How Much Can Hardware Help Routing?

ALLAN BORODIN

University of Toronto, Toronto, Ont., Canada

PRABHAKAR RAGHAVAN

IBM Almaden Research Center, San Jose, California

BARUCH SCHIEBER

IBM T. J. Watson Research Center, Yorktown Heights, New York

AND

ELI UPFAL

IBM Almaden Research Center, San Jose, California and Weizmann Institute, Rehovot, Israel

Abstract. We study the extent to which complex hardware can speed up routing. Specifically, we consider the following questions. How much does adaptive routing improve over oblivious routing? How much does randomness help? How does it help if each node can have a large number of neighbors? What benefit is available if a node can send packets to several neighbors within a single time step? Some of these features require complex networking hardware, and it is thus important to investigate whether the performance justifies the investment. By varying these hardware parameters, we obtain a hierarchy of time bounds for worst-case permutation routing.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems; G.3 [**Probability and Statistics**]

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Multi-port, packet routing, permutation routing, randomized routing algorithms, single-port

---

E. Upfal's research at the Weizmann Institute was supported by The Norman D. Cohen Professorial Chair of Computer Science.

Authors' present addresses: A. Borodin, Department of Computer Science, University of Toronto, Sanford Fleming Building, 10 King's College Road, Toronto, Ontario, Canada M5S 3G4, e-mail: borodin@theory.toronto.edu; P. Raghavan, IBM Research Division, Almaden Research Center, San Jose, CA; B. Schieber, IBM Research Division, T. J. Watson Research Center, Yorktown Heights, NY; E. Upfal, IBM Research Division, Almaden Research Center, San Jose, CA, and Department of Applied Mathematics, Weizmann Institute, Rehovot, Israel.

1. *Introduction*

The availability of novel hardware features in some technologies may enable the realization of networks that are more powerful than existing ones. For instance, optical technology can support networks in which each node has a large number of neighbors and can communicate with many of them simultaneously. (See, e.g., Green [1991] and Ramaswami [1993].) We study the extent to which such powerful hardware features can reduce routing time.

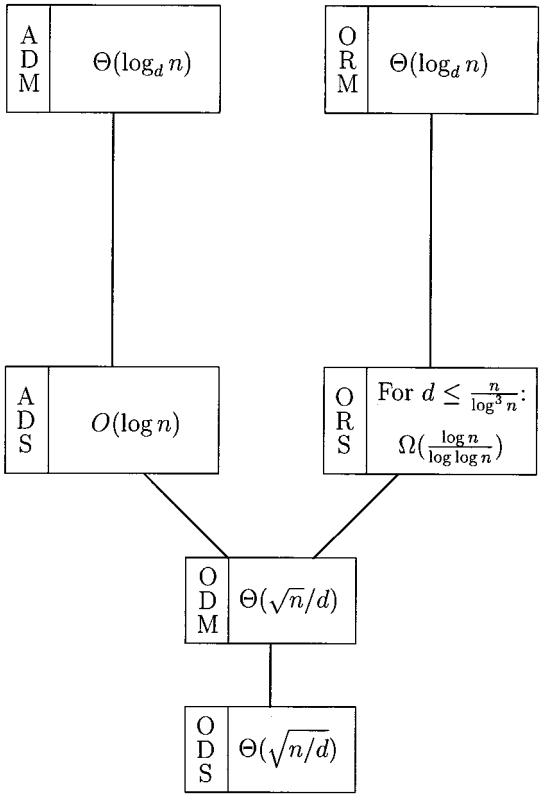
We study the *permutation routing* problem: Each of n nodes in a parallel computation network has a packet that is to be sent to another node, and each node receives exactly one packet. Routing proceeds in synchronous steps. The network is modeled as an undirected graph each of whose edges has bidirectional links; thus, an edge can carry one packet in each direction during each step. (We remark that all our positive results hold for partial permutations as well.)

An (n, d)-*routing scheme* $S_{n,d}$ consists of a network of n nodes each of degree at most d, together with a permutation routing algorithm for the network. Let $T(S_{n,d})$ denote the maximum, over all instances, of the number of steps when the scheme $S_{n,d}$ is used. Let $T(n, d)$ denote the minimum, over all (n, d)-routing schemes $S_{n,d}$, of $T(S_{n,\,d})$. We study $T(n, d)$ as a function of n and d. The availability of a large node degree d in some technologies (e.g., optical networks) raises an obvious question: how does $T(n, d)$ depend on d? In this context, we study a number of "hardware" issues. (1) How does the time for *oblivious routing* (in which the possible paths followed by a packet depend only on its own source and destination) compare with the time for *adaptive routing*? Adaptive routing usually requires more complicated routing hardware and protocols. (2) What is the benefit of algorithms that allow nodes to send out packets along more than one outgoing link in a single time step? While this could reduce the overall time for routing, it means that the communications ports have to be more complex. (3) What is the power of randomization in networks with these features? (For randomized algorithms, we define $T(n, d)$ to be the number of steps within which routing is guaranteed to terminate with probability $1 - o(1)$; note that for any "reasonable algorithm" the expected running time is at most one step more.)

The reader familiar with the routing literature may immediately obtain a succinct picture of our results from Figure 1. In this figure and elsewhere throughout the paper, the letters A and O distinguish adaptive (A) and oblivious (O) routing schemes. The letters R and D distinguish randomized and deterministic schemes. The letters M and S distinguish multi-port and single-port routing. In multi-port routing, a node is allowed to send a packet along each of its outgoing edges simultaneously, whereas in single-port routing only one outgoing edge may be active at any time. (Note that this is the least restrictive model of single-port routing. In more restrictive models, we may also limit the number of packets coming into a node in a step.)

Clearly, there is a hierarchy among the models we consider: the ARM model is the strongest model, while the ODS model is the weakest. Our work determines the structure of this hierarchy, proving some models to be as strong as ARM, and showing others to be strictly weaker.

Much is known about permutation routing in a "sparse" network. In particular, for oblivious routing, we know that $\sqrt{n}$ steps are asymptotically optimal for deterministic routing on the butterfly. (See Borodin and Hopcroft [1985] and



NOTE: Unless indicated otherwise, all logarithms are base e.
FIG. 1. The routing hierarchy.

Kaklamanis et al. [1991].) On the other hand, Valiant's two stage randomized algorithm runs in optimal $\Theta(\log n)$ time[1] for sparse networks [Aleliunas 1982; Upfal 1992; Valiant and Brebner 1981]. For adaptive routing, there are $\Theta(\log n)$ deterministic routing algorithms based on AKS sorting [Ajtai et al 1993] that can be implemented on a constant degree network [Leighton 1985], as well as simpler schemes using the multibutterfly network [Leighton and Maggs 1989; Upfal 1992]. The optimality of the $O(\log n)$ time routing algorithms for constant degree networks follows easily from the diameter bound: in any n node, degree d network, the diameter is at least $\log_d n$. For high degree networks, when can we meet this $\Omega(\log_d n)$ bound?

1.1. SUMMARY OF RESULTS. In Section 2, we consider oblivious routing. Somewhat unexpectedly, we show that oblivious randomized single-port (ORS) routing faces an intrinsic bottleneck: for any ORS scheme there is an instance requiring $\Omega(\log n/\log\log n)$ steps even when d is nearly n (where the diameter bound is just a constant). Thus, schemes such as Valiant's cannot be improved much beyond the $O(\log n)$ running time bound, even in very dense networks. We complement this with results showing that if we strengthen the ORS model by

[1] All logarithms are base e unless indicated otherwise.

allowing multi-port routing, this bottleneck can be overcome. Specifically, for every n and d, we give an ORM scheme for routing that achieves the diameter bound of $\Theta(\log_d n)$ steps (Section 2.3). We complete our study of oblivious routing by giving tight bounds for oblivious deterministic routing, for both the single-port (ODS) and the multi-port (ODM) cases. In Section 3.1, we show that we can meet the diameter bound of $\Omega(\log_d n)$ in the adaptive deterministic multi-port (ADM) model.

In summary, our major new results are: On the positive side, we have ADM and ORM (and thus ARM) routing schemes that meet the diameter bound. On the negative side, we show that the diameter bound cannot be achieved by ODM and ORS (and thus ODS) schemes. All the bounds we have are essentially tight for the models studied. The preliminary conference version of this work [Borodin et al. 1993] included a sketch of an ARS algorithm that does not meet the diameter bound. The running time of that algorithm can be stated (for simplicity) as $(\log \log n)^{O(1)} \log_d n$. Because of the apparent nonoptimality of that scheme, we omit it here and plan to reconsider ARS routing in a separate paper. The only model we leave completely unresolved (between the diameter bound of $\Omega(\log_d n)$ and the multibutterfly upper bound of $O(\log n)$) is the ADS model. A recurring feature highlighted by our results is that the single-port model presents difficult challenges that were obscured in previous work on low-degree networks, where the time bounds grew with node degrees.

2. *Oblivious Routing*

2.1. Oblivious Deterministic Routing—Tight Bounds. Borodin and Hopcroft [1985] proved that for any n and d, and for any deterministic single-port oblivious routing algorithm, there is a permutation requiring $\Omega(\sqrt{n/d})$ steps. A modification of this argument shows that for any n and d, and for any deterministic multi-port oblivious routing algorithm, there is a permutation requiring $\Omega(\sqrt{n}/d)$ steps [Kaklamanis et al. 1991]. We show these bounds to be tight.

2.1.1. *Single-Port Case*. In this case, the network is a Cartesian product of two graphs: a square mesh of size $\sqrt{n/d} \times \sqrt{n/d}$, and the complete graph K_d on d nodes. Clearly the network has n nodes and degree $d + 3$. It is useful to view the network as a mesh of cliques; thus, the edges of the network can be partitioned into *mesh edges* and *clique edges*. Address each node by a triplet (r, c, d), where r is its row address and c its column address (in the mesh), and d is its clique address. Routing is accomplished by alternating *local steps* that only use clique edges, and *systolic* steps that only use mesh edges. In a local step, each node v checks whether the destination column address of the packet it currently holds is the same as its own column address; if so, the packet is sent to the node whose clique address is the destination clique address. If not, the packet is sent out in the following systolic step. In a systolic step, each node v checks whether the destination column address of the packet it currently holds is the same as its own column address; if not, the packet is sent along the row towards its column. Otherwise, if the column address of the packet is the same as the column address of v (and, in this case, the clique addresses are also identical), then the packet is sent along the column towards its row.

Consider a packet p. Before it is sent along a clique edge, it incurs no delay while traveling along the row. In turning to the clique edge, and in moving along the column edges it may be delayed only by other packets whose destination column is the same as the destination column of p. Each such delay can be charged to a unique packet by a "chain of delays" argument. For example, suppose p is delayed by q which is later delayed by r and r then goes on without any delays to its destination. Then p's delay, as well as q's delay, is charged to r.[2] Thus, the total delay of packet p is bounded by $\sqrt{n/d}$. The time bound is now easy to establish.

2.1.2. MULTI-PORT CASE. We make use of the following fact. The Cartesian product of two complete graphs K_d on d nodes (which we denote by K_d^2) is a network on which any permutation on d^2 nodes can be routed in two steps using an oblivious deterministic multi-port algorithm. To see this, address each node by a pair (d_1, d_2), where d_1 is its first clique address and d_2 is its second clique address. Routing from node (d_1, d_2) to node (e_1, e_2) is accomplished in two steps: In the first step, the packet is sent to node (e_1, d_2) and in the second step the packet is sent to (e_1, e_2). It is easy to see that in case of a permutation there is no edge congestion, and thus the routing can be done in two time steps. Our network is now the product of a square mesh of size $\sqrt{n}/d \times \sqrt{n}/d$, and K_d^2. The degree of this network is $(d - 1) + (d - 1) + 4 = 2d + 2$. By extending the algorithm and analysis of the single-port case, the time bound follows.

2.2. OBLIVIOUS RANDOMIZED SINGLE-PORT—LOWER BOUND. The lower bound uses the von Neumann minimax principle, using random inputs on an oblivious deterministic single-port (ODS) scheme to guarantee a bad input for any oblivious randomized single-port (ORS) scheme. Aiello et al. [1991] give a result that resembles Theorem 2.2.1 below. However, it should be noted that their result assumes a bit-serial model and that a packet carries a certain amount of routing information. Our result makes no such assumptions, and is purely a statement of expected maximum congestion.

THEOREM 2.2.1. *For any n-node network of degree $d \leq n/\log^3 n$, and any ODS scheme, the expected routing time for a random permutation (with each permutation chosen with uniform probability) is $\Omega(\log_d n + \log n/\log\log n)$.*

We remark that the limiting condition $d \leq n/\log^3 n$ occurs in Lemma 4, and in fact any d that is $o(n/\log^2 n)$ will suffice for its proof. It is plausible that the theorem holds for any d asymptotically smaller than $n/\log n$.

By the von Neumann minimax principle [Yao 1977], we immediately have:

COROLLARY 2.2.2. *For any n-node network of degree $d \leq n/\log^3 n$, and any ORS scheme, there is a permutation for which the expected routing time is $\Omega(\log_d n + \log n/\log\log n)$.*

PROOF OF THEOREM 2.2.1. The $\log_d n$ term is the diameter lower bound. The second term is due to the expected congestion at some node. Note that since in a single-port model a node is not allowed to send more than one packet at a time, the congestion at a node provides a lower bound on the routing time. We show that the expected maximum congestion is $\Omega(\log n/\log\log n)$.

[2] For a detailed discussion of the chain of delays argument, see Felperin et al. [1996].

In an oblivious deterministic algorithm, the route between any source-destination pair is fixed. For (almost) every source-destination pair (u, v), we select a specific internal node in the route from u to v, in a way described later. We call this node the *assigned* node of (u, v), and denote it $V(u, v)$.

For any permutation π and two nodes u and w define

$$I_{uw}(\pi) = \begin{cases} 1 & \text{if } w = V(u, \pi(u)). \\ 0 & \text{otherwise.} \end{cases}$$

Let π be a random permutation. Define $P_{uw} = \Pr\{I_{uw}(\pi) = 1\}$. Thus, $\Sigma_u P_{uw}$ gives us a lower bound on the expected congestion at w.

We show that there are sufficiently many nodes w for which $\Sigma_u P_{uw} \geq \delta$, for a positive constant δ. Then, we show that for a particular such node w, there is a reasonably large probability that $\Sigma_u I_{uw} \geq \log n/4 \log \log n$. This will ensure the desired expected maximum congestion is as desired.

Fix a node u. We show how to assign $V(u, v_i)$, for most $v_i \neq u$. This assignment will have the following properties:

(1) A node w may be the assigned node of at most $(d + 1) \log n$ pairs (u, v_i).
(2) If a node w is the assigned node of some pair (u, v_i), then it is the assigned node of at least $\log n$ such pairs.
(3) For at least $n - d \log n$ nodes v_i, $V(u, v_i)$ is defined.

LEMMA 2.2.3. *There exists an assignment with the above three properties.*

PROOF. Fix u and consider the possible destinations $v_1, v_2, \ldots, v_{n-1}$ with $v_i \neq u$. We show a procedure that determines the assignments of $V(u, v_i)$. The procedure has two stages.

In the first stage, we iterate the following. Select an unmarked node w that is an internal node in at least $\log n$ routes that have not been deleted (in a manner described next), if such exists. Mark w, and assign it to $\log n$ of these routes (chosen arbitrarily). Delete these $\log n$ routes. The first stage ends when there is no such node w.

In the second stage, follow each of the undeleted routes (u, v_i), from v_i back to u. Assign (u, v_i) to the first marked node encountered on the way, if such exists.

Now, we prove that this procedure indeed gives the desired assignment. A marked node w is assigned exactly $\log n$ routes in the first stage. Next, we bound the number of routes it is assigned in the second stage. At most d of these routes may be routes that begin at one of the neighbors of w. The rest of these routes must contain at least one of the unmarked neighbors of w as an internal node. For each unmarked neighbor, we may have at most $\log n - 1$ such routes. Hence, their number is bounded by $d(\log n - 1)$. We conclude that the total number of routes assigned to w is between $\log n$ and $(d + 1) \log n$.

To bound the number of unassigned routes at the end of the procedure, observe that each unassigned route either begins at a neighbor of u or contains an unmarked neighbor of u as an internal node. Thus, the number of these routes is bounded by $d(\log n - 1) + d = d \log n$. □

Given an assignment with these properties, P_{uw} satisfies the following.

(1) For any two nodes u and w, either

$$P_{uw} = 0, \text{ or } \frac{\log n}{n} \leq P_{uw} \leq (d+1)\frac{\log n}{n}.$$

(2) $\Sigma_w \Sigma_u P_{uw} = \Sigma_u \Sigma_w P_{uw} \geq \Sigma_u \frac{n - d\log n}{n} = n - d\log n.$

If there exists some w such that $\Sigma_u P_{uw} \geq \log n$, then clearly the congestion bound follows. Suppose that for all nodes w, $\Sigma_u P_{uw} < \log n$.

We fix a node w satisfying $\Sigma_u P_{uw} \geq \delta$ and analyze the probability (for a random permutation) that the congestion $\Sigma_u I_{uw}$ exceeds $\log n/4 \log\log n$. Intuitively, we are throwing n "balls" at a bin w. The probability of a "hit" is at most $(d+1)\log n/n$, and the expected number of hits is $\Omega(1)$. Our goal is to get a lower bound on the probability that the maximum number of hits is $\Omega(\log n/\log\log n)$. As will be formalized later, this bound is achieved when $\Omega(n/((d+1)\log n)$ balls are thrown with hit probability $(d+1)\log n/n$, and the rest have zero hit probability.

There is a small problem with the "ball and bin" intuition: it does not account for the fact that the hits must correspond to a (partial) permutation. That is, the hits must correspond to different destination nodes. (Notice that they correspond to different source nodes.) To overcome this problem we use the property that if $P_{uw} > 0$, then $P_{uw} \geq \log n/n$. Suppose that there is a "balls to bins" assignment with more than h hits for some $h \leq \log n$. Each such hit corresponds to some source node u, with $P_{uw} \geq \log n/n$. Thus, there are at least $\log n$ nodes v such that w is $V(u, v)$. Since $h \leq \log n$, we can match at least one different destination for each source. This matching restricts the number of possible assignment, thus P_{uw} may be now as small as $1/n$ (if it is not zero).

LEMMA 2.2.4. *Let $x_1, \ldots, x_k$ be independent* 0/1 *random variables. Let $Pr\{x_i = 1\} = p_i$, $1 \leq i \leq k$, $\Sigma_{i=1}^k p_i \geq \delta$, for some fixed $\delta > 0$, and $(1/n) \leq p_i \leq (d+1)\log n/n$. Then, for $d \leq n/\log^3 n$*

$$Pr\left\{\sum_{i=1}^{k} x_i \geq \frac{\log n}{c \log\log n}\right\} \geq n^{-2/c}.$$

To prove this lemma, we first prove the following more general lemma.

LEMMA 2.2.5. *Let $x_1, \ldots, x_k$ be independent* 0/1 *random variables. Let $Pr\{x_i = 1\} = p_i$, $1 \leq i \leq k$, $\Sigma_{i=1}^k p_i \geq \delta$, for some fixed $\delta > 0$, and $a \leq p_i \leq b$. Then, for any $Y \leq \delta/2b \log(b/a)$,*

$$Pr\{\sum_{i=1}^{k} x_i \geq Y\} \geq \frac{1}{2}\left(\frac{1}{2Y}\right)^Y \left(\frac{\delta}{e \log(b/a)}\right)^Y.$$

PROOF. Without loss of generality assume that $\log(b/a)$ is a positive integer, and $\delta < 1$ (larger δ can only increase the probability we compute). Let $P(1) = \Sigma_{a \leq p_i \leq ae} p_i$, and for $j = 2, \ldots, \log(b/a)$, let $P(j) = \Sigma_{ae^{j-1} < p_i \leq ae^j} p_i$. Since $\Sigma_{j=1}^{\log(b/a)} P(j) \geq \delta$ there exists $s \in \{1, \ldots, \log(b/a)\}$ such that $P(s) \geq \delta/\log(b/a)$. Let $x_{s_1}, \ldots, x_{s_m}$ be the m variables with probabilities in this interval. Observe that $P(s) \leq mb$ and thus $m \geq \delta/(b \log(b/a))$. Also, for some $1 \leq i \leq m$, $p_{s_i} \geq \delta/(m \log(b/a))$. Since all the

probabilities in this interval are within a factor of e, for all $1 \le i \le m$, $p_{s_i} \ge \delta/(em \log(b/a))$. Let $B(n, p)$ denote a random variable with the binomial distribution with parameters n and p. We get

$$\Pr\{\sum_{\ell=1}^{m} x_{s_\ell} \ge Y\} \ge \Pr\left\{B\left(m, \frac{\delta}{em \log(b/a)}\right) \ge Y\right\} \ge \binom{m}{Y}\left(\frac{\delta}{em \log(b/a)}\right)^Y \times \left(1 - \frac{\delta}{em \log(b/a)}\right)^{m-Y} \ge \left(\frac{m-Y}{Y}\right)^Y\left(\frac{\delta}{em \log(b/a)}\right)^Y \times \left(1 - \frac{(m-Y)\delta}{em \log(b/a)}\right) \ge \left(\frac{1}{2Y}\right)^Y\left(\frac{\delta}{e \log(b/a)}\right)^Y \frac{1}{2}.$$

The last step follows since

$$m \ge \frac{\delta}{b \log(b/a)} \text{ and } Y \le \frac{\delta}{2b \log(b/a)}$$

and therefore $Y \le m/2$. □

Proof of Lemma 2.2.4. To apply Lemma 2.2.5, we consider only balls with positive hit probabilities. Thus, we have k independent 0/1 random variables, where $\Pr\{x_i = 1\} = p_i$, $1 \le i \le k$, $\sum_{i=1}^{k} p_i \ge \delta$ and $1/n \le p_i \le (d+1)\log n/n$. Then, setting $d \le n/\log^3 n$ (to get that

$$\frac{\log n}{c \log\log n} \le \frac{n\delta}{2e(d+1)\log n \log((d+1)\log n)},$$

for sufficiently large n), we get the desired bound. □

Consider a node w for which $\sum_u P_{uw} \ge \delta$, call it a *good* node. By our assumption $\sum_u P_{uw} < \log n$. Thus, for any fixed $\delta < 1$, the number of good nodes is at least $(n(1-\delta) - d \log n/\log n = \Omega(n/\log n)$. By Lemma 2.2.4, the probability of a good node w having a congestion of $\log n/4 \log\log n$ is at least $1/\sqrt{n}$.

We would like to conduct the following trials until the first success: pick a good node w and check whether it has a congestion of $\log n/4 \log\log n$. By the above argument, we know that the success probability of the *first* trial is at least $1/\sqrt{n}$. Later we show that this bound holds for the first $n/\log n$ trials. Hence, the probability that in the first $t = n/\log n$ trials one of the nodes has congestion $\log n/(4 \log\log n)$ is at least $1 - (1 - 1/\sqrt{n})^t > 1 - (1 - 1/\sqrt{n})^{\sqrt{n}} \approx 1 - 1/e$. Thus, the expected maximum congestion is $\Omega(\log n/\log\log n)$.

Fix some $t \le n/\log n$. It remains to argue that the success probability of the tth trial given that the previous $t-1$ trials failed is at least $1/\sqrt{n}$. Each of the previous $t-1$ nodes that is not congested constrains the destinations of fewer than $\log n/\log\log n$ source nodes. If we set the destinations for these source nodes and eliminate the traffic originating at them, then the total expected congestion $\sum_u \sum_w P_{uw}$ is reduced by at most $(t-1)\log n/\log\log n$; that is, it remains $\Omega(n)$. It follows that there are still $\Omega(n/\log n)$ good nodes, even after this traffic has been removed. Note that after removing this traffic, I_{uw} cannot

decrease for the good nodes w. Thus, the probability that the good node w has congestion $\log n/4 \log\log n$ is at least $1/\sqrt{n}$. □

To conclude this section, we prove a high probability lower bound using a variation of the von Neumann minimax principle [Yao 1977].

THEOREM 2.2.6. *For any n-node network of degree $d \le n/\log^3 n$, and any ORS scheme, there is a permutation for which with probability at least $1 - n^{-c}$ (for any constant c) the routing time is $\Omega(\log_d n + \log n/\log\log n)$.*

PROOF. Let $\mathcal{A}$ be the set of all ODS schemes, and let S_n be the symmetric group on n elements. For an ODS scheme $A \in \mathcal{A}$ and a permutation $\pi \in S_n$, define

$$T_{A,\pi} = \begin{cases} 1 & \text{if } A \text{ routes } \pi \text{ in at least } \frac{1}{2}(\log_d n + \log n/\log\log n) \text{ time steps.} \\ 0 & \text{otherwise.} \end{cases}$$

Consider a scheme $A \in \mathcal{A}$. From the proof of Theorem 2.2.1, it follows that with probability $1 - (1 - 1/\sqrt{n})^{n/\log n} \ge 1 - n^{-c}$ (for any constant c) A routes a random permutation (chosen uniformly) in at least $\frac{1}{2}(\log_d n + \log n/\log\log n)$ time steps. This implies that $\sum_{\{\pi \in S_n\}} T_{A,\pi} \ge (1 - n^{-c})n!$.

Consider any ORS scheme R. This scheme can be viewed as a distribution over the ODS schemes. For $A \in \mathcal{A}$, let p_A be the probability that the ORS scheme R is identical to the ODS scheme A. It follows that for a fixed permutation π the probability that R routes π in at least $\frac{1}{2}(\log_d n + \log n/\log\log n)$ time steps is $\sum_{\{A \in \mathcal{A}\}} p_A \cdot T_{A,\pi}$. We claim that for at least one permutation π: $\sum_{\{A \in \mathcal{A}\}} p_A \cdot T_{A,\pi} \ge 1 - n^{-c}$.

To obtain a contradiction, assume that for all permutations π: $\sum_{\{A \in \mathcal{A}\}} p_A \cdot T_{A,\pi} < 1 - n^{-c}$. Then, $\sum_{\{\pi \in S_n\}} \sum_{\{A \in \mathcal{A}\}} p_A \cdot T_{A,\pi} = \sum_{\{A \in \mathcal{A}\}} p_A \sum_{\{\pi \in S_n\}} T_{A,\pi} < (1 - n^{-c})n!$. However, since $\sum_{\{A \in \mathcal{A}\}} p_A = 1$, this implies that for at least one scheme $A \in \mathcal{A}$ $\sum_{\{\pi \in S_n\}} T_{A,\pi} < (1 - n^{-c})n!$. A contradiction. The theorem follows. □

2.3. OBLIVIOUS RANDOMIZED MULTI-PORT—UPPER BOUND. In this section, we analyze Valiant's oblivious randomized multi-port algorithm on the d-way wrap-around butterfly network.

2.3.1. *The Network.* Let $n = m \log_d m$ and suppose that $\log_d m$ is an integer. The d-way butterfly has $\log_d m$ layers, each with m nodes. Number the layers $0, \ldots, \log_d m - 1$, and the nodes in each layer by $0, \ldots, m - 1$. A location in the network is characterized by a pair (ℓ, x), where ℓ is the layer number, and x is the node number in the layer. Let $x_0, \ldots, x_s$ denote the base d representation of the number x. A node (ℓ, x) is connected to d nodes in layer $\ell + 1 \pmod{(\log_d m)}$. The numbers of these d nodes are $x_0, \ldots, x_{\ell-1}, *, x_{\ell+1}, \ldots, x_s$, where $* = 0, \ldots, d - 1$.

2.3.2. *The Algorithm.* Consider a packet at origin (ℓ_O, x_O) with destination (ℓ_D, x_D). The packet travels to its destination in three stages. In the first stage, the packet makes $\log_d m$ "random" transitions, where in each random transition the packet leaves its current location by an outgoing edge chosen at random from the d outgoing edges of this node. Each random choice is independent of

previous choices of this packet, and of the choices for other packets. At the end of the first stage, the packet is at a random node in the same stage as its origin. In the second stage the packet takes another $(\ell_D - \ell_O)$ "random" transitions. At the end of the second stage, the packet is at random node in same stage as its destination node. In the third stage, the packet reaches its destination in another $\log_d m$ (deterministic) transitions. Since a node can use all its outgoing edges simultaneously, we assume that each outgoing edge has its own queue. The queues are priority queues. The *priority number* of a packet in the first stage is the number of edges already traversed. In the second stage, it is $\log_d m$ plus the number of edges already traversed in that stage, and in the third stage it is $3 \log_d m$ minus the number of edges traversed in the third stage. Packets with lower priority number have higher priority and ties are broken arbitrarily.

THEOREM 2.3.2.1. *There is an ORM scheme that routes an arbitrary permutation on an n node d-way butterfly in $O(\log_d n)$ steps with high probability.*

PROOF. We analyze the delay of each stage separately. We start with the first stage. Our analysis uses the critical delay sequence method [Uptal 1984]. Given an execution of the algorithm, we define a critical delay sequence $\mathcal{D} = e_1, \ldots, e_{\log_d m}$, for the first stage of the algorithm with respect to this execution. The last edge in the sequence, $e_{\log_d m}$, is one of the last edges to transmit packets with priority $\log_d m$ in this execution. If $e_{i+1} = v \to w$ then e_i is one of the last edges to transmit packets with priority i or less amongst e_{i+1} and the d ingoing edges of node v.

Let t_i denote the time at which edge e_i of the critical delay sequence finished transmitting all packets of priority i or less. Let f_i denote the number of packets with priority i that traversed edge e_i of the critical delay sequence.

Clearly $t_i \le t_{i-1} + f_i$, and the run-time of the first stage is bounded by

$$t_{\log_d m} - t_0 = \sum_{i=1}^{\log_d m} t_i - t_{i-1} \le \sum_{i=1}^{\log_d m} f_i,$$

where we define $t_0 = 0$.

A delay sequence is any sequence of edges $e_1, \ldots, e_{\log_d m}$, such that for every $1 \le i < \log_d m$ either $e_i = e_{i+1}$, or $e_{i+1} = v \to w$ and e_i is one of the d ingoing edges of node v. Note that the set of delay sequences includes the critical delay sequence(s). Let $f_i = g_i + h_i$, where g_i counts packets that were not counted in $\Sigma_{j<i} f_j$ (i.e., g_i counts packets that were counted first in edge e_i). Since in the first stage each packet chooses its outgoing edge at each stage independently at random, for any given edge (whether or not the edge is in the critical delay sequence)

$$E[g_i] \le d^{i-1} \frac{1}{d^{i-1}} \frac{1}{d} = \frac{1}{d},$$

and the distribution of g_i is stochastically bounded by $B(d^i, 1/dd^i)$. Thus, by Hoeffding's theorem [Hoeffding 1958], $\Sigma_{i=1}^{\log_d m} g_i$ is stochastically dominated by $B(n, 1/md)$.

Let G be the maximum over all delay sequences of $\Sigma_{i=1}^{\log_d m} g_i$. There are no more than $n(d+1)^{\log_d m} \le n^3$ possible delay sequences; thus

$$\Pr\{G > 2e \log_d n\} \le n^3 \sum_{k>2e \log_d n} \Pr\left\{B\left(n, \frac{1}{md}\right) \ge k\right\}$$
$$\le n^3 \sum_{k>2e \log_d n} \binom{n}{k}\left(\frac{1}{md}\right)^k$$
$$\le n^3 \sum_{k>2e \log_d n} \left(\frac{\log_d m}{k}\right)^k \left(\frac{1}{d}\right)^k = o(1).$$

Let H be the maximum over all delay sequences of $\sum_{i=1}^{\log_d m} h_i$. Next, we bound H given that $G \le 2e \log_d n$. Consider a packet p, first counted in f_i. Since the packet is taking a random path, the number of transitions it makes until it leaves a given path is stochastically bounded by a geometric distribution with probability of success $1 - (1/d)$. Due to the structure of the butterfly network, once a packet leaves a path, it cannot return to it in that stage. Thus, the probability that G packets traversed a total of H edges in the path is bounded by the probability of less than G successes in $H + G - 1$ trials each with success probability $1 - (1/d)$. We get that for $d \ge 10$

$$\Pr\{H > 4e \log_d n | G \le 2e \log_d n\} \le n^3 \sum_{k>4e \log_d n} \Pr\left\{B\left(k + G, \frac{1}{d}\right) > k\right\}$$
$$\le n^3 \sum_{k>4e \log_d n} 2^{G+k}\left(\frac{1}{d}\right)^k = o(1).$$

Let F be the maximum over all delay sequences of $\sum_{i=1}^{\log_d m} g_i + h_i$. Thus, we get

$$\Pr\left\{\sum_{i=1}^{2 \log_d m} F > 6e \log_d n\right\}$$
$$\le \Pr\{G > 2e \log_d n\} + \Pr\{H > 4e \log_d n | G \le 2e \log_d n\}$$
$$= o(1).$$

To analyze the second stage we need to define the delay sequence slightly differently. Given an execution of the algorithm, let p be one of the last packets to traverse an edge in the second stage. Let b be the priority number of this packet in its last transition, and let e_b be the last edge traversed by p. Note that $b \le 2 \log_d m$. Define a delay sequence $\mathcal{D} = e_{\log_d m+1, \ldots,} e_b$, with respect to this execution, starting from e_b backwards. For $i < b$, if $e_{i+1} = v \to w$, then e_i is one of the last edges to transmit packets with priority i or less among e_{i+1} and the d ingoing edges of node v. Define f_i, g_i and h_i as before. The remainder of the proof is similar to the analysis of the delay sequence for the first stage. The bound on the delay in the third stage is obtained by its symmetry to the first stage. □

3. *Adaptive Routing*

3.1. ADAPTIVE DETERMINISTIC MULTI-PORT. We construct an n-input n-output, degree d network that routes any end-to-end permutation in $O(\log_d n)$

steps. Our construction generalizes the multibutterfly routing scheme [Upfal 1992] to networks of large degree and small diameter. The scheme of Upfal [1992] already gives a (single-port) bound of $O(d \log_2 n)$ for all $d \geq 4$. We therefore concentrate on the case of large d, any $d > d_0$, for some large constant d_0. (To simplify the presentation, no attempt is made to obtain the best constants.)

3.1.1. *The Network*. The basic building block of the network is a $\sqrt{d}$-way m-splitter. A $\sqrt{d}$-way m-splitter has one set of m input nodes and $\sqrt{d}$ output sets, each with $m/\sqrt{d}$ nodes. Each input has $\sqrt{d}$ edges to each of the $\sqrt{d}$ output sets. The edges connecting the input set to each of the output sets define an expander graph with the following properties:

(1) Even if for each input set we arbitrarily erase half of the edges to each output set from that input set, each set X of at most $m/10d$ inputs is connected to more than $\sqrt{d}|X|/10$ outputs in each output set.
(2) For a given set of inputs X, let $\Gamma(X, \sqrt{d}/4, i)$ denote the set of vertices in the ith output set with at least $\sqrt{d}/4$ neighbors in X. We require that for each set X of at most $m/16e$ inputs, $|\Gamma(X, \sqrt{d}/4, i)| < |X|/\sqrt{d}$.

The network has $2 \log_d n + 1$ layers. The vertices at layer $0 \leq i \leq 2\log_d n$ are partitioned into $(\sqrt{d})^i$ sets of $m_i = n/(\sqrt{d})^i$ vertices. Each of the sets in layer i is an input set of a $\sqrt{d}$-way m_i-splitter. The output sets of that splitter are $\sqrt{d}$ sets of size m_{i+1} in layer $i + 1$.

LEMMA 3.1.1.1. *There exists an expander graph with the above properties*.

PROOF. It is enough to show the existence of the desired graph between the set of inputs and one set of outputs. Choose a random bipartite graph with m vertices in one side, each of degree $\sqrt{d}$, and $m/\sqrt{d}$ vertices on the other side, each with degree d. The probability that Property (1) fails is bounded by

$$\sum_{k \leq m/(10d)} \binom{m}{k} \binom{\frac{m}{\sqrt{d}}}{\frac{k\sqrt{d}}{10}} \binom{\sqrt{d}}{\sqrt{d}/2}^k \left(\frac{dk}{10m}\right)^{k\sqrt{d}/2},$$

which is $o(1)$.

The probability that Property (2) fails is bounded by

$$\sum_{k \leq m/(16e)} \binom{m}{k} \binom{\frac{m}{\sqrt{d}}}{\frac{k}{\sqrt{d}}} \binom{k\sqrt{d}}{\frac{k\sqrt{d}}{4}} \left(\frac{k}{m}\right)^{k\sqrt{d}/4},$$

which is also $o(1)$. □

3.1.2. *The Algorithm*. Nodes at odd levels of the multibutterfly transmit in odd stages, while nodes at even levels transmit in even stages. A stage has three

steps. In the first step, each node in a transmitting level sends a request message to all its neighbors in output sets to which it has packets to transmit. (Note that the given permutation determines the output set for each packet.) A node in a receiving level that receives fewer than $\sqrt{d}/4$ messages, and currently stores fewer than $\sqrt{d}/4$ packets, replies in the second step with a "ready" message to its neighbors in the previous layer. In the third step, each node in a transmitting level sends packets to some of the nodes that reply with a "ready" message. Suppose that a node in a transmitting level has to transmit k packets to a specific output set, and suppose that k' of its neighbors in this output set replied with a "ready" message. Then, the node selects $\min\{k, k'\}$ neighbors out of them and sends each one a distinct packet out of the k packets.

3.1.3. *Analysis*. Consider a splitter in a stage in which the inputs of the splitter are transmitting packets to the outputs. Fix an output set Y of that splitter. Let k (respectively, k') be the number of packets that need to traverse output set Y and that are stored at the beginning (respectively, end) of that stage in input nodes of that splitter. Let ℓ be the number of packets stored in output set Y at the beginning of this stage.

LEMMA 3.1.3.1. *For sufficiently large d,*

$$k' < \frac{20}{\sqrt{d}}(k + \ell).$$

PROOF. At the beginning of a stage, no node has more than $\sqrt{d}/2$ packets. Thus, if a node stores packets at the end of a stage, at least $\sqrt{d}/2$ of its neighbors in output set Y either received more than $\sqrt{d}/4$ requests, or started the stage with at least $\sqrt{d}/4$ packets.

At most $m/\sqrt{d}$ packets traverse nodes in output set Y. For sufficiently large d, $m/\sqrt{d} \leq m/16e$. Thus, by Property (2) of the expander graphs, the number of nodes in output set Y that received more than $\sqrt{d}/4$ requests is at most $k/\sqrt{d}$.

Let IY be the set of input nodes that, at the end of the stage, store packets that need to be transmitted to output set Y, and let $|IY| = p'$. We claim that $p' \leq m/10d$. Suppose that $p' > m/10d$, then IY has at least $(m/10\sqrt{d} - k/\sqrt{d})$ neighbors in output set Y each storing at least $\sqrt{d}/4$ packets. Since $k \leq m/\sqrt{d}$, the number of such neighbors is at least $(m/10\sqrt{d} - m/d)$. But then for sufficiently large d, the total number of packets that pass output set Y is at least $\sqrt{d}/4 \cdot (m/10\sqrt{d} - m/d) > m/\sqrt{d}$, a contradiction.

Since the input nodes in IY still store packets at the end of the stage, we know that at least half of the neighbors of each node in IY did not accept packets at this stage. Since $p' \leq m/10d$, by the expansion property (Property (1)) IY has at least $p'\sqrt{d}/10$ neighbors in output set Y that did not accept packets at this stage. There are no more than $k/\sqrt{d}$ output nodes that received more than $\sqrt{d}/4$ requests, and no more than $4\ell/\sqrt{d}$ of output nodes stored more than $\sqrt{d}/4$ packets at the beginning of the stage. Thus, $p'\sqrt{d}/10 \leq k/\sqrt{d} + 4\ell/\sqrt{d}$, or

$$p' \leq \frac{40}{d}(k + \ell).$$

Since no node stores more than $\sqrt{d}/2$ packets $k' \leq 20(k + \ell)/\sqrt{d}$. □

Denote by X_i^t the number of packets in layer i after the execution of stage t.

COROLLARY 3.1.3.2. *If layer i is transmitting packets at stage t, then*

$$X_i^t \leq \frac{20}{\sqrt{d}} (X_i^{t-1} + X_{i+1}^{t-1}) .$$

We analyze the progress of the routing algorithm in terms of a potential function. The analysis is a simplified version of the proof in Leighton and Maggs [1989]. Let $w = d^{1/4}$. The *potential* of a packet after stage t is w^i, if after the execution of that stage the packet is in stage $2 \log_d n - i$ of the network. Let $\Phi(t)$ denote the sum of the potentials of the packets that have *not* reached their destinations after stage t. Clearly $\Phi(0) = nw^{2\log_d n}$, and the routing terminates at the first stage τ such that $\Phi(\tau) < 1$.

Assume that $t + 1$ and i have the same parity (i.e., either both odd or both even). Then, layer i is transmitting at stage $t + 1$, and by Corollary 3.1.3.2

$$X_i^{t+1} \leq \frac{20}{\sqrt{d}} (X_i^t + X_{i+1}^t).$$

Layer $i + 1$ is receiving at that stage, and clearly, $X_{i+1}^{t+1} \leq X_i^t + X_{i+1}^t$.

After the next stage, we get that

$$X_i^{t+2} \leq X_{i-1}^{t+1} + X_i^{t+1} \leq X_{i-2}^t + X_{i-1}^t + \frac{20}{\sqrt{d}} (X_i^t + X_{i+1}^t).$$

Similarly,

$$X_{i+1}^{t+2} \leq \frac{20}{\sqrt{d}} (X_{i+1}^{t+1} + X_{i+2}^{t+1}) \leq \frac{20}{\sqrt{d}} \left(X_i^t + X_{i+1}^t + \frac{20}{\sqrt{d}} (X_{i+2}^t + X_{i+3}^t) \right).$$

Thus, for sufficiently large d, and for all i whether even or odd,

$$X_i^{t+2} \leq X_{i-2}^t + X_{i-1}^t + \frac{20}{\sqrt{d}} \left(X_i^t + X_{i+1}^t + \frac{20}{\sqrt{d}} X_{i+2}^t \right) .$$

Substituting this bound into the potential function, we get that

$$\Phi(t) = \sum_{i=0}^{2\log_d n - 1} X_i^t w^{2\log_d n - i},$$

and

$$\begin{aligned} \Phi(t + 2) &\leq \sum_{i=0}^{2 \log_d n - 1} \left(X_{i-2}^t + X_{i-1}^t + \frac{20}{\sqrt{d}} \left(X_i^t + X_{i+1}^t + \frac{20}{\sqrt{d}} X_{i+2}^t \right) \right) w^{2\log_d n - i} \\ &\leq \sum_{i=0}^{2\log_d n - 1} \left(\frac{1}{w^2} + \frac{1}{w} + \frac{20}{\sqrt{d}} + \frac{20w}{\sqrt{d}} + \frac{400w^2}{d} \right) X_i^t w^{2\log_d n - 1}, \end{aligned}$$

where the last inequality is obtained by rearranging the sum.

Thus, the potential function is decreased by a factor of at least

$$\frac{1}{w^2} + \frac{1}{w} + \frac{20}{\sqrt{d}} + \frac{20w}{\sqrt{d}} + \frac{400w^2}{d} = \Omega(d^{-1/4})$$

over any two stages, and for $\tau = O(\log_d n)$, $\Phi(\tau) < 1$. Thus, we prove:

THEOREM 3.1.3.3. *There is an ADM scheme on an n-input n-output degree d multibutterfly* (*of depth* $2\log_d n$) *that routes any end-to-end permutation in* $O(\log_d n)$ *steps*. *Using the technique in Upfal* [1992] *this scheme readily extends to yield a scheme that routes any global permutation of* $n' = 2n\log_d n$ *packets* (*one at each node of the network*) *in* $O(\log_d n)$ *steps*.

4. *Further Work*

Our results clearly highlight the problem of devising and analyzing single-port schemes. For ADS routing, we have no results that go beyond what is known for small degree. We know of an apparently sub-optimal and relatively complex ARS scheme (see the sketch in Borodin et al. [1993]). It is still an open problem to find a simple and optimal ARS scheme and/or to derive a lower bound showing that the diameter bound cannot be achieved for large degree. (It is plausible that there might be an $\Omega(\log \log n)$ lower bound for degree $d = n^{\epsilon}$.) Our algorithms for single-port routing require nodes to receive more than one packet in a step, a weakness that should be addressed. Our work does not consider the maximum queue size; a natural dichotomy to be studied would be algorithms with bounded versus unbounded queue size.

ACKNOWLEDGEMENTS. We are grateful to Alok Aggarwal and Don Coppersmith for enlightening discussions. We also appreciate the many helpful suggestions of an anonymous referee.

REFERENCES

AIELLO, B., LEIGHTON, F. T., MAGGS, B., AND NEWMAN, M. 1991. Fast algorithms for bit-serial routing on a hypercube. *Math. Syst. Theory 29*, 253–271.

AJTAI, M., KOMLÓS, J., AND SZEMERÉDI, E. 1983. Sorting in $c \log n$ parallel steps. *Combinatorica 3*, 1, 1–19.

ALELIUNAS, R. 1982. Randomized parallel communication. In *Proceedings of the 1st Annual ACM–SIGOPS Symposium on Principles of Distributed Computing* (Ottawa, Ont., Canada, Aug. 18–20). ACM, New York, pp. 60–72.

BORODIN, A., AND HOPCROFT, J. E. 1985. Routing, merging, and sorting on parallel models of computation. *J. Comput. Syst. Sci. 30*, 130–145.

BORODIN, A., RAGHAVAN, P., SCHIEBER, B., AND UPFAL, E. 1993. How much can hardware help routing? In *Proceedings of the 25th Annual Symposium on Theory of Computing* (San Diego, Calif., May 16–18). ACM, New York, pp. 573–582.

FELPERIN, S., RAGHAVAN, P., AND UPFAL, E. 1996. A theory of wormhole routing in parallel computers. *IEEE Trans. Comput. 45*, 704–713.

GREEN, P. 1991. The future of fiber-optic computer networks. *IEEE Comput. 24,* 78–89.

HOEFFDING, W. 1958. On the distribution of the number of successes in independent trials. *Ann. Math. Stat. 27*, 713–721.

KAKLAMANIS, C., KRIZANC, D., AND TSANTILAS, T. 1991. Tight bounds for oblivious routing in the hypercube. *Math. Syst. Theory 24*, 223–232.

LEIGHTON, F. T. 1985. Tight bounds on the complexity of parallel sorting. *IEEE Trans. Comput. C-34*, 344–354.

LEIGHTON, F. T., AND MAGGS, B. 1989. Expanders might be practical: Fast algorithms for routing around faults in multibutterflies. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*. IEEE, New York, pp. 384–389.

PELEG, D., AND UPFAL, E. 1989. The token distribution problem. *SIAM J. Comput. 18*, 229–243.

RAMASWAMI, R. 1993. Multiwavelength lightwave networks for computer communications. *IEEE Commun. Mag. 31*, 2 (Feb.), 78–88.

UPFAL, E. 1984. Efficient schemes for parallel communication. *J. ACM 31*, 3 (July), 507–517.

UPFAL, E. 1992. An $O(\log N)$ deterministic packet routing scheme. *J. ACM, 39*, 1 (Mar.), 55–70.

VALIANT, L. G., AND BREBNER, G. J. 1981. Universal schemes for parallel communication. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, Milwaukee, Wis., May 11–13). ACM, New York, pp. 263–277.

YAO, A. C-C. 1977. Probabilistic computations: Towards a unified measure of complexity. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*. IEEE, New York, pp. 222–227.

RECEIVED FEBRUARY 1995; REVISED JULY 1996; ACCEPTED AUGUST 1997