Distrib. Comput. (1997) 10: 189–197

DISTRIBUTED COMPUTING

Constant time per edge is optimal on rooted tree networks*

Michael Mitzenmacher**

Digital Systems Research Center, 130 Lytton Avenue, Palo Alto, CA 94301, USA (e-mail: michaelm@pa.dec.com)

Received: September 1996 / Accepted: April 1997

Abstract. We analyze the relationship between the expected packet delay in rooted tree networks and the distribution of time needed for a packet to cross an edge using convexity-based stochastic comparison methods. For this class of networks, we extend a previously known result that the expected delay when the crossing time is exponentially distributed yields an upper bound for the expected delay when the crossing time is constant [20] using a different approach. An important aspect of our result is that unlike most other previous work, we do not assume Poisson arrivals. Our result also extends to a variety of service distributions, and it can be used to bound the expected value of all convex, increasing functions of the packet delays. An interesting corollary of our work is that in rooted tree networks, if the expectation of the crossing time is fixed, the distribution of the crossing time that minimizes both the expected delay and the expected maximum delay is constant. Our result also holds in multicasting rooted tree networks, where a single message can have several possible destinations.

Besides offering a useful analysis on this restricted class of networks, we also provide a small improvement to the bounding technique. Surprisingly, this improvement is also applicable to previously developed comparison methods, leading to an improvement in the upper bounds for greedy routing on butterfly and hypercube networks given by Stamoulis and Tsitsiklis [20].

Key words: Queueing theory – Stochastic comparisons – Increasing convex ordering – Tree networks

1 Introduction

We consider the problem of bounding the average packet delay in dynamic packet routing networks. One way to formulate the problem is to consider the packet-routing network as a queueing network, where edges in the graph that models the network behave as servers. In many real-life networks, it takes a fixed constant time for a packet to cross an edge; however, most queueing theory results require the assumption that the service time be exponentially distributed. Stamoulis and Tsitsiklis developed a method to tackle this discrepancy by comparing the two types of networks [20]. They showed that in all layered Markovian networks the expected delay when the service times are exponentially distributed provides an upper bound for the expected delay when the service times are constant with the same mean.

We apply an alternative, weak stochastic ordering that achieves more general results on a more restricted class of networks: *rooted tree networks*. Two rooted tree networks can be compared if the service distribution in one is "*more variable*" than that of the other, in a sense we shall define in Sect. 2. This comparison method has several advantages over previous techniques: it does not require Poisson arrivals; the method can be used for many classes of service distributions, not just those corresponding to constant or exponentially distributed services; the comparison can be used on many non-Markovian networks; and the results can provide bounds not only for the expected delay, but for increasing convex functions of the delay as well.[1]

From our comparison method, we develop a slight improvement in the bounding technique. Instead of comparing the network where service times are constant to the network where service times are exponentially distributed, we use a network where there are edges of both types. Surprisingly, this technique can also be used with the comparison approach of Stamoulis and Tsitsiklis. As a result we can improve their upper bounds on the performance of greedy routing in hypercube and butterfly networks.

1.1 The model

We briefly describe the model. The underlying network is a rooted tree, where packets enter at the root and proceed away from the root until they reach their destination and

* An earlier version of this paper appeared in the 1996 ACM Symposium on Parallel Algorithms and Architectures.

** The work was supported in part by the Office of Naval Research and in part by NSF Grant CCR-9505448. Most of this work was done while the author was a student at U.C. Berkeley.

[1] In this paper, increasing is meant to be synonymous with *non-decreasing*, and is different from *strictly increasing*.

exit the system. The edges of the network are represented by servers, and the time to cross an edge is referred to as the *crossing time* or the *service time*. Packets are served First Come-First Served (FCFS) at each queue. The *delay* of a packet is the difference between the time the packet enters and the time the packet exits the network. We associate the following parameters with the ith packet that arrives to the system:

- A final destination d_i.
- A random variable X_i that represents the time between the $(i-1)$st and ith packet arrival.
- A vector of random variables $S_{i,j}$ that represent the service time the ith packet requires at server j if it visits that server.[2]

For convenience we assume the first arrival occurs at time 0. The destinations d_i are assumed to be generated by some process independent of all X_i and $S_{i,j}$; we call this process the *routing discipline*. We enforce the requirement that the random variables X_i and $S_{i,j}$ be independent unless otherwise noted, and similarly the random variables associated with packets i and i' are independent unless otherwise noted. Note, however, that the *distributions* of X_i and $S_{i,j}$, as well as the destination d_i, can depend on the packet number i.

We note that this model is strong enough to handle, for example, any *Markovian* routing discipline on the network. A routing discipline is Markovian if the probability that, after completing service at queue j, the next destination of a packet is queue k (or that the packet leaves the network) is dependent only on j and is independent of its previous history and the state of the system. For example, on rooted tree networks, if the packet destinations are determined by some fixed distribution, then the network can be modelled with a Markovian routing discipline, and vice versa. Our model also includes many non-Markovian routing disciplines, however. For example, final packet destinations might run through the leaves in some fixed cyclic order, as in a round-robin scheme.

Although rooted tree networks are perhaps the simplest type of network topology, they provide natural representations of several real systems. For example, video transmission from a single server to several destinations can be represented by rooted tree networks. Also, in many systems, there may be small tree-shaped subnetworks that receive all outside traffic through a single source; for example, a university may have one direct server link to the Internet, from which external mail is distributed to the various departments. Such a subnetwork can be modeled by a rooted tree network. This method may also be useful for studying the behavior of some distributed data structures, such as distributed search trees.

The main result (Theorem 8) will require defining the proper notation. The following corollary of our main result, however, appears to be interesting in its own right:

Corollary. *Let the service times in a rooted tree network be $S_{i,j}$. Suppose also that we may vary the $S_{i,j}$ subject to keeping $E[S_{i,j}]$ fixed. Then the expected delay for each packet is minimized when $S_{i,j} = E[S_{i,j}]$, that is, when the $S_{i,j}$ are constant random variables. Similarly, the expected value of the maximum delay for the first k packets is minimized when the $S_{i,j}$ are constant random variables.* □

Thus we find not only that constant time servers are better than servers with exponentially distributed service time in rooted tree networks, but that for these networks, constant service time is the best possible.

Our results also apply to *multi-casting* rooted tree networks. In a multi-casting system, a single message can have several destinations. The model does not need to be dramatically changed for such a system; the only difference is that at a node a packet may instantaneously split into multiple copies, to traverse multiple paths in the network at the same time. We provide a method for determining upper bounds on the expected delay on certain multi-casting systems under the same conditions we use for the standard routing problem.

1.2 Previous work

Stochastic comparison techniques have been used previously primarily on single queues, for example in [15, 21]. These techniques have also been applied and generalized to more complicated processes [1, 13, 14, 19, 22]. Indeed, although our work was derived independently, it strongly resembles the work of Niu [13], who examined tandem queues using a similar approach. A good modern treatment of the subject is given by Shaked and Shanthikumar [19]. In the computer science literature, comparison results have primarily been based on the work of Stamoulis and Tsitsiklis [20]. Using coupling and stochastic comparison they showed that if the underlying network is layered and Markovian, and servers use a First Come First Served (FCFS) policy, then changing the servers in the network from constant time to exponential time with the same mean does not decrease the expected time a packet spends in the network [20]. This work was applied to greedy routing on array networks in [11] and generalized to non-layered networks by Harchol-Balter and Wolfe in [4]. Methods for lower bounds are also suggested in [20] and [11]. This approach, however, has certain limitations: it requires Poisson arrivals into the network, the comparison holds only between constant and exponential service times, and the bounds apply only to expected delay. Results similar to but more general than those of [20] and [4] were also discovered independently by Righter and Shanthikumar in [14] using more general stochastic comparisons.

Other approaches besides stochastic comparison have also proven successful. Better bounds can often be obtained by examining specific networks, as was done by Leighton in [9] and later by Kahale and Leighton in [5]. Recently, a new approach for modeling routing networks has been suggested by Borodin et al., where one assumes that packets are injected into the network by an adversary. This method has proven useful for showing the stability of networks under many different routing policies given reasonable conditions on the adversary, and it can also give loose bounds on the expected delay [2].

[2] Although in packet routing servers usually model just the edges, we may also have servers at each node to model work there as well.

The rest of the paper is organized as follows: in Sect. 2, we will provide the necessary background on the stochastic order relation we use to prove our results. We will prove the main theorem in Sect. 3, and then examine its implications through various corollaries in Sect. 4. In Sect. 5, we offer an improvement to the standard bounding technique and discuss its application to hypercube and bufferfly networks. We clarify the reasons for the restriction to rooted tree networks in Sect. 6

2 The increasing convex ordering

Our work will require using a stochastic order relation similar to the concept of stochastic domination of random variables. We present the necessary background, based primarily on the treatment by Ross; the proofs can all be found either in [17, pp. 270–279] or [19, Sect. 2.A].

Definition. A function $f: \mathbf{R} \to \mathbf{R}$ is *convex* if

$$f(\lambda x_1 + (1-\lambda)x_2) \leqq \lambda f(x_1) + (1-\lambda) f(x_2)$$

for all x_1, x_2, and $0 < \lambda < 1$.

Definition. For random variables X and Y, we say that X is greater than Y *with respect to the increasing convex ordering*, and write $X \geqq_{icx} Y$, if $E[h(X)] \geqq E[h(Y)]$ for all increasing, convex functions h for which the expectations exist. If X and Y have cumulative probability distributions F and G respectively, we may also write $F \geqq_{icx} G$ in place of $X \geqq_{icx} Y$.

The partial order relation $X \geqq_{icx} Y$ should be contrasted with the standard notion of stochastic domination: to say that X is stochastically larger than Y, or $X \geqq_{st} Y$, is equivalent to $E[h(X)] \geqq E[h(Y)]$ for all increasing functions h. Hence $X \geqq_{st} Y$ immediately implies that $X \geqq_{icx} Y$, and the $\geqq_{icx}$ relation can be considered a relaxation of the standard notion of stochastic domination.

Following [17], we shall also use the following more convenient terminology: if $X \geqq_{icx} Y$ then we shall say that X *is more variable than* Y. The following lemma comparing the moments of random variables indicates why this phrasing appears appropriate:

Lemma 1 *If* $X \geqq_{icx} Y$, *then* $E[X^k] \geqq E[Y^k]$ *for* $k \geqq 1$. *Also, if* $E[X] = E[Y]$, *then* $X \geqq_{icx} Y$ *implies* $Var(X) \geqq Var(Y)$. □

The following pictorial condition provides perhaps more intuition for the increasing convex ordering.

Lemma 2 *Suppose that for two random variables* X *and* Y *with finite means* $E[X] \leqq E[Y]$ *and cumulative distribution functions* F *and* G, *respectively, there exists a* δ *such that*

$F(x) \leqq G(x)$ *for* $x \leqq \delta$, *and*

$F(x) \geqq G(x)$ *for* $x > \delta$.

Then $X \leqq_{icx} Y$. □

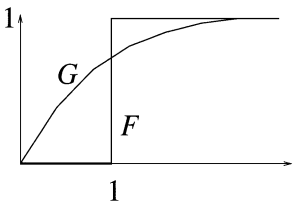

Fig. 1. An application of Lemma 2. The constant random variable with distribution F is less variable than the random variable with distribution G, assuming that they both have the same mean, since the distribution functions cross at a single point

See, for example, Fig. 1. Note that if $E[X] = E[Y]$, the pictorial condition suggests that more of the weight of Y's distribution lies further from the mean than X's, so it stands to reason that Y will be more variable than X.

Since our primary motivation for using these methods is to compare systems that have exponentially distributed service times to systems with other service times, we define classes of random variables that can be ordered against exponential random variables.

Definition. A nonnegative random variable X is *new better than used in expectation* ($NBUE$) if

$$E[X - a | X > a] \leqq E[X] \ \forall a \geqq 0.$$

X is said to be *new worse than used in expectation* ($NWUE$) if

$$E[X - a | X > a] \geqq E[X] \ \forall a \geqq 0.$$

In particular, a constant random variable is NBUE (see [17, Proposition 8.6.1, p. 273]). The class of NBUE random variables will be useful because they are easily compared to exponentially distributed random variables in this partial ordering.

Lemma 3 *If* X *is* $NBUE$ ($NWUE$) *with mean* μ, *and* Y *is exponentially distributed with mean* μ, *then* $X \leqq_{icx} Y$ ($X \geqq_{icx} Y$). □

Thus an exponentially distributed random variable is more variable than a constant one, as one would hope! In fact, as one would expect, of all the random variables with a fixed mean, a constant random variable is the smallest in the increasing convex ordering.

Lemma 4 *If* X *is a random variable with mean* μ, *and* Y *is a constant random variable of mean* μ, *then* $X \geqq_{icx} Y$.

In the networks we consider, we will be able to express the time a packet spends in the system as a function of several random variables, including the service times. Thus we must also consider how this stochastic ordering behaves when we take functions of random variables.

Lemma 5 *If* $X_1, X_2, \ldots, X_n$ *are independent and* $Y_1, Y_2, \ldots, Y_n$ *are independent, and* $X_i \geqq_{icx} Y_i$ *for* $1 \leqq i \leqq n$, *then*

$$g(X_1, \ldots, X_n) \geqq_{icx} g(Y_1, \ldots, Y_n)$$

for all functions g *that are increasing and convex in each argument.* □

The following lemma, combined with Lemma 5, shows that any function g designed by composing addition and maximum operations will be increasing and convex. This fact will be crucial to the proof of the main result.

Lemma 6 *The functions* $\max(X, Y)$ *and* $X + Y$ *are convex and increasing in* X *and* Y. *Also, if* $g(x)$ *and* $h(x)$ *are convex and increasing functions, then so is* $g(h(x))$. □

Finally, the following technical facts regarding the increasing convex ordering will also be useful.

Lemma 7

- *If* $E[X] = E[Y]$, *then* $X \geqq_{icx} Y$ *implies* $-X \geqq_{icx} -Y$.
- *For random variables* X, Y, *and* Θ, *if* $\{X \mid \Theta = \theta\} \geqq_{icx} \{Y \mid \Theta = \theta\}$ *for all* θ *in the support of* Θ, *then* $X \geqq_{icx} Y$.
- *Suppose* $\{X_i\}$ *and* $\{Y_i\}$ *are sequences of random variables that converge in distribution to* X *and* Y *respectively, such that*

  $E[X_i] \to E[X]$ *and* $E[Y_i] \to E[Y]$.

  Then if $X_i \geqq_{icx} Y_i$ *for all* i, *then* $X \geqq_{icx} Y$.

3 More variability increases delay

We are now ready to state and prove our main theorem, which is a natural extension of known results for the case of a single queue. (See, for example, [17, p. 274]).

Theorem 8 *Consider two rooted tree networks* Q_1 *and* Q_2 *with the same underlying topology and routing discipline. Let* Q_1 (Q_2) *have packet interarrival times* X_i^1 (X_i^2), *and let* $S_{i,j}^1$ ($S_{i,j}^2$) *be the service times of the* i*th packet at server* j *in* Q_1 (Q_2). *Let* T_k^1 (T_k^2) *be the departure time of the* k*th packet to arrive in* Q_1 (Q_2), *and let* D_k^1 (D_k^2) *be the delay of the* k*th packet in the system. If* $X_i^1 \geqq_{icx} X_i^2$ *and* $S_{i,j}^1 \geqq_{icx} S_{i,j}^2$ *for all* i, j, *then* $T_k^1 \geqq_{icx} T_k^2$ *for all* k. *Furthermore, if* $E[X_i^1] = E[X_i^2]$ *for all* i, *then* $D_k^1 \geqq_{icx} D_k^2$ *for all* k.

We shall discuss specific ramifications of Theorem 8 after the proof.

Proof. Without loss of generality, let the servers be numbered in increasing order from the root. We begin by coupling the networks so that the routing choices made are the same for both networks; that is, without loss of generality we may assume that for all i the ith packet has the same destination in each network. This assumption is valid because if

$$\{T_k^1 \mid d_1 = \alpha_1, d_2 = \alpha_2, \ldots, d_k = \alpha_k\} \geqq_{icx} \{T_k^2 \mid d_1 = \alpha_1, d_2 = \alpha_2, \ldots, d_k = \alpha_k\},$$

then it follows from Lemma 7 that $T_k^1 \geqq_{icx} T_k^2$, under the assumption that Q_1 and Q_2 have the same routing discipline. A similar statement is true for D_k^1 and D_k^2.

To prove that $T_k^1 \geqq_{icx} T_k^2$, from Lemma 5 it suffices to show that the departure time in both systems of the kth packet can be expressed as the same increasing convex function of a finite number of the X_i and the $S_{i,j}$. For convenience we drop the superscripts when the equation holds in both systems.

First, note that the service time of the kth packet cannot depend on any $S_{i,j}$ with $i > k$ or any of the X_i with $i > k$. Then the departure time depends only on finitely many of the random variables.

Now consider any of the random variables $S_{i,j}, X_i$ that T_k may depend on. Consider the exit time as a function of one of these random variables Z, with all the other random variables instantiated. It is clear that as Z increases the exit time can only increase, and that the exit time as a function of Z has the followng form: it consists of two piecewise linear segments, the first of which is constant (if the kth packet is not held up by an increase in Z) and the second of which has non-negative slope (if the kth packet is held up by an increase in Z). Hence the exit time is convex in Z.

More concretely, let $T_{i,j}^1$ and $T_{i,j}^2$ be the exit time of packet i from queue j in the two systems. We show by induction on i and j that $T_{i,j}^1 \geqq_{icx} T_{i,j}^2$. Let i' be the packet that completes service at queue j before i, and let j' be the queue which served i and i' before queue j. Both i' and j' are well defined, since the network is a tree and the routing decisions are the same for both systems. Then packet i begins service at queue j either after i' finishes service or as soon as i arrives, yielding the recurrence

$$T_{i,j} = \max(T_{i',j}, T_{i,j'}) + S_{i,j}.$$

Inductively, from Lemmas 5 and 6, one can show that $T_{i,j}^1 \geqq_{icx} T_{i,j}^2$. Note that $T_{i',j}$ and $T_{i,j'}$ are not independent; the induction shows that $T_{i,j}$ can be written as a function of the X_i and $S_{i,j}$ built up from max and addition operations. The base case for each queue corresponds to the first packet through the queue, which can be handled similarly.

A similar recurrence holds for the delays $D_{i,j}$, where $D_{i,j}$ is the delay of packet i up to the point of leaving j:

$$D_{i,j} = \max\left(D_{i',j} - \sum_{k=i'+1}^{i} X_k, D_{i,j'}\right) + S_{i,j}.$$

Hence, inductively, we find that $D_{i,j}$ is a convex increasing function in the $S_{i,j}$ and $-X_i$. This has an interesting interpretation: as the interarrival time X_i decreases, the delay increases; or, the sooner a packet arrives, the longer it has to wait.

Now, if $E[X_i^1] = E[X_i^2]$, then by Lemma 7, $-X_i^1 \geqq_{icx} -X_i^2$. Hence, with this additional hypothesis,

$$D_k^1 = T_k^1 - \sum_{i=1}^{k} X_i^1 \geqq_{icx} T_k^2 - \sum_{i=1}^{k} X_i^2 = D_k^2,$$

proving the theorem. □

Remark. In the similar proof by Niu for tandem queueing systems, he shows that the delays can be ordered according to the increasing convex ordering only when the arrival process is the same [13, Theorem 3]. For the case where the interarrival processes can be ordered, that is, $X_i^1 \geqq_{icx} X_i^2$, but the processes do not have the same distribution, he proves only a weaker statement comparing the expectations of the delays [13, Theorem 2]. Our result is stronger because we demonstrate that the delays are convex and increasing in $-X_i$. Niu's result appears to have led to a misconception that the delays cannot be ordered

in the variability ordering [23, p. 514]; our proof shows that the delays can be so ordered. □

We note that the proof of Theorem 8 cannot be extended in its current form to networks where packets may arrive from more than one entry point. For example, consider a node with external arrivals from outside the network and internal arrivals from other nodes in the network. The time of the first arrival at such a node would be the *minimum* of the first arrival time from outside the network and the first arrival time inside the network, and this is no longer a convex function of the appropriate random variables. In general, the expression for the time at which the ith packet enters such a node would not be expressible solely using addition and maximum functions, and thus the restricted network configuration is necessary. A more detailed counter-example is given in Sect. 6.

From Theorem 8 and Lemma 3, we derive a useful corollary.

Corollary 9 *In a rooted tree networks where the service times are NBUE (NWUE), the expected time a packet spends in the system is at most (at least) the expected time when the service times are exponentially distributed with the same mean. Furthermore, the same results hold for any convex increasing function of the delays.* □

Remark. Previously, the primary use of results of this type has been to show that, in equilibrium, the expected delay when arrivals are Poisson and the service times are exponentially distributed provides an upper bound for the case where arrivals are Poisson and service times are constant. This conclusion follows from Theorem 8 and Lemma 7, using the fact that the delays D_i converge to the delay of a packet in equilibrium. As the expected time a packet spends in the system in equilibrium given Poisson arrivals and exponential service times can generally be computed explicitly in Markovian networks (for example, see [6] or [23, Sect. 6.3]), this bounding technique can give useful upper bounds. Our result shows that these bounds can be generalized on Markovian rooted tree networks. For example, the upper bound also holds for the case where the service times are only NBUE and/or the interarrival process is not Poisson, but only NBUE; and the exponential/Poisson case is a lower bound when the service times and/or arrival process are NWUE. These techniques may also provide useful bounds in discrete time settings as well, through comparisons with the corresponding discrete time networks [10].

The fact that not just the expected delay, but any convex increasing function in the delay, can be similarly bounded is also important because in many settings, such as video or audio transmission, the variance in the delay can be as important as the delay itself. Also, bounding the higher moments can yield probabilistic bounds on the delay that are more useful than just the bounds on the expected delay. □

Using Theorem 8 and Lemma 4, we may conclude that constant service times are optimal for rooted tree networks.

Corollary 10 *Let the service times in a rooted tree network be $S_{i,j}$. Suppose also that we may vary the $S_{i,j}$ subject to keeping $E[S_{i,j}]$ fixed. Then the expected delay for each packet is minimized when $S_{i,j} = E[S_{i,j}]$, that is, when the $S_{i,j}$ are constant random variables. Similarly, the expected value of the maximum delay (or any increasing convex function of the delays) of the first k packets is minimized when the $S_{i,j}$ are constant random variables.* □

For rooted tree networks, this result strengthens previous results based on the techniques of Stamoulis and Tsitsiklis ([20] and [4]), which show that constant service times are better than exponential service times on Markovian networks. Our comparison applies to a wider range of service distributions, albeit on a restricted class of networks.

A counter-example developed by Harchol-Balter and Wolfe in [4] demonstrates that constant service times are not always optimal (in terms of minimizing the expected delay) in non-Markovian networks, even under the assumption of Poisson arrivals. Harchol-Balter and Wolfe raise the following question: for what class of networks does replacing servers with exponentially distributed service times by servers with constant service times reduce the expected time a packet spends in the system? We suggest an interesting refinement:

Question: For which networks (under a suitably general interarrival distribution) does replacing servers using constant service time with servers using any other service distribution with the same mean increase the expected time a packet spends in the system?

We have shown that rooted tree networks lie in this class.

4 Extensions

With some simple modifications, the proof of Theorem 8 can be extended to handle several other interesting cases. For example, we briefly explain how the proof can be extended in some cases of dependent interarrival times and dependent service times.

Corollary 11 *The result of Theorem 8 also holds in the case where $X_i^1 = X_i^2$ in distribution and the variables X_i are dependent.*

Proof. The proof of Theorem 8 is extended simply by coupling the arrival process as well as the destinations. That is, we prove that

$$\{T_k^1 \mid d_1 = \alpha_1, \ldots, d_k = \alpha_k, X_1 = x_1, \ldots, X_k = x_k\} \geqq_{icx} \{T_k^2 \mid d_1 = \alpha_1, \ldots, d_k = \alpha_k, X_1 = x_1, \ldots, X_k = x_k\}.$$

This suffices by Lemma 7. In this case, for any fixed values for the X_i, the $T_{i,j}$ are convex increasing functions of the $S_{i,j}$. The proof for the $D_{i,j}$ is similar. □

Corollary 12 *Suppose that the ith packet has a length l_i determined by a random variable L_i, where the L_i are independent. Also suppose that $S_{i,j} = \phi_j(l_i)$ for some increasing convex functions ϕ_j for all i. Then the results of Theorem 8 hold if we replace the hypothesis $S_{i,j}^1 \geqq_{icx} S_{i,j}^2$ for all i, j with $L_i^1 \geqq_{icx} L_i^2$.*

Proof. The proof of Theorem 8 carries over, except that $T_{i,j}$ and $D_{i,j}$ are now convex increasing functions in the L_i. □

Corollaries 11 and 12 demonstrate that the comparison can be useful even when realistic restrictions on the independence of certain variables apply. Corollary 12 seems particularly interesting, since it is often difficult to analyze systems where service times are dependent. Indeed, in practice the issue is often circumvented by assuming the service times at different queues are independent as an approximation; this idea appears to date back to Kleinrock [7].

Our results also extend easily to multi-casting networks.

Corollary 13 *The results of Theorem* 8, *Corollary* 9, *and Corollary* 10 *also hold for multi-casting rooted tree networks.*

Proof. The proof of Theorem 8 is easily modified to handle multi-casting rooted tree networks; one must couple the packet destinations, and then consider each duplicate of a packet separately. □

In the case where arrivals are Poisson and the packet destinations are given by a fixed distribution, this result can be used in a manner similar to the results of [20] to obtain an upper bound on the expected delay of packets in a multi-casting rooted tree system in many cases. Actually, one could also achieve this result by modifying the argument of [20], but for this class of networks our results are much more general.

5 Leaf edges

As we mentioned in the remark after Corollary 9, the case where packets take exponential time to cross an edge provides an upper bound on the expected delay for the case where packets take constant time to cross an edge, and this leads to a computable upper bound on the expected delay for Markovian rooted tree systems with Poisson arrivals.[3] Here we provide a small improvement to this approach by leaving the crossing times of some edges constant. In fact, we can apply this improvement to the bounds obtained using the techniques developed by Stamoulis and Tsitsiklis as well.

We begin by considering just Markovian rooted tree networks. In a single queue, if the external arrivals are Poisson and all service times are exponential, then the departures also form a Poisson process [6, Theorem 2.1]. Hence, if the arrivals in a Markovian rooted tree network are Poisson and the service times are exponential, the arrivals at and departures from each queue form a Poisson process as well. This is the fact we will use to improve the bounds.

Consider the queue edges connected to the leaves of the rooted tree; call these edges *leaf edges*. We compare the network Q_1 where all crossing times are constant to the network Q_2 where the crossing times for all non-leaf edges are exponentially distributed and the crossing times for all leaf edges are constant. We assume that for each edge the expected time to cross the edge is the same for all packets in both Q_1 and Q_2. By Theorem 8, the expected time a packet spends in Q_1 is bounded above by the expected time a packet spends in Q_2. (See Fig. 2.)

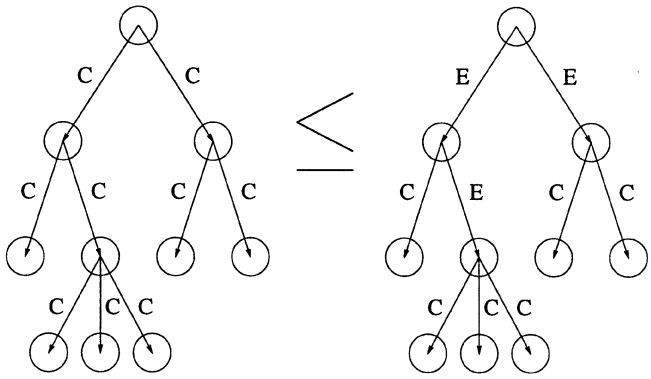

Fig. 2. Tree bounds: replace non-leaf edges with exponential servers, while leaf edges have constant servers

The advantage of leaving the time to cross the leaf edges constant is that the expected time of a packet in such a system in equilibrium can still be explicitly computed. This is because all the non-leaf edges correspond to queues with Poisson arrivals and exponential service times, while leaf edges correspond to queues with Poisson arrivals and constant service times; the expected time a packet spends in both types of systems can be determined by standard queueing theory. By the Pollaczek-Khinchin formula [8], the expected time $E[T]$ a packet spends in an M/G/1 queue (a queue with Poisson arrivals and service distribution S) in equilibrium is given by

$$E[T] = \frac{\lambda E[S^2]}{2(1-\lambda E[S])} + E[S]. \tag{1}$$

Hence, if the routing is Markovian, we can explicitly calculate the expected time in Q_2 queue by queue, and hence bound Q_1. Note by equation (1) that this bound is strictly better than the bound one would obtain if the crossing time for leaf edges were exponentially distributed. Also, this approach does not depend on the service time of leaf edges in Q_1 being constant; as before, they can be NBUE, and using the Pollaczek-Khinchin formula we can again compute a bound given the variance of the service distribution. Similarly, the bound also holds if the interarrival distribution is NBUE as well, and the approach can be used for any convex increasing function in the delay.

A natural question is whether this same improvement can be applied with the Stamoulis-Tsitsiklis technique that bounds all Markovian networks, not just rooted trees. For such networks we define a *leaf edge* to be any edge such that after crossing that edge, the packet must leave the system. In the standard Stamoulis-Tsitsiklis approach, all edges in the network are initially represented by constant time servers, and for the comparison they are replaced by Processor Sharing (PS) servers, which divide the total available service among all waiting packets equally. A delay argument then shows that the expected time a packet spends in the PS network is at least the

[3] In this section, we assume that for each edge, the crossing time for every packet has the same expectation.

expected time a packet spends in the constant server network. Assuming arrivals are Poisson and using standard results ([6, Theorem 3.7]), one has that the expected time in the PS network is the same as in a network where service times are exponentially distributed [4, 20]. This leads to computable bounds. We improve these bounds by replacing all non-leaf edges with PS servers, but leave all leaf edges with constant time servers. In this case as well, if the arrivals to the network are Poisson then in equilibrium the input stream to leaf edges will be Poisson [6, Theorem 3.7].

This leads to an improvement on the bounds for the expected delay of greedy routing on hypercube and butterfly networks given in [20]. (The technique also applies to the bounds on greedy routing on array networks given in [11], but these bounds have previously been improved by Kahale and Leighton [5].) The interested reader is referred to [20] for the full details of the underlying models; here we briefly describe the new bounds. For convenience we use the notation of [20] in this comparison. First consider hypercube networks of dimension d, where for each packet each dimension must be crossed with probability p (when $p = 1/2$, destinations are uniformly distributed), and dimensions are crossed in some fixed order (this corresponds to *greedy routing*). It take constant time 1 to cross an edge in any dimension. Packets are generated at each node as a Poisson process of rate λ, and the load ρ for each edge is thus $\rho = \lambda p$. The expected time a packet spends in the system in equilibrium is T.

Stamoulis and Tsitsiklis [20, Proposition 11] derive an upper bound of

$$T \leqq \frac{dp}{(1-\rho)}.$$

Here edges that cross the final dimension are leaf edges, and hence we can improve this bound to

$$T \leqq \frac{(d-1)p}{(1-\rho)} + p\left(1 + \frac{\rho}{2(1-\rho)}\right).$$

Note that in high traffic, that is in the limit as $\rho \to 1$, this improves the bound by a factor of $1 - \dfrac{1}{2d}$.

We can also improve the bounds for butterfly networks. The d-dimensional butterfly has $d+1$ levels and $(d+1)2^d$ nodes. Packets are generated at the first level for a destination in the $(d+1)$st level, with each node in the first level generating packets as Poisson process of rate λ. There are two types of edges between levels: *straight* and *vertical*. The probability p represents the probability a straight edge is taken by a packet crossing a level; otherwise the packet crosses the vertical edge. When $p = \frac{1}{2}$, the destinations are uniformly distributed. The load is $\rho = \lambda \max\{p, 1-p\}$.

Stamoulis and Tsitsiklis [20, Proposition 16] derive an upper bound of

$$T \leqq \frac{dp}{1-\lambda p} + \frac{d(1-p)}{1-\lambda(1-p)}.$$

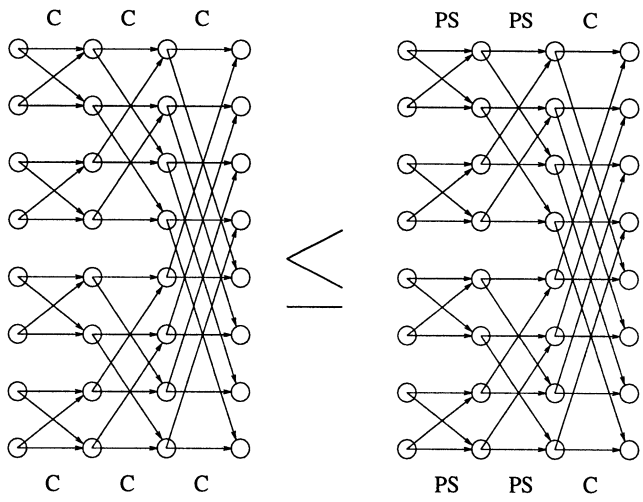

Fig. 3. Butterfly bounds: replace all but the last layer with PS servers

Here edges crossing the final level are leaf edges (see Fig. 3), and hence we can show

$$T \leqq \frac{(d-1)p}{1-\lambda p} + p\left(1 + \frac{\lambda p}{2(1-\lambda p)}\right) + \frac{(d-1)(1-p)}{1-\lambda(1-p)} + p\left(\frac{\lambda(1-p)}{2(1-\lambda(1-p))}\right).$$

Again, in the limit as $\rho \to 1$, this improves the bound by a factor of $1 - \dfrac{1}{2d}$.

Remark. Although we have suggested leaving just the crossing times for leaf edges constant to find upper bounds, it is possible in some networks that tighter bounds could be achieved by leaving more or other edges constant as well. For example, consider a *chain* of queue edges with constant crossing times, such that arrivals enter at the first queue and proceed through the entire chain of queues before exiting the system. (Note that packets may not leave the system before finishing the final stage.) The expected time a packet spends in a chain can be determined if arrivals are Poisson [3, 18], and hence if the network contains a chain of queue edges, one may achieve better bounds by leaving the crossing times for the edges of the chain fixed. □

6 The difficulty of interactions

In networks more complicated than rooted trees, delay in one area may prove beneficial by preventing congestion and allowing packets from other areas to get through. The rooted tree networks prove simple to compare because the interaction between packets is limited: if a packet is delayed, it can only delay other packets.

A simple example with minimal interaction between streams of packets demonstrates how important this lack of interaction is to the generality of our result. Consider a system with three edges, as given in Fig. 4, with packets a and b arriving at nodes A and B respectively at time 0,

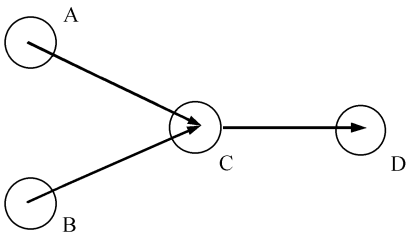

Fig. 4. A simple network with interference

and both exiting at node D. For convenience we consider only two possibilities: either it takes constant time 1 to cross edges AC and BC, or the time to cross is either 0 or 2, decided by a fair coin toss. The time to cross edge CD is always 1. Packets arriving at the same time at node C are ordered by a fair coin toss as well. A simple calculation shows that the expected time $E[T]$ before each packet leaves is then:

$$E[T] = \begin{cases} 2.5 & \text{if } a \text{ and } b \text{ both take time 1 on } AC,\ BC \\ 2 & \text{if one takes time 1 and the other does not} \\ 2.25 & \text{if } a \text{ and } b \text{ both take time 0 or 2 randomly.} \end{cases}$$

Note that in this case the worst choice is to use constant time per edge, because the packets will then interfere at C. The example can be extended to a dynamic variation by having packets arrive at A and B, say, every three time units.

The example suggests thinking of such systems as a multi-player game, where each player is a packet that can control the distribution of its service time while keeping the mean fixed. It is interesting that in this natural two-player example an optimal solution requires the players to adopt different strategies. In a rooted tree network, the optimal game strategy is trivial, as the stochastic comparison technique we have described has shown.

It is also important in our results that the route of a packet be independent of the state of the system. Indeed, Ross provides a simple counter-example based on a system of two queues where entering packets are served at one of the two queues and exit the system [16]. Suppose that packets arrive in batches of three, batches are separated by a suitably long interval of time, and each packet proceeds to the queue with fewer packets already waiting (ties decided arbitrarily). We compare the following two systems: in the first system all services take constant time 1, while in the second system all services take time 0 or 2, decided by a fair coin flip. Then the expected time of the third packet in each batch is greater in the first system, although the service times are less variable. One can easily generalize this counter-example to the case where service times are either constant or exponentially distributed and arrivals form a renewal process. The counter-example does not apply, however, if the arrival process is Poisson; moreover, there is some evidence that in the case of Poisson arrivals, in systems where packets go to the shortest queue, constant service times are better than exponential service times [12, Chapter 4]. This special case is an interesting open question.

7 Conclusion

We have applied a stochastic ordering relation in order to understand how service time distribution affects the expected time a packet spends in a packet routing network. Our method provides surprisingly general results on rooted tree networks; for example, unlike previous comparison methods, it does not depend on a Poisson arrival process. Insight gained from the case of tree networks also leads to improved bounds on the hypercube and butterfly using the analysis of Stamoulis and Tsitsiklis.

We remain hopeful that these or similar methods may be applicable to a wider class of networks. This may require applying further alternative stochastic ordering relations less stringent than the standard notion of stochastic domination and more complex comparison methods.

Acknowledgments. The author would like to thank Micah Adler, John Byers, Nabil Kahale, Michael Luby, Ketan Patel, and Alistair Sinclair for helpful discussions.

References

1. Baccelli F, Massey WA, Towsley D: Acyclic fork-join queueing networks. J Assoc Comput Mach 36: 615–642 (1989)
2. Borodin A, Kleinberg J, Raghavan P, Sudan M, Williamson D: Adversarial queueing theory. Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, pp 376–385 (1996)
3. Friedman HD: Reduction methods for tandem queueing systems. Oper Res 13: 121–131 (1965)
4. Harchol-Balter M, Wolfe D: Bounding delays in packet-routing networks. In: Proceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing, pp 248–257 (1995)
5. Kahale N, Leighton T: Greedy dynamic routing on arrays. In: Proceedings of the Sixth Annual ACM/SIAM Symposium on Discrete Algorithms, pp 558–566 (1995)
6. Kelly FP: Reversibility and stochastic networks. Wiley, New York 1979
7. Kleinrock L: Communication nets. McGraw-Hill, New York 1964
8. Kleinrock L: Queueing systems, Vol II. Computer applications. Wiley New York 1976
9. Leighton FT: Average case analysis of greedy routing algorithms on arrays. In: Proceedings of the Second Annual ACM Symposium on Parallel Algorithms and Architectures, pp 2–10 (1990)
10. Marsan MA: On some discrete time queueing systems. Alta Frequenza 49: 285–292 (1980)
11. Mitzenmacher M: Bounds on the greedy routing algorithm for array networks. In: Proceedings of the Sixth Annual ACM Symposium on Parallel Algorithms and Architectures, pp 248–259, (1994) (to appear in: J Comput Syst Sci)
12. Mitzenmacher M: The power of two choices in randomized load balancing. PhD thesis, University of California, Berkeley (1996)
13. Niu S. C: On the comparison of waiting times in tandem queues. J Appl Probab 18: 707–714 (1981)
14. Righter R, Shanthikumar J: Extremal properties of the FIFO discipline in queueing networks. J Appl Probab 29: 967–978 (1992)
15. Rolski T, Stoyan D: On the comparison of waiting times in GI/G/1 queues. Oper Res 24: 197–200 (1976)
16. Ross SM: Average delay in queues with non-stationary Poisson arrivals. J Appl Probab 15: 602–609 (1978)
17. Ross SM: Stochastic models. Wiley, New York 1983
18. Rubin I: Communication networks: Message path delays. IEEE Trans Profess Group Inform Theory 20: 738–745 (1974)

19. Shaked M, Shantikumar J: Stochastic orders and their applications. Academic Press, New York 1994
20. Stamoulis GD, Tsitsiklis JN: The efficiency of greedy routing in hypercubes and butterflies. IEEE Trans Commun 42(11): 3051–3061 (1994) [An early version appeared in: Proceedings of the Second Annual ACM Symposium on Parallel Algorithms and Architectures, pp 248–259 (1991)]
21. Stoyan D: Comparison method for queues and other stochastic models. Wiley, New York 1983
22. Szekli R: Stochastic ordering and dependence in applied probability. Springer, Berlin Heidelberg New York 1995
23. Wolff R: Stochastic modeling and the theory of queues. Prentice-Hall, New Jersey, 1989

Michael Mitzenmacher received his B.A. in mathematics and computer science from Harvard University, a Certificate of Advanced Study in mathematics from Cambridge University, and his Ph.D. in computer science from the University of California at Berkeley. He joined Digital Systems Research Center in 1997. His research interests include randomized algorithms, on-line algorithms, communication networks, and queueing networks.