# Fast Evaluation of Ensemble Transients of Large IP Networks [*]

*Catalin T. Popescu* *A. Udaya Shankar*
`cpopescu@cs.umd.edu` `shankar@cs.umd.edu`
Department of Computer Science
University of Maryland, College Park

CS-TR-3903
May 11, 1998

**Abstract**

We extend a numerical approximate solution method (the Z-iteration) to time-dependent open networks of M(t)/M(t)/1/$\infty$ and M(t)/M(t)/1/K queues, and apply the method to obtain transient performance metrics of large IP networks. The method generates a set of coupled differential equations, one for each queue in the network. The equations are numerically unstable under certain conditions (e.g., large bandwidths and buffers), and we present techniques to overcome this problem. The resulting numerical procedure is accurate and very fast. For example, a 20-second evolution for a 1000-node network with high-speed links ($\approx 10^4$packets/sec) and large buffers ($\approx 10^4$packets) was obtained in 18 minutes on an Ultra Sparc, whereas simulation would take days.

## 1 Introduction

Evaluating the performance of large IP networks is an important problem, but an adequate technique does not currently exist. Analytical approaches cannot handle many critical network features. Packet-level simulation is popular but it is extremely slow when bandwidths are high and buffers are large, even for networks of moderate size. Flow-level simulation is faster but its accuracy is questionable [because it makes the unrealistic assumption that the interval between successive changes in the network-wide flow pattern is large enough so that steady state holds for most of the interval].

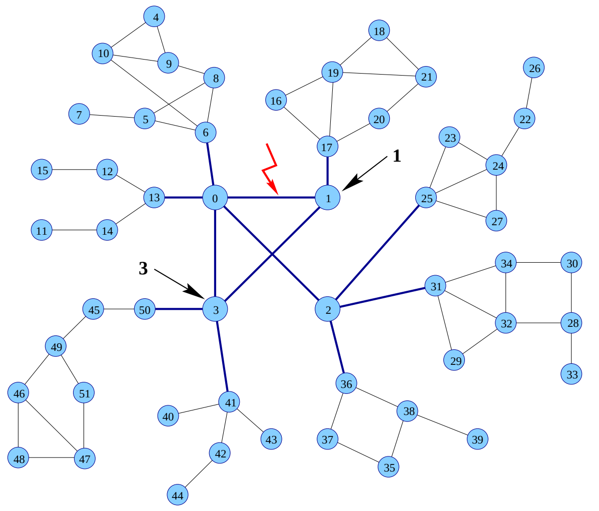
We take an alternative approach, based on transient metrics. We are interested in observing the behavior of a network when a perturbation is applied, such as a node/link failure, load change, routing change, etc. We would like to know, for example, which links become congested and how quickly, how long an overload can be maintained before the blocking probability exceeds a specific value, and such like.

Time-dependent queuing systems are a natural modeling tool for investigating such issues. Their arrival and service rates can vary over time to reflect external or feedback control, and solving them yields transient responses to network and workload changes. The difficulty, of course, is that these systems are not tractable by traditional approaches. Analytical and simulation approaches fail for the same reasons mentioned above. Numerical solutions are unmanageable for realistic networks because the state space becomes enormous.

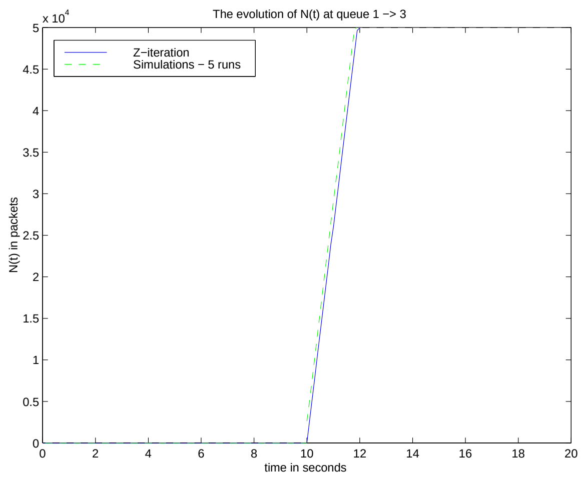
[*]This work is supported partially by ARPA contract number DABT6396C0075 and DoD contract number MDA90497C3015 to the University of Maryland. It should not be interpreted as representing the opinions or views of ARPA, DoD, or the U.S. Government.

The Z-iteration is an attempt to address this problem. It is a numerical approximate solution method that yields the time evolution of ensemble metrics (e.g. instantaneous queue size distribution) at a cost several orders cheaper than simulation. The first version of the Z-iteration [3, 4] handled single multi-class multi-resource queues, adequate for analyzing connection-oriented networks with strict access control and resource reservation. In this paper, we develop another formulation that also handles networks of time-dependent queues, appropriate for modeling classical datagram networks.

Figure 1 illustrates an application of this Z-iteration. The network in Figure 1(a) has 52 nodes, 66 links, and 457 end-to-end user classes subject to shortest-hop routing. At time $t = 10$ sec, the link marked by the arrow fails and routes are updated. Figure 1(b) shows the evolution of the instantaneous average queue size at the link from nodes 1 to 3, obtained from Z-iteration and from simulation. The Z-iteration took 18 seconds, compared with 2.5 hours for packet-level simulation (averaging over 5 runs).



(a)



(b)

Figure 1: A network and example evolution

The rest of the paper is organized as follows. Section 2 describes how we model a datagram network by a network of time-dependent queues. Section 3 describes the Z-iteration for single queues and for networks of queues. Section 4 examines numerical stability issues. Examples are presented in Section 5. We conclude in Section 6.

## 2 Network Model

We consider a datagram network with some routing algorithm. The links have fixed bandwidths and the routers allocate fixed-size buffers for each outgoing link. The traffic consists of end-to-end *user classes*, each defined by source node, destination node, and time-dependent packet generation rate. We assume fixed-size packets. We focus on networks with high-speed links ($\approx 10^4$packets/sec) and large link buffers ($\approx 10^4$packets), similar to the IP networks of today.

We model each outgoing link of a node by a M/M/1/K queue with service rate equal to the link bandwidth and max queue size K equal to the link buffer space. The routing between the queues is determined by the network topology, user classes, and routing tables.

To illustrate, suppose node A has an outgoing link A1 to node B, and node B has outgoing links, B1, B2, and B3. Then in the queueing model, the output of A1 can go to B1, B2, B3, or be absorbed within node B, as shown in Figure 2. The probability $r_{\mathrm{A1,B1}}$ of going to B1 is given by the fraction of the total user class rates in A1 that are forwarded to B1 (note that the path of a user class is determined by the routing tables). For example, if the traffic that arrives at queue A1 consists of two classes, one with rate 5000 pkts/sec and another with rate 3000 pkts/sec, and node B routes the first class to B1 and the second class to B2, than the routing probability for B1 is $5000/(5000+3000) = 5/8$, for B2 is $3000/(5000+3000) = 3/8$, and for B3 is 0.

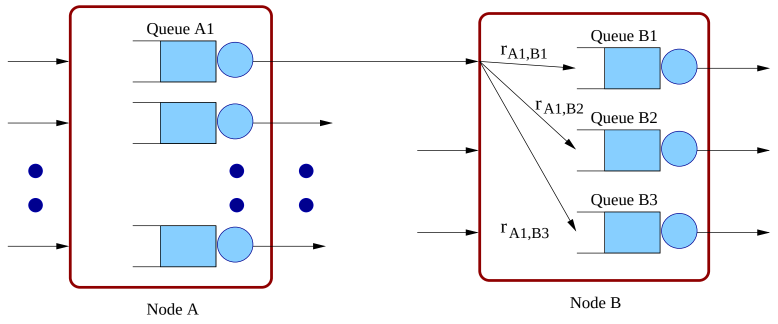


Figure 2: The queue model for a link from node A to node B.

## 3 Z-iteration

The Z-iteration is an efficient numerical approximation method that computes instantaneous ensemble metrics of time-dependent queuing systems. It is based on functional approximations of relationships between instantaneous metrics by the corresponding steady-state relationships. These approximations allow the evolution of the metrics to be defined by a small number of differential equations, rather than the large number of Chapmann-Kolmogorov equations (which are as many as the maximum queue size).

Table 1 gives the notation we use for a queue. Instantaneous parameters refer to a queue with time-varying arrival and service rates. Steady-state parameters refer to a queue in steady state, with constant arrival and service rates.

### 3.1 Old Version

The first version of the Z-iteration was based on the following flow equation, obtainable from the Chapmann-Kolmogorov equations:

$$\frac{dN(t)}{dt} = \lambda(t)[1 - B(t)] - \mu(t)U(t) \tag{1}$$

The idea is to express $B(t)$ and $U(t)$ in terms of $N(t)$, resulting in a single scalar differential equation for $N(t)$. It turns out that the relationship between $B(t)$ and $N(t)$ is very well approximated by the relationship between steady-state $B$ and steady-state $N$. The same holds for the relationship between $U(t)$ and $N(t)$.

Thus we want functions expressing $B$ and $U$ in terms of $N$. What is available, however, are functions expressing $N$, $B$, and $U$ in terms of the steady-state traffic intensity $\rho$ ($= \lambda/\mu$). We denote these functions by $F_{\mathrm{N}}(\cdot)$, $F_{\mathrm{B}}(\cdot)$, and $F_{\mathrm{U}}(\cdot)$, respectively. For example, for a M/M/1/$\infty$ queue we have [2]

$$N = F_{\mathrm{N}}(\rho) = \frac{\rho}{1-\rho}$$

| | |
|---|---|
| $\lambda(t)$ | instantaneous arrival rate at time $t$ |
| $\mu(t)$ | instantaneous service rate at time $t$ |
| $N(t)$ | instantantaneous average queue size at time $t$ |
| $B(t)$ | instantaneous blocking probability at time $t$ |
| $U(t)$ | instantaneous utilization at time $t$ |
| $z(t)$ | instantaneous virtual traffic intensity at time $t$ $(\neq \lambda(t)/\mu(t))$ |
| $\lambda$ | steady-state arrival rate |
| $\mu$ | steady-state service rate |
| $\rho$ | steady-state traffic intensity $(= \lambda/\mu)$ |
| $N$ | steady-state average number of customers |
| $B$ | steady-state blocking probability |
| $U$ | steady-state utilization |
| $F_{\mathrm{N}}(\cdot)$ | function yielding $N$ in terms of $\rho$ |
| $F_{\mathrm{B}}(\cdot)$ | function yielding $B$ in terms of $\rho$ |
| $F_{\mathrm{U}}(\cdot)$ | function yielding $U$ in terms of $\rho$ |

Table 1: Notation

$$\begin{aligned} B &= F_{\mathrm{B}}(\rho) = 0 \\ U &= F_{\mathrm{U}}(\rho) = \rho \end{aligned} \tag{2}$$

For a $M/M/1/K$ queue, we have [2]:

$$\begin{aligned} N &= F_{\mathrm{N}}(\rho) = \frac{\rho}{1-\rho} - \frac{(K+1)\rho^{(K+1)}}{1-\rho^{(K+1)}} \\ B &= F_{\mathrm{B}}(\rho) = \frac{1-\rho}{1-\rho^{(K+1)}}\,\rho^K \\ U &= F_{\mathrm{U}}(\rho) = \frac{1-\rho^K}{1-\rho^{(K+1)}}\,\rho \end{aligned} \tag{3}$$

For M/M/1/$\infty$, we can invert $F_{\mathrm{N}}(\rho)$ and so obtain $B$ and $U$ in terms of $N$, specifically, $B = 0$ and $U = N/(N+1)$. But in general, including the case of blocking queues, we cannot invert $F_{\mathrm{N}}(\rho)$ analytically. So instead we inverted numerically, using another approximation as follows:

- $U$ is computed from $N$ *assuming a non-blocking system.*
- $B$ is computed as the fixed point of $B = F_{\mathrm{B}}(\rho)$ and $\rho = U/(1-B)$ (obtained by equating the inflow $\lambda(1-B)$ to the outflow $\mu U$). The resulting value of $\rho$ is simply the steady-state traffic intensity value consistent with $B$ and $N$.

This approach works very well for M(t)/M(t)/1/$\infty$ and M(t)/M(t)/K/K queues, and for M(t)/M(t)/1/K queues when $\lambda(t) < \mu(t)$. But it does not work for networks of these queues or for M/M/1/K queues when $\lambda(t) > \mu(t)$.

## 3.2 New Version

We now develop a formulation of the Z-iteration that eliminates the non-blocking assumption used in the numerical inversion operation above. This version works for open networks of M(t)/M(t)/1/K and M(t)/M(t)/1/$\infty$ queues.

As mentioned above, formulas for $N$, $B$ and $U$ are usually in terms of $\rho$. This suggests that we introduce an instantaneous version of $\rho$, which we refer to as the *instantaneous virtual traffic* *intensity*, denoted by $z(t)$, and develop a differential equation for $z(t)$ rather than for $N(t)$. Then $N(t)$, $U(t)$, and $B(t)$ can be approximated by

$$\begin{aligned} N(t) &= F_{\mathrm{N}}(z(t)) \\ B(t) &= F_{\mathrm{B}}(z(t)) \\ U(t) &= F_{\mathrm{U}}(z(t)) \end{aligned} \tag{4}$$

Although $z(t)$ is fictitious, it has a natural interpretation: at any time $t$, it is the amount of traffic intensity that if applied constantly would result in steady-state $N$, $B$, and $U$ equal to $N(t)$, $B(t)$, and $U(t)$, repsectively. In fact, $z(t)$ is just a more accurate version of the iterate $\rho$ that appears in the numerical fixed-point inversion in the old version. *Note that $z(t)$ is not equal to $\lambda(t)/\mu(t)$.*

To obtain a differential equation for $z(t)$, we start with the differential equation for $N(t)$

$$\frac{dN(t)}{dt} = \lambda(t)[1 - B(t)] - \mu(t)\ U(t);$$

Replacing $dN(t)/dt$ by $(dN(t)/dz(t))(dz(t)/dt)$, $dN(t)/dz(t)$ by $dF_{\mathrm{N}}(z)/dz$, $B(t)$ by $F_{\mathrm{B}}(z(t))$, and $U(t)$ by $F_{\mathrm{U}}(z(t))$, we obtain

$$\frac{dz(t)}{dt} = \frac{1}{dF_{\mathrm{N}}(z)/dz}[\lambda(t)(1 - F_{\mathrm{B}}(z(t))) - \mu(t)F_{\mathrm{U}}(z(t))] \tag{5}$$

Thus we have a scalar differential equation whose solution yields the evolution of $z(t)$. Plugging $z(t)$ into equation (4) yields evolutions of $N(t)$, $U(t)$, and $B(t)$.

Equation (5) can be instantiated for any type of M/M/$\cdot$ queue. For a M/M/1/$\infty$ queue, we obtain

$$\frac{dz(t)}{dt} = (1 - z(t))^2(\lambda(t) - \mu(t)z(t)) \tag{6}$$

For a M/M/1/K queue, we obtain

$$\frac{dz(t)}{dt} = \frac{(1 - z(t)^{(K+1)})(1 - z(t)^K)(1 - z(t))^2}{(1 - z(t)^{(K+1)})^2 - (K+1)^2 z(t)^K (1 - z(t))^2}(\lambda(t) - \mu(t)z(t)) \tag{7}$$

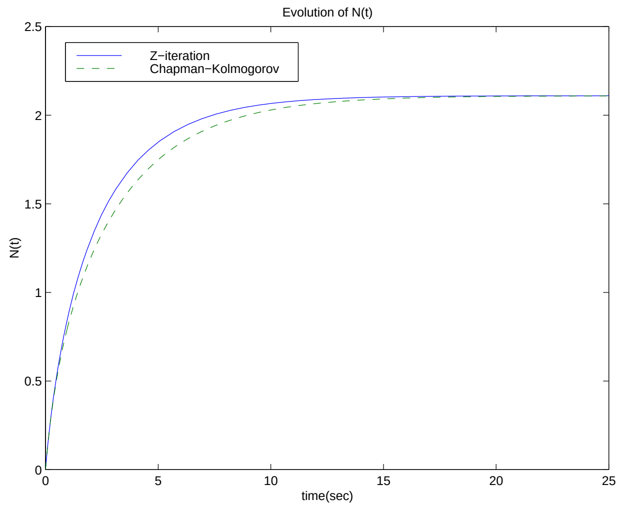
**Example (M/M/1/K)** We apply the method to M/M/1/K queues driven by constant $\lambda$ and $\mu$. The queues are initially empty. The accuracy of the results is demonstrated by comparing against direct solution of the Chapman-Kolmogorov equations. Figure 3 has plots of $N(t)$ and $U(t)$ for $\lambda = 1.5$, $\mu = 2.0$ and $K = 7$. Figure 4 shows $N(t)$ and $B(t)$ for $\lambda = 2.0$, $\mu = 1.5$ and $K = 10$. ∎

## 3.3 Networks of Queues

We extend the Z-iteration to networks of M(t)/M(t)/1/$\infty$ and M(t)/M(t)/1/K queues. Here, a departure from queue $i$ is routed to queue $j$ with a time-dependent probability $r_{ij}(t)$, and leaves the network with probability $1 - \sum_j r_{ij}(t)$. The arrivals to queue $i$ consist of external arrivals $\lambda_i(t)$ (from outside the network) and departures from queues in the network routed to queue $i$. The total arrival rate of queue $i$, denoted $\lambda_i^*(t)$, is given by

$$\lambda_i^*(t) = \lambda_i(t) + \sum_{j=1}^{n} r_{ji}(t)\ \mu_j(t)\ U_j(t). \tag{8}$$

Any arriving customer can be blocked, except, of course, customers feeding back from queue $i$. The differential equation for $z_i(t)$, the instantaneous virtual traffic intensity of queue $i$, is



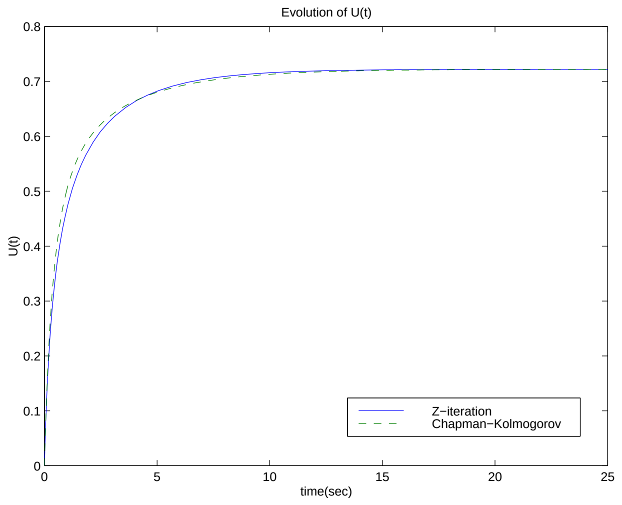


Figure 3: Results for M/M/1/K queue with $\lambda = 1.5$, $\mu = 2.0$, $K = 7$

obtained by appropriately modifying equation (6) or (7). If queue $i$ is a M/M/1/$\infty$ queue, we have

$$\frac{dz_i(t)}{dt} = (1 - z_i(t))^2[\lambda_i(t) + \sum_{j=1;j\neq i}^{n} r_{ji}(t)\mu_j(t)F_{\mathrm{U}_j}(z_j(t)) - \mu_i(t)(1 - r_{ii}(t))z_i(t)] \tag{9}$$

If queue $i$ is a M/M/1/K queue, we have

$$\begin{aligned}\frac{dz_i}{dt} &= \frac{(1 - z_i^{(K_i+1)})(1 - z_i^{K_i})(1 - z_i)^2}{(1 - z_i^{(K_i+1)})^2 - (K_i + 1)^2 z_i^{K_i}(1 - z_i)^2} \\ &\quad \times [\lambda_i(t) + (\sum_{j=1;j\neq i}^{n} r_{ji}\mu_j(t)F_{\mathrm{U}_j}(z_j)) - \mu_i(t)(1 - r_{ii})z_i]\end{aligned} \tag{10}$$

**Example (network of M/M/1/K queues)** We apply the method to networks of M/M/1/K queues driven by constant $\lambda$ and $\mu$. The networks are initially empty. The accuracy of the results is demonstrated by comparing against simulation. The first example is the network shown in Figure 5(a). Evolutions of instantaneous average queue size for the two queues are shown in Figure 5(b) and (c). A second example, dealing with a tandem network with feedback, is shown in Figure 6. ∎

From experimentation we find that the method works very well for open queueing networks but fails for closed queueing networks. To work well with closed queueing networks, we need the standard normalization constant (for computing $B$, $U$ and $N$).

## 4 Numerical Issues

Recall that we model IP networks by networks of M/M/1/K queues. We are interested in IP networks with large link speeds ($\approx 10^4$packets/sec), large link buffers ($\approx 10^4$packets), and high utilizations, which translates into large values for K, $\lambda$, and $\mu$. This makes the system of differential equations for $z_i(t)$ *extremely stiff*, giving rise to problems of numerical stability and

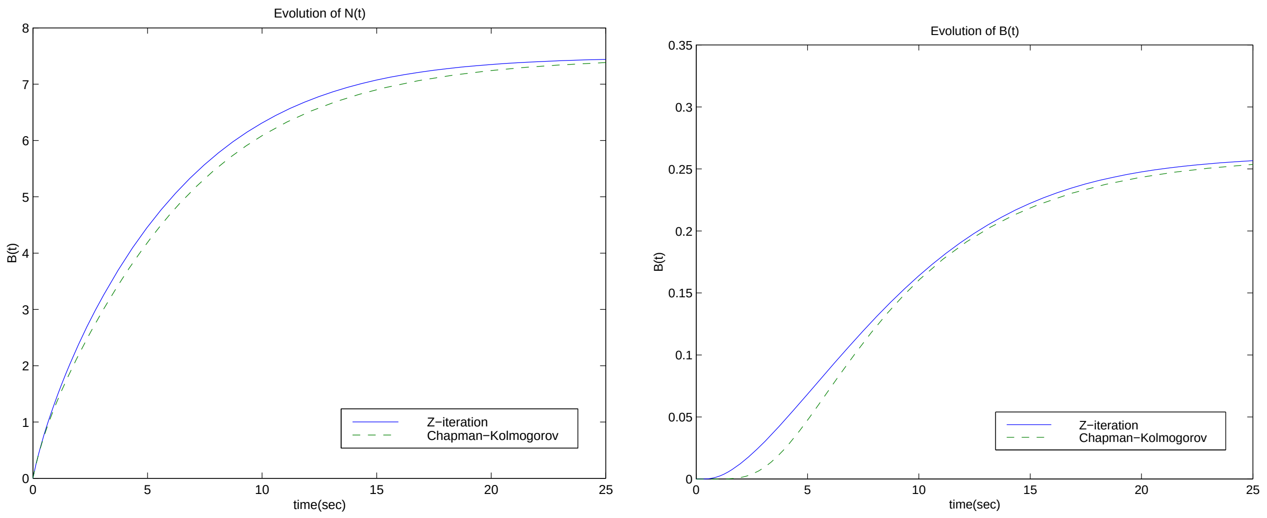


Figure 4: Results for M/M/1/K queue with $\lambda = 2.0$, $\mu = 1.5$, $K = 10$

convergence. Applying regular solving methods to the differential equations as stated above will produce an incorrect answer or no answer at all, and applying stiff equation solvers will produce a correct answer but *extremely slowly* (for a good coverage of differential equation solvers, see [5]).

In this section, we describe how to overcome these problems for networks of M(t)/M(t)/1/K queues. Consider the differential equation (10). Observe that for high $K$ ($\approx 10^3$), we have $z^K \approx 0$ for $z < 1-\epsilon$ and $z^K \approx \infty$ for $z > 1+\epsilon$, for some $\epsilon$. Using this, equation (10) becomes

$$\frac{dz_i}{dt} = \begin{cases} (1-z_i)^2[\ \lambda_i(t) + (\ \sum_{j=1;j\neq i}^{n} r_{ji}\mu_j \min(z_j,1)) - \mu_i(t)(1-r_{ii})z_i\ ] & \text{for } z_i < 1-\epsilon \\ \frac{(1-z_i)^2}{z_i}[\ \lambda_i(t) + (\sum_{j=1;j\neq i}^{n} r_{ji}\mu_j \min(z_j,1)) - \mu_i(t)(1-r_{ii})z_i\ ] & \text{for } z_i > 1+\epsilon \\ \text{as in equation (10)} & \text{otherwise} \end{cases} \tag{11}$$

The formulas for $N$, $B$ and $U$ are similarly modified. For example, for $N$ we have:

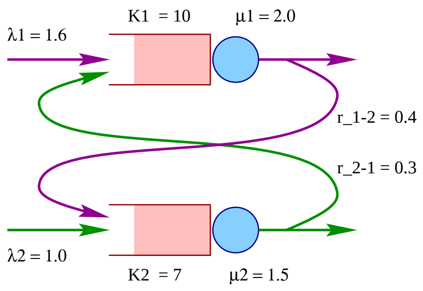
$$N_i = \begin{cases} \frac{z_i}{1-z_i} & \text{for } z_i < 1-\epsilon \\ K_i + \frac{z_i}{1-z_i} & \text{for } z_i > 1+\epsilon \\ \text{as in equation (3)} & \text{otherwise} \end{cases} \tag{12}$$

These modifications are not sufficient. We need to stop using equation (10) around $z_i = 1.00$. To do so, we first find an $\epsilon$ for which $N$, computed using $z = 1-\epsilon$ and (12) is equalt to the $N$ computed using $z = 1+\epsilon$ and (12). That is,

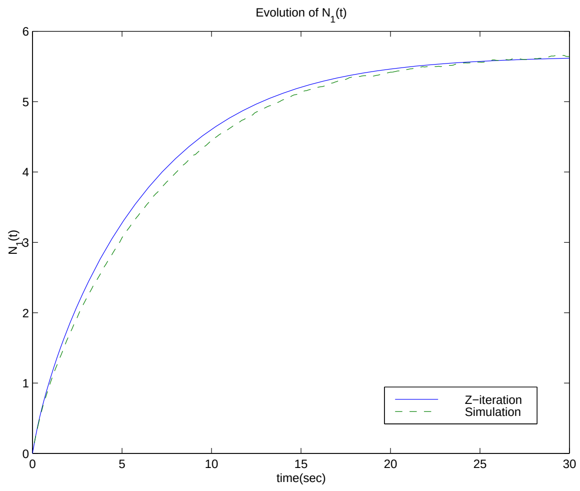
$$K + \frac{1+\epsilon}{1-(1+\epsilon)} = \frac{1-\epsilon}{1-(1-\epsilon)} \tag{13}$$

which yields $\epsilon = 2/K$. So we make the computation jump over the interval $[1-\epsilon,\ 1+\epsilon]$ as follows: whenever $1 - 2/K_i < z_i < 1$ would hold, we set $z_i = 1 + 2/K_i$; whenever $1 < z_i < 1 + 2/K_i$ would hold, we set $z_i = 1 - 2/K_i$. Outside this interval, we continue using equation (11).

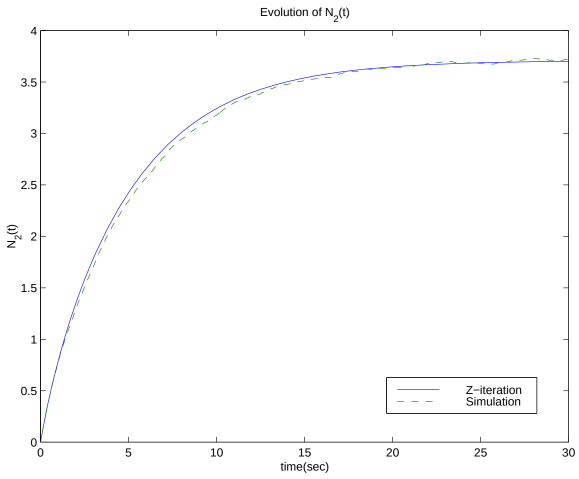
This is still not good enough. For large $\lambda_i^*$ and $\mu_i$, we get a lot of large fluctuations when $z_i(t)$ comes close to the value $\lambda_i^*(t)/\mu_i(t)$. This is because $dz_i(t)/dt$ becomes highly negative (positive) when $z_i(t)$ is slightly higher (lower) than $\lambda_i^*(t)/\mu_i(t)$. To overcome this problem, we exploit a monotonicity property.



(a)



(b)



(c)

Figure 5: A simple network and the results obtained for it.

Consider the evolution of $z_i$. At any moment $t$, $z_i$ tends to evolve monotonically (increasing or decreasing) from its current value to $\lambda_i^*(t)/\mu_i(t)$. So we introduce a "slope" flag which indicates the sign of the slope of $z_i$, i.e., positive when $z_i$ is increasing and negative when $z_i$ is decreasing. The initial value of the slope flag is determined by the sign of $dz_i(0.00)/dt$. While $\lambda^*(t)/\mu(t)$ does not change, the flag does not change. At each time step in the numerical solution, if we obtain a negative $dz_i(t)/dt$ and the slope flag is positive, we do the following: if $\lambda_i^*(t)/\mu_i(t)$ has decreased from its value at the previous time step, set the slope flag to negative, otherwise set $dz_i(t)/dt$ to 0. We proceed in a similar way if we obtain a positive $dz_i(t)/dt$ and the slope flag is negative.

This strategy is also used when modifying $z_i$ around 1.00. whenever $1-2/K_i < z_i < 1$ would hold, we set $z_i = 1+2/K_i$ only if the slope flag is plus; whenever $1 < z_i < 1+2/K_i$ would hold, we set $z_i = 1-2/K_i$ only if the slope flag is negative.

Finally, some implementation issues. We use the standard Runge-Kutta method, of fixed order, choosing a proper step size (around $10^{-2}$). We take care that $step \cdot max_i z_i < max \Delta z$ holds, so that the step can be smaller when some $dz_i/dt$ is large. The modifications of the slope flags and the jump over 1.00 are handled outside the inner Runge-Kutta loop.

The combination of all these techniques yields a very fast and accurate solver (see run times in following section).

## 5 Evolutions of IP Networks

This section describes evolutions for two IP networks, one of 52 nodes and one of 1000 nodes. We used the Transit-Stub model in the GT-ITM (Georgia Tech Internetwork Topology Models) [1] to generate the network topoplogies.

The network of 52 nodes is displayed in Figure 7. Figures 8 and 9 show the evolution of the instantaneous queue size at various links. Each figure has two curves, one obtained by the Z-iteration and one by averaging 5 simulation runs. The Z-iteration took 18 seconds of computation time on an Ultra Sparc, whereas each simulation run took 30 minutes for a total of 2.5 hours for the 5 simulations.

The plots for links 1-17 and 3-1 in the figures clearly show that many more simulations need to be averaged to obtain decent confidence intervals, probably around 20 simulations which would require about 10 hours (as opposed to the Z-iteration's 18 seconds). Even though the simulations in the other plots in the figures appear to be of adequate confidence, this is a misleading conclusion resulting from the large scaling in these plots. For instance, plots of the blocking probabilities on these links would show variation similar to that in the plots for links 1-17 and 3-1.

The 1000-node network is "shown" in Figure 10, with certain parts relevant to the evolutions highlighted. The subsequent figures show the evolution of the instantaneous queue size at various links, obtained by the Z-iteration. It required about 18 minutes of computation time. We did not attempt a simulation of this as that would have taken far too long (at least many days for one run).

## 6 Conclusions and Future Work

Queuing systems are a natural way of modeling computer networks and many other systems. One usually obtains steady-state metrics of queueing models, because this is relatively tractable for many kinds of queueing systems. But steady-state metrics do not offer answers to many interesting questions. Transient evolutions, on the other hand, can provide answers to most questions that one would like to ask in evaluating a system. They also allow for realistic modeling of critical events such as network routing updates, unlike steady-state models.

Transient metrics are usually very hard to obtain, unmanageable by analytical methods and time-consuming by simulation. But the Z-iteration changes this premise, allowing very fast computation of some very useful transient metrics. It translates a queuing network with $N$ nodes into a system of $N$ coupled differential equations. Looking at the results and the run time, we conclude that it offers the power of simulation at a fraction of cost for these transient metrics.

One area of future work is to extend the results here to multi-class queueing networks. These would be needed for modeling QoS datagram networks (incorporating buffer and bandwidth reservation), like the next generation of IP networks.

Another area of future work is to develop a performance evaluation procedure. Using the Z-iteration, one can obtain evolutions of ensemble transient metrics. But this by itself does not amount to evaluation. We need to develop a collection of scenarios of adequate coverage for a given network, obtain their evolutions and resulting metrics, interpret the data, and reach conclusions.

## References

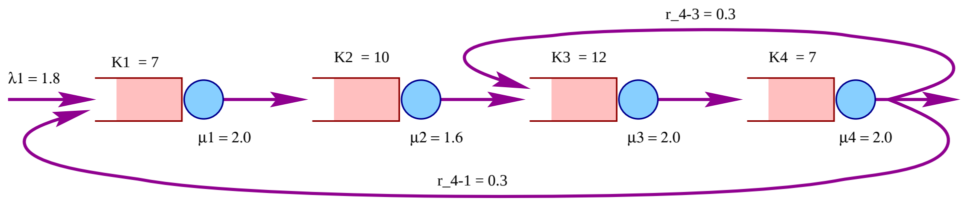
[1] Ken Calvert Ellen W. Zegura and S. Bhattacharjee. How to Model an Internetwork. In *IEEE Infocom '96*, San Francisco, CA, 1996.

[2] L. Kleinrock. *Queueing Systems*, volume I and II. New York: Wiley, 1976.

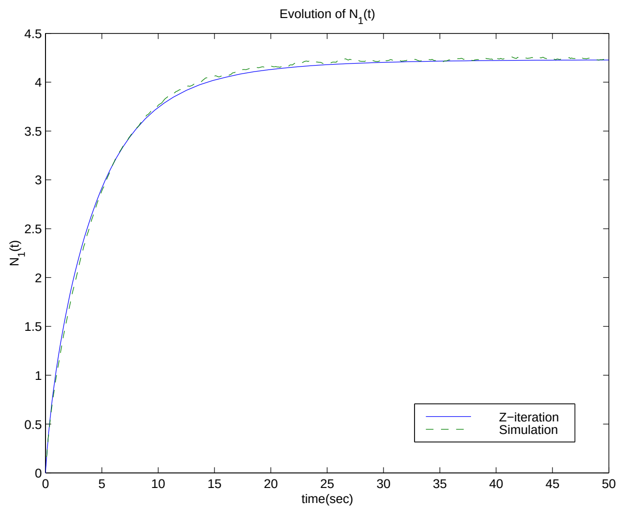
[3] I. Matta and A.U. Shankar. Z-Iteration: A Simple Method for Throughput Estimation in Time-Dependent Multi-Class Systems. In *ACM SIGMETRICS/PERFORMANCE '95*, pages 126–135, Ottawa, Canada, May 1995.

[4] I. Matta and A.U. Shankar. Fast Time-Dependent Evaluation of Integrated Services Networks. *Computer Networks and ISDN System – Special Issue on Modeling of Wired and Wireless ATM*, 1998. To appear. Preliminary version in Proc. IEEE ICNP '94, 1994.

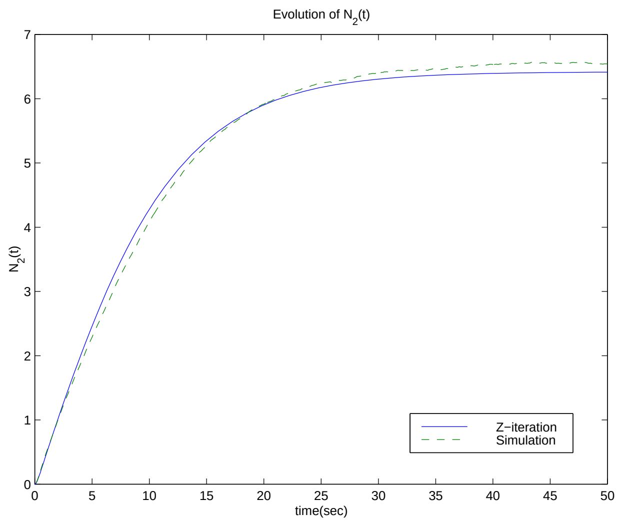
[5] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis.* Springer-Verlag, second edition, 1993.



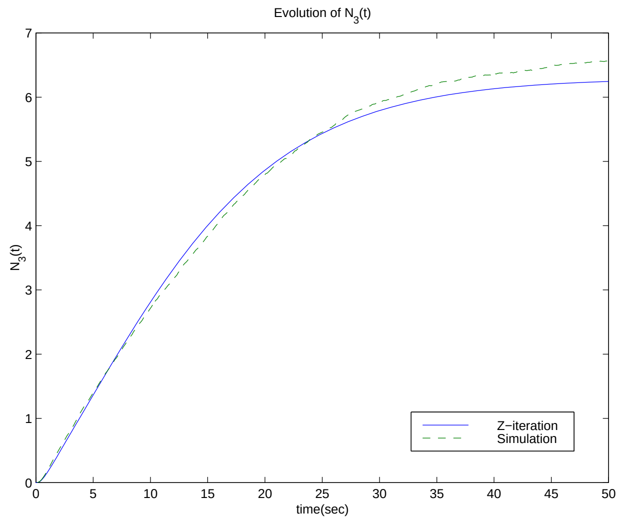
(a)



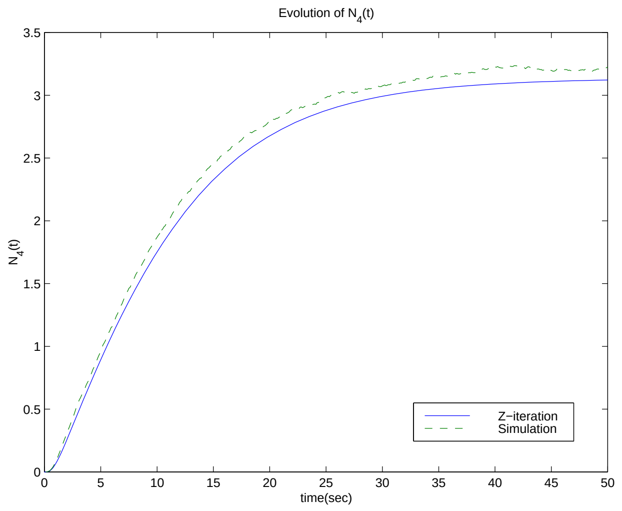
(b)



(c)

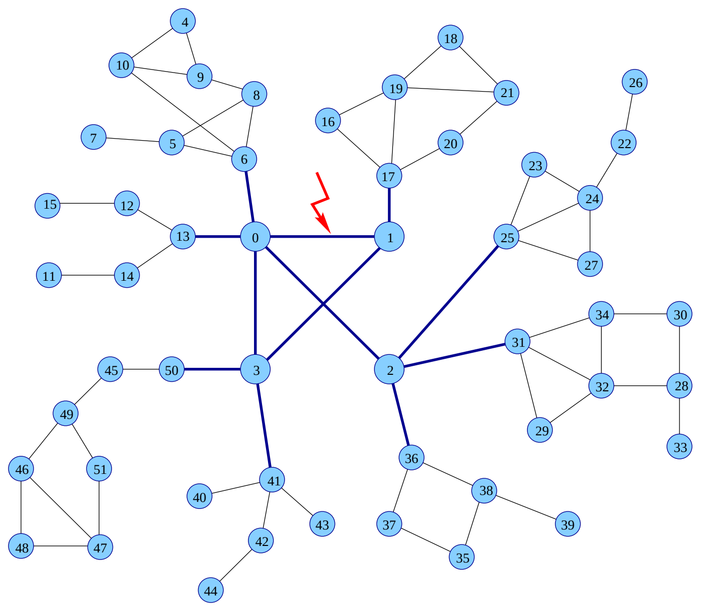


(d)



(e)

Figure 6: A Tandem network with feedback, and its' metrics



Network consists of 52 nodes and 66 links organized as 1 backbone (nodes 0, 1, 2, 3) and 8 LANs (attached via access links at nodes 13, 6, 17, 25, 31, 36, 41, 50).

Buffer capacity for outgoing link queues
backbone/access link: 50,000 pkts
LAN link: 25,000 pkts

Bandwidth of links
backbone/access link: 30,000 pkts/sec
LAN link: 15,000 pkts/sec

Workload
457 end-to-end connections, each with poisson traffic of 1500 pkts/sec

Routing
single shortest-hop route for each connection

Perturbation event
at time 10 sec, the link indicated by arrow fails
routes are recomputed, and changes in link queues are plotted.

Figure 7: Network 1: "small" network of 52 nodes.

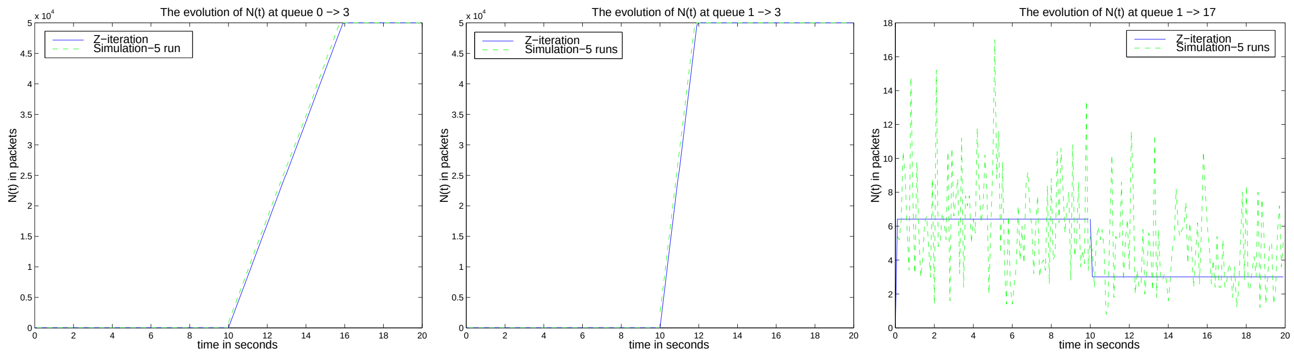


Figure 8: Network 1: instantaneous queue size evolution for links 0-3, 1-3 and 1-17.

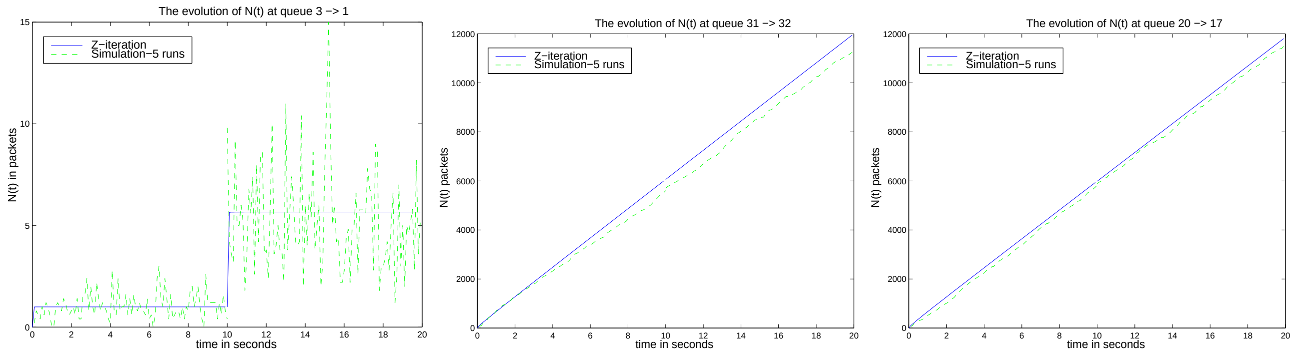
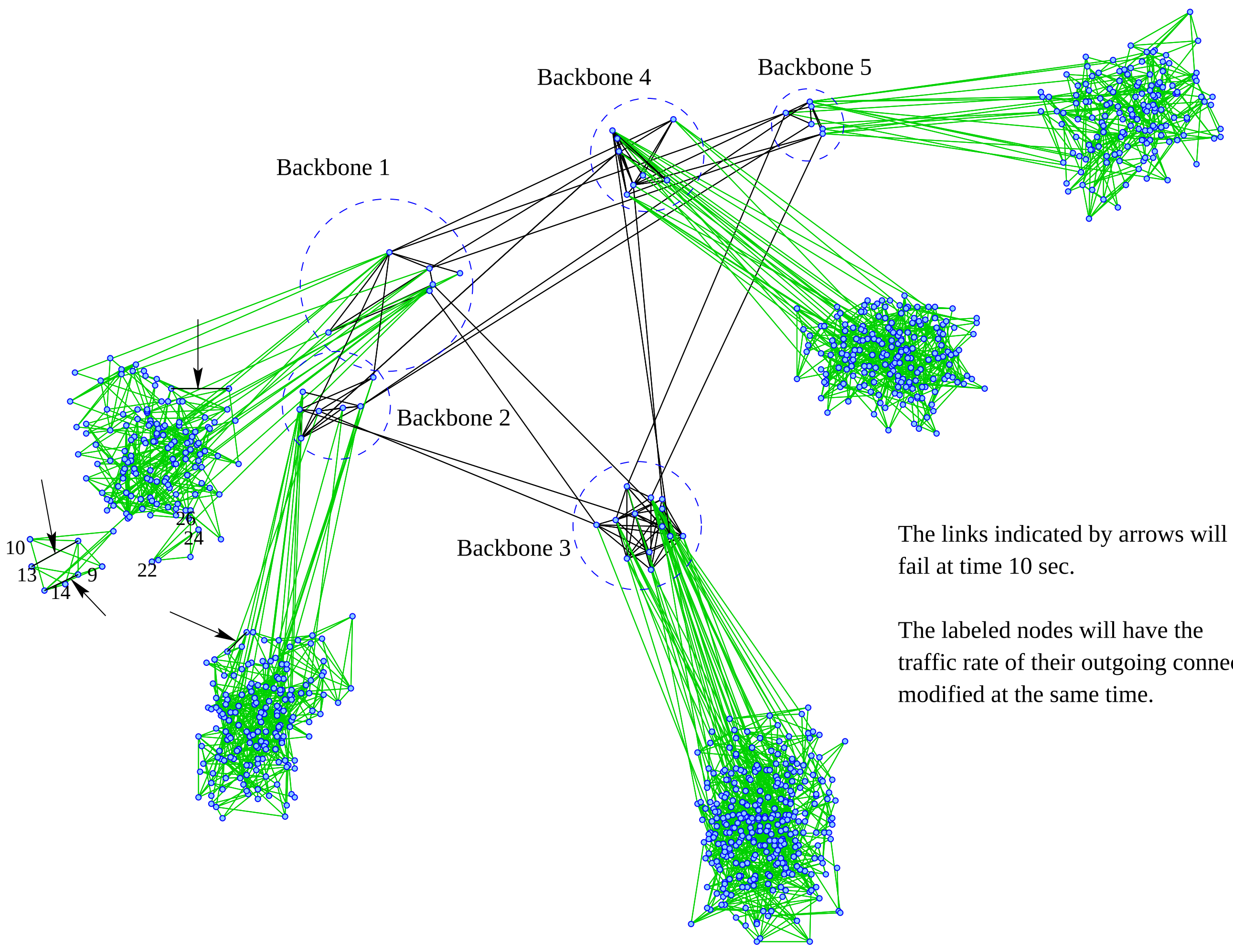


Figure 9: Network 1: instantaneous queue size evolution for links 3-1, 31-32 and 20-17.



The links indicated by arrows will fail at time 10 sec.

The labeled nodes will have the traffic rate of their outgoing connections modified at the same time.

Figure 10: Network 2: overview.

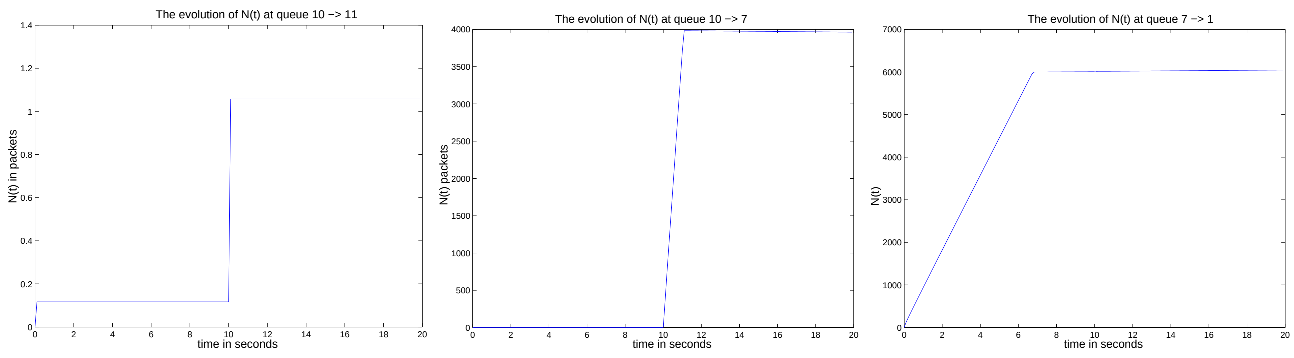


Figure 11: Network 2: instantaneous queue size evolution for links 10-11, 10-7 and 7-1.

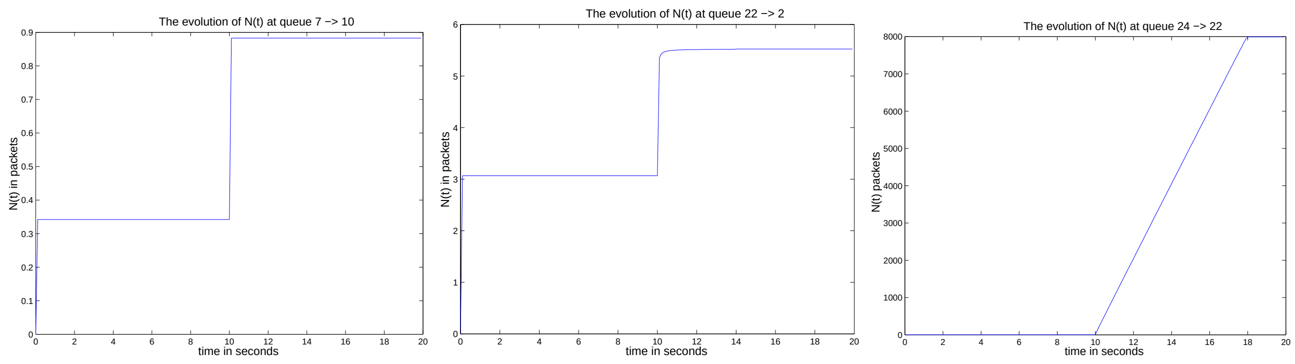


Figure 12: Network 2: instantaneous queue size evolution for 7-10, 22-2 and 24-22.

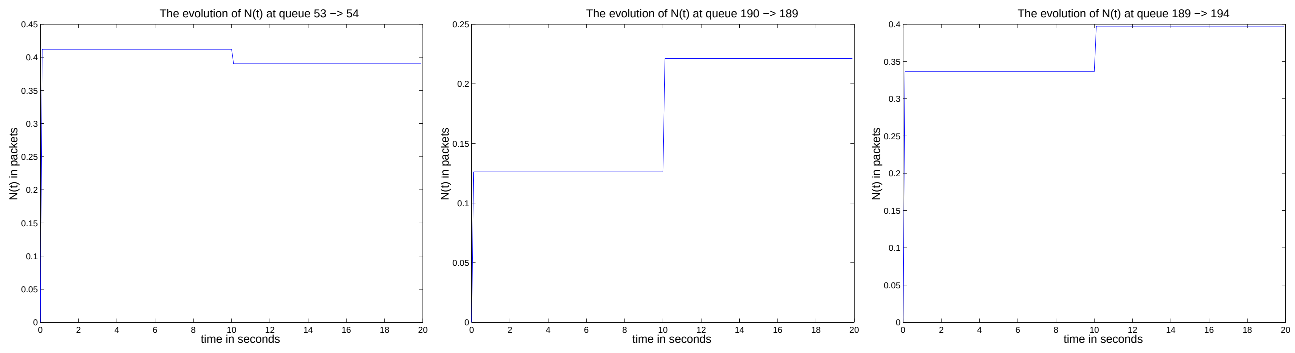


Figure 13: Network 2: instantaneous queue size evolution for links 53-54, 190-189 and 189-194.