Bounds on the Greedy Routing Algorithm for Array Networks

Michael Mitzenmacher*
Computer Science Division
U.C.Berkeley
CA 94720
email: mitzen@cs.berkeley.edu

Abstract

We analyze the performance of greedy routing for array networks by providing bounds on the average delay and the average number of packets in the system for the dynamic routing problem. In this model packets are generated at each node according to a Poisson process, and each packet is sent to a destination chosen uniformly at random. Our bounds are based on comparisons with computationally simpler queueing networks, and the methods used are generally applicable to other network systems. A primary contribution of the paper is a new lower bound technique that also improves on the previous lower bounds by Stamoulis and Tsitsiklis for heavily loaded hypercube networks. On heavily loaded array networks, our lower and upper bounds differ by only a small constant factor.

We further examine extensions of the problem where our methods prove useful. For example, we consider variations where edges can have different transmission rates or the destination distribution is non-uniform. In particular, we study to what extent optimally configured array networks outperform the standard array network.

1 Introduction

1.1 Statement of the Problem

In this paper, we consider the important problem of dynamic routing in an n by n array mesh network. Our model can be described as follows: nodes (or processors) on an array mesh generate packets at random times. Each packet must be routed to a unique destination that is chosen uniformly at random from the nodes on the array. (For convenience, we allow a packet's destination to be the same as its starting point; the arguments can be modified easily.)

We make standard assumptions about the network for our analysis. The unit of transmission is a packet, which transverses an edge in the network in unit time. Only one packet can be in transit across an edge at any time, although a node with multiple input/output connections can receive and transmit more than one packet at any instant. Packets are buffered when an edge on their path is busy, and nodes are assumed to have infinite buffer capacity. We model arrivals to the network by Poisson processes at each node.

We will analyze the *greedy routing* method, an intuitively powerful paradigm whereby packets travel on natural shortest paths between nodes. On a mesh network, greedy routing corresponds to packets first being routed to the correct column and then to the correct row. Greedy routing has proven successful because it is simple to implement and generally effective in practice. As we shall show, it also proves amenable to theoretical analysis.

1.2 Previous Work

Leighton's previous work on greedy routing for arrays and tori motivates this study [8, 9]. Leighton finds probabilistic bounds on the maximum delay and buffer size, given the following assumptions: packets are generated only at discrete time intervals, packets are generated with a fixed constant probability at each node at each discrete unit of time, and packets with the furthest to travel in each direction are sent first. His derivations also yield results for first-in first-out (FIFO) service, but only at low arrival rates. Recently, similar work improving on the combinatorial analysis has been done by Kahale and Leighton [3].

Primarily we follow a different method of analysis based on the work of Stamoulis and Tsitsiklis in [12], which uses comparison methods. Such methods have been used for other stochastic models, but primarily for single queues. Some examples of various applications can be found in [13]. Stamoulis and Tsitsiklis present sufficient conditions for when the average delay in a network with FIFO service and unit service times can be bounded above by a comparison with a modified, simpler network, and then apply their methods to greedy routing on hypercube and butterfly networks. They also find lower bounds on the average delay a packet experiences in these networks. Here, we deviate from their methods and demonstrate a stronger lower bound technique also based on comparisons that is both powerful and easily generalizable.

Recently, this problem has been studied using the Jackson open queueing network model in [1]. This model differs from others in that the transmission time across an edge, instead of being constant, is exponentially distributed with unit mean [2]. Although this model is less realistic, it proves easier to analyze using standard queueing theory techniques. We shall relate our results closely to this model. One interpretation of these results is that the average delay given by

*Supported by the ONR

the Jackson open queueing network model (i.e., when transmission times are exponentially distributed) yields an upper bound for the average delay when transmission takes constant time for this and other similar problems.

1.3 Summary of Results

We provide new, rigorous upper and lower bounds for the average time a packet spends in a mesh network. The upper bound is derived by comparing the array mesh network to a similar network with a different service policy, which has a computationally simple equilibrium distribution, and is based on work by Stamoulis and Tsitsiklis in [12]. The new lower bound technique we develop is also based on a comparison argument that does not seem to appear in previous literature. On the array, as the network load reaches capacity, our upper and lower bounds differ by a factor of at most either 3 or 6, depending on whether the array has an even or odd number of nodes. Our lower bound technique also improves on the results of Stamoulis and Tsitsiklis for heavily loaded hypercube networks.

The analysis also applies to interesting generalizations of the standard problem. For example, we also examine the problem of bounding the average delay when one can vary the transmission rates of the wires, subject to a fixed linear constraint that corresponds intuitively to the cost of the network. Besides yielding an upper bound, our analysis shows how much more traffic an optimally configured array network can handle over the standard configuration. While the standard configuration of an n by n array is stable for external arrival rates up to $4/n$ (for even n), we prove that the optimally configured network is stable for arrival rates up to $6/(n+1)$. We also consider special situations where the destination distribution is non-uniform, but depends on the distance of the destination from the packet source, as well as other common extensions.

The paper proceeds as follows: in section 2, we provide some basic definitions that describe the problem. In section 3, we apply the work of Stamoulis and Tsitsiklis to the array and describe the relation of our result to the Jackson open queueing network model. In section 4, we derive a rigorous lower bound on the average delay as well as provide a useful approximation. We consider generalizations of the problem in section 5, including non-uniform destination distributions and varying transmission times. We conclude with a discussion of possible future directions.

2 Definitions and Background

2.1 Definitions for the Model

We begin with the definitions we will use throughout the paper. We first describe the topology of an array network. The underlying graph consists of an n by n array of nodes. (For convenience we only consider square arrays; rectangular arrays are easily handled similarly.) Nodes are connected by directed edges to their neighbors in the same row and column. Note that two directed edges connect each pair of neighbors, one in each direction. These edges correspond to an input and an output wire for each pair in the obvious manner. For the underlying graph $G = (V, E)$, we shall label the nodes by the ordered pairs (i, j), where i represents the node row and j represents the node column, and $1 \leq i, j \leq n$. We assume the node $(1, 1)$ lies in the upper left-hand corner. Edges are denoted by ordered pairs of nodes.

In our standard model packets are generated at the nodes as independent Poisson processes with rate λ. Their destinations are uniformly distributed over all nodes in the network. Packets travel along the directed edges, with at most one packet on any edge at any given time. Nodes have infinite buffer space available to hold waiting outgoing packets. The packets are sent out according to the FIFO discipline. Packets move to their destination greedily, first to the correct column along only row edges and then to the correct row along only column edges.

We model this network by considering an associated queueing network, $\mathcal{Q}$, where each directed edge is a FIFO server with unit service time. Since edges represent queues, we use the terms interchangably throughout the paper. Our upper bound uses in comparison a similar queueing network, where the servers are *Processor Sharing*, or PS. Under the PS discipline, all customers queued at a server receive an equal proportion of the available service simultaneously.

In order to compare networks, we require the concept of stochastic domination. We say a random variable Y stochastically dominates a random variable X, and write $X \leq_{st} Y$, if $\Pr[X > \alpha] \leq \Pr[Y > \alpha]$ for all α.

Certain classes of queues will naturally arise as we develop our bounds. For an M/D/1 queue, the arrivals to the queue constitute a Poisson process, the service time required by the arrivals are constant, and there is only one server. Similarly, a queue is said to be of type M/M/1 if the above conditions hold but the service times are exponentially distributed. Notice that although the external arrivals to each queue in the array system we have described form a Poisson process, the arrival process of all packets to a queue does not. Thus the queues in our standard system are neither M/M/1 nor M/D/1 queues.

Finally, we make note of variables used throughout the paper. The variable λ refers to rate at which packets are generated at each node, while λ_e refers to the total rate of packet arrivals on edge e. The service rate at an edge e will be denoted by ϕ_e; notice that in the standard problem ϕ_e is always 1. We define the *network load*, ρ, to be $\rho = \max_{e \in E} \lambda_e / \phi_e$. For single M/M/1 and M/D/1 queues, it is well known that when $\rho < 1$ the queue is *stable*, meaning that there is a unique equilibrium distribution and that such quantities as the average delay and queue size are well defined [6]. We shall generally assume stability for the networks analyzed in this paper without further concern; indeed, the upper bounds given demonstrate the stability of the networks for $\rho < 1$.

The variable T represents the average delay, by which we mean the average time a packet spends in the system from generation to arrival at its destination. The average distance a packet travels, by which we mean the number of edges it passes through, will be denoted by $\bar{n}$. A simple counting argument reveals that $\bar{n} = \frac{2}{3}(n - \frac{1}{n})$ in the two-dimensional n by n array. Sometimes we will also wish to refer to the average distance, excluding packets with the same source and destination. We will refer to this average as $\bar{n}_2$, which is $2n/3$ in the two-dimensional array.

2.2 Product-form Networks

The comparison that yields the upper bound proves useful because under the PS discipline the network becomes a *product-form network*. A network is product-form if in the equilibrium distribution each queue appears as though it were an independent process with Poisson arrivals [14]. For the PS network, under the invariant distribution the number of packets at each queue has a geometric distribution with mean $\frac{\lambda_e}{\phi_e - \lambda_e}$. The λ_e can be determined either by solving a system of equations [6], or by using the technique of [1], so computing the expected number at each queue (and hence in the system) is a simple exercise. The average delay is then easily derived using Little's Law [10]

$$N = T \sum \lambda,$$

where in the above equation N is the average number of packets in the system, T is the average time a packet spends in the system, and $\sum \lambda$ is the overall rate at which packets enter the system.

As it happens, the equilibrium distribution under the PS discipline is the same as the equilibrium distribution for the equivalent *Jackson network.* This relationship will be explained more fully in section 3.3 and used to compare the upper and lower bounds.

3 An Upper Bound

3.1 A Bounding Theorem

We begin by reviewing the results of Stamoulis and Tsitsiklis [12], and then demonstrate that their result applies to array networks. Before providing the relevant theorem, we briefly explain the intuition. Stamoulis and Tsitsiklis define a *sample path* of a system to be all the information regarding packet arrival times (from outside the system) and routing information. (Technically, they require that this information is in a special form, such as "the fifth packet arriving at queue 5 continues along to queue 7.") They show that for certain networks, the PS network is a *delayed version* of a standard FIFO network. That is, given any fixed sample path, if we look at the ordered list of times when packets exit the system from a queue, every time for the PS network is at least as large as the corresponding time in the FIFO network. In this respect, the PS network looks like a slowed-down version of the original FIFO network. This leads to the following theorem:

Theorem 1 (Stamoulis and Tsitsiklis) *Let Q be a queueing network satisfying the following properties with unit service times and FIFO servers:*

- *The network is layered. That is, the arcs are labeled with numbers from the set $\{1, \ldots, N\}$ for some N, and any packet crossing an arc labeled i thereafter only crosses arcs with labels $j, j > i$ until exiting the network.*
- *Routing is Markovian. In other words, the probability distribution of the next arc to be crossed depends only on the arc just traversed, instead of on the complete path the packet has taken.*
- *The external arrival streams at each arc are independent Poisson streams with a fixed rate (which may depend on the arc).*

Let $\tilde{Q}$ be an identical network with PS servers. Then if $N(t)$ (resp $\tilde{N}(t)$) denotes the (random) total number of packets present in Q (resp $\tilde{Q}$) at time t, then

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

This theorem bounds the average number of items in the system, and with Little's Law one can bound the average delay. Note that, as mentioned in Section 2.2, $\tilde{N}(t)$ is easy to compute, so the bound is also meaningful. It would be even more desirable to bound each individual queue the same way, but the argument only holds for the system as a whole.

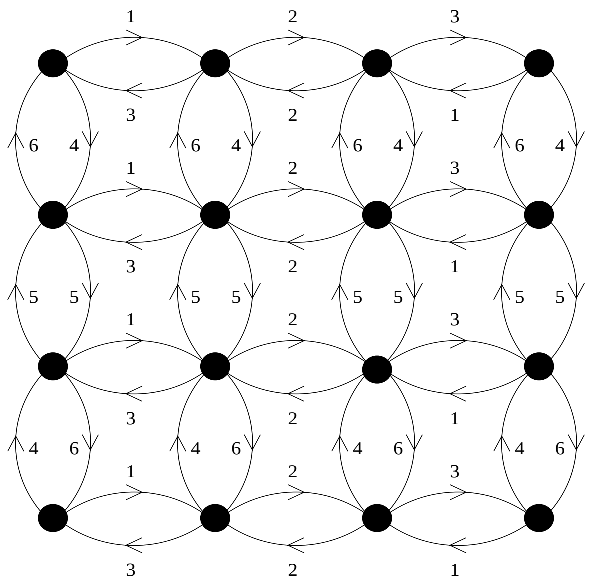


Figure 1: Layering the Array

3.2 The Conditions of the Theorem

We now demonstrate that the conditions of the theorem hold for the array. In our model the packets generated at each node are Poisson processes, and newly generated packets correspond to external arrivals. Hence the external arrival streams at each arc are Poisson processes, so the third condition is trivially satisfied. We must also demonstrate that the array is layered and the greedy routing scheme is Markovian.

Although the array is not layered under general routing schemes, it is under the greedy routing scheme with the defined topology. An example of a labeling that layers the array is given in Figure 1. The labeling suggests the following lemma:

Lemma 2 *The following labeling layers the array under greedy routing:*

edge	*label*
$((i,j),(i,j+1))$	j
$((i,j+1),(i,j))$	$n-j$
$((i,j),(i+1,j))$	$n+i-1$
$((i+1,j),(i,j))$	$2n-i-1$

Proof: The proof is immediate, since packets first head left or right and then only up or down. ■

To show the routing process is Markovian, we demonstrate a Markov process that moves a packet along a row/column such that it stops uniformly at every node. From this it is clear that there is a Markov process simulating greedy routing with uniform destinations.

Lemma 3 *Given a linear array of n elements, there is a Markov chain that simulates an entering packet being transferred with uniform probability to every position on the array.*

Proof: Label the nodes $1, \ldots, n$ in the obvious fashion. Suppose a packet enters at node k. It remains at its entry point with probability $1/n$; otherwise, it moves left with probability $\frac{k-1}{n}$ and right with probability $\frac{n-k}{n}$. Now suppose the packet is moving to the left. After each move, a packet at node j stops with probability $1/j$ and continues to the left otherwise. By symmetry the case to the right can be handled similarly. This clearly defines a Markov process, and it is a simple exercise to show that each packet goes to each possible destination with probability $1/n$. ∎

Corollary 4 *Greedy routing on an array is Markovian.*

From our previous discussion, we can conclude the following theorem:

Theorem 5 *The expected number of packets in an array network with unit service times under the PS service model yields an upper bound for the expected number of packets of the equivalent array network with unit service times under the FIFO model. Similarly, the same holds true for the average delay experienced by a packet.*

3.3 Relation to the Jackson model

The Jackson open queueing network model differs from the standard model in that transmission times (i.e., service times) are exponentially distributed with mean 1. The equilibrium distribution of the Jackson model is the same as the PS service model with unit edge time. This follows from a sequence of standard results from queueing theory [4, 6, 14].

This model was recently examined by Harchol-Balter and Black to approximate greedy routing behavior [1]. We can interpret our result as saying that for the array, the Jackson model provides an upper bound on the average lifetime of a packet in the system for the standard (unit-time transmission) model. As described in [1] and noted as far back as [7], the arrival rates at a queue edge can be easily determined combinatorially.

Theorem 6 (Harchol-Balter and Black) *The total arrival rate of packets at an edge directed from (i,j) is given by the following table:*

Direction	*Rate*
Left	$\frac{\lambda}{n}(j-1)(n-j+1)$
Right	$\frac{\lambda}{n}j(n-j)$
Up	$\frac{\lambda}{n}(i-1)(n-i+1)$
Down	$\frac{\lambda}{n}i(n-i)$

From Theorems 5 and 6 and Little's Law we can derive a formula for the upper bound:

Theorem 7 *The average time a packet spends in an array network with constant unit edge transmission times is bounded above by*

$$T \leq \frac{1}{\lambda n^2} \sum_{e \in E} \frac{\lambda_e}{1-\lambda_e} = \frac{4}{\lambda n} \sum_{i=1}^{n-1} \frac{1}{\frac{n}{\lambda i(n-i)} - 1}.$$

This relation between the upper bound and the Jackson open network model will continue to prove useful throughout the paper.

4 An Approximation and Lower Bounds

4.1 Previous lower bound methods

Stamoulis and Tsitsiklis developed a lower bound technique for routing on butterfly and hypercube networks [12]. Essentially, the bounds are achieved by looking at a subset of the edges that satisfies the condition that a packet only crosses one edge from that subset. For example, one can examine edges crossing a single dimension of the hypercube or the first layer in the butterfly. Equivalent results for the array follow by considering an appropriate subset of edges. We state the results here, without proof:

Theorem 8 *For any routing scheme on the array, the average delay T satisfies*

$$T \geq f\left[1 + \frac{\rho}{2n(1-\rho)}\right],$$

where $f = 1/2$ if n is even, $f = 1/2 - 1/n^2$ if n is odd.

For any oblivious routing scheme, the average delay T satisfies

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

where f is as above.

Of course, we also have trivial lower bound $T \geq \bar{n}$. This is immediate, since a packet experiences a unit delay at each edge it passes through.

We develop a different method for lower bounds on greedy routing that is easier to generalize and yields better bounds for heavily loaded networks. The asymmetry of the array will in fact allow us to achieve a very strong lower bound when the network is near capacity. We begin by examining a good approximation for the average delay.

4.2 A useful approximation

As discussed in Section 3.3, the upper bound we found for the average delay in the array network derives from the equivalent Jackson queueing network. The equilibrium distribution of this system, as we have noted, is product-form; it is as though the queues were independent M/M/1 queues.

Since we know the service times are actually constant, it seems reasonable to consider an approximation where in equilibrium each queue is an independent M/D/1 queue. In effect, we simply assume initially that all queues are independent; this type of idea seems to have first been considered by Kleinrock as far back as in [5]. Of course this assumption is unwarranted, but in practice, it proves accurate in many situations. Moreover, this approximation will provide a useful intermediate step in establishing our lower bounds.

The average time a packet spends in the queue in this case can be derived from the Pollaczek-Khinchin mean value formula [6]. Let λ_d be the arrival rate at an M/D/1 queue, N_d be the expected number of packets in the queue in equilibrium, and S be the random variable representing the service time for a packet. Then we have (for a stable system):

$$N_d = E[S]\lambda_d + \frac{\lambda_d^2 E[S^2]}{2(1-\lambda_d E[S])}.$$

In our standard case, $E[S] = 1$, yielding:

$$N_d = \lambda_d + \frac{\lambda_d^2(1 + \mathrm{Var}[S])}{2(1-\lambda_d)}.$$

When the service time is constant, the Var$[S]$ term disappears, but when the service times are exponentially distributed, the variance in the service time is 1. Thus for small arrival rates, the expected number is almost the same, regardless of the service; for large arrival rates, however, the expected number of packets may differ by almost a factor of 2 between the two models. In fact, regardless of the value of $E[S]$, the value of $E[S^2]$ differs by a factor of 2 between the cases where S is constant and S is exponentially distributed, and hence N_d differs by at most a factor of 2 as well. Applying Little's Law, we have the following lemma:

Lemma 9 *The average delay of a packet under the Jackson queueing network model is at most twice the average delay of a packet in the equivalent system of independent M/D/1 queues, where corresponding queues have the same arrival rate.*

The corresponding approximation for the array network, based on a system of M/D/1 queues, is:

$$T \approx (4/\lambda n) \sum_{i=1}^{n-1} \frac{(\lambda i(n-i))[(n-\lambda i(n-i))^2 + n^2]}{2n^2(n-\lambda i(n-i))}.$$

In practice, simulation results suggest that this is a much better approximation than the upper bound. Table 1 compares the above estimate for T and the results from a small set of simulations. As one might expect, the estimate is more accurate for lightly loaded networks, where there is less interference and hence dependence. Interestingly, in heavily loaded networks assuming independence overestimates T, suggesting that the dependence inherent in the network actually helps performance.

This suggestion has recently been borne out by the work of Kahale and Leighton in [3]. They show that the average delay T, for a fixed ρ, is at most some constant greater than $\bar{n}$, the average distance a packet travels. The approximation for T above yields that the difference between T and $\bar{n}$ is linear in n for a fixed ρ. In this light, the discrepancy between the simulation results and the approximation is to be expected. We believe that the approximation may still be useful as a rough estimate, especially when the network size or the arrival rate is small.

4.3 A new lower bound technique

We now demonstrate a new lower bound for the average delay under greedy routing based on a comparison with a network of M/D/1 queues. The method can be applied to various queueing networks and routing schemes, such as tori, hypercubes, or butterfly networks under greedy routing with uniformly distributed destinations. Although this technique does not provide the best lower bounds at low arrival rates, for high arrival rates we achieve better bounds for both the array and the hypercube than previous methods.

We first provide a general version of the lower bound, and follow up with a specialization of the bound for Markovian networks. Recalling that the upper bound was derived from a delayed version of the network, we develop a *rushed* version to find a lower bound. The trick is to send a copy of a packet to all the queues it will visit immediately, and have each duplicate exit the system after it has been served by the single queue. Intuitively, this system will work faster than the standard system, since queues receive their packets immediately. However, the expected number of packets in the system increases by a factor corresponding to the number of duplicates of a packet.

Theorem 10 *Let $\mathcal{Q}$ be a queueing network with unit service times, FIFO servers, and Poisson external arrivals. Let d be the maximum number of distinct services required by any packet over all possible source and destination pairs. Let $\tilde{\mathcal{Q}}$ be a corresponding set of queues with unit service times and FIFO servers, each having independent Poisson arrivals at a rate equal to the total arrival rate for the corresponding queue in $\mathcal{Q}$. If $N(t)$ (resp $\tilde{N}(t)$) denotes the (random) total number of packets present in $\mathcal{Q}$ (resp. $\tilde{\mathcal{Q}}$) at time t, then*

$$E[\tilde{N}(t)] \leq E[N(t)]d, \ \forall t \geq 0.$$

n	ρ	T (Sim.)	T (Est.)
5	0.2	3.545	3.256
	0.5	4.176	3.722
	0.8	6.252	5.984
	0.9	8.867	8.970
	0.95	12.172	12.877
	0.99	20.333	21.384
10	0.2	6.929	6.711
	0.5	7.748	7.641
	0.8	10.652	12.183
	0.9	14.718	18.444
	0.95	21.034	28.014
	0.99	63.950	77.309
15	0.2	10.289	10.123
	0.5	11.192	11.518
	0.8	14.563	18.329
	0.9	19.226	27.718
	0.95	26.867	41.990
	0.99	68.220	103.312
20	0.2	13.649	13.523
	0.5	14.589	15.383
	0.8	18.191	24.465
	0.9	20.041	36.983
	0.95	31.771	56.015
	0.99	77.283	141.127

Table 1: Simulation vs. M/D/1 estimate

Proof: The theorem will come about from a series of comparisons of similar queueing networks. We assume without loss of generality that a packet's complete route through $\mathcal{Q}$ is determined according to the correct probabilities at its generation time. For convenience we also assume that a packet visits each queue at most once. (This restriction could easily be removed.) Note that d can be thought of as the maximum distance a packet travels in the natural manner.

We first examine a new system, $\mathcal{Q}_1$, associated with the network $\mathcal{Q}$. When a packet is generated in $\mathcal{Q}$, for each queue in $\mathcal{Q}$ that the packet will travel through a copy is generated at each corresponding queue in $\mathcal{Q}_1$. Packets do not move from queue to queue in $\mathcal{Q}_1$; instead, when a packet finishes being serviced, it simply leaves the system. Of course different copies of a packet may complete service at their respective queues at different times.

Each queue in $\mathcal{Q}_1$, when examined in isolation, acts as an M/D/1 queue, with arrival rate λ_e. The queues are not independent; however, the expected number of packets in each queue is just that of an M/D/1 queue with Poisson arrivals. By the linearity of expectations, the expected number of packets in $\mathcal{Q}_1$ is the same as the expected number in $\tilde{\mathcal{Q}}$, that is:

$$E[\tilde{N}(t)] = E[N_1(t)],$$

where $N_1(t)$ has the obvious meaning.

We now consider a fixed *sample path*, where here a sample path consists of all information regarding packet arrival times and destinations. We would like to say that any packet p is finished in $\mathcal{Q}_1$, in the sense that all copies of it have been serviced at all the appropriate queues, before p exits in $\mathcal{Q}$. This statement, however, may not be true, since packets may be serviced in different orders in the two networks. In particular, a packet q that gets serviced after p at some queue in $\mathcal{Q}$ may arrive in the system before p does, in which case q is serviced before p at that queue in $\mathcal{Q}_1$.

To avoid this complication, we consider a new system $\mathcal{Q}_2$ that acts exactly as $\mathcal{Q}_1$ with the additional constraint that the servers must handle packets in the same order as $\mathcal{Q}$. We allow edges in $\mathcal{Q}_2$ to remain idle even if there are packets queued, if this is necessary, to satisfy this constraint. Note that, in the worst case, $\mathcal{Q}_2$ services packets at the same times that $\mathcal{Q}$ does. As the order in which packets are serviced does not affect the number in the queue, and adding idle time only increases the number of packets in the queue, the expected number of packets in $\mathcal{Q}_2$ is still at least the expected number in $\mathcal{Q}_1$; that is,

$$E[N_1(t)] \leq E[N_2(t)],$$

where $N_2(t)$ has the obvious meaning.

We now bound the expected number of packets in the network $\mathcal{Q}$. For any fixed sample path, a packet cannot be in $\mathcal{Q}_2$ unless its corresponding packet is still in $\mathcal{Q}$. Thus we can think of $\mathcal{Q}$ as a delayed version of $\mathcal{Q}_2$, except that $\mathcal{Q}_2$ may contain multiple copies of a packet. As the number of copies of each packet is at most d, the expected number of packets in $\mathcal{Q}_2$ is at most d times the expected number of packets in $\mathcal{Q}$; that is,

$$E[N_2(t)] \leq E[N(t)]d.$$

Combining the determined inequalities yields $E[\tilde{N}(t)] \leq E[N(t)]d$, as was to be shown. ∎

By Lemma 9 and Little's Law, this yields a lower bound for the average delay that is within a factor of $4n-4$ of the upper bound for the array.

Although Theorem 10 is sufficient for our model, note that the theorem can be generalized to systems that are not FIFO and have varying service times. The same idea holds; a similar system that immediately receives a copy of a packet at each queue the packet visits will be faster, at the expense of having more packets in the system. Taking into account this factor provides a lower bound. In particular, note that unlike the result on upper bounds, this proof also holds for non-Markovian systems, such as toroidal meshes.

For general networks, Theorem 10 appears to be the best possible. For a single queue, for instance, $E[\tilde{N}(t)] = E[N(t)]d$, and for a linear array of M/D/1 queues, $E[\tilde{N}(t)] \approx E[N(t)]d$. However, we can improve the results for Markovian networks.

Definition 11 *For each queue e in a Markovian queueing network, let d_e be the expected number of distinct services a packet queued at e has left before reaching its destination (including the service at e). We define the* maximum expected remaining distance *of the network, $\bar{d}$, by $\bar{d} = \max d_e$, where the maximum is taken over all queues in the network.*

In an n by n array network under greedy routing, the maximum expected remaining distance is achieved by a packet located at node $(1,1)$ and headed right. In this case $\bar{d} = n - 1/2$.

Theorem 12 *Let $\mathcal{Q}$ be a Markovian queueing network as in Theorem 10, with maximum expected distance $\bar{d}$, and let $\tilde{\mathcal{Q}}$ be a corresponding set of queues as in Theorem 10. Then*

$$E[\tilde{N}(t)] \leq E[N(t)]\bar{d}, \ \ \forall t \geq 0.$$

Proof: The proof is entirely the same as Theorem 10 up to the last paragraph, where we bounded the expected number of packets in the network $\mathcal{Q}$. We now make more careful use of the fact that $\mathcal{Q}_2$ is a delayed version of $\mathcal{Q}$ by noting that a packet cannot be in $\mathcal{Q}_2$ *at queue e* unless its corresponding packet is still in $\mathcal{Q}$ *and has not yet completed service in e*. Thus, for each packet in $\mathcal{Q}$, the number of copies in $\mathcal{Q}_2$ corresponding to that packet is at most the remaining number of services that packet has yet to complete. Let $R(t)$ be a random variable representing the number of services yet to complete over all packets in $\mathcal{Q}$ at time t. We have

$$E[N_2(t)] \leq E[R(t)].$$

Since the network $\mathcal{Q}$ is Markovian, the expected remaining number of services before a packet reaches its destination depends only on its current location, and in particular is independent of the number of packets in the network. In fact, regardless of a packet's location, its expected remaining number of services is at most $\bar{d}$. Thus, we have that

$$E[R(t)] \leq E[N(t)]\bar{d}.$$

The result follows. ∎

By Lemma 9 and Little's Law, this yields a lower bound for the average delay that is within a factor of $2n-1$ of the upper bound.

4.4 The strength of the lower bound

For most networks, the bound given in Theorem 12 is clearly not tight. Using the maximum expected remaining distance to bound the expected remaining number of services appears excessive. However, the example of a linear array of queues again demonstrates that the bound given is essentially the best possible in general. One might also think that $\bar{d}$ could be replaced by $\bar{n}_2$, the expected number of queues a packet travels through. Indeed, the author made this mistake in an earlier version of the paper [11]. However, the remaining distance a packet has left to travel depends on its location, and the distribution of packet locations is not, in general, independent of the number of packets.

In an array network, intuition suggests that the queues in the middle of the array should have higher expected queue sizes, since the number of packets passing through them is larger than for other queues. Thus one would expect that the lower bound of Theorem 12 is very weak for the array. One might well expect that the expected remaining number of services per packet would be well under $\bar{n}_2$. Simulations show this to be the case. Let R be the expected remaining number of services in equilibrium, and $\bar{r}$ be the $E[R]/E[N]$. Table 2 presents some estimates for $\bar{r}$ for array networks of various sizes. The simulations suggest that $\bar{r}$ is indeed less than $\bar{n}_2$, and that $\bar{r}/\bar{n}_2 < 0.7$ for large enough n.

It seems possible that better bounds on the expected remaining number of services could be found for the array by making use of the underlying topology. Such a bound would be directly translatable into a stronger lower bound on the average delay by Theorem 12. Indeed, any results on the distribution of the remaining number of services would be interesting; the value corresponds to the amount of work necessary to empty a system.

n	$\bar{n}$	ρ	$\bar{r}$ (Sim.)
5	3.333	0.2	2.568
		0.5	2.574
		0.8	2.600
		0.9	2.610
		0.99	2.613
10	6.667	0.2	4.665
		0.5	4.694
		0.8	4.746
		0.9	4.775
		0.99	4.776
15	10	0.2	6.755
		0.5	6.796
		0.8	6.875
		0.9	6.913
		0.99	6.924
20	13.333	0.2	8.841
		0.5	8.887
		0.8	8.982
		0.9	9.041
		0.99	9.029

Table 2: Simulation measurement of $\bar{r}$

4.5 Application of the lower bounds

Stamoulis and Tsitsiklis make careful note of the difference between their upper and lower bounds at high loads, that is, as ρ approaches 1. This case is significant since it corresponds to the worst case for network performance. Theorems 10 and 12 improve on their results for the hypercube and match their result for the butterfly for this case.

Following their lead, we consider a hypercube of dimension d, where the destination distribution is such that node of distance k from the node of entry is a packet's destination with probability $p^k(1-p)^{d-k}$. Note that when $p = 1/2$, this distribution is uniform over the nodes of the network. For smaller values of p, packets tend to travel to nearer neighbors, whereas for larger values, packets tend to reach more distant neighbors. Under greedy routing, the system can be thought of as a Markovian network where each packet considers each dimension in some canonical order and crosses an edge in each dimension with probability p.

The previous bounds for the hypercube yield that

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

In particular, since dp is fixed for a given network, in the limit as ρ approaches 1, their bounds on T for a d-dimensional hypercube differ by a factor of $2d$ for all values of p. Since this lower bound is derived by primarily examining only edges crossing one dimension, this factor makes intuitive sense; a factor of 2 arises from the difference between M/M/1 and M/D/1 queues, while the factor of d corresponds to considering only one dimension of edges. Our lower bound improves on this result. The maximum expected remaining distance stems from a packet that is queued to cross an edge in the first dimension and is $1 + p(d-1)$. By Theorem 12, as ρ approaches 1 our upper and lower bounds differ by a factor of $2(dp+1-p)$, which is less than $2d$ for all $p \in (0,1)$. As p approaches 0 the factor separating the upper and lower bounds approaches 2, and it is bounded by a constant for $p = O(1/d)$. In the more usual case of $p = 1/2$, the upper and lower bounds differ by a factor of $d+1$.

For the butterfly consisting of d levels, all packets go through d edges. By Theorem 10, in the limit as ρ approaches 1 our lower bound is within a factor of $2d$ of the upper bound. This matches the results of Stamoulis and Tsitsiklis, as one would expect [12].

The lower bounds for both of these networks could be improved by better bounds on the expected remaining number of services in equilibrium. However, at this time we know of no stronger bounds for these topologies.

4.6 Improving the lower bound in high traffic

The lower bound of Theorem 12 is somewhat disappointing, in that its separation from the upper bound for the array is a factor linear in n. We improve our result so that as ρ approaches 1 the difference is a constant factor. Let us call a queue *saturated* if $\lambda_e/\phi_e = \rho$ and *unsaturated* otherwise. The key is that only saturated edges are important as $\rho \to 1$. Intuitively, this is because the saturated queues grow much larger than all the others. Examining only saturated edges will allow us to reduce the number of copies of a packet we consider in the network $\mathcal{Q}_1$.

We consider the subnetwork of the array network given by the saturated edges. As in Theorems 10 and Theorem 12, we will find a lower bound on the number of packets in the array network; however, this time we only consider packets that cross a saturated edge.

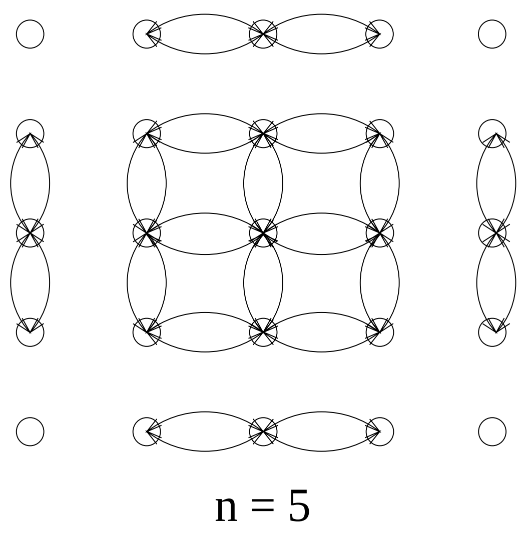
Definition 13 *For each queue e in a Markovian queueing network, let s_e be the expected number of distinct services from saturated servers a packet queued at e has left before reaching its destination (including the service at e). We define the* maximum expected remaining saturated distance *of the network, $\bar{s}$, by $\bar{s} = \max s_e$, where the maximum is taken over all queues in the network.*

Theorem 14 *Let $\mathcal{Q}$ be a network such that for any unsaturated queue e, λ_e/ϕ_e is bounded away from 1 as $\rho \to 1$. Let s be the maximum number of saturated queues a packet can traverse. In the limit as ρ goes to 1, the expected delay is within a factor $2s$ of the upper bound of Theorem 6. If the network is Markovian, then the expected delay is within a factor of $2\bar{s}$ of the upper bound.*

Proof: We sketch the proof, which is similar to Theorems 10 and 12. Consider the original network $\mathcal{Q}$. It is clear that the average delay of a packet can only decrease if we assume that crossing an unsaturated edge incurs no delay. We may think of a modified version of $\mathcal{Q}$, call it $\mathcal{S}$, that offers no delay at unsaturated edges. In this case, the number of packets in the system can be found by examining only the queues at the saturated edges; other edges are assumed to be empty.

We can now proceed as in Theorems 10 and 12 to lower bound the expected number of packets in $\mathcal{S}$. By Little's Law this will provide us with a lower bound on the average delay in $\mathcal{S}$. (Keep in mind that the total arrival rate into $\mathcal{S}$ is still λn^2!) The modifications are simple; when a packet enters $\mathcal{S}$, we introduce a copy of the packet at each saturated edge that it will cross in the corresponding network.

This effectively bounds the expected number of packets in the saturated queues to within a factor of s of the expected number if the system were composed of independent M/D/1 queues. It is simple to show that as ρ goes to 1 the expected number of packets at unsaturated M/D/1 queues is bounded, while the expected number of packets at saturated M/D/1 queues is unbounded. For Markovian



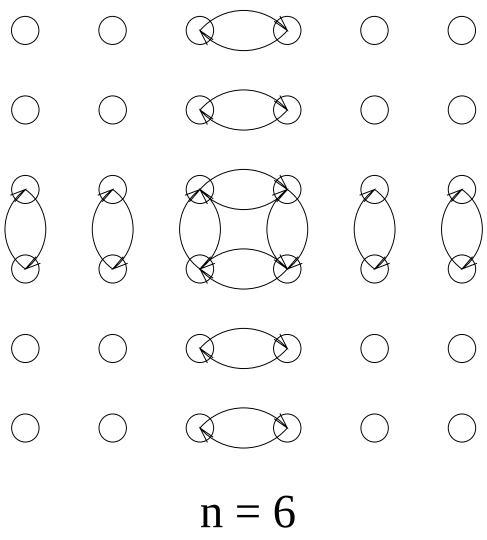


Figure 2: Examples of Saturated Edges in Array Networks

networks, one can similarly examine the expected number of remaining services through saturated queues to replace s by $\bar{s}$. By Lemma 9 and Little's Law, the theorem follows. ■

Theorem 14 applies to array networks, as the expected number of packets at unsaturated queues is bounded by a function of n. Whether n is even or odd makes a significant difference in our lower bound, since a packet can go through at most 2 saturated edges when n is even, and up to 4 when n is odd; see Figure 2. Indeed, by simple combinatorial calculations, one finds that $\bar{s} = 3/2$ when n is even, and $\bar{s} < 3$ when n is odd. (In fact $\bar{s} \to 3$ as $n \to \infty$.) Thus under high loads the upper and lower bounds we have found differ by a factor of 3 when n is even and at most 6 when n is odd. Although this difference seems unusual, it may reflect a real phenomenon in array networks. For example, when n is even the network is stable for $\lambda < 4/n$, but for odd n we must have $\lambda < 4n/(n^2 - 1)$ for stability.

Note that Theorem 14 yields no improvement over Theorem 12 for the hypercube or butterfly, since all queues are saturated by symmetry.

n	r_s (Sim.)
5	1.875
10	1.250
15	2.106
20	1.230
25	2.209

Table 3: Simulation measurement of r_s

As with Theorem 12, we expect that the lower bound of Theorem 14 is far from tight, as $\bar{s}$ overestimates the expected remaining number of services at saturated queues per packet. Let R_s be a random variable corresponding to the remaining number of services at saturated queues at equilibrium, and let $r_s = E[R_s]/E[N]$. In Table 3 some estimates for r_s found by simulation are given. The estimates reflect the value for r_s observed for $\rho = 0.99$; simulations for other values of ρ suggest the dependence of r_s on the arrival rate is minimal.

5 Extensions

5.1 Variable transmission rates

The techniques we have used prove applicable to a number of extensions of the original problem. The first generalization we consider involves varying transmission times across the wires. For example, since edges along the periphery of the array receive less traffic, one might wish to place slower wires there than in the center of the array to build a system with a better performance to cost ratio. How should one build the network to optimize performance?

The problem for the Jackson network model has been considered extensively. The results yield upper bounds that can be applied to the case where service times are constant, using a variation of the proof of Theorem 5.

As an example, we consider the case where service rates are bound by a linear constraint. Imagine that the cost for a server of a given rate is linear; that is, a service rate of ϕ_j for the jth server costs $d_j\phi_j$. Assuming we have a total of D (dollars) to spend on the network, then the service is bound by the constraint:

$$\sum_j d_j\phi_j = D.$$

Theorem 15 *Given a Jackson network, suppose the service rates are subject to the overall constraint $\sum_j d_j\phi_j = D$, where ϕ_j is the service rate of the jth queue and $D > \sum_j \lambda_j$. Then if λ_j is the overall arrival rate at queue j, the optimal allocation to minimize the mean number of customers in the network (and thus the average delay) is*

$$\phi_j = \lambda_j + \frac{\sqrt{\lambda_j d_j}}{\sum_k \sqrt{\lambda_k d_k}} \frac{D - \sum_k \lambda_k d_k}{d_j},$$

where the sum is over all queues in the network.

The theorem is a simple application of Lagrange multipliers. Recall that $N = \sum_j \lambda_j/(\phi_j - \lambda_j)$. One forms the Lagrangian

$$N' = \sum_j \lambda_j/(\phi_j - \lambda_j) + \beta\left[\sum_j d_j\phi_j - D\right]$$

and satisfies the equations $\frac{\partial N'}{\partial \phi_j} = 0$ for all j. (See, for example, [7] or [4].) Note that when all the $d_e = 1$, so improving any queue can be done with equal expense, the optimal allocation corresponds to first allocating each queue just enough service capability to handle its arrival load, and then distributing the remaining money proportionally to the square root of the arrival rates.

Since in equilibrium in the Jackson model all queues are independent, one can easily determine the average number of packets in the system in equilibrium, and hence the average delay from Little's Law. We apply this analysis to the array. Define D^* by $D^* = D - \sum_{e \in E} \lambda_e d_e$. An interpretation of D^* is the extra money available after assigning each queue the minimum service rate necessary for stability. Then the average delay is given by:

$$T = \frac{\bar{n}}{D^*} \left[\sum_{e \in E} \sqrt{\frac{\lambda_e d_e}{\lambda}} \right]^2 .$$

As previously noted, the expression for T above is an upper bound for the case where service requirements are discrete instead of exponentially distributed. It is not clear, however, that the allocation described in Theorem 15 remains optimal for this case; indeed, this seems an interesting open question. Using Theorem 10, one also has a lower bound within $2n-1$ of the upper bound. Note that one cannot apply the argument of Theorem 12, since all queue sizes become unbounded as the arrival rate increases to capacity.

In a system with optimal service rates, the average delay tends to infinity as D^* approaches 0; however, the system is stable for any positive value of D^*. From this we show that, as one might expect, a modified network can handle a higher rate λ of incoming packets. If all $d_j = 1$, then for the original array network $D = 4n(n-1)$, and thus $D^* = 4n(n-1) - \sum_{e \in E} \lambda_e$. We use the identity that the sum of the arrival rates at each node equals the average distance traveled by a packet multiplied by the total external arrival rate. (See, for example, [7].) Thus

$$D^* = 4n(n-1) - \sum_{e \in E} \lambda_e = 4n(n-1) - \bar{n}(n^2 \lambda),$$

and hence D^* is positive whenever

$$\lambda < \frac{4}{\bar{n}}(1 - \frac{1}{n}) = \frac{6}{n+1}.$$

If transmission capacity is optimally distributed, then the array will remain stable under arrival rates of $\lambda < 6/(n+1)$, as opposed to $4/n$. Since this stability condition holds when the transmission times are exponentially distributed, it also holds in the model where transmission times are constant, since the first model yields an upper bound for the second. This condition is necessary for stability for all nondeterministic arrival schemes.

One can similarly find solutions for the problem when other, non-linear constraints are imposed, or where costs vary from edge to edge. Natural constraints depend on the relationship between cost and transmission speed. Further examples of the method can be found in [7]. In practice, one might instead wish to choose transmission rates from a finite set of possibilities. Although this method does not yield an optimum solution for this problem, it can provide a suitable first approximation.

5.2 Further extensions

We can to some extent remove the assumption that a packet's destination is uniform over the array. Theorem 1 requires only that the routing process can be considered Markovian. Thus, for example, one could have the packet move along each row/column in some direction, stopping movement in that direction at each point with probability 1/2, except at the edge of the array (where the packet must stop). This corresponds to a distribution where packets are more likely to travel to nearby destinations. Theorem 12 also applies in this case, and Theorem 10 can be used when the routing is not Markovian.

The methods presented here easily extend to array networks in higher dimensions under the greedy routing paradigm. The derivation seems relatively straightforward; one can explicitly determine the arrival rates at individual queues combinatorially or by solving a large system of equations, as described in [1].

Finally, the results here also hold asymptotically for slotted time, where the time axis is not continuous but instead consists of slots of some fixed duration τ. Arrivals in this model are assumed to come in batches, the number of arrivals at a slot being a Poisson random variable with mean $\lambda\tau$. It is clear that the average time in this case is within τ of the average time in the continuous case; one can simply imagine that the packets instead arrived over the interval as a Poisson process, already having incurred some delay of at most τ. A more detailed argument can be found in [12].

6 Open Problems

Leighton also examines toroidal networks, which appear very similar to array networks [8, 9]. Although the lower bound techniques described in this paper apply to this case, the methods of Stamoulis and Tsitsiklis do not. In fact any network containing a ring of directed edges cannot be layered, and the greedy routing scheme on the torus is clearly not Markovian. An upper bound for the average delay on toroidal networks thus remains open. Similarly, one might consider a randomized version of greedy routing, where packets randomly decide whether to move to the correct row or the correct column. The same approximation and lower bounds given for the standard greedy routing algorithm apply, but the upper bound argument fails. We note that in simulations the randomized greedy routing scheme performs slightly worse than the standard scheme; thus further study of it may be only of academic interest.

Of course there remains room to tighten the bounds given in this paper. It is not yet clear how much better the lower bounds can be tightened, but it seems likely that they could be improved by a constant factor on array, hypercube, and butterfly networks by suitably bounding the expected remaining distance the packets have left to travel. Also, it would be a significant improvement if we could compare the original network directly to a system of M/D/1 queues, without introducing the copies of Theorem 12. Finally, it seems undesirable that the bounds we achieve depend on whether n is odd or even. Perhaps another technique could remove this distinction.

Acknowledgments

The author would like to thank Mor Harchol-Balter for suggesting the problem, particularly the extension to varying transmission times, and Abhiram Ranade for helpful discussions. The author also thanks the anonymous referee for pointing out the error in Theorem 10 in an earlier version of the paper.

References

[1] M. Harchol-Balter and P. Black. Queueing analysis of oblivious packet-routing networks. In *Proceedings of the Fifth Annual ACM/SIAM Symposium on Discrete Algortihms*, pages 583–592, 1990.

[2] J. Jackson. Jobshop-like queueing systems. *Management Science*, 10:131–142, 1963.

[3] Nabil Kahale and Tom Leighton. Greedy dynamic routing on arrays. In *Proceedings of the Sixth Annual ACM/SIAM Symposium on Discrete Algortihms*, pages 558–566, 1995.

[4] F.P. Kelly. *Reversibility and Stochastic Networks.* John Wiley and Sons, 1979.

[5] Leonard Kleinrock. *Communication Nets.* McGraw-Hill, Inc., 1964.

[6] Leonard Kleinrock. *Queueing Systems, Volume I: Theory.* John Wiley and Sons, 1976.

[7] Leonard Kleinrock. *Queueing Systems, Volume II: Computer Applications.* John Wiley and Sons, 1976.

[8] F.T. Leighton. Average case analysis of greedy routing algorithms on arrays. In *Proceedings of the Second Annual ACM Symposium on Parallel Algortihms and Architecutres*, pages 2–10, 1990.

[9] F.T. Leighton. *Introduction to Parallel Algorithms and Architectures.* Morgan Kaufmann, 1992.

[10] J.D.C. Litle. A proof of the queueing formula $l = \lambda w$. *Operations Research*, 9:383–387, 1961.

[11] Michael Mitzenmacher. Bounds on the greedy routing algorithm for array networks. In *Proceedings of the Sixth Annual ACM Symposium on Parallel Algortihms and Architectures*, pages 248–259, 1994.

[12] G.D. Stamoulis and J.N. Tsitsiklis. The efficiency of greedy routing in hypercubes and butterflies. In *Proceedings of the Second Annual ACM Symposium on Parallel Algortihms and Architectures*, pages 248–259, 1991.

[13] Dietrich Stoyan. *Comparison Method for Queues and Other Stochastic Models.* John Wiley and Sons, 1983.

[14] J. Walrand. *An Introduction to Queueing Networks.* Prentice-Hall, 1988.