n+(K+1)\log n]$. Every packet that has already made a choice of its destination by time t_0-S is out of the system by time t_0 and thus has wave number at most $w_0+A\log n$. On the other hand, packets that enter the network after time $t_0+nS(A(2nS+1)+K+1)\log n$ has wave number $\geq w_0+(A(2nS+1)+K+1)\log n$. Hence $\mathcal{E}_\tau$ depends only on the destination of packets enter the network at times in the range $[\tau-2AnS\log n-S, \tau+(A(2nS-1)+K+1)nS\log n]$. We can thus take $b=5$ in the main theorem. To define a suppose a packet Π becomes active at time τ. Then all packets currently in the network will have left it by the time $\tau+S$. If Π has not left the queue by this time then Π will certainly be eligible and can now enter. Thus we can take $a=2$. Thus, all the conditions of Theorem 2.1 are satisfied, and we have proved:

Theorem 3.1 *There is a constant C, such that the above algorithm is stable for any inter-arrival distribution with expectation at least C. The expected time a packet spends in the network is $O(\log n)$, and in the case of geometric inter-arrival time the expected time a packet spends in the system is $O(\log n)$.*

After running the algorithm for a long time, wave numbers could be come very large. To avoid the storage of very large numbers, wave numbers can be stored mod $2A\log n$ and eligibility defined to take account of this in the obvious way.

4. Greedy dynamic routing on a butterfly with bounded buffers and injection rate $O(1/\log n)$

We present in this section a simple greedy algorithm ("pure queueing protocol") that can sustain an inter-arrival distribution with expectation $\Omega(\log n)$ using buffers of size $q=O(1)$ in the routing switches. The algorithm and analysis is based on the static result of Maggs and Sitaraman [9].

We first describe the behavior of switches in the network:

- Packets are selected from buffers in FIFO order.
- A switch V alternates between the two switches W, W' feeding it. If at time $t-1$ switch V received a packet from switch W, at time t it

first checks switch W'. If the buffer of W' is non-empty then W' send a packet to V, otherwise V returns to switch W.

The dynamic algorithm uses a token based flow control mechanism. Each input has $m = O(1)$ tokens. A token can be in one of three modes: *enabled, used* or *suspended.* Initially all tokens are in enabled mode. To inject a packet into the network the input needs an enabled token. A packet is sent with a token and the mode of the token switches to used mode. When a packet is delivered the token (acknowledgment) is returned to the input node. Let t_s be the last time a given token was sent with a packet, let t_r be the last time it returns to the input node. If $t_r - t_s \leq K \log n$ then the token becomes enabled again at time $t_s + K \log n$. If $t_r - t_s > K \log n$ then the token mode is switched to suspended mode for $3mnS = O(n^3)$ steps (S is defined in Lemma 4.1). Then it is switched back to enabled mode (K is a constant fixed in the proof). This flow mechanism guarantees that an input cannot inject more than m packets in each interval of $K \log n$ steps, and that the input does not inject new packets when the network is congested.

We use a separate butterfly network $\Gamma'(n)$ to route tokens back to their sources. The output nodes of $\Gamma(n)$ are identified with the inputs for $\Gamma'(n)$. A token from input j of $\Gamma(n)$ which reaches output k of $\Gamma(n)$ must travel from input k of $\Gamma'(n)$ to output j of $\Gamma'(n)$.

Lemma 4.1 *Under this protocol no packet takes more than $S = 4qn^2 + 2K \log n$ steps to complete its service once it has become active.*

Proof: Our network $\Gamma(n) \cup \Gamma'(n)$ has $2 \log n$ levels. We prove by induction on i that if a buffer B at level i is non-empty (level 0 is the output level of $\Gamma'(n)$ which is the input level of $\Gamma(n)$) then after at most 2^i steps the front of the queue moves onto the next level. This is clear for $i = 0$ and our protocol ensures that after at most $2 \times 2^{i-1}$ steps the switch will be able to move the front of B. Let μ be some packet waiting in B. FIFO selection then ensures that a packet spends at most $q2^i$ time steps at level i. Thus a token takes at most $2qn^2$ time steps to reach its output once it obtains a packet. By the same argument, a packet has to wait at most $2qn^2 + K \log n$ time steps to obtain a token, once it has become active. □

The proof in [9] is based on a *delay tree* argument. To apply this proof technique here we analyze the delays in $\Gamma(n)$ and $\Gamma'(n)$ separately. Note that since we assume that a processor has sufficient buffer space to receive all packets sent to it, delays in one network do not propagate to the other network. In our setting the delay tree in $\Gamma(n)$ is defined as follows: Fix a packet Π which is one of the first m packets of an input queue at time τ. Let $M(\Pi)$ be the set of packets that were in the network during any step t in which Π was in the network. A node v of $\Gamma(n)$ is *full* with respect to Π if at least q packets in $M(\Pi)$ traversed v during this time. The *spine* $SP(\Pi)$ of Π's delay tree $DT = DT(\Pi)$ is the path in $\Gamma(n)$ from the input node, j say, to Π's output node back to the output node of $\Gamma'(n)$ labeled j. Let $F(\Pi)$ be the set of *full* nodes with respect to Π in the network $\Gamma(n)$. $DT(\Pi)$ consists of $SP(\Pi)$ plus any node reachable from it by a path consisting entirely of nodes in $F(\Pi)$. (Paths are directed from nodes of the spine down to lower numbered levels.) The number of packets on the delay tree $DT(\Pi)$, denote by $h_1(\Pi)$, is the sum over all the nodes of the tree, of the number of packets in $M(\Pi)$ visiting each of the nodes. (Note that packets can be counted several times in this count). A similar construction gives $DT'(\Pi)$, a delay tree of Π in $\Gamma'(n)$. Let $h_2(\Pi)$, denote the number of packets on DT', and let $h(\Pi) = h_1(\Pi) + h_2(\Pi)$.

By Theorem 2.1 in [9] the time a packet Π spends in the network is bounded by $\log n + h_1(\Pi)$, and the time it takes the corresponding token to return to the source is $\log n + h_2(\Pi)$.

Let

$$T = 2K \log n,$$

and define the event:

$\mathcal{E}_\tau$: There exist delay trees DT and DT' such that considering all the packets and tokens that are in the network at time τ, and the packets and tokens entering the network during the interval $[\tau, \tau + T]$,

$$h(\Pi) \geq (K - 1) \log n.$$

(For each pair of trees DT and DT' we imagine a packet Π that followed $SP(DT)$, and a token that followed $SP(DT')$, both were in the network during the interval $[\tau, \tau + T]$. We compute the maximum of $h(\Pi)$ over all DT and DT'.)

Clearly $\neg\mathcal{E}_\tau$ implies that any packet that is among the first m packets in its queue at time τ will be delivered, and its token returned to the source, before time $\tau + T$, since a packet waits no more than $K \log n$ till it has an enabled token, and then its routing takes no more than $K \log n$ steps.

Lemma 4.2 *For any $\alpha > 0$ there exists $K = K(\alpha)$ such that if $\tau - 2K\log n \leq t \leq \tau - K\log n$ then*

$$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{H}_t) \leq n^{-\alpha},$$

provided $\mathcal{H}_t \subseteq \bar{\mathcal{E}}_t$.

Proof: Under the conditions of the lemma, any packet and token in the network at time t will have left the network by time τ. The destinations of packets generated in the interval $[t+1, \tau+T]$ are random and independent of $\mathcal{H}_t$. Once a token changes to a used state, it requires at least $K\log n$ steps in order to become enabled again. Thus each input node can inject at most $4m$ new packets between times $t+1$ and $\tau+T$, each having a new random destination.

Theorem 2.5 of [9] shows that if each input node injects 1 packet to a random destination then for some $K_0 = K_0(\alpha)$ and q sufficiently large

$$\mathbf{Pr}(\exists\Pi: \ h_1(\Pi) \geq K_0 \log n) = O(n^{-\alpha}).$$

So if each input injects at most $4m$ packets with random destinations into the network then

$$\mathbf{Pr}(\exists\Pi: \ h_1(\Pi) \geq 4mK_0 \log n) = O(n^{-\alpha}).$$

A symmetric argument proves the same bound for $h_2(\Pi)$. Choosing $K = 4mK_0 + 1$ yields the lemma. $\Box$

Let $\tau_0 = \tau - 2mnS$ and

$$\mathcal{B}_\tau = \bigcap_{t=\tau_0}^{\tau} \mathcal{E}_t.$$

Corollary 4.1

$$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{H}_{\tau_0}) \leq \mathbf{Pr}(\mathcal{B}_\tau \mid \mathcal{H}_{\tau_0}) + 2mSn^{1-\alpha}.$$

Lemma 4.3

$$\mathbf{Pr}(\mathcal{B}_\tau \mid \mathcal{H}_{\tau_0}) \leq n^{-\alpha}.$$

Proof: The network at time τ contains at most m packets per input node. We must must find a way to be able to treat the destinations of these packets as random given $\mathcal{H}_{\tau_0}$. For input i and time t let $D_{i,t}$ be the set of m destinations associated with the tokens for i. The occurrence of events $\mathcal{E}_t, t \in [\tau_0, \tau]$

can be determined from the sets $D_{i,t}, t \in [\tau_0, \tau + T]$. Imagine that initially i.e. at time τ_0 each token has a destination and it gets a new one whenever it returns from a trip through the network. We will view the sequence $D_{i,t}, t = \tau_0, \ldots, \tau + T$ as being produced as follows: We start with an arbitrary set of destinations $D_{i,\tau}$. Each token j has *due time* d_j for its next change of destination. At time d_j the destination of token j is replaced with a new random destination and d_j is replaced by $d_j + K \log n$ unless either (i) j is late and becomes suspended at some time in the interval $[d_j, d_j + S]$ or (ii) there is no active packet waiting for j at time $d_j + K \log n$, in which case j gets a new due date determined by the arrival distribution and *not* by the workings of the network, which we treat as an adversary. This adversary is limited to arbitrarily choosing tokens to make late. We can assume that the adversary makes its decision on token j at time d_j.

Suppose that there is a time $t \in [\tau_0, \tau - 2K \log n]$ such that the adversary decides not to make any token late during the interval $[t, t + S]$. Then at the end of this time interval the destinations of unsuspended tokens are completely random and so the (conditional) probability $\mathbf{Pr}(\mathcal{E}_{t+S}) \leq n^{-\alpha}$. On the other hand, if the adversary tries to delay at least one token in every such interval then there will be no tokens in the network at time $\tau' = \tau_0 + mnS$ and $\bar{\mathcal{E}}_{\tau'}$ will occur, unless there are fewer than m new arrivals at some input in the period $[\tau, \tau + mnS]$. The probability of the latter is negligible and the lemma follows. $\Box$

Combining Corollary 4.1 and Lemma 4.3 we obtain

$$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{H}_{\tau - 2mnS}) = O(mSn^{1-\alpha}).$$

Taking $T = 2K \log n$, $m = O(1)$, $a = b = 3$, $\alpha = 16$ and applying Theorem 2.1 we obtain

Theorem 4.1 *There is a constant C, such that the above algorithm is stable for any injection rate with expected inter-arrival time greater than $C \log n$. The expected time a packet spends in the network is $O(\log n)$. In the case of geometric inter-arrival time the expected time a packet spends in the system is $O(\log n)$.*

5. Greedy dynamic routing on a butterfly with buffers of size $O(\log n)$ and constant injection rate

The algorithm in the previous section sustains an injection rate which is only up to $O(1/\log n)$ of the network capacity. We now present a greedy algorithm that is stable for any inter-arrival distribution with expectation bounded by some constant C, thus a constant fraction of the network capacity. This algorithm, however, requires buffers of size $q = O(\log n)$.

The algorithm and analysis is based on the static result in [8] Section 3.4.4. When a packet Π is injected into the network it receives a random priority number $r(\Pi)$ chosen uniformly at random from the interval $[1, ..., 8eK \log n]$ (K is a constant fixed in the proof). Packets are generally selected from the buffers according to their random priority numbers. This does not *guarantee* that a packet will get through the network in a reasonable time, since it is conceivable that under this protocol a packet could stay in the network for a very long time. Consequently, once every $10K \log n$ steps we change the selection rule for one step so that packets are selected according to age and FIFO order. We call this a *special step*.

The algorithm uses the same token based flow control mechanism as the one described in the previous section, only the values of K and S will change. It also uses a second network $\Gamma'(n)$ to route the tokens back to the inputs. The tokens inherit the priorities of their packets, for the return phase, as well as their age status.

This time m is of order $\log n$.

Lemma 5.1 *Under this protocol no packet takes more than $S = 40Kqn^2 \log n + K \log n$ steps to complete its service once it has become active.*

Proof: An argument similar to that given in the proof of Lemma 4.1 ensures that a packet's token arrives back at its input within $4qn^2$ special steps of getting a packet. $\Box$

Let

$$T = 2K \log n$$

and define the event:

$\mathcal{E}_\tau$: There is a delay sequence (in $\Gamma(n) \cup \Gamma'(n)$) of length $K \log n$ at some time in the interval $[\tau, \tau + T]$.

The definition of a delay sequence is similar to that given in Section 3. A full definition can be found for example on p550 of [8]. It is basically a path P from an input node to an output node (back to an input node) plus a sequence of nodes $v_1, v_2, \ldots, v_p$ of P. The nodes are in order on P but there can be repetition in the sequence. There is a sequence of packets $\Pi_1, \Pi_2, \ldots, \Pi_p$ where Π_i is required to go through node v_i. Finally, there is the condition that $r(\Pi_i) \leq r(\Pi_{i+1})$ for $1 \leq i < p$. Because of our special steps we have to relax this last condition for $\lceil p/(10K \log n) \rceil$ values of i.

Assume first that the buffers are unbounded. Then (see e.g. the proof of Theorem 3.26 in [8]) the event $\neg\mathcal{E}_\tau$ implies that a packet that received an enabled token in the interval $[\tau, \tau + K \log n]$ is delivered within $K \log n$ steps, i.e. before time $\tau + T$. In these circumstances each token becomes enabled at least once in the interval $[\tau, \tau + K \log n]$, and so the first m packets in each queue at time τ are delivered by time $\tau + T$. However, if no packet was delayed more than $K \log n$ steps, then no buffer had more than $K \log n$ packets at any step in that interval and we get the same performance as if each buffer had size $q = K \log n$. Thus with this queue size, $\neg\mathcal{E}_\tau$ implies that the first m packets in each queue at time τ are delivered by the time $\tau + T$.

Using the almost identical argument to that given in the previous section we can prove that for any $\beta > 0$ we can choose $K = K(\beta)$ such that

$$Pr(\mathcal{E}_\tau \mid \mathcal{H}_{\tau-2mnS}) \leq n^{-\beta}.$$

Taking $T = 2K \log n$, $m = O(\log n)$, $a = b = 3$, $\beta = 13$ and applying Theorem 2.1 we obtain

Theorem 5.1 *There is a constant C, such that the above algorithm is stable for any injection rate with expected inter-arrival time greater than C. The expected time a packet spends in the network is $O(\log n)$. In the case of geometric inter-arrival time the expected time a packet spends in the system is $O(\log n)$.*

6. Dynamic routing on a two-dimensional mesh with bounded buffers under injection rate $O(1/n)$

Our dynamic algorithm is based on the Greedy Algorithm of Section 1.7 of Leighton [8].

A packet takes the shortest one-bend route from origin to destination, first (left or right) on its origin row to its destination column, then up or down on that column to the packet's destination.

We assume that a switch can receive up to four packets per step, one from each incoming edge, and send four packets per step, one through each outgoing edge. A switch maintains a buffer for each outgoing edge. When there is a space in a buffer the switch receives packets to that buffer according to the following rule: Except for special steps, packets that have to travel farthest have the highest priority. There will be a special step every $10Kn$ steps, where K is a sufficiently large constant ($K \geq 2e^2$ will do). In such a step, priority is given to age, oldest first. Packets of the same age are dealt with in lexicographic order of pair (origin,destination).

The algorithm uses a token based admission control mechanism similar to that in the previous two sections, with one packet per input i.e. $m = 1$. Let t_s be the last time a given token was sent with a packet, and let t_r be the last time it returns to its input node. If $t_r - t_s \leq Kn$ then the token becomes enabled again at time $t_s + Kn + Z$, where Z is a random number chosen uniformly from $[0, Kn]$. If $t_r - t_s > Kn$ then the token mode is switched to suspended mode until time $t_r + 3n^2S + Z$ steps, then it is switched again to enabled mode, where S is defined in Lemma 6.1 below and Z is chosen as above.

We use a separate network to route tokens back to their sources and the routing mirrors that of the main network.

This flow mechanism guarantees that an input cannot inject more than one packet within each interval of Kn steps. Furthermore, the probability that a token becomes enabled at any fixed time is at most $1/(Kn)$.

Lemma 6.1 *Under this protocol no packet takes more than $S = 100Kn^6$ steps to reach its destination once it has become active.*

Proof: Consider a packet P. Let P_0 be its predecessor in the queue. P_0 does not leave the queue until it has an enabled token. At that time there are no more than n^2 other packets in the network. Consider the progress of the highest priority old packet Π in the network. If Π is moving along a column then it moves at every special time step. If it is moving along a row, then it could fail to move because further along that row there is contention for a column buffer at that special time step. Π waits at most $Kn + n^2$ special steps before making another move. This is because the packet waiting to move along the column in question will be the oldest packet trying to get into the column edge buffer. Thus after $Kn + Kn^2 + n^3$ special steps Π will have reached its destination column and will reach its final destination within a further n special steps. So after at most $Kn^3 + Kn^4 + n^5$ special steps, P_0 will be the highest priority packet in the network and will be delivered within a further $Kn + Kn^2 + n^3$ special steps. Thus P_0 gets to its destination at most $Kn + Kn^2 + (K+1)n^3 + Kn^4 + n^5 \leq 2n^5$ special steps after leaving the queue. The used token comes back after at most another $2n^5$ special steps and after at most $2Kn$ steps is re-activated. Finally, after at most another $2n^5$ special steps the packet P is delivered. The sum of these delays is less than $100Kn^6$. $\square$

Next we turn to the definition of the event $\mathcal{E}_\tau$. Our analysis is based on the the technique in [7] as described in [8] [Sect. 1.7.2]. As in [7] we relate the execution of the algorithm to an artificial execution on a *wide-channel* model in which an arbitrary number of packets can traverse an edge at any step, and no packet is ever delayed. We assume that the execution on the wide-channel starts at time τ.

Let α and q be constants to be defined later, let

$$d_0 = (\alpha + 3)\log n + \log(4K) + 1 \ .$$

This serves as a suitable high probability upper bound on the delay of any given packet. We now define the events $\mathcal{E}_\tau$ as the union of the following events:

1. There is a row edge e (in the network that routes the packets), a $t \geq 0$, and an interval $[t_0, t_0 + t + d_0] \subseteq [\tau, \tau + Kn]$, such that at least $t + d_0 - 1$ packets traverse edge e in that interval in the wide-channel model.

2. There is a column edge e (in the network that routes the packets), a $t \geq 0$, and an interval $[t_0, t_0 + t + 2d_0] \subseteq [\tau, \tau + Kn]$, such that at least $t + d_0 - 1$ packets traverse edge e in that interval in the wide-channel model.

3. A routing buffer (in the network that routes the packets) has q packets in some step in the interval $[\tau, \tau + Kn]$.

4. There is a row edge e (in the mirror network that routes the tokens), a $t \geq 0$, and an interval $[t_0, t_0 + t + d_0] \subseteq [\tau, \tau + Kn]$, such that at least $t + d_0 - 1$ tokens traverse edge e in that interval in the wide-channel model.

5. There is a column edge e (in the mirror network that routes the tokens), a $t \geq 0$, and an interval $[t_0, t_0+t+2d_0] \subseteq [\tau, \tau+Kn]$, such that at least $t+d_0-1$ tokens traverse edge e in that interval in the wide-channel model.

6. A routing buffer (in the network that routes the tokens) has q tokens in some step in the interval $[\tau, \tau+Kn]$.

We say that a packet was delayed d steps in traversing an edge if there is a d step gap between the time it traverses the edge in the wide-channel model and the time it traverses the edge in the standard model. Leighton's analysis in [7] is based on the following fact (see Corollary 1.9 and Lemma 1.10 in [8]): If buffers are unbounded and the farthest to go packet always has the highest priority, then a packet is delayed d steps in traversing a row edge e only if there is an interval of $t+d$ steps such that $t+d$ packets cross edge e in that interval in the wide-channel model.

This must be modified to account for the special steps. Then we can only postulate that at least $t+d-\lceil (t+d)/(10Kn) \rceil$ packets cross e. Similarly, if a packet is delayed d steps in crossing a column edge e then, assuming no packet is delayed more than d_0 steps on a row, there is an interval of $t+d_0+d$ steps in which at least $t+d-\lceil (t+d)/(10Kn) \rceil$ packets cross edge e in the wide-channel model (see [8] for a detailed proof). Thus we have the following corollary that satisfies requirement (2.b) in the general theorem:

Corollary 6.1 *The event $\neg\mathcal{E}_\tau$ implies that any packet with an enabled token at time τ is delivered within the next $2n+2d_0 \leq Kn$ steps.*

Proof: Any packet with an enabled token is delivered within $2n$ steps in the wide-channel model. In the standard model its additional delay is at most $2d_0$. $\square$

It follows that if $\mathcal{E}_\tau$ does not occur then any packet which is active at time τ is delivered within $T = 3Kn$ time steps.

We now bound the probability of the event $\mathcal{E}_\tau$ given $\mathcal{H}_{\tau-2n^2S}$. Once we have proved the equivalent of Lemma 4.2 we can make the same arguments as in Corollary 4.1 and Lemma 4.3, with $\tau_0 = \tau - 2n^2S$. The effect of adding Z is similar to that of waiting for a packet to arrive in the butterfly examples i.e. it is outside the control of the adversary. We will argue next that for any $\alpha > 0$ there is a $K = K(\alpha)$ such that if $\tau - 2Kn \leq t \leq \tau - Kn$ then

$$\mathbf{Pr}(\mathcal{E}_\tau \mid \mathcal{H}_t) = O(n^{-\alpha}), \tag{10}$$

provided $\mathcal{H}_t \subseteq \bar{\mathcal{E}}_t$.

For an edge e and an interval $[t_1, t_2] \subseteq [\tau, \tau + Kn]$ let $\mathcal{E}_0(e, t_1, t_2, r)$ be the event that in the wide-channel model r packets cross e during time interval $[t_1, t_2]$. Note that every token is used at most once in the interval and its destination can be treated as random.

Case 1: e is a row edge:

$$\mathbf{Pr}(\mathcal{E}_0(e, t_1, t_2, r)) \leq \binom{n}{r} \left(\frac{t_2 - t_1}{Kn}\right)^r \leq \left(\frac{e(t_2 - t_1)}{rK}\right)^r .$$

(The nodes on the row under consideration have a total of n tokens. Each token is used at most once in the interval. Choose r of them to transport the packets of interest. The probability that a token becomes enabled at any fixed time is at most $1/(Kn)$.)

Case 2: e is a column edge:

$$\mathbf{Pr}(\mathcal{E}_0(e, t_1, t_2, r)) \leq \binom{n^2}{r} \left(\frac{t_2 - t_1}{Kn^2}\right)^r \leq \left(\frac{e(t_2 - t_1)}{rK}\right)^r .$$

(There is a total of n^2 tokens. Each token is used at most once in the interval. Choose r of them to transport the packets of interest. The probability that a token becomes enabled at any fixed time is at most $1/(Kn)$ and the probability that the token uses a particular column is $1/n$.)

Thus the probability that there is a row edge e, a $t \geq 0$, and an interval $[t_0, t_0 + t + d_0] \subseteq [\tau, \tau + Kn]$, such that $t + d_0$ packets traverse edge e in that interval in the wide-channel model is bounded by

$$2Kn^3 \sum_{t \geq 0} \left(\frac{e(d_0 + t)}{(d_0 + t - 1)K}\right)^{d_0 + t - 1} \leq 4Kn^3 \left(\frac{e}{K}\right)^{d_0 - 1} \leq n^{-\alpha}$$

for $K \geq e^2$. (There are Kn possible values for t_0 and $\leq 2n^2$ edges.)

Similarly the probability that there is a column edge e, a $t \geq 0$, and an interval $[t_0, t_0 + t + 2d_0] \subseteq [\tau, \tau + Kn]$, such that $t + d_0$ packets traverse edge e in that interval in the wide-channel model is bounded by

$$2Kn^3 \sum_{t \geq 0} \left(\frac{e(2d_0 + t)}{(d_0 + t - 1)K}\right)^{d_0 + t - 1} \leq 2Kn^3 \sum_{t \geq 0} \left(\frac{2e}{K}\right)^{d_0 + t} \leq 4Kn^3 \left(\frac{2e}{K}\right)^{d_0 - 1} \leq n^{-\alpha} ,$$

provided that $K \geq 2e^2$.

Remark 1 *It follows (see [8, Sect. 1.7.2, Lemma 1.10]) that with probability at least $1 - 2n^{-\alpha}$, for every edge w and time interval I of d_0 steps, there is a step in I in which w is empty.*

Next we bound the probability that any buffer is full in the interval $[\tau, \tau + Kn]$. From Remark 1 we can assume that some row edge has q packets in its buffer only if there is a window of d_0 steps in which some input tries to inject at least q packets. The probability of this is at most

$$Kn^3 \binom{d_0}{q} \left(\frac{1}{Cn}\right)^q = O(n^{-\alpha})$$

for sufficiently large q.

From Remark 1 we can further assume that a column edge has q packets in its buffer only if there is a window of d_0 steps in which q packets turn at e, Assuming no row delay of d_0 or more, the probability of this is bounded by

$$Kn^3 \binom{n}{q} \left(\frac{2d_0}{Kn^2}\right)^q = O(n^{-\alpha})$$

for sufficiently large q (see [8, Sect. 1.7.2, Theorem 1.13]). (The factor Kn^3 bounds the number of (interval $I = [t_1, t_2]$, edge e) pairs. There are $\binom{n}{q}$ choices of token. There is a probability $1/n$ that its destination uses e. There is a probability of at most $2d_0/(Kn)$ that it becomes enabled at a time which means it would cross e during $[t_1 - d_0, t_2]$ in the wide-channel model.)

This completes the proof of (10).

Taking $T = 3Kn$, $m = 1$, $a = b = 9$, $\alpha = 50$ and applying Theorem 2.1 we obtain

Theorem 6.1 *There is a constant C, such that the above algorithm is stable for any injection rate with expected inter-arrival time greater than Cn. The expected time a packet spends in the network is $O(n)$. In the case of geometric inter-arrival time the expected time a packet spends in the system is $O(n)$.*

7. Adversarial model

In order to avoid probabilistic assumptions on the input, Borodin *et al.* [1] defined the *adversarial* input model. Instead of probabilistic assumptions, restrictions are placed on the amount of required traffic through each edge. More precisely, for an edge e of the network and a time interval I we let $\theta(e, I)$ denote the number of messages arriving during interval I whose input-output path contains e. An adversary has *injection rate* α if for all e and I:

$$\theta(e, I) \leq \alpha |I| \tag{11}$$

where $|I|$ is the length of I.

Surprisingly, the main results of this paper can be extended to this model.

7.1 Butterfly with constant injection rate α and bounded buffers

We will show how to extend the result of Section 3 to the adversarial model. We assume that (11) holds for some (sufficiently small) $\alpha > 0$.

When a packet arrives it now adds a random offset between 1 and $c \log n$ to its wave number. We route packets in the way previously described. The only issue is the likelihood of a long delay sequence. As described previously, we have a set of buffers $s_j, j \in J$ and a set of wave numbers $W = [w, w + A \log n]$. For each buffer we have a wave number $w_j \in W$, where $w_{j+1} \leq w_j$ and there is a set P_j of packets which want to use this edge. Let $\mathcal{F}_j$ be the event: there exists a choice $\Pi_j \in P_j$ such that

(i) Π_j has wave number w_j and

(ii) $\Pi_j \notin \{\Pi_1, \Pi_2, \ldots, \Pi_{j-1}\}$.

Our assumptions about input rate imply that $\mathbf{Pr}(\mathcal{F}_j) \leq \beta = \alpha(c + A)/c < 1$. More importantly, we have

$$\mathbf{Pr}(\mathcal{F}_j \mid \mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{j-1}) \leq \beta. \tag{12}$$

Condition on the occurrence of $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{j-1}$, let $P'_j = P_j \setminus \{\Pi_1, \Pi_2, \ldots, \Pi_{j-1}\}$ be the current set of choices for Π_j. The choice of wave number offset by packets is done independently and so the probability of $\mathcal{F}_j$ does not increase. Inequality (12) is enough to prove the unlikelihood of long delay sequences. The event $\mathcal{E}_\tau$ can be defined in the same way as before. Thus we prove the following theorem:

Theorem 7.1 *There is a constant $\alpha > 0$, such that for any adversary with injection rate α the system is stable, and the expected time a packet spends in the system is $O(\log n)$.*

7.2 Butterfly with $O(1/\log n)$ injection rate and bounded buffers

Let Π be the first packet that takes time $D + \log n$ to reach its destination after becoming active at time t. From [9] we know that the delay tree of Π is hit at least D times. We need only consider hits from messages in the time interval $[t - D - \log n, t + D + \log n]$. By assumption there are at most

$$\left(\frac{(2D + 2\log n + 1)\alpha}{\log n}\right)\left(\frac{D}{q} + \log n\right)$$

such hits. So if $D = \Delta \log n$ and

$$2(\Delta + 1)\alpha\left(\frac{\Delta}{q} + 1\right) < \Delta, \tag{13}$$

then Π cannot be delayed by more than D. Now for small α and large q, $\Delta = 3\alpha$ satisfies (13) and we conclude that every packet is delivered in time $(1 + 3\alpha)\log n$.

7.3 Butterfly with constant injection rate α and buffers of size $O(\log n)$

In this case we will have to use our stability theorem (Theorem 2.1). Putting $p = 2\alpha$ we know that in time interval of length $[\tau, \tau + K\log n]$, no more than $2\alpha K \log n$ packets will arrive at any input. Also, the existence of a delay sequence depends only on the priority calculation, since we are guaranteed that no edge sees more than $\alpha K \log n$ packets in the interval (removing the necessity to prove Theorem 3.2.5 of [8].

7.4 Mesh with injection rate $\alpha/n, \alpha < 1/2$ and bounded buffers

This case is straightforward. The assumption (11) implies that in the wide-channel model, fewer packets cross an edge e than are required for the first two possibilities of event $\mathcal{E}_\tau$, assuming $d_0 > 1/(1-2\alpha)$. We can take $q = d_0$ since for every edge e we can argue that in any interval of d_0 steps there is a step in which e is empty.

References

[1] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan and D. P. Williamson Adversarial queuing theory. *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pp. 376–385, 1996.

[2] A. Z. Broder and E. Upfal. Dynamic Deflection Routing in Arrays. *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pp. 348–355, 1996.

[3] M. Harcol-Balter and P. Black. Queuing analysis of oblivious packet routing networks. *Procs. of the 5th Annual ACM-SIAM Symp. on Discrete Algorithms.* Pages 583–592, 1994.

[4] M. Harcol-Balter and D. Wolf. Bounding delays in packet-routing networks. *Procs. of the 27th Annual ACM Symp. on Theory of Computing,* 1995, pp. 248–257.

[5] N. Kahale and T. Leighton. Greedy dynamic routing on arrays. *Procs. of the 6th Annual ACM-SIAM Symp. on Discrete Algorithms.* Pages 558–566, 1995.

[6] L. Kleinrock. *Queuing Systems Volume I: Theory,* John Wiley, New York, NY, 1975.

[7] T. Leighton. Average case analysis of greedy routing algorithms on arrays. *Procs. of the Second Annual ACM Symp. on Parallel Algorithms and Architectures.* Pages 2–10, 1990.

[8] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures.* Morgan-Kaufmann, San Mateo, CA 1992.

[9] B. M. Maggs and R. K. Sitaraman. Simple algorithms for routing on butterfly networks with bounded queues. *Proc. of the 24th Annual ACM Symp. on Theory of Computing.* Pages 150–161, 1992.

[10] M. Mitzenmacher. Bounds on the greedy algorithms for array networks. *Procs. of the 6th Annual ACM Symp. on Parallel Algorithms and Architectures.* Pages 346–353, 1994.

[11] A. G. Ranade. How to emulate shared memory. *Procs. of the 28th Annual Symp. on Foundations of Computer Science*, pages 185–194, October 1987.

[12] C. Scheideler and B. Voecking Universal continuous routing strategies. *Procs. of the 8th Annual ACM Symp. on Parallel Algorithms and Architectures.* 1996.

[13] G. D. Stamoulis and J. N. Tsitsiklis. The efficiency of greedy routing in hypercubes and butterflies. *Procs. of the 6th Annual ACM Symp. on Parallel Algorithms and Architectures.* Pages 346–353, 1994.

[14] E. Upfal. Efficient schemes for parallel communication. *Journal of the ACM*, 31 (1984):507–517.