October 1990 LIDS–P–1999

THE EFFICIENCY OF GREEDY ROUTING IN HYPERCUBES AND BUTTERFLIES †

by

George D. Stamoulis ‡ and John N. Tsitsiklis ‡

Abstract

We analyze the following problem: Each node of the d-dimensional hypercube independently generates packets according to a Poisson process with rate λ. Each of the packets is to be sent to a randomly chosen destination; each of the nodes at Hamming distance k from a packet's origin is assigned an a priori probability $p^k(1-p)^{d-k}$. Packets are routed under a simple greedy scheme: each of them is forced to cross the hypercube dimensions required in increasing index-order, with possible queueing at the hypercube nodes. Assuming unit packet length and no other communications taking place, we show that this scheme is stable (in steady-state) if $\rho < 1$, where $\rho \overset{\text{def}}{=} \lambda p$ is the load factor of the network; this is seen to be the broadest possible range for stability. Furthermore, we prove that the average delay T per packet satisfies $T \leq \frac{dp}{1-\rho}$, thus showing that an average delay of $\Theta(d)$ is attainable for any fixed $\rho < 1$. We also establish similar results in the context of the butterfly network. Our analysis is based on a stochastic comparison with a product-form network.

† Research supported by the NSF under Grant ECS-8552419 and by the ARO under Grant DAAL03-86-K-0171.

‡ Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Mass. 02139, USA; e-mail: stamouli@lids.mit.edu, jnt@lids.mit.edu.

1. INTRODUCTION

1.1 Problem Definition — Summary of the Results

During the execution of parallel algorithms in a network of processors, it is necessary that processors communicate with each other in order to exchange information. This is accomplished by routing messages through the underlying interconnection network. In the present paper, we consider a well-known problem: the nodes (i.e., processors) of the hypercube network generate packets at random time instants; each packet has a single destination, which is selected at random. We discuss a simple greedy scheme for routing these packets and we analyze its steady-state stability and delay properties. The results to be derived extend to the butterfly network.

We consider the d-dimensional binary hypercube (or d-cube); e.g., see [BeT89]. This network consists of 2^d nodes, numbered from 0 to $2^d - 1$. Associated with each node z is a binary identity $(z_d, \ldots, z_1)$, which coincides with the binary representation of the number z. For $j \in \{1, \ldots, d\}$, we denote by e_j the node numbered 2^{j-1}; that is, all entries of the binary identity of e_j equal 0 except for the jth one (from the right), which equals 1. For two nodes z and y, we denote by $z \oplus y$ the vector $(z_d \oplus y_d, \ldots, z_1 \oplus y_1)$, where $\oplus$ is the symbol for the XOR operation. Each arc of the d-cube is directed and connects two nodes whose binary identities differ in a single bit; see Fig. 1a, where the 3-cube is depicted. That is, arc (z, y) exists if and only if, for some $m \in \{1, \ldots, d\}$, $z_i = y_i$ for $i \neq m$ and $z_m \neq y_m$; this is equivalent to $y = z \oplus e_m$ for some $m \in \{1, \ldots, d\}$. Such an arc is said to be of the mth type; the set of arcs of the mth type is called the mth dimension. Note that (z, y) stands for a unidirectional arc pointing from z to y; of course, if arc (z, y) exists, so does arc (y, z). Each node z has d neighbors, namely $z \oplus e_1, \ldots, z \oplus e_d$; thus, the d-cube has $d2^d$ arcs. The Hamming distance between two nodes z and y is defined as the number of bits in which their binary identities differ; it is denoted by $H(z, y)$. Any path from z to y contains at least as many arcs as the Hamming distance between z and y. Moreover, there always exist paths that contain exactly that many arcs; these paths are shortest. It is easily seen that the diameter of the d-cube equals d.

The underlying assumptions for communications are as follows: Each piece of information is transmitted as a packet with unit transmission time. Only one packet can traverse an arc at a time; all transmissions are error-free. Each node may transmit packets through all of its output ports and at the same time receive packets through all of its input ports. Each node has infinite buffer capacity. Finally, for analytical convenience, the time axis is taken to be continuous. (The case of slotted time is briefly treated in §3.4.)

Routing a packet from a node z to another node y may be accomplished optimally by transmitting the packet along one of the shortest paths from z to y; this takes time equal to $H(z, y)$, provided that the packet does not encounter any contention en route. This is the simplest conceivable communication task. Under more general tasks, several packets are to be transmitted at the same time, with the set of origin-destination pairs having a special structure. (e.g., the permutation task, where each node transmits one packet, with different packets having different destinations; see also §1.2.) By exploiting this special structure, one may devise an algorithm to perform such a task efficiently. Such routing problems are called static, because they consider tasks to be performed only once, in the absence of other transmissions. In the dynamic version of the routing problem, it is assumed that multiple tasks are generated over an infinite time-horizon; moreover, different tasks may interfere with each other. As discussed below, most of the literature on routing deals with static problems.

In the present paper, we consider a dynamic problem with a simple structure:

Each node of the d-cube generates packets according to a Poisson process with rate λ; different nodes generate their packets independently of each other. Each packet has a single destination, which is selected randomly; in particular, we assume that

$$\Pr[\text{a packet generated by node } x \text{ is destined for node } z] = p^{H(x,z)}(1-p)^{d-H(x,z)} , \qquad (1)$$

where $p \in (0,1]$; different packets make their selections independently of each other.

Notice that the problem just defined is invariant under translation; that is, if each hypercube node is renamed from x to $x \oplus y^*$ (where y^* is a fixed d-bit string), then the statistics of the various random variables are not affected.

It is seen from (1) that for $p = \frac{1}{2}$ the destination distribution is uniform; that is, each node (including its origin) is equally likely to be chosen as a packet's destination. This is the case usually considered in the literature (see §1.2); in most of the related works, a packet's origin is not a permissible destination; however, it is easily seen that our results (when rescaled appropriately) also apply to this case. Also note that for $p < \frac{1}{2}$ the destination distribution favors nodes at shorter distance from a packet's origin; in this case, packet transmissions tend to be more localized.

As will be proved in §2.1, the inequality

$$\rho \overset{\text{def}}{=} \lambda p < 1$$

is a necessary condition for stability; ρ will be called the load factor of the system. Therefore, it is of particular interest to devise a routing scheme that is guaranteed to be stable for all $\rho < 1$. Moreover, it is desirable that such a scheme does not introduce excessive delay; in particular, it is required that for each $\rho < 1$ there exists some C_ρ (which does not depend on d) such that the average delay T per packet does not exceed $C_\rho d$. (Here, T is defined as the steady-state average time elapsing until a packet reaches its destination.) This requirement on the delay is motivated by the fact that, in the abscence of other transmissions, it would take at most dp time units (on the average) to route a packet to its (random) destination; thus, it is desirable that the additional delay due to contention does not increase this quantity by more than a multiplicative factor depending on the load of the network.

The simplest approach to our routing problem is for each packet to choose a shortest path leading to its destination and attempt to traverse this path as fast as possible. Although it is intuitively clear that such greedy schemes may possibly be efficient, their performance has not been analyzed rigorously in the literature. In this paper, we prove that a particular greedy scheme is very efficient. The routing scheme to be analyzed is as follows: Consider a packet originating at node x and destined for node z; this packet will be routed through that shortest path (from x to z) in which the hypercube dimensions are crossed in increasing index-order. (Such paths are often referred to as canonical.) For example, a packet travelling from node $(0,0,0,0)$ to node $(1,0,1,1)$ in the 4-cube would follow the path

$$(0,0,0,0) \to (0,0,0,1) \to (0,0,1,1) \to (1,0,1,1).$$

It will be proved that this simple routing scheme is stable for all $\rho < 1$, which is the broadest possible stability region. Moreover, it will be established that, for $\rho < 1$, the delay T induced by the scheme satisfies

$$dp + p\frac{\rho}{2(1-\rho)} \leq T \leq \frac{dp}{1-\rho};$$

of particular interest is the upper bound on the delay, which guarantees that, for any fixed ρ, each packet reaches its destination in an average time $\Theta(d)$. Notice also that under heavy traffic (i.e., for $\rho \to 1$) the delay T increases as $\frac{1}{1-\rho}$. It will be established that such a behavior under heavy traffic is optimal for any fixed d; indeed, it will be proved that $\lim_{\rho\to 1}[(1-\rho)T] > 0$ under any legitimate routing scheme.

The results above may be easily extended to the d-dimensional butterfly. This crossbar network is an "unfolded" version of the d-cube; see §4.1 and [BeT89]. In this context, it is assumed that packets are generated at one of the fronts of the butterfly and destined for a randomly chosen node at the opposite front; the destination distribution is identical to that presented in (1), except for the fact that x and z belong to opposite fronts of the butterfly. Notice that crossing the dimensions in increasing index-order is the only legitimate choice of paths for the butterfly. Thus, the scheme simply reduces to greedy routing; this will be seen to be stable for all $\rho < 1$, where ρ is now defined as $\rho \overset{\text{def}}{=} \lambda \max\{p, (1-p)\}$; moreover, for $\rho < 1$, the average delay T satisfies

$$d + p\frac{\lambda p}{2(1-\lambda p)} + (1-p)\frac{\lambda(1-p)}{2[1-\lambda(1-p)]} \leq T \leq \frac{dp}{1-\lambda p} + \frac{d(1-p)}{1-\lambda(1-p)}.$$

Again, the delay T is $\Theta(d)$ for any fixed $\rho < 1$, which is the optimal order of magnitude; also, the behavior of T under heavy traffic will be seen to be optimal, for any fixed d.

To the best of our knowledge, these results are new. Moreover, our analysis provides the first proof that some routing scheme (on either the d-cube or the butterfly) is stable for all $\rho < 1$ while satisfying the requirement for $\Theta(d)$ average delay; proving that greedy routing has these properties has been a long-standing open question in the routing literature. Also, this is the first routing scheme for which the bounds on the delay are expressed in simple formulae involving the system's parameters ρ and d. Finally, the approach for deriving the aforementioned results is new as well: it is established that the hypercube (resp., the butterfly) behaves as a queueing network with deterministic servers (each corresponding to an arc) and with Markovian routing among the various servers; then, by using sample path arguments, it is shown that the delay induced by this queueing network is dominated by that corresponding to a product-form network. This kind of approach relies on the assumption of Poisson arrivals; nevertheless, we hope that our analysis will be suggestive of the efficient perfomance of greedy routing under more general packet-generating processes; in fact, the conditions for stability derived in our analysis are much more general.

1.2 Survey of Previous Work

As already mentioned, there exists considerable literature on algorithms for communication tasks in various interconnection networks. However, several of the related articles analyze static problems, where each task has to be performed only once, and no other packet transmissions are taking place at the same time. In particular, for the hypercube network, Bertsekas et al. [BOSTT89] have devised optimal algorithms for a variety of communication tasks. Previously, Saad and Schultz [SaS85], as well as Johnsson and Ho [JoH89], had constructed optimal or nearly optimal algorithms for hypercubes, under somewhat different assumptions on packet transmissions. The interested reader may find more references in these three papers and in [BeT89].

The communication tasks considered in the aforementioned papers as well as the respective algorithms do not employ any randomization. In his famous paper [Val82], Valiant has demonstrated how to use randomization in order to perform a deterministic task. In particular, in the context of the d-cube, he considered the permutation task and showed that it may be accomplished in time $\Theta(d)$ with high probability, by using a randomized two-phase algorithm. In the first phase, each packet chooses a random intermediate destination (with all nodes being equiprobable) and is sent there; in the second phase, each packet travels from its intermediate destination to its actual destination. In a later paper, Valiant and Brebner [VaB81] modified this algorithm, thus simplifying considerably the analysis. In particular, they assumed that, in each of the two phases, packets cross the various hypercube dimensions in increasing index-order. These paths coincide with the ones used in the scheme analyzed in the present paper. It was established (for both permutation algorithms) that there exists some constant R such that the completion time is at most Rd with high probability. Notice, however, that these algorithms make inefficient use of the communication resources of the network; indeed, the average traffic per arc is $O(\frac{1}{d})$ packets per time unit. This performance was improved later by Chang and Simon [ChS86] as well as by Valiant [Val88]. Each of these articles presents an algorithm for routing d permutations on the d-cube in $O(d)$ time (with high probability); these algorithms result in an average traffic of $O(1)$ packets per arc and time unit. [ChS86] also contains a scheme for routing continuously batches of permutations, by pipelining.

The dynamic routing problem of this paper has been dealt with in several articles, which we discuss below; all of them consider the case of uniform destination distribution. Abraham and Padmanabhan [AbP86] have constructed an approximate model for this problem, under various assumptions on the buffer capacity of the nodes. In particular, they assume that packets advance in the respective paths independently of each other; the model involves some parameters, which are determined by solving a system of non-linear equations. Greenberg and Hajek [GrH89] have analyzed this problem under the assumption that deflection routing is used instead of shortest path routing; that is, packets may be temporarily misrouted, rather than stored or dropped. The analysis in [GrH89] is approximate too. Varvarigos [Var90] has formulated a Markov chain model for evaluating the performance of deflection routing, and has investigated its stationary statistics numerically. Note that all three [AbP86], [GrH89] and [Var90] are dealing with the hypercube network. The same problem has been analyzed in the context of the Manhattan network (square mesh) by Greenberg and Goodman [GrG86], with their analysis being again approximate. Recently, Leighton [Lei90] proved that greedy routing in the square mesh has very satisfactory average performance. Problems similar to ours were also analyzed by Mitra and Cieslak [MiC86], as well as by Hajek and Cruz [HaC87], in the context of the extended Omega network (which is a crossbar switch); it was assumed (in both papers) that, for each individual packet, transmission times over the various arcs are independent and exponentially distributed random variables. This assumption (called "Kleinrock's independence assumption") simplifies the analysis considerably; in fact, our problem would have been trivial under this assumption, because the underlying networks would have been of the Jackson type. Nevertheless, such an independence assumption appears to be unrealistic for our problem.

Finally, another dynamic routing problem, was analyzed by Stamoulis and Tsitsiklis in [StT90], where it was assumed that packets generated at random instants and at random nodes of the hypercube must be broadcast to all nodes.

2. PRELIMINARY RESULTS FOR THE HYPERCUBE

2.1 The Necessary Condition for Stability

First, we start with an observation to be used several times in the analysis. Consider a fixed packet $\mathcal{P}$ generated at node x. Let $\mathcal{B}_i$ denote the event that packet $\mathcal{P}$ will choose a destination z such that $z_i \neq x_i$; notice that if event $\mathcal{B}_i$ occurs, then $\mathcal{P}$ will have to cross an arc of the ith dimension in order to reach its destination. It is a straightforward consequence of the definition of the destination distribution [see (1)] that the following is true:

Lemma 1: For any fixed packet $\mathcal{P}$, events $\mathcal{B}_1, \ldots, \mathcal{B}_d$ are mutually independent, with $\Pr[\mathcal{B}_i] = p$ for $i = 1, \ldots, d$. Independence prevails both with and without conditioning on the origin of the packet. ■

Lemma 1 essentially implies the following: In order to choose the binary identity of its destination, packet $\mathcal{P}$ flips each of the bits of the identity of its origin x; each bit-flip is performed with probability p, independently of the others. Notice also that the average number of bit-flips performed equals dp; therefore, under any routing scheme, each packet will have to traverse at least dp hypercube arcs on the average.

Next, we derive the necessary condition for stability. The average total number of packets generated in the network per unit time equals $\lambda 2^d$. Thus, by the conclusion of the previous paragraph, it is seen that during each time unit an average total demand for at least $\lambda 2^d dp$ packet transmissions is generated in the system. Since at most $d2^d$ packet transmissions may take place per unit time, it follows that the system can be stable only if $\lambda 2^d dp \leq d2^d$. Thus, we obtain the following necessary condition for stability under any routing scheme:

$$\rho \overset{\text{def}}{=} \lambda p \leq 1 , \tag{2}$$

where ρ is the load factor of the system. This terminology is appropriate, because when $\rho \approx 1$ all hypercube arcs are almost always busy, even if no redundant packet transmissions take place. Notice that (2) is a necessary condition for stability under more general arrival processes. Furthermore, this condition can be strengthened to $\rho < 1$, unless all arrival processes are deterministic.

2.2 Lower Bounds on the Delay

First, we establish a universal lower bound on the steady-state average delay T per packet; that is, a bound that applies to any routing scheme. Recall that T is defined as the stationary average of the time elapsing between the moment a packet is generated until it reaches its destination.

Proposition 2: The average delay T per packet induced by any routing scheme satisfies

$$T \geq \max\{dp, p\mathcal{D}(2^d;\rho)\} = \Omega\left(dp + p\frac{\rho}{2^d(1-\rho)}\right) , \qquad \forall \rho < 1 ,$$

where $\mathcal{D}(2^d;\rho)$ is the average delay for the $M/D/2^d$ queue with unit service time and arrival rate $2^d\rho$. ■

Proof: Consider a fixed packet $\mathcal{P}$ generated at node x; if its random destination satisfies $z_1 \neq x_1$ (that is, if event $\mathcal{B}_1$ occurs for $\mathcal{P}$), then $\mathcal{P}$ will not reach its destination until it traverses at least one arc of the 1st dimension. Let W be the average time until a packet crosses the 1st dimension, with the convention that packets that never do so contribute zero to this average; clearly, $T \geq W$. It is straightforward to see that the value of W can only decrease if we introduce the following conditions:

(a) Each packet for which event $\mathcal{B}_1$ has not occured never crosses the 1st dimension.

(b) Each packet for which event $\mathcal{B}_1$ has occured, is available upon its generation at all nodes; moreover, such a packet will only cross the first available arc of type 1.

Under these assumptions, the 2^d arcs of the 1st dimension operate as an $M/D/2^d$ queue. The input stream of this queue consists of all packets for which event $\mathcal{B}_1$ occurs; by Lemma 1, this stream is Poisson with rate $\lambda 2^d p = 2^d \rho$. The average delay induced by this queue equals $\mathcal{D}(2^d;\rho)$; since only a fraction p of the packets "joins" this $M/D/2^d$ queue, we have

$$W \geq p\mathcal{D}(2^d;\rho)\,. \tag{3}$$

Recall now that $T \geq W$ and $T \geq dp$ (see §2.1); these facts together with (3) imply that

$$T \geq \max\{dp, p\mathcal{D}(2^d;\rho)\}\,. \tag{4}$$

Furthermore, it is known [Bru71] that

$$\mathcal{D}(2^d;\rho) \geq 1 + \frac{\rho}{2^{d+1}(1-\rho)}\,;$$

combining this with (4), it follows that

$$T = \Omega\left(\max\left\{dp, p + p\frac{\rho}{2^{d+1}(1-\rho)}\right\}\right) = \Omega\left(dp + p\frac{\rho}{2^d(1-\rho)}\right)\,,$$

where we have also used the inequality $\max\{\alpha_1,\alpha_2\} \geq \frac{1}{2}(\alpha_1+\alpha_2)$. The proof of the result is now complete. **Q.E.D.**

The universal lower bound of Proposition 2 shows that $\lim_{\rho\to 1}[(1-\rho)T] > 0$, for any fixed d, under any routing scheme. As far as asymptotics with respect to d are concerned, the bound appears to be loose, due to the presence of the factor $\frac{1}{2^d}$. Below, we establish a sharper lower bound applying to a restricted but fairly broad class of routing schemes.

As suggested by the proof of Proposition 2, a scheme that comes close to attaining the universal lower bound for the delay T (if there exists such a scheme) would schedule transmissions adaptively. This claim is further supported by Proposition 3, which establishes a lower bound on T under oblivious schemes. Under an oblivious scheme, each packet selects its path independently of the existing traffic and insists on traversing the selected path (see [BoH82]); we also assume that all rules for path selection are time-independent. We now present the lower bound on the delay induced by oblivious schemes.

Proposition 3: The average delay T per packet induced by any oblivious routing scheme satisfies

$$T = \Omega\left(dp + p\frac{\rho}{1-\rho}\right)\,. \qquad \blacksquare$$

Proof: This proof is similar to that of Proposition 2. We consider a node x and an arc $(y, y\oplus e_1)$; under any oblivious scheme, the following is true: For each packet generated at x, arc $(y, y\oplus e_1)$ is the first arc of type 1 to be crossed by such a packet with probability $q_{x,y}$, independently of all other events occuring in the network; moreover, there holds

$$\sum_{y=0}^{2^d-1} q_{x,y} \geq p, \qquad \forall x \in \{0,\ldots,2^d-1\}\,, \tag{5}$$

because it is with probability p that some packet generated by node x will necessarily cross an arc of the 1st dimension. Let W be the average time until a packet crosses the 1st dimension for the first time, with the convention that packets that never do so contribute zero to this average; clearly, $T \geq W$. For any oblivious routing scheme, the value of W can only decrease if we introduce the following conditions:

(a) Each packet to cross the 1st dimension is only delayed at the first time it does so.

(b) Each packet to cross arc $(y, y \oplus e_1)$ is available at node y upon its generation.

Under these conditions, each arc $(y, y \oplus e_1)$ is fed by a group of 2^d Poisson streams. We denote by r_y the total arrive rate of the compound Poisson stream; obviously, we have

$$r_y = \lambda \sum_{x=0}^{2^d-1} q_{x,y} , \qquad \forall y \in \{0, \ldots, 2^d - 1\} . \tag{6}$$

Clearly, arc $(y, y \oplus e_1)$ behaves as an $M/D/1$ queue with unit service time. Therefore (see [Kle75]), the average delay W_y per packet joining this queue is given as follows:

$$W_y = 1 + \frac{r_y}{2(1 - r_y)} .$$

Using this, we obtain

$$W \geq \frac{1}{\lambda 2^d} \sum_{y=0}^{2^d-1} r_y W_y = \frac{1}{\lambda 2^d} \sum_{y=0}^{2^d-1} r_y \left[1 + \frac{r_y}{2(1 - r_y)} \right] . \tag{7}$$

Combining (5) and (6), we have

$$\sum_{y=0}^{2^d-1} r_y \geq \lambda 2^d p . \tag{8}$$

Notice now that $r[1 + \frac{r}{2(1-r)}]$ is a convex and increasing function of r; therefore, in light of (8), the right-hand quantity in (7) is minimized when $r_y = \lambda p$ for all $y \in \{0, \ldots, 2^d - 1\}$. Thus, it follows that

$$W \geq \frac{1}{\lambda 2^d} \sum_{y=0}^{2^d-1} \lambda p \left[1 + \frac{\lambda p}{2(1 - \lambda p)} \right] = p \left[1 + \frac{\rho}{2(1 - \rho)} \right] .$$

This together with the facts $T \geq W$ and $T \geq dp$ proves that

$$T \geq \max \left\{ dp, p \left[1 + \frac{\rho}{2(1 - \rho)} \right] \right\} ,$$

and the result follows. **Q.E.D.**

Proposition 3 implies that, for a fairly broad class of schemes, the universal lower bound on the delay T (see Proposition 2) is rather loose. Suppose now that, under oblivious routing, we allow packets generated at node x to take into account the routing decisions taken by packets previously generated at x. It is an interesting open question to investigate whether Proposition 3 still holds. We believe it does, because each packet has a very limited knowledge of the routing decisions taken within the entire network. If this is indeed the case, then a scheme violating the lower bound given by Proposition 3 should involve centralized coordination and/or adaptive routing.

It is worth noting that Propositions 2 and 3 hold for any destination distribution that is invariant under translation; that is, when the probability that a packet originating at node x is destined for node z equals $f(x \oplus z)$, which depends only on $x \oplus z$. In this case, ρ is defined as

$$\rho \overset{\text{def}}{=} \max\{\rho_1, \ldots, \rho_d\} ;$$

ρ_j is the load factor for the jth dimension and equals

$$\rho_j \overset{\text{def}}{=} \lambda \sum_{\{y: y_j = 1\}} f(y) .$$

In this case, (2) is still a necessary condition for stability under any routing scheme.

2.3 Simple Non-Greedy Schemes

In this subsection, we discuss simple routing schemes based on pipelining successive instances of an algorithm used in static routing. For simplicity, we assume that the destination distribution is uniform (i.e., $p = \frac{1}{2}$).

As already mentioned in §1.2, in the first phase of the permutation algorithm of [VaB81], each packet selects a destination at random (with all nodes being equiprobable) and travels there. This algorithm has the following property: there exists a constant $R > 1$ such that the first phase takes time less than Rd (and close to this value) with high probability. Consider now the following routing scheme for our problem:

> At time $t = 0$, each node selects one of its packets; all selected packets are routed as in the first phase of [VaB81]. These packets arrive at their respective destinations at time t_1, where $t_1 \leq Rd$ with high probability. At time t_1, each node selects another one of its packets, and the selected packets are again routed as in [VaB81] etc.

Under this scheme, each packet generated by some node x joins an $M/G/1$ queue. The arrival rate for this queue is λ; the service time is distributed as the duration of the first phase of the algorithm in [VaB81], which is close to Rd with high probability. (For simplicity, we ignore the overhead required for termination detection.) Hence, node x routes one of its packets every Rd time units (approximately); thus, stability may prevail only if $\lambda Rd \leq 1$, or equivalently $\rho \leq \frac{1}{2Rd}$. Therefore, for any fixed ρ, the simple scheme described becomes unstable for large d.

A potential remedy to this undesirable performance is to pipeline successive instances of an efficient static algorithm for d permutations, such as those in [ChS86] and [Val88]. Such an approach would lead to a routing scheme which would be stable for $\rho < \rho^*$ with ρ^* being some small constant; e.g., using the algorithm of [ChS86] would lead to $\rho^* \approx 0.005$, which is very small compared to the upper bound given by (2).

All of the schemes described above are non-greedy, i.e. they involve idling: it often occurs that packets wait at their respective origins, while some of the arcs to be traversed are idle. As will be seen in §3, avoidance of this idling phenomenon improves performance dramatically.

3. THE MAIN RESULTS FOR THE HYPERCUBE

In this section, we analyze an efficient greedy routing scheme for the hypercube network. As already mentioned in §1.1, the scheme is as follows: Each packet proceeds towards its destination by crossing the dimensions required in increasing index-order. To clarify matters, consider a packet $\mathcal{P}$ generated at node x and destined for node z; let $i_1, \ldots, i_k$ be the entries in which the binary identities of x and z differ, with $i_1 < i_2 < \cdots < i_k$; then, packet $\mathcal{P}$ follows the path

$$x \to x \oplus e_{i_1} \to x \oplus e_{i_1} \oplus e_{i_2} \to \cdots \to x \oplus e_{i_1} \oplus \cdots \oplus e_{i_k} = z\,.$$

(This path is unique for each origin-destination pair.) It should be noted that packets advance at their respective paths as fast as possible; that is, no idling occurs (hence the characterization "greedy"). Also, whenever several packets present at a node y wish to traverse the same arc, then priority is given to the one that arrived at y the first.

The routing scheme presented above is the non-idling version of one of the schemes described in §2.3 (namely, of that based on the algorithm of [ValB81]); moreover, the scheme is oblivious (see §2.2). It will be seen in §3.1 that, under this scheme, the hypercube is equivalent to a queueing network with certain useful properties. The analysis in §§3.2 and 3.3 deals with the performance of this equivalent queueing network.

3.1 The Equivalent Queueing Network

It is straightforward that, under our routing scheme, the d-cube may be viewed as a queueing network, with $d2^d$ deterministic FIFO "servers"; each "server" has unit service duration and corresponds to a hypercube arc. This equivalent queueing network (to be referred to as $\mathcal{Q}$) has the following properties:

Property A: The external arrival stream at any arc $(x, x \oplus e_i)$ is Poisson with rate $\lambda p(1-p)^{i-1}$; streams corresponding to different arcs are mutually independent.

To see this, consider a packet $\mathcal{P}$ generated at node x of the d-cube; with probability $p(1-p)^{i-1}$ the destination of $\mathcal{P}$ satisfies $z_1 = x_1, \ldots, z_{i-1} = x_{i-1}$ and $z_i \neq x_i$ (see Lemma 1). Since packets cross the hypercube dimensions in increasing index-order, it follows that each of the packets generated by node x will join the queue for arc $(x, x \oplus e_i)$ with probability $p(1-p)^{i-1}$.

Property B: After crossing arc $(y, y \oplus e_i)$, a packet will never traverse again an arc $(z, z \oplus e_j)$ with $j \in \{1, \ldots, i\}$. Thus, the equivalent network $\mathcal{Q}$ is a levelled network; that is, its "servers" are organized in d levels, with the ith level comprising all arcs $(y, y \oplus e_i)$ for $y \in \{0, \ldots, 2^d - 1\}$, i.e. all arcs of the ith dimension. Upon "service completion" at a certain level, a packet either joins a queue at a higher level or it departs from the network.

In Fig. 1a, we presented the 3-cube, while, in Fig. 1b, we present the equivalent network $\mathcal{Q}$.

Property C: Routing is Markovian. In particular, upon crossing arc $(y, y \oplus e_i)$, a packet takes one of the following actions: either it joins the queue at arc $(y \oplus e_i, y \oplus e_i \oplus e_j)$ with probability $p(1-p)^{j-i-1}$ for $j = i+1, \ldots, d$; or it departs from the network with probability $(1-p)^{d-i}$. After crossing arc $(y, y \oplus e_d)$, a packet departs from the network with probability 1. Different packets take their routing decisions independently of each other.

The validity of this property requires some clarification; in particular, in light of Property B, we need to show the following result:

Lemma 4: Consider a fixed packet $\mathcal{P}$, which has just crossed arc $(y, y \oplus e_i)$; there holds

$$\Pr[\mathcal{P} \text{ will cross } (y \oplus e_i, y \oplus e_i \oplus e_j) \mid \mathcal{P} \text{ has crossed } (y, y \oplus e_i)] = p(1-p)^{j-i-1}, \text{ for } i < j \leq d. \quad (9)$$

■

Proof: Let x denote the origin of packet $\mathcal{P}$. Clearly, in order to prove (9) it suffices to prove the following:

$$\Pr[\mathcal{P} \text{ will cross } (y \oplus e_i, y \oplus e_i \oplus e_j) \mid \mathcal{P} \text{ has crossed } (y, y \oplus e_i) \text{ and } \mathcal{P} \text{ originated at } x] = p(1-p)^{j-i-1},$$

for any permissible origin x of $\mathcal{P}$. Notice, however, that $\mathcal{P}$ will skip the dimensions $i+1, \ldots, j-1$ and cross the jth dimension next if and only if events $\mathcal{B}_{i+1}, \ldots, \mathcal{B}_{j-1}$ did not occur while $\mathcal{B}_j$ occured. Hence, proving (9) is equivalent to proving the following:

$$\Pr[\overline{\mathcal{B}}_{i+1}, \ldots, \overline{\mathcal{B}}_{j-1}, \mathcal{B}_j \mid \mathcal{P} \text{ has crossed } (y, y \oplus e_i) \text{ and } \mathcal{P} \text{ originated at } x] = p(1-p)^{j-i-1}, \quad (10)$$

where $\overline{\mathcal{B}}_\ell$ is the complement of event $\mathcal{B}_\ell$. Since hypercube dimensions are crossed in increasing index-order, knowledge of the origin of $\mathcal{P}$ and of the fact that $\mathcal{P}$ has just crossed arc $(y, y \oplus e_i)$ provides information only on the first i bits of the destination of $\mathcal{P}$; thus, (10) follows from the independence result of Lemma 1. **Q.E.D.**

According to the proof of Lemma 4, propagation of a packet $\mathcal{P}$ on the hypercube may also be visualized as follows: Upon generation, $\mathcal{P}$ decides whether or not to cross dimension 1; the probability that it decides positively equals p. If it does so, then it takes its step on this dimension and then it decides whether or not to cross dimension 2; if it does not decide to cross dimension 1, then it considers crossing dimension 2 etc.

3.2 The Sufficient Condition for Stability of the Routing Scheme

In the previous subsection, we established that, under the routing scheme analyzed, the hypercube is equivalent to a queueing network $\mathcal{Q}$ with Markovian routing. In this subsection, we derive a sufficient condition for stability of the routing scheme. First, we prove the following result:

Proposition 5: The total arrival rate at any arc of the d-cube equals $\lambda p = \rho$. ■

Proof: By symmetry among the hypercube nodes, all arcs belonging to the same dimension j have the same total arrival rate θ_j. Furthermore, the total arrival rate for the jth dimension equals $2^d \lambda p$, because each of the packets generated within the d-cube crosses the jth dimension for an expected number of p times. Hence, we have $2^d \theta_j = 2^d \lambda p$, which gives $\theta_j = \lambda p = \rho$ for all $j \in \{1, \ldots, d\}$.

Q.E.D.

Notice now that the equivalent network $\mathcal{Q}$ has the following properties:

(a) Each "server" is fed externally by a Poisson process. Arrival processes corresponding to different "servers" are independent.

(b) Service times are bounded. Also service times corresponding to different packets and/or different "servers" are (trivially) independent.

(c) Routing is Markovian.

These properties allow us to apply Theorem 2A of [Bor87]; it thus follows that network $\mathcal{Q}$ is <u>stable</u> if the total arrival rate for each "server" is less than unity. By stability it is meant that the stationary distribution of the network's state is well-defined and is independent of the initial state. (*) Recalling the equivalence of $\mathcal{Q}$ with the hypercube (under our greedy routing scheme) and using Proposition 5, we reach the following conclusion:

Proposition 6: The greedy routing scheme under analysis is stable for all $\rho < 1$. ■

In light of the necessary condition for stability $\rho < 1$ (see §2.1), it is seen that the routing scheme under analysis has <u>optimal</u> stability properties. In fact, Theorem 2A of [Bor87] applies to more general arrival processes; therefore, Proposition 6 is rather general.

3.3 The Bounds for the Delay Induced by the Scheme

In this subsection, we establish both upper and lower bounds for the average delay T induced by the routing scheme under analysis. Starting with the upper bound (which is the most interesting result), we will show that $T \leq \frac{dp}{1-\rho}$ for all $\rho < 1$. The basic idea for proving this result is as follows:

If the service discipline at the "servers" of the equivalent network $\mathcal{Q}$ is <u>changed</u> from FIFO to <u>Processor Sharing</u> (PS), then the average delay per packet <u>increases</u>; under the PS discipline, $\mathcal{Q}$ becomes a <u>product-form</u> network, and its delay is easily computed.

(*) In this paper, we use the term "stable" as in [Wal88], rather than the term "ergodic" used in [Bor87].

Recall that under the PS discipline all customers present at a server receive an equal proportion of service simultaneously; see [Wal88], p. 354. For example, consider a deterministic PS server, with unit service rate; assume that it has two customers to serve, with the first customer arriving at time 0 and the second at time $\frac{1}{4}$; upon arrival of the second customer, the first one has $\frac{3}{4}$ units of service remaining; however, due to the presence of the second customer, she will be served at rate $\frac{1}{2}$; thus, she will depart at time $\frac{1}{4} + 2\frac{3}{4} = \frac{7}{4}$; similarly, it can be seen that the second customer will depart at time 2. Notice that we are using the term "service rate" for a PS server (rather than the term "service duration"), because the time duration for which a customer receives service depends on previous and future arrivals.

The proof of the upper bound on the delay T makes use of several lemmas that establish sample path results; these we present next.

Lemma 7: Let there be a deterministic FIFO server with unit service duration. For a fixed sequence $t_1, t_2, \ldots$ of arrival times, let $D_1, D_2, \ldots$ denote the corresponding sequence of departure times. Similarly, let $\tilde{D}_1, \tilde{D}_2, \ldots$ be the departure times for a deterministic PS server, with unit service rate, fed by the same input stream. There holds

$$D_i \leq \tilde{D}_i \, , \quad \text{for } i = 1, \ldots \qquad \blacksquare$$

Proof: Clearly, we have $D_1 = t_1 + 1$. In the context of the PS server, the 1st customer will depart at time $t_1 + 1$ only if no other customers arrive until that time; otherwise, the server will be slowed down, and the 1st customer will depart later than $t_1 + 1$. It follows that

$$\tilde{D}_1 \geq t_1 + 1 = D_1 \, . \tag{11}$$

It is well-known that the PS discipline is work-conserving; see [Wal88], pp. 353-354. That is, the unfinished work $W(t)$ at time t is the same for both the FIFO and the PS servers considered. By definition of $W(t)$, we have

$$D_i = t_i + W(t_i-) + 1 \, , \quad \text{for } i = 1, \ldots \tag{12}$$

We now consider the ith arrival at the PS server, where $i \geq 2$. If $W(t_i-) = 0$, then reasoning similarly as in proving (11), it follows that $\tilde{D}_i \geq t_i + 1 = D_i$. Assume now that $W(t_i-) \neq 0$; it is straightforward that customers depart from a deterministic PS server in the order they arrive; hence, the ith customer may depart only after an amount $W(t_i-) + 1$ of work has been finished by the server. Therefore, we have

$$\tilde{D}_i \geq t_i + W(t_i-) + 1 = D_i \, ,$$

where we have also used (12). The proof of the lemma is now complete. **Q.E.D.**

Let there be two streams of events, one occuring at times $\tau_1, \tau_2, \ldots$ and the other at times $\tau'_1, \tau'_2, \ldots$ If $\tau_i \leq \tau'_i$ for $i = 1, \ldots$, then the latter stream of events will be said to be a delayed version of the former. For example, as implied by Lemma 7, for any fixed arrival stream, the departing stream of a deterministic PS server is a delayed version of the one of the corresponding FIFO server.

Lemma 8: Let there be a deterministic FIFO server with unit service duration. Let $D_1, D_2, \ldots$ (resp. $D'_1, D'_2, \ldots$) be the sequence of departure times corresponding to a fixed sequence $t_1, t_2, \ldots$ (resp. $t'_1, t'_2, \ldots$) of arrival times. If $t_i \leq t'_i$ for $i = 1, \ldots$, then

$$D_i \leq D'_i\,, \quad \text{for } i = 1, \ldots \qquad \blacksquare$$

Proof: There holds

$$D_1 = t_1 + 1 \qquad \text{and} \qquad D_i = \max\{D_{i-1}, t_i\} + 1 \ \text{ for } \ i = 2, \ldots\,;$$

similarly,

$$D'_1 = t'_1 + 1 \qquad \text{and} \qquad D'_i = \max\{D'_{i-1}, t'_i\} + 1 \ \text{ for } \ i = 2, \ldots$$

Using these facts and the assumption $t_i \leq t'_i$ for $i = 1, \ldots$, the result follows by a straightforward inductive argument. **Q.E.D.**

The result to be established next is based on Lemmas 7 and 8; generalizing this result will lead to the upper bound on the delay induced by our greedy routing scheme. We consider the queueing network $\mathcal{G}$ depicted in Fig. 2a. This consists of three deterministic FIFO servers with unit service duration, denoted by S_1, S_2 and S_3. Customers completing service at S_1 or S_2 either depart from the network or they join the queue at S_3; routing decisions are Markovian. Obviously, $\mathcal{G}$ is a levelled network (see §3.1). We define a sample path ω of $\mathcal{G}$ as the following collection of information:

(a) The external arrival times at servers S_1, S_2 and S_3.

(b) The routing decision taken by the ith customer upon service completion at S_1 (resp. S_2) for $i = 1, \ldots$

Clearly, given a sample path ω, network $\mathcal{G}$ evolves in a deterministic fashion. The result to be proved is as follows:

Lemma 9: Let $\tilde{\mathcal{G}}$ be a network identical to $\mathcal{G}$ except for the fact that PS service discipline applies for the servers of $\tilde{\mathcal{G}}$ (instead of FIFO); see Fig. 2b. For a particular sample path ω, let $B(t)$ [resp. $\tilde{B}(t)$] denote the number of customers departing from $\mathcal{G}$ (resp. $\tilde{\mathcal{G}}$) during the interval $[0, t]$; there holds

$$B(t) \geq \tilde{B}(t)\,, \qquad \forall t \geq 0\,. \qquad \blacksquare$$

Proof: First, we consider a network $\mathcal{G}'$ obtained from $\mathcal{G}$ by changing the service discipline only at S_1 and S_2 (from FIFO to PS); see Fig. 2c.

We define as the output stream of a server the stream of customers completing service therein, including those that do not depart from the network. Notice that server S_1 is not affected at all by the presence of the other two servers; the same statement applies for server S_2. Therefore, applying Lemma 7, it is seen that the output stream of server S_1 in $\mathcal{G}'$ is a delayed version of that corresponding to S_1 of $\mathcal{G}$. Recalling also that the routing decisions of customers completing service are the same for networks $\mathcal{G}$ and $\mathcal{G}'$, it follows that the substream of customers departing from $\mathcal{G}'$ at S_1 is a delayed version of the corresponding substream in $\mathcal{G}$. Similar statements apply for the streams stemming from S_2.

Next, we consider the stream feeding S_3 in $\mathcal{G}'$; this stream is a delayed version of that feeding S_3 in $\mathcal{G}$, because each arrival at S_3 of $\mathcal{G}'$ corresponds to an arrival at S_3 of $\mathcal{G}$ that occurs <u>no later</u>. [Recall the aforementioned "comparison" of the output streams of S_1 (resp. S_2) in the two networks and the coupling of the routing decisions.] Therefore, applying Lemma 8, the output stream from S_3 of $\mathcal{G}'$ is a delayed version of that corresponding to S_3 of $\mathcal{G}$. The former output stream is delayed <u>further</u> when the service discipline at S_3 of $\mathcal{G}'$ is changed from FIFO to PS. This modification (which yields network $\tilde{\mathcal{G}}$) does not affect the streams of customers departing from the 1st level. Therefore, for each of the servers of $\tilde{\mathcal{G}}$, its departing stream is a delayed version of that of the corresponding server of $\mathcal{G}$; this proves the result in question.

Its should be noted that customers joining S_3 may get out of order when changing the service discipline; thus, a <u>particular</u> customer may depart earlier from $\tilde{\mathcal{G}}$ than from $\mathcal{G}$. Nevertheless, this does not affect the validity of the lemma. **Q.E.D.**

Next, we generalize Lemma 9. In the context of the network $\mathcal{Q}$, a sample path ω is defined as the collection of information comprising all external arrival times and all routing decisions. Notice that routing decisions at each "server" are identified by the <u>order</u> they are taken, not by the identity of the packets deciding; e.g. "the 1st packet to cross arc $(e_1 \oplus e_2, e_1)$ will advance to $(e_1, e_1 \oplus e_3)$, the 2nd such packet will depart" etc. Such an identification of the routing decisions is legitimate due to the fact that routing in $\mathcal{Q}$ is Markovian. Similarly as in Lemma 9, we denote as $\tilde{\mathcal{Q}}$ the network obtained from $\mathcal{Q}$ after changing the service discipline of all "servers" from FIFO to PS.

Lemma 10: For a particular sample path ω, let $B(t)$ [resp. $\tilde{B}(t)$] denote the number of packets that have departed from $\mathcal{Q}$ (resp. $\tilde{\mathcal{Q}}$) during the interval $[0, t]$; there holds

$$B(t) \geq \tilde{B}(t), \qquad \forall t \geq 0. \qquad \blacksquare$$

Outline of the Proof: This proof is done by extending the argument used in proving Lemma 9. In particular, one has to replace the FIFO "servers" by PS ones, on a level-by-level basis, starting from the 1st level and moving one level at a time. At the jth step of this process, all streams stemming from levels $1, \ldots, j-1$ remain the same, while all streams stemming from levels $j, \ldots, d$ are delayed. The only subtle point of this proof lies on the fact that packets may get out of order at certain steps; see also the proof of Lemma 9. Nevertheless, this creates no difficulty, due to the assumed coupling of routing decisions. If one insists on tracing the path followed by a particular packet [say the first to arrive at "server" $(0, e_1)$] it may occur that this <u>changes</u> at some step of the process described above; this is of no importance, because the "comparison" of the various streams still applies, even though the streams may consist of different packets at each step. **Q.E.D.**

Now that we have established Lemma 10, we can easily prove the following result:

Proposition 11: Let $N(t)$ [resp. $\tilde{N}(t)$] denote the (random) total number of packets present in network $\mathcal{Q}$ (resp. $\tilde{\mathcal{Q}}$). There holds

$$N(t) \leq_{\mathrm{st}} \tilde{N}(t), \qquad \forall t \geq 0 . \qquad \blacksquare$$

Proof: On a sample path basis, there holds $N(t) = B(t) - A(t)$, where $A(t)$ [resp. $B(t)$] is the number of arrivals at (resp. departures from) network $\mathcal{Q}$ during $[0, t]$; a similar relation holds for network $\tilde{\mathcal{Q}}$. Using Lemma 10, we have $N(t) \leq \tilde{N}(t)$ on a sample path basis. Relaxing the coupling of the arrival processes and the routing decisions in the two networks, we obtain the stochastic inequality in question. **Q.E.D.**

Notice that Proposition 11 (and Lemma 10) applies for <u>all</u> levelled networks with Markovian routing and deterministic FIFO servers (possibly with different service times); in particular, if the FIFO discipline is changed to PS, then the total number of customers in such a network increaces in the stochastic sense.

Next, we present the main result of this subsection.

Proposition 12: The delay T of the greedy routing scheme under analysis satisfies

$$T \leq \frac{dp}{1-\rho}, \qquad \forall \rho < 1 . \qquad \blacksquare$$

Proof: As established in [Wal88], pp. 93-94, network $\tilde{\mathcal{Q}}$ is of the <u>product form</u>, provided that it is stable. Since the total arrival rate for each "server" equals ρ (as was the case under the FIFO discipline), the steady-state probability that a particular "server" of $\tilde{\mathcal{Q}}$ hosts n packets equals $(1-\rho)\rho^n$. Therefore, the steady-state average total number $\tilde{N}$ of packets present in $\tilde{\mathcal{Q}}$ equals $\tilde{N} = d2^d \frac{\rho}{1-\rho}$. This together with Proposition 11 implies that the average total number N of packets present in network $\mathcal{Q}$ (in steady-state) satisfies

$$N \leq d2^d \frac{\rho}{1-\rho} . \tag{13}$$

Recall now the equivalence of network $\mathcal{Q}$ with the d-cube under the greedy routing scheme analyzed. By Little's law, the average delay T induced by this scheme satisfies

$$T = \frac{N}{\lambda 2^d} = \frac{Np}{\rho 2^d} . \tag{14}$$

This together with (13) proves the result. **Q.E.D.**

Next, we comment on the number of packets stored per hypercube <u>node</u>. The steady-state average number of packets per node equals $\frac{N}{2^d}$; this satisfies $\frac{N}{2^d} \leq d\frac{\rho}{1-\rho}$. Thus, it is seen that, for any fixed ρ, the average size of the queue built at each node is $O(d)$. In fact, one can show that the total number of packets within the d-cube is $O(d2^d)$ with high probability. Indeed, by Proposition 11 and the product-form property of $\tilde{Q}$, the limiting random variable $\lim_{t\to\infty} N(t)$ is stochastically dominated by the sum of $d2^d$ independent geometrically distributed random variables with expected value $\frac{\rho}{1-\rho}$. Using the Chernoff bound, it follows that, for $t \to \infty$, $N(t) \leq d2^d \frac{\rho}{1-\rho}(1+\epsilon)$ with high probability, for any $\epsilon > 0$.

As a final result, we present the lower bound on the delay T.

Proposition 13: The delay T of the greedy routing scheme under analysis satisfies

$$T \geq dp + p\frac{\rho}{2(1-\rho)}\,, \qquad \forall \rho < 1\,. \qquad \blacksquare$$

Proof: Let N_j denote the stationary average number of packets in the queue for an arc of the jth dimension. Since each dimension comprises 2^d arcs, there holds

$$N = \sum_{j=1}^{d} 2^d N_j\,. \tag{15}$$

Each arc of the 1st dimension is only fed by a Poisson stream with rate $\rho < 1$; using the expression for the average size of an $M/D/1$ queue (see [Kle75]), it follows that

$$N_1 = \rho + \frac{\rho^2}{2(1-\rho)}\,. \tag{16}$$

Recall now that, for $j \geq 2$, arc $(x, x \oplus e_j)$ has a total arrival rate of ρ; since each packet stays at an arc for at least one time unit, we have $N_j \geq \rho$ for $j = 2, \ldots, d$. Combining this with (15) and (16), we obtain

$$N \geq d2^d\rho + 2^d\frac{\rho^2}{2(1-\rho)}\,;$$

this together with (14) proves the result. **Q.E.D.**

It should be noted that another lower bound for T follows immediately from Proposition 3, which is applicable because our routing scheme is oblivious; the lower bound of Proposition 13 is sharper by a factor of at most 2.

Next, notice that, by Propositions 12 and 13, we have (for fixed p)

$$\frac{p}{2} \leq \lim_{\rho\to 1}[(1-\rho)T] \leq dp\,.$$

It is an interesting open question to close the gap in the above inequality. It is conjectured that for all $p \in (0,1)$, the upper bound is tight (within a factor independent of d). This conjecture is based on the fact that, for $p \in (0,1)$, each packet $\mathcal{P}$ faces <u>additional</u> contention for each dimension it crosses; that is, $\mathcal{P}$ contends with packets that had not entered the path of $\mathcal{P}$ up to this point. On the other hand, it is easily seen that the lower bound is tight for $p = 1$. Indeed, in this case, each packet generated at node x is destined for node $\bar{x}$, where each entry of the binary identity of $\bar{x}$ is the complement of the corresponding entry of x; thus, by crossing the hypercube dimensions in increasing index-order, packets generated at different nodes follow disjoint paths; this easily gives that $T = d + \frac{\rho}{2(1-\rho)}$.

3.4 The Case of Slotted Time

In the analysis so far, it was assumed that the time axis is continuous. In the present subsection, we briefly discuss the case of slotted time. In particular, we assume that the time axis is divided in slots of duration τ; all nodes are synchronized to the same clock. Since packets are taken to have unit transmission times, we may assume, without loss of generality, that $\tau \leq 1$ and, in particular, that $\frac{1}{\tau}$ is integer; otherwise, there will be some waste due to the fact that packets do not "fit" exactly to time slots. Furthermore, it is assumed that each node of the hypercube generates a new batch of packets at the beginning of each slot, namely at each time $k\tau$ with $k \in \{0, 1, \ldots\}$. The batch size has Poisson distribution with expected value $\lambda\tau$; thus, the input traffic intensity is the same as in the case of continuous time. Notice that batches generated at different times and/or different nodes have independent sizes. Again, each packet has a single destination, which is chosen according to the rule used in the case of continuous time [see (1)]; packets are to be routed to their respective destinations under the greedy scheme introduced in the beginning of this section.

In order to analyze the routing problem under the new assumptions, we shall consider the slotted-time version of the equivalent network $\mathcal{Q}$. The new network will be denoted by $\overline{\mathcal{Q}}$; it has the same properties as $\mathcal{Q}$, except for the fact that new arrivals occur in the way presented above and that all events occur synchronously. We consider a sample path ω of network $\mathcal{Q}$ (see §3.3); it is apparent that a sample path $\overline{\omega}$ of $\overline{\mathcal{Q}}$ can be obtained from ω as follows: For each arc $(x, x \oplus e_i)$, we take the set of external arrivals occuring under ω during the interval $[k\tau, (k+1)\tau)$ and we consider them as the batch of external arrivals feeding $(x, x \oplus e_i)$ at time $k\tau$ under the sample path $\overline{\omega}$. Thus, all external arrival streams are "advanced" in time, that is each new customer arrives under $\overline{\omega}$ no later than under ω. Henceforth, we assume that the two networks $\mathcal{Q}$ and $\overline{\mathcal{Q}}$ are coupled in the way presented above. Let $\overline{N}(k\tau)$ be the total number of customers present in $\overline{\mathcal{Q}}$ at time $k\tau$. By considering the streams stemming from each of levels of the two networks (starting from the 1st level), it may be proved that

$$\overline{N}(k\tau) \leq N(k\tau) + X_k ,$$

where X_k equals the total number of external arrivals occuring in continuous time during the interval $[k\tau, (k+1)\tau)$. Using this result, one can upper bound the various performance measures corresponding to the case of slotted time; e.g., the new value $\overline{T}$ of the average delay per packet can be seen to satisfy

$$\overline{T} \leq \frac{dp}{1-\rho} + \tau , \qquad \forall \rho < 1.$$

4. GREEDY ROUTING ON THE BUTTERFLY NETWORK

In this section, we extend the results derived for the hypercube to the butterfly network. First, we briefly describe the basic properties of this network.

4.1 The Butterfly Network

The d-dimensional butterfly is an "unfolded" version of the d-cube. It consists of $(d+1)2^d$ nodes, organized in $d+1$ levels, with each level having 2^d nodes. In particular, for $j \in \{1, \ldots, d+1\}$, the nodes of the jth level are denoted by $[x; j]$ where $x \in \{0, \ldots, 2^d - 1\}$. For $j \neq d+1$, each node $[x; j]$ is connected to two nodes, namely $[x; j+1]$ and $[x \oplus e_j; j+1]$; see Fig. 3a, where the 2-butterfly is depicted. Therefore, there exist two types of arcs:

(a) Arcs of the form $[x; j] \rightarrow [x; j+1]$, which are referred to as straight arcs; for notational convenience, arc $[x; j] \rightarrow [x; j+1]$ will be denoted by $(x; j; \mathrm{s})$.

(b) Arcs of the form $[x; j] \rightarrow [x \oplus e_j; j+1]$, which are referred to as vertical arcs; for notational convenience, arc $[x; j] \rightarrow [x \oplus e_j; j+1]$ will be denoted by $(x; j; \mathrm{v})$.

The butterfly network is a crossbar switch; packets are assumed to be generated at the 1st level and destined for the $(d+1)$st level. It is easily seen that for each origin-destination pair $[x; 1]$ and $[z; d+1]$ there corresponds a unique path, which consists of d arcs. In particular, let $i_1, \ldots, i_k$ be the entries in which the binary identities of x and z differ, with $i_1 < i_2 < \cdots < i_k$. Then, the path from $[x; 1]$ to $[z; d+1]$ contains exactly k vertical arcs, namely

$$(x; i_1; \mathrm{v}), \ (x \oplus e_1; i_2; \mathrm{v}), \ \ldots, \ (x \oplus e_1 \cdots \oplus e_{i_{k-1}}; i_k; \mathrm{v});$$

the remaining $d-k$ arcs of the path are straight arcs. Notice that these k vertical arcs correspond to the arcs traversed by a packet travelling from x to z in the d-cube, when dimensions are crossed in increasing index-order.

4.2 Preliminary Results

The dynamic routing problem to be analyzed is essentially the same as that in the context of the d-cube. That is, each node of the 1st level independently generates packets according to a Poisson process with rate λ; all packets have unit transmission time. Each packet has a single destination in the $(d+1)$st level; this destination is selected randomly, according to the following rule:

$$\Pr\left[\text{a packet generated by node } [x; 1] \text{ is destined for node } [z; d+1]\right] = p^{H(x,z)}(1-p)^{d-H(x,z)},$$

where $p \in [0, 1]$; recall that $H(x, z)$ denotes the Hamming distance of the binary representations of x and z. Again, different packets take their selections independently of each other. Notice that, for $p = \frac{1}{2}$, the destination distribution is uniform over the nodes of the $(d+1)$st level; that is, each such node is equally likely to be chosen as a packet's destination.

First, we note that a result analogous to Lemma 1 applies; however, in the present context, $\mathcal{B}_j$ corresponds to the event that a packet has to traverse a vertical arc stemming from the jth level. Furthermore, notice that arcs $(x; 1; \mathrm{s})$ and $(x; 1; \mathrm{v})$ may only be traversed by packets generated by node x. Therefore, packets to traverse arc $(x; 1; \mathrm{v})$ form a Poisson stream with rate λp; similarly, packets to traverse arc $(x; 1; \mathrm{s})$ form a Poisson stream with rate $\lambda(1-p)$. Recalling that all packets have unit transmission time, it follows that the inequalities $\lambda p < 1$ and $\lambda(1-p) < 1$ are both <u>necessary</u> conditions for stability of any routing scheme. Combining these conditions, we obtain the following result: Stability may prevail only if

$$\rho \overset{\text{def}}{=} \lambda \max\{p, 1-p\} < 1\,. \tag{17}$$

Notice that, for given λ, the maximum value of ρ occurs for $p = \frac{1}{2}$. For $p > \frac{1}{2}$, the vertical arcs become the bottleneck of the system; for $p < \frac{1}{2}$, the straight arcs become the bottleneck of the system; see also Proposition 15.

Next, we present a <u>universal</u> lower bound on the average delay T per packet.

Proposition 14: Under <u>any</u> routing scheme, there holds

$$T \geq d + p\frac{\lambda p}{2(1-\lambda p)} + (1-p)\frac{\lambda(1-p)}{2[1-\lambda(1-p)]}\,. \qquad \blacksquare$$

Proof: When no idling occurs, the value W_{v} (resp. W_{s}) of the average delay induced by arc $(x; 1; \mathrm{v})$ [resp. $(x; 1; \mathrm{s})$] equals that of an $M/D/1$ queue with arrival rate λp [resp. $\lambda(1-p)$] and unit service duration; when idling occurs, these delay values are larger. Thus, we have [Kle75]

$$W_{\mathrm{v}} \geq 1 + \frac{\lambda p}{2(1-\lambda p)} \qquad \text{and} \qquad W_{\mathrm{s}} \geq 1 + \frac{\lambda(1-p)}{2[1-\lambda(1-p)]}\,. \tag{18}$$

Note that after a packet arrives at the 2nd level, it requires at least $d-1$ more time units until it reaches its destination; thus, it is seen that, under any routing scheme, the average delay T per packet satisfies

$$T \geq d - 1 + pW_{\mathrm{v}} + (1-p)W_{\mathrm{s}}\,.$$

This together with (18) proves the reult. **Q.E.D.**

Equation (17) as well as Proposition 14 demonstrate the limitations applying to the performance of any routing scheme. The scheme to be analyzed below is the simplest possible:

Packets are routed in a <u>greedy</u> fashion; that is, each packet advances at its respective path as fast as possible. When several packets contend for the same arc, then priority is allotted on a FIFO basis.

In fact, given that there is only one path per origin-destination pair, greedy routing is the most natural scheme arising in the context of the butterfly. It will be shown in §4.3 that this simple scheme is very efficient.

4.3 Performance Analysis of Greedy Routing

Similarly with the hypercube (see §3.1), under greedy routing, the butterfly may be viewed as a queueing network $\mathcal{R}$ with $d2^{d+1}$ deterministic FIFO "servers"; each of them has unit service duration and corresponds to an <u>arc</u>. In Fig. 3b, we present the network $\mathcal{R}$ corresponding to the 2-dimensional butterfly. The main properties of the equivalent network $\mathcal{R}$ are as follows:

Property A: $\mathcal{R}$ is a <u>levelled</u> network; it consists of d levels, with the jth level comprising all arcs $(x; 1; \mathrm{s})$ and $(x; 1; \mathrm{v})$. In fact, each packet receives one time unit of "service" at each level, contrary to the network described in §3.1, where a packet might skip some of the levels.

Property B: Routing is <u>Markovian</u>. In particular, after traversing arc $(y; j; \mathrm{s})$ [resp. $(y; j; \mathrm{v})$], where $j \neq d$, a packet takes one of the following two actions: either it joins the queue for arc $(y; j+1; \mathrm{s})$ [resp. $(y \oplus e_j; j+1; \mathrm{s})$] with probability $1-p$; or it joins the queue for arc $(y; j+1; \mathrm{v})$ [resp. $(y \oplus e_j; j+1; \mathrm{v})$] with probability p. After crossing arc $(y; d; \mathrm{s})$ [resp. $(y; d; \mathrm{v})$], a packet departs from the network with probability 1. Different packets take their routing decisions independently of each other.

This property may be established by reasoning similarly as in Lemma 4; recall that an independence result, analogous to Lemma 1, applies for the butterfly as well (see §4.1).

Next, we investigate the stability properties of our greedy routing scheme; for this purpose, we need the following result:

Proposition 15: The total arrival rate at each arc $(x; j; \mathrm{s})$ equals $\theta_{\mathrm{s}} = \lambda(1-p)$. Also, the total arrival rate at each arc $(x; j; \mathrm{v})$ equals $\theta_{\mathrm{v}} = \lambda p$. ∎

Proof: We fix some $j \in \{1, \ldots, d\}$. By symmetry, all straight (resp. vertical) arcs of the jth level have the same total arrival rate $\theta_{\mathrm{s}}^{(j)}$ [resp. $\theta_{\mathrm{v}}^{(j)}$]. As already mentioned, each packet crosses some straight (resp. vertical) arc of the jth level with probability $1-p$ (resp. p); also the total arrival rate over all arcs of the jth level equals $\lambda 2^d$, because each packet crosses exactly one arc of this level. Therefore, we have $\lambda 2^d(1-p) = 2^d \theta_{\mathrm{s}}^{(j)}$ and $\lambda 2^d p = 2^d \theta_{\mathrm{v}}^{(j)}$, which proves the result. **Q.E.D.**

Similarly as in §3.2, the sufficient condition for stability of the equivalent network $\mathcal{R}$ (and of the greedy routing scheme) is obtained by applying Theorem 2A of [Bor87]; this condition is as follows:

Proposition 16: Greedy routing on the butterfly is stable if

$$\lambda p < 1 \qquad \text{and} \qquad \lambda(1-p) < 1 ,$$

or equivalently $\rho \overset{\mathrm{def}}{=} \lambda \max\{p, 1-p\} < 1$. ∎

In light of the necessary condition for stability in (17), it is seen that greedy routing in the butterfly has <u>optimal</u> stability properties.

Finally, we establish the upper bound for the average delay T per packet induced by greedy routing.

Proposition 17: There holds

$$T \leq \frac{dp}{1-\lambda p} + \frac{d(1-p)}{1-\lambda(1-p)}\,, \qquad \forall \rho < 1\,. \qquad \blacksquare$$

Proof: By Little's law, we have

$$T = \frac{N}{\lambda 2^d}\,, \tag{19}$$

where N is the average total number of packets present in the equivalent network $\mathcal{R}$ in steady-state. We now consider the network $\tilde{\mathcal{R}}$, which is identical to $\mathcal{R}$ except for the fact that all of its "servers" operate under a PS discipline; let $\tilde{N}$ be the corresponding average total number of packets. Since $\mathcal{R}$ is a levelled network with Markovian routing, we can apply Proposition 11; see also the comment on the generality of that result, following its proof. Therefore, we have

$$N \leq \tilde{N}\,. \tag{20}$$

In the stable case (i.e., for $\rho < 1$), network $\tilde{\mathcal{R}}$ is of the product form [Wal88], pp. 93-94. Recalling also Proposition 15, it follows that the stationary probability that a particular "server" $(x; j; \mathrm{v})$ [resp. $(x; j; \mathrm{s})$] of $\tilde{\mathcal{R}}$ hosts n packets equals $(1-\lambda p)(\lambda p)^n$ (resp. $[1-\lambda(1-p)][\lambda(1-p)]^n$). Since there exist $d2^d$ "servers" of each of the two types, it follows that

$$\tilde{N} = d2^d \frac{\lambda p}{1-\lambda p} + d2^d \frac{\lambda(1-p)}{1-\lambda(1-p)}\,. \tag{21}$$

This together with (19) and (20) proves the result. **Q.E.D.**

Next, we comment on the number of packets stored per node of the butterfly; first, notice that only the nodes of levels $1, \ldots, d$ have to store packets. An overall estimate of the expected number of packets per node is provided by the quantity $\frac{N}{d2^d}$, which satisfies

$$\frac{N}{d2^d} \leq \frac{\lambda p}{1-\lambda p} + \frac{\lambda(1-p)}{1-\lambda(1-p)} \stackrel{\text{def}}{=} q_\rho\,.$$

This estimate is quite favorable because it suggests that the "overall" average queue-size per node is $O(1)$ for any fixed ρ. However, it is not guaranteed that this bound holds for the average number of packets stored by the nodes of each individual level. It is conjectured that this is actually the case; the following result provides strong evidence for this claim: for any $j \in \{1, \ldots, d\}$, the total number of packets stored by the nodes of levels $1, \ldots, j$ does not exceed $j2^d q_\rho(1+\epsilon)$ with high probability, for any $\epsilon > 0$. This result may be proved by applying stochastic domination between the first j levels of networks $\mathcal{R}$ and $\tilde{\mathcal{R}}$, and using the product-form property of $\tilde{\mathcal{R}}$.

Using Propositions 14 and 17 and the definition of ρ it follows that

$$\frac{1}{2}\max\{p, 1-p\} \leq \lim_{\rho \to 1}[(1-\rho)T] \leq d\max\{p, 1-p\}\,.$$

It is an interesting open question to close the gap in the above inequality. Similarly as for the hypercube (see the end of §3.3), it is conjectured that the upper bound is tight for all $p \in (0,1)$; for $p = 0$ and for $p = 1$, the lower bound is tight, because packets originating at different nodes follow disjoint paths.

Finally, it should be noted that the case of slotted time can be treated as in §3.4.

5. CONCLUDING REMARKS

In this paper, we analyzed a problem where the nodes of the hypercube network generate packets at random time instants, according to independent Poisson processes. Each packet has unit transmission time and is destined for a randomly selected node; in a special case, the destination distribution is uniform. We considered a simple greedy routing scheme, where each packet crosses the hypercube dimensions required in increasing index-order. We proved that this scheme has optimal stability properties and, when stable, it induces an average delay $T = \Theta(d)$ per packet; the bounds on the average delay were given in simple closed-form expressions. Our analysis was based on a new approach, which relates the behavior of the hypercube (under the routing scheme considered) to that of a queueing network with Markovian routing. Using the same idea, we extended the results to the butterfly network, thus proving the efficiency of greedy routing in this context.

It would be of interest to analyze the problem under an arbitrary destination distribution. For this case, it may be profitable to "mix" the packets by first sending each of them to a random intermediate node, as is done for the permutation task in [VaB81] and [Val82]. Such a "mixing" may result in improved delay properties under medium traffic, at the expense of reducing the maximum traffic that may be sustained by the system.

In an even more general version of the problem analyzed, it may be assumed that each packet is destined for a different subset of nodes; it may also be assumed that the packets received by a node influence the packet-generating process of this node as well as the lengths and destinations of the new packets. This situation arises in the distributed execution of iterative algorithms. Analyzing this general problem seems to be a rather challenging and interesting direction for further research.

Acknowledgement: The authors are grateful to Dr. Teunis Ott for mentioning the possibility of comparing the FIFO and PS disciplines.

REFERENCES

[AbP86] S. Abraham and K. Padmanabhan, "Performance of the Direct Binary n-Cube Network for Multiprocessors", *Proceedings of the 1986 International Conference on Parallel Processing.*

[BeT89] D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall.

[BOSTT89] D.P. Bertsekas, C. Ozveren, G.D. Stamoulis, P. Tseng, and J.N. Tsitsiklis, "Optimal Communication Algorithms for Hypercubes", Report LIDS-P-1847, Laboratory for Information and Decision Systems, M.I.T.

[BoH82] A. Borodin and J.E. Hopcroft, "Routing, Merging and Sorting on Parallel Models of Computation", *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pp. 338-344.

[Bor87] A.A. Borovkov, "Limit Theorems for Queueing Networks, Part I", *Theory Probab. Appl.*, vol. 31, pp. 413-427.

[Bru71] S.L. Brumelle, "Some Inequalities for Parallel-Server Queues", *Operations Research*, vol. 19, pp. 402-413.

[ChS86] Y. Chang and J. Simon, "Continuous Routing and Batch Routing on the Hypercube", *Proceedings of the 5th ACM Symposium on Principles of Distributed Computing*, pp. 272-281.

[GrG86] A.G. Greenberg and J. Goodman, "Sharp Approximate Models of Adaptive Routing in Mesh Networks", preprint.

[GrH89] A.G. Greenberg and B. Hajek, "Deflection Routing in Hypercube Networks", preprint.

[HaC87] B. Hajek and R.L. Cruz, "Delay and Routing in Interconnection Networks", In A.R. Odoni, L. Bianco, and G. Szago (Eds.), *Flow Control of Congested Networks*, Springer-Verlag.

[JoH89] S.L. Johnsson and C.-T. Ho, "Optimum Broadcasting and Personalized Communication in Hypercubes", *IEEE Trans. Comput.*, vol. 38, pp. 1249-1267.

[Kle75] L. Kleinrock, *Queueing Systems, Vol. I: Theory*, John Wiley.

[Lei90] F.T. Leighton, "Average Case of Greedy Routing Algorithms on Arrays", *preprint.*

[MiC87] D. Mitra and R.A. Cieslak, "Randomized Parallel Communications on an Extension of the Omega Network", *J. ACM*, vol. 34, pp. 802-824.

[SaS85] Y. Saad and M.H. Schultz, "Data Communication in Hypercubes", Dept. of Computer Sciences, Research Report YALEU/DCS/RR-428, Yale University.

[StT90] G.D. Stamoulis and J.N. Tsitsiklis, "Efficient Routing Schemes for Multiple Broadcasts in Hypercubes", Report LIDS-P-1948, Laboratory for Information and Decision Systems, M.I.T.; to appear in *Proceedings of the 29th IEEE Conference on Decision and Control.*

[VaB81] L.G. Valiant and G.J. Brebner, "Universal Schemes for Parallel Communication", *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pp. 263-277.

[Val82] L.G. Valiant, "A Scheme for Fast Parallel Communication", *SIAM J. Comput.*, vol. 11, pp. 350-361.

[Val88] L.G. Valiant, "General Purpose Parallel Architectures", Report TR-07-1988, Aiken Computation Laboratory, Harvard University.

[Var90] E.A. Varvarigos, "Optimal Communication Algorithms for Multiprocessor Computers", Report CICS-TH-192, Center for Intelligent Control Systems, M.I.T.

[Wal88] J. Walrand, *An Introduction to Queueing Networks*, Prentice-Hall.

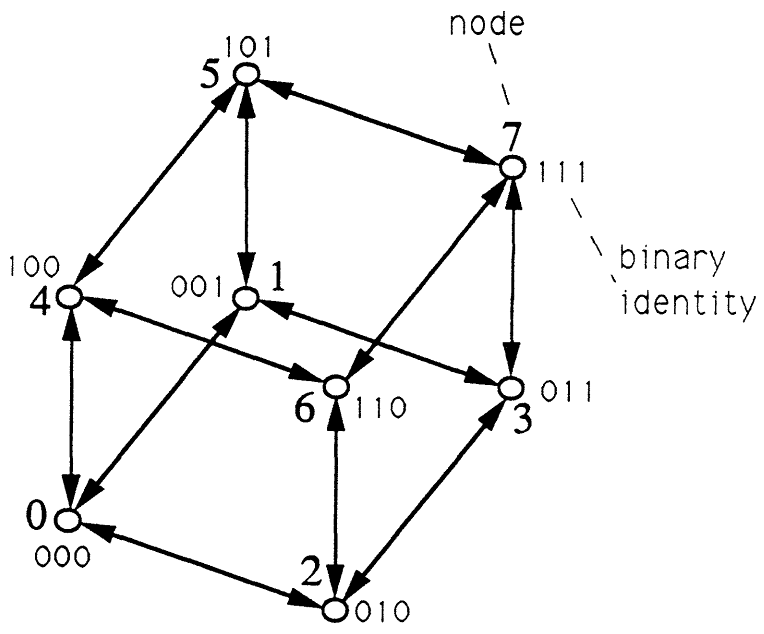

Figure 1a: The 3-dimensional hypercube.

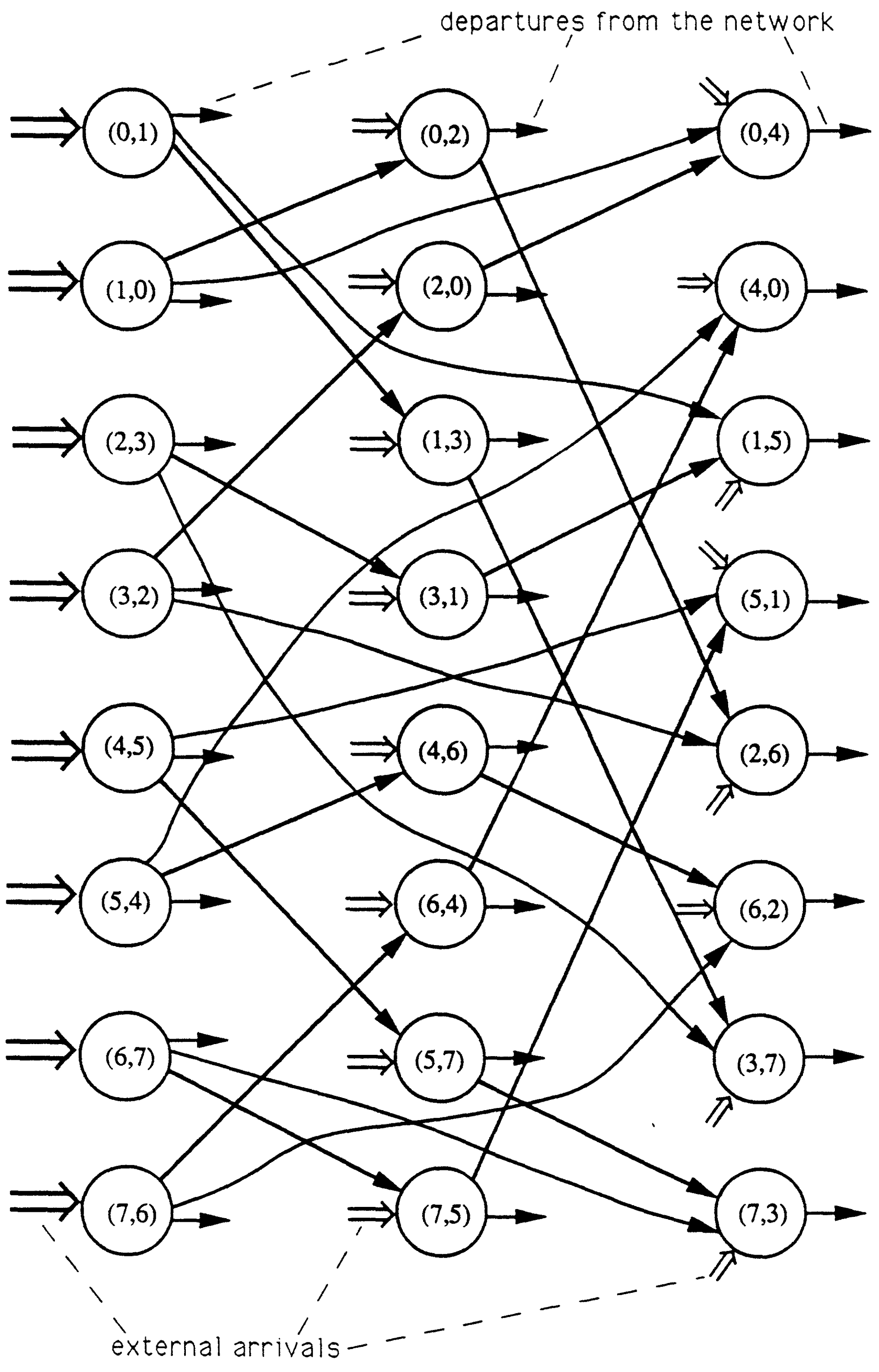

Figure 1b: The equivalent network $\mathcal{Q}$ for the 3-dimensional hypercube.

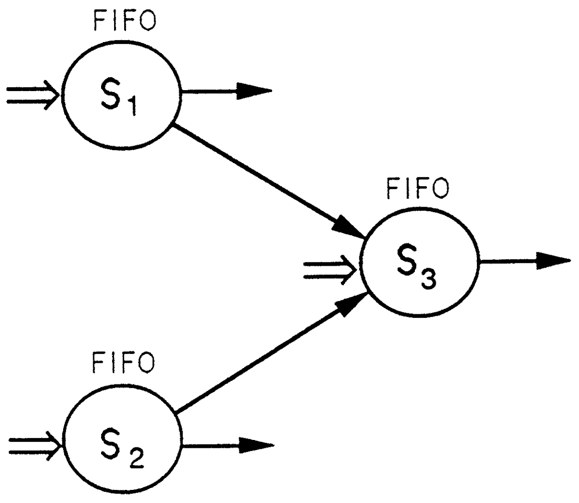

Figure 2a: Network $\mathcal{G}$.

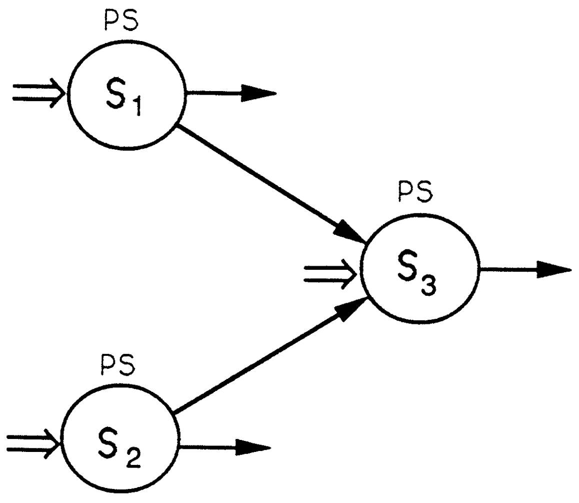

Figure 2b: Network $\tilde{\mathcal{G}}$.

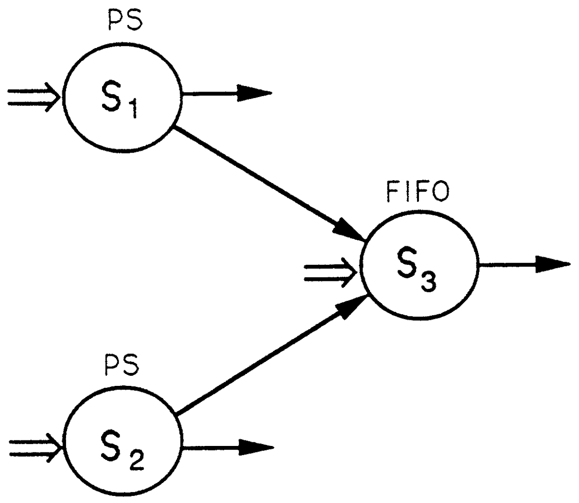

Figure 2c: Network $\mathcal{G}'$.

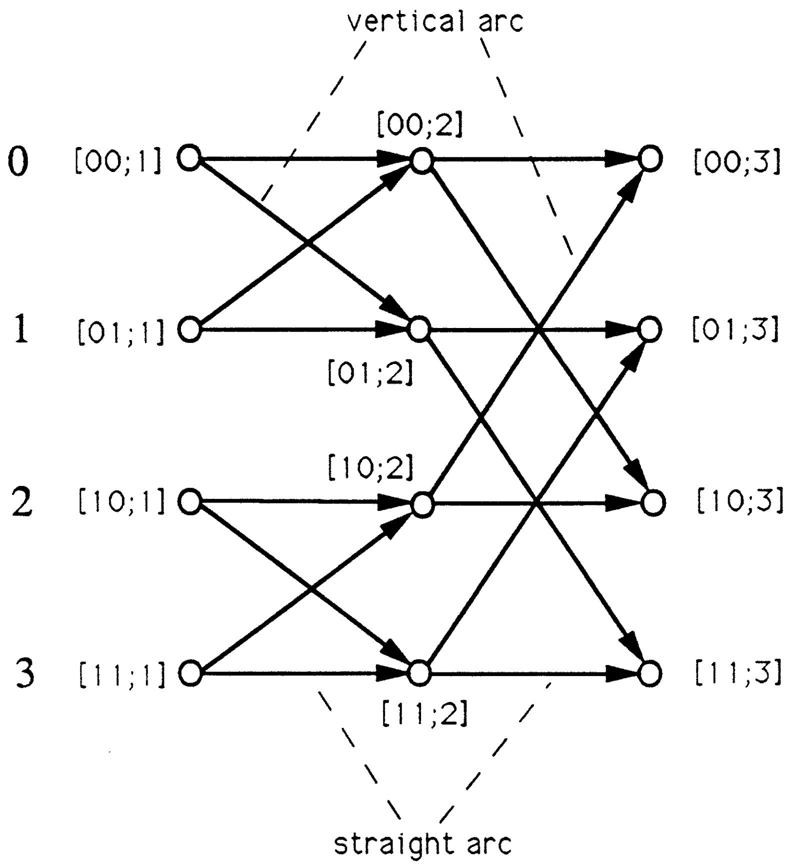

Figure 3a: The 2-dimensional butterfly.

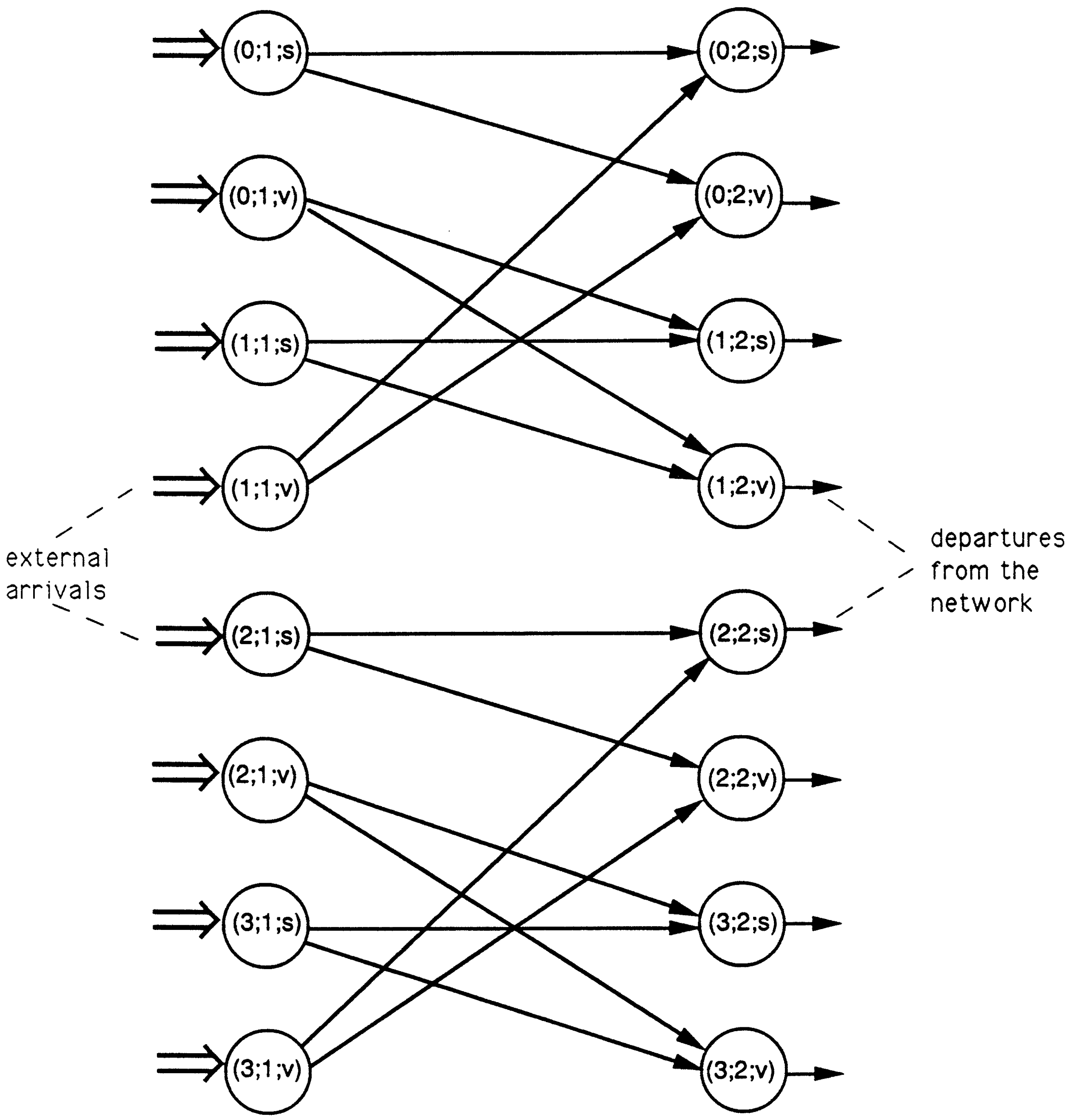

Figure 3b: The equivalent network $\mathcal{R}$ for the 3-dimensional butterfly.