Report Number: WUCS-89-04

1989-04-23

Load Balancing Algorithms for Jacksonian Networks with Acknowledgement Delays

Andreas D. Bovopoulos and Aurel A. Lazar

Load balancing algorithms for Jacksonian networks are derived. The state of the network is represented by the total number of packets for which the source has not yet received an acknowledgement, The networks studied are subject to the state independent routing and, state dependent and state independent flow control. The objective is to maximize the throughput of the network so that the end-to-end expected packet time delay does not exceed an upper bound. The optimal flow control is shown to be a window type, while the routing policy balances the traffic inside the network. Several load balancing algorithms are... Read complete abstract on page 2.

Load Balancing Algorithms for Jacksonian Networks with Acknowledgement Delays

Andreas D. Bovopoulos and Aurel A. Lazar

Complete Abstract:

Load balancing algorithms for Jacksonian networks are derived. The state of the network is represented by the total number of packets for which the source has not yet received an acknowledgement, The networks studied are subject to the state independent routing and, state dependent and state independent flow control. The objective is to maximize the throughput of the network so that the end-to-end expected packet time delay does not exceed an upper bound. The optimal flow control is shown to be a window type, while the routing policy balances the traffic inside the network. Several load balancing algorithms are evaluated.

LOAD BALANCING ALGORITHMS FOR JACKSONIAN NETWORKS WITH ACKNOWLEDGEMENT DELAYS

Andreas D. Bovopoulos
Aurel A. Lazar

WUCS-89-04

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

To appear in the Proceedings of IEEE Infocom '89, April 23-29, 1989, Ottawa, Canada

Load Balancing Algorithms for Jacksonian Networks with Acknowledgement Delays

Andreas D. Bovopoulos[*] *and Aurel A. Lazar*[**]

ABSTRACT

Load balancing algorithms for Jacksonian networks are derived. The state of the network is represented by the total number of packets for which the source has not yet received an acknowledgement. The networks studied are subject to state independent routing and, state dependent and state independent flow control. The objective is to maximize the throughput of the network so that the end-to-end expected packet time delay does not exceed an upper bound. The optimal flow control is shown to be of a window type, while the routing policy balances the traffic inside the network. Several load balancing algorithms are evaluated.

1. Introduction

If the resources of a packet switched network are insufficient to handle the incoming traffic, the packet time delay increases and congestion problems may arise. In order to insure that incoming packets receive timely delivery to their destination, a controller might monitor and control the network in such a way that all accepted packets receive the requested quality of service. Upon delivery of a packet to its destination, an acknowledgement packet is sent from destination to source *with a nonzero acknowledgement delay.*

Our model for this class of congestion problems consists of a Jacksonian network with a forward as well as an acknowledgement network. (Note that if the acknowledgement delay is negligible, the acknowledgement network need not be included in the model.) Assuming that the source sends packets into the network with rate c, load balancing algorithms are derived. These algorithms maximize the average throughput such that the expected packet time delay in the forward network does not exceed an upper bound. The problems analyzed in this paper can be classified as *centralized, synchronous, and based on partial observations.* Related work in the area of load balancing is presented in [BOV87], [BOV88a], [BOV88b].

This paper is organized as follows. In section 2, the problem formulation is introduced. In section 3, the load balancing algorithms for Jacksonian networks *with state dependent flow control and state independent routing* are found. In section 4, the load balancing problem of a Jacksonian network *with both state independent flow control and routing* is derived. In section 5, the introduced load balancing algorithms are applied to specific examples.

2. The Statement of the Problem

A user wishes to optimally utilize the resources of a Jacksonian network consisting of a forward and an acknowledgement network (see Fig. 2.1).

Each of the I processors of the forward network has an infinite number of buffers and serves packets with an exponential service rate. There are J processors in the acknowledgement network. Let μ^i be the service rate of the i^{th} processor, $i \in I+J$. Let $\mathrm{R} = [r^{ij}]$ be the $(I+J+1)\times(I+J+1)$ routing matrix ($0 \le i \le I+J$, $0 \le j \le I+J$). Using this notation, packets join the network at node i with probability r^{0i}. Upon completion of service at node i, packets leave the network with probability r^{i0} or are routed from node i to node j with probability r^{ij}. We assume that the topology of the network does not change with time *and that, at the time a packet reaches its destination, an acknowledgement packet begins its way from destination to source.*

Let

$$M \overset{def}{=} I+J \quad .$$

The evolution of the queueing network is described by the stochastic vector
$\mathrm{Q}_t = (Q_t{}^1, \cdots, Q_t{}^{M-1}, Q_t{}^M)$, where $Q_t{}^i$ refers to the number of packets at node i, $1 \le i \le M$.

Let k_i be the number of packets in processor i, for all i, $1 \le i \le M$. The state space of the system is given by

* *Department of Computer Science, Washington University, Saint Louis, MO 63130, e-mail: andreas@patti.wustl.edu.*
** *Department of Electrical Engineering, Center for Telecommunications Research, Columbia University, New York, NY 10027, e-mail: aurel@ctr.columbia.edu*

To appear in the Proceedings of IEEE Infocom '89, 23-27 April 1989, Ottawa, Canada

$$E = \{k_1 + \cdots + k_M\} \quad ,$$

where $0 \le k_i$, $i = 1, \cdots, M$.

$E\gamma$ and $E\tau$ are the expected throughput and expected forward time delay, respectively, of the network. The user maximizes its throughput such that the expected time delay of its packets via the forward network does not exceed a given upper bound T, that is,

$$\max_{E\tau \le T} E\gamma \quad . \tag{2.1}$$

3. Load Balancing of a Jacksonian Network with Acknowledgement Delays

In the sequel the load balancing problem of a Jacksonian network with nonzero acknowledgement delays is studied.

In order to maximize the throughput of the network in such a way that the expected time delay of the user's packets does not exceed a given upper bound, a prime optimization method [LUE84] of solving the problem is followed. If, at most N packets are permitted to enter the network, the original Jacksonian network (depicted in Fig. 2.1) is identical to the network depicted in Fig. 3.1, which in turn can be transformed into a first order equivalent Jacksonian network (Fig. 3.2).

Let

$$E_N = \left\{ k_1 + \cdots + k_M | \sum_{j=1}^{M} k_j \le N, \right\} \quad ,$$

where $0 \le k_i$, $i = 1, \cdots, M$.

$E\gamma_N$ and $E\tau_N$ are the expected forward throughput and expected time delay, respectively, given that at any given moment no more that N packets can be in the network.

Let the $1 \times (M+1)$ matrix $\Theta \overset{def}{=} [\theta^0 \ \theta^1 \ \cdots \ \theta^M]$ be the solution of the traffic flow equations

$$\Theta = \Theta \mathrm{R} \quad ,$$

where $\theta^0 = 1$. Let

$$g^1_{l_1} \overset{def}{=} \sum_{k_1 + k_2 + \cdots + k_I = l_1} \prod_{j=1}^{I} \left(\frac{\theta^j}{\mu^j}\right)^{k_j} \quad , \tag{3.1}$$

for all l_1, $1 \le l_1 \le N$, where $0 \le k_i$, for $i = 1, \cdots, I$.

If l_1 is the total number of packets in processors $1, 2, \cdots, I$, then the Norton equivalent, symbolized by $\nu^1_{l_1}$, is given by

$$\nu^1_{l_1} \overset{def}{=} \frac{g^1_{l_1 - 1}}{g^1_{l_1}} \quad . \tag{3.2}$$

If, in addition, l_2 is the total number of packets in processors $I+1, I+2, \cdots, I+J$ then the Norton equivalent of the processors serving the acknowledgement packets is symbolized by $\nu^2_{l_2}$ and given by the equation

$$\nu^2_{l_2} = \frac{g^2_{l_2 - 1}}{g^2_{l_2}} \quad , \tag{3.3}$$

where

$$g^2_{l_2} = \sum_{k_{I+1} + \cdots + k_{I+J} = l_2} \prod_{j=I+1}^{I+J} \left(\frac{\theta^j}{\mu^j}\right)^{k_j} \quad . \tag{3.4}$$

The state space of the equivalent forward network (Fig. 3.2) is

$$E^1_N = \left\{ k_1 + \cdots + k_I | \sum_{i=1}^{I} k_i \le N \right\} \quad ,$$

where $0 \le k_i$, for all i, $i = 1, \cdots, I$.

The controller of the *equivalent acknowledgement network* is a processor which is the first order equivalent of the original controller together with the acknowledgement network. This controller does not have a maximum service rate c (the maximum service rate of the original controller) but rather has a state dependent maximum service rate c_{l_1}, for all l_1, $l_1 \in E^1_N$. This is the maximum feasible value of the Norton equivalent of the system consisting of the acknowledgement network and the controller. Notice that because this value corresponds to the throughput of a closed network, its maximum value is achieved ([BOV88b], Proposition 2.8.1) when the controller serves with maximum rate c. Given that there are l_1 packets in the forward network, the maximum value of the equivalent controller is

$$c_{l_1} \overset{def}{=} \frac{\sum_{n=1}^{N-l_1} \prod_{l=1}^{n} (\frac{c}{\nu^2_l}) \nu^2_n}{1 + \sum_{n=1}^{N-l_1} \prod_{l=1}^{n} (\frac{c}{\nu^2_l})} = \frac{\sum_{n=1}^{N-l_1} c^n g^2_{n-1}}{1 + \sum_{n=1}^{N-l_1} c^n g^2_n} \quad . \tag{3.5}$$

for all $l_1 \in E^1_N$.

The Norton equivalent function of a Jacksonian network is a concave increasing function with respect to the number of packets ([SHA86],[WAL88]). Thus, c_{N-l_1} is a concave increasing function of l_1, for all $l_1 \in E^1_N$.

Let λ_k^*, for all $k \in E_N^1$, refer to the service rate of the equivalent acknowledgement network in Fig. 3.2. λ_k^* is given by

$$\lambda_k^* = \nu_{N-k}^2 \quad , \tag{3.6}$$

where $k \in E_N^1$.

Definition 2. $\lambda^* = (\lambda_k^*), 0 \le k \le N$, denotes the control of the equivalent network. Note that the class of admissible controls satisfies the peak constraint

$$0 \le \lambda_k^* \le c_k \quad , \tag{3.7}$$

where $k \in E_N^1$. For the case in which the routing parameters are state independent and the network is subject to acknowledgement delays, the optimal flow control is a window flow control and can be derived using the linear program presented in the Appendix I.

In [SHA86] and [WAL88] it is proven that the Norton equivalent of a single class Jacksonian network with one input and many outputs is a concave increasing function with respect to the number of packets k. The Norton equivalent ν_k of the part of the network to which a time delay constraint is applied is therefore concave increasing with respect to the number of packets k. Consequently, ([BOV88b], Proposition 3.6.2), we conclude that *the optimal flow control policy of the equivalent network of Fig. 3.2 is a window flow control, with at most one random point corresponding to the last packet of the window.*

The window flow control policy of the equivalent network can be translated into a flow control policy of the network depicted in Fig. 3.1. It follows from Equation (3.5) that a window flow control policy of the equivalent network corresponds to a window flow control policy of the original network. If the last packet of the window of the equivalent network arrives with an arrival rate less than the maximum defined by (3.5), then the actual controller of the network operates with a window of the same size and sends the last packet with an arrival rate less than c. The actual arrival rate is computed using (3.5). These observations are summarized in the following proposition.

Proposition 3.1 *The optimal flow control of the equivalent network of a Jacksonian network with acknowledgement delay is given by*

$$\lambda_k^* = \begin{cases} c_k & \text{if } 0 \le k \le L-2 \\ 0 < \lambda_{L-1}^* \le c_k & \text{if } k = L-1 \\ 0 & \text{if } L \le k \le N-1 \end{cases} ,$$

which corresponds to a window flow control policy of the form

$$\lambda_k = \begin{cases} c & \text{if } 0 \le k \le L-2 \\ 0 < \lambda_{L-1} \le c & \text{if } k = L-1 \\ 0 & \text{if } L \le k \le N-1 \end{cases} .$$

The actual value of λ_{L-1} can be computed from the optimal values of λ_k^*, for all k, $k = 0, 1, \cdots, L-1$, and Equation (3.5).

In the remainder of this section, the load balancing problem is addressed. Observe that if at most N packets are permitted to enter the network, the optimal flow control is a window flow control with at most one random point, corresponding to the rate of the last packet in the window. An iterative prime optimization algorithm (based on the feasible direction techniques described in [LUE84]) is introduced in order to compute the load balancing policy. If the controller has complete information about the state of the forward network, it can make the optimal flow control decisions. Since an increase in the rate with which packets are acknowledged improves the accuracy of the controller's information concerning the state of the forward network, any increase in the rate with which the acknowledgement packets return to the controller improves the performance of the network.

The load balancing algorithm presented next is based on the previous ideas.

Iterative Algorithm 1:

Let $L(j)$ be the maximum i for which $\lambda_i \neq 0$, (*i.e.*, the window size after the j^{th} iteration).

Let $E\gamma(j)$ be the expected throughput when the network is subject to the routing parameters and flow control derived during the j^{th} iteration.

Step 0 : An arbitrary feasible routing matrix is assigned. Set $L(0) = 1$, and $n = 1$.

The n^{th} iteration of the algorithm has the following steps :

n^{th} Iteration :

Step 1 : For the current window flow control (of size $L(n-1)$) and given routing parameters in the network, solve the following problem. Allow $L(n-1)$ packets to enter the network with the maximum arrival rate. Then, using the Flow Deviation Algorithm [KOB83], improve the routing parameters in the acknowledgment network by keeping the routing parameters in the forward network unchanged and solving the following convex nonlinear optimization problem:

$$\min \frac{1}{E\gamma(n)} \quad ,$$

under the constraints:

$$\sum_{j \in IN(i)} \theta^j = \sum_{j \in OUT(i)} \theta^j \quad ,$$

for every $i = 0, I+1, \cdots, M$, where :

IN(i) is the set of channels incoming to node i of the acknowledgement network,

OUT(i) is the set of channels outgoing from node i of the acknowledgement network.

Step 2 : For the current window flow control (of size $L(n-1)$) and given routing parameters in the acknowledgement network, solve the following problem. Allow $L(n-1)$ packets to enter the network with maximum arrival rate. Then, using the Flow Deviation Algorithm [KOB83], improve the routing parameters in the forward network by keeping the routing parameters in the acknowledgement network unchanged and solving the following convex nonlinear optimization problem:

$$\min \frac{1}{E\gamma(n)} \quad ,$$

under the constraints:

$$\sum_{j \in IN(i)} \theta^j = \sum_{j \in OUT(i)} \theta^j \quad ,$$

for every $i = 1, 2, \cdots, I$, where :

IN(i) is the set of channels incoming to node i of the forward network,

OUT(i) is the set of channels outgoing from node i of the forward network.

Step 3: For the routing parameters computed in Steps 1 and 2, update the flow control with the use of the linear program presented in Appendix I. The solution of the linear program gives a window of size L.

Step 4: Let p_k for all k, $k = 0, 1, \cdots, L$, be the probability that there are k packets in the acknowledgement network under the flow control policy computed in Step 3. Further, let $\theta^j(k)$, for all j, $j = I+1, I+2, \cdots, M$, and for all k, $k = 1, 2, \cdots, L$, be the solution of the traffic flow equation which maximizes the Norton equivalent of the acknowledgement network, given there are k packets in the acknowledgement network. Then set $\hat{\theta}^j = (\sum_{k=1}^{L} p_k \theta^j(k))(\sum_{k=1}^{L} p_k)^{-1}$ for all j, $j = I+1, I+2, \cdots, M$. For the set of traffic flows defined by $\hat{\theta}^j$ for all j, $j = I+1, I+2, \cdots, M$, update the flow control using the iterative algorithm presented in Appendix I. The solution of the linear program gives a window of size L^*.

Step 5: Let p_k for all k, $k = 0, 1, \cdots, L^*$, be the probability that there are k packets in the forward network under the flow control policy computed in Step 4. Further, let $\theta^j(k)$, for all j, $j = 1, 2, \cdots, I$, and for all k, $k = 1, 2, \cdots, L^*$, be the solution of the traffic flow equation which maximizes the Norton equivalent of the forward network, given there are k packets in the forward network. Then set $\hat{\theta}^j = (\sum_{k=1}^{L^*} p_k \theta^j(k))(\sum_{k=1}^{L^*} p_k)^{-1}$ for all j, $j = 1, 2, \cdots, I$. For the set of traffic flows defined by $\hat{\theta}^j$, for all j, $j = 1, 2, \cdots, I$, update the flow control using the iterative algorithm presented in Appendix I. The solution of the linear program gives a window of size $L^{\#}$.

Step 6 : Let $L(n) = L^{\#}$. If $|E\gamma(n) - E\gamma(n-1)| < \epsilon$, where ϵ is a properly chosen tolerance, stop; the computed routing parameters as well as the window $L(n)$ are optimal. Else, $n = n+1$, and repeat all the steps of the iteration.

Intuitively, the near optimal solution of the maximization problem

$$\max_{\lambda, \mathbf{R}, E\tau \le T} E\gamma$$

is achieved by an iterative procedure, each step of which solves the optimization problem

$$\max_{\lambda, E\tau \le T} \max_{\mathbf{R}} E\gamma \quad . \tag{3.8}$$

The previous equation suggests a practical way of controlling a network. Specifically, it suggests that for a given *flow control* on the transport layer, we optimize the *routing* on the network layer. Note that the time delay constraint is dealt with on the transport layer.

4. State Independent Load Balancing of a Jacksonian Network with Acknowledgement Delays

In this section the class of Jacksonian networks which are subject to state independent flow control and which operate with acknowledgement delays are analyzed.

The acknowledgement packets are served by dedicated processors (*i.e.*, the processors $I+1, \cdots, I+J$) as before.

Let $\mathcal{P}$ be the set of all directed paths connecting the origin and destination nodes. Furthermore, let y^k be the flow on path k, for all k, $k \in \mathcal{P}$. Then,

$$\lambda = \sum_{k \in \mathcal{P}} y^k \quad . \tag{4.1}$$

The total load at node i amounts to

$$\theta^i = \sum_{\substack{all\ paths\ p \\ utilizing\ server\ i}} y^p \quad . \qquad (4.2)$$

The expected number of packets in the forward network is given by

$$EQ = \sum_{j=1}^{I} \frac{\theta^j}{\mu^j - \theta^j} \quad . \qquad (4.3)$$

The problem being analyzed can be formulated as the following convex nonlinear optimization problem:

$$\max \sum_{k \in \mathcal{P}} y^k \quad ,$$

under the following constraints:

$$0 \le \sum_{k \in \mathcal{P}} y^k \le c \quad ,$$

$$\sum_{j=1}^{I} \frac{\theta^j}{\mu^j - \theta^j} - \sum_{k \in \mathcal{P}} y^k T \le 0 \quad ,$$

$$\theta^j \le \mu^j \quad , \qquad (4.4)$$

for every node, $j = 1, 2, \cdots, M$.
Constraint (4.4) is never active. Consequently, a feasible direction technique may ignore its presence.

In the sequel we develop an algorithm which is based on the Kuhn-Tucker conditions and which solves the previous convex nonlinear optimization problem. Let ϵ_0 be the Lagrange multiplier corresponding to the time delay constraint. Further, let ϵ_1 be the Lagrange multiplier corresponding to the upper bound of the arrival rate. The Kuhn-Tucker conditions can be written as:

$$-1 + \epsilon_0\left(\frac{\partial EQ}{\partial y^k} - T\right) + \epsilon_1 = 0 \quad , \qquad (4.5)$$

for all k, $k \in \mathcal{P}$,

$$\epsilon_1\left(\sum_{k \in \mathcal{P}} y^k - c\right) = 0 \quad , \qquad (4.6)$$

$$\epsilon_0\left(EQ - \sum_{k \in \mathcal{P}} y^k T\right) = 0 \quad , \qquad (4.7)$$

and

$$\epsilon_l \ge 0 \quad , \qquad (4.8)$$

for all l, $l = 0, 1$.

From equation (4.5), we find that

$$(1 + \epsilon_0 T) = \epsilon_1 + \epsilon_0 \frac{\partial EQ}{\partial y^k} \quad , \qquad (4.9)$$

for all k, $k \in \mathcal{P}$.

If $E_T < T$, then $\epsilon_0 = 0$ and $\epsilon_1 = 1$, which implies that $\sum_{k \in \mathcal{P}} y^k = c$. If $\sum_{k \in \mathcal{P}} y^k < c$, then $(\frac{\partial EQ}{\partial y^k})^{-1} = \frac{\epsilon_0}{1+\epsilon_0 T} > 0$. If $\sum_{k \in \mathcal{P}} y^k = c$, then $(\frac{\partial EQ}{\partial y^k})^{-1} \le \frac{1+\epsilon_0 T}{\epsilon_0} > 0$.

From the previous equations we find that for all k, $k \in \mathcal{P}$

$$\left(\frac{\partial EQ}{\partial y^k}\right)^{-1} \begin{cases} = \frac{\epsilon_0}{1+\epsilon_0 T} & \text{if } \sum_{k \in \mathcal{P}} y^k < c \\ \ge \frac{\epsilon_0}{1+\epsilon_0 T} & \text{if } \sum_{k \in \mathcal{P}} y^k = c \end{cases} \qquad (4.10)$$

The previous expressions can be used for the creation of an effective feasible direction optimization technique.

Observe that a set of path flows is feasible if it does not violate the constraint $0 \le \sum_{k \in \mathcal{P}} y^k \le c$, if it does not saturate any of the processors in the network, and if the expected time delay is acceptable.

Iterative Algorithm 2:

Initially, all the path flows are equal to zero.
Step 1: *If possible, increase the path flows to values that are feasible; otherwise, stop.*
Step 2: *Redistribute the flows so that the Kuhn-Tucker conditions are validated; go to Step 1.*

5. Applications

In this section the algorithms presented in the previous section are applied and thoroughly examined in an example.

A Network of Parallel Processors

In the sequel the optimal flow control and routing of a network of parallel processors is desired. The network is depicted in Fig. (6.1). The service rates of the processors are $\mu^1 = 2$ *packets/sec*, $\mu^2 = 1$ *packet/sec*, and $\mu^3 = .5$ *packets/sec*. Packets arrive into the network with state dependent arrival rate λ_k where 0 *packets/sec* $\le \lambda_k \le 8$ *packets/sec*. Observe that the first step of Iterative Algorithm 1 requires the computation of the optimal routing of the traffic in the network. The computation is done with a flow deviation algorithm [KOB83]. In Fig. (5.2) the optimal value of the parameters θ^k for all k, $k = 1, 2, 3$, as a function of the number N of circulating packets is depicted. Observe that if the number of packets circulating in the network is low, a larger percentage of packets is directed towards the faster servers. As the number of circulating packets increases, the load in the network tends to become balanced. In Step 5 of Iterative Algorithm 1, we compute the traffic flow parameters which maximize the value of the Norton equivalent of the forward network. Those parameters are depicted in Fig. (5.3) for different values of the number of circulating packets. In Fig. (5.4) the maximum state dependent arrival rates and optimal flow control parameters of the equivalent network are depicted for $T = 1.4$ *sec*. In Fig. (5.5) the network of the three parallel processors is shown to be be subject to state independent flow control. In Fig. (5.6) we derive the optimal state independent path flows under state independent flow control for $T = 1.4$ *sec*. In Fig. (5.7) for the previous described network of the three parallel processors we compute the performance of the network as a function of the maximum achieved throughput versus the upper bound of the accepted expected time delay of the packets in the forward network, as a function of the acknowledgement delay. In column A, we compute the performance of the network under state independent flow control. In column B1, we compute the performance of the network under state dependent flow control with acknowledgement delays given by $\mu_4 = \mu_5 = 4.0$ *packets/sec*. In column B2, we compute the performance of the network under state dependent flow control with acknowledgement delays given by $\mu_4 = \mu_5 = 4000.0$ *packets/sec*. In column B3, we compute the performance of the network under state dependent flow control with instanteneous acknowledgement delays. *Notice that control policies that use more information result in a better utilization of the network resources.*

6. Conclusions

In this paper the load balancing problem of a Jacksonian network was investigated. The state of the network was represented by the total number of packets for which the source had not yet received an acknowledgement. Two classes of networks were considered. Both classes were subject to state independent routing. For the first class, the flow control was assumed to be state dependent, whereas for the second class, the flow control was assumed to be state independent. The objective was to maximize the throughput of the network such that the end-to-end expected time delay of the packets did not exceed an upper bound. Load balancing algorithms were investigated. The optimal flow control was shown to be a window flow control, and the derived routing policy balanced the traffic inside the network. A detailed example of an application of the introduced load balancing algorithms was also presented.

7. Appendix

Optimal Flow Control of a Jacksonian Network

Using the general methodology introduced in [BOV88b] and [BOV87], we formulate the flow control problem.

When there are k packets in the network, the probability that an incoming packet joins the network is $\alpha(k, k+1)$. The probability that an incoming packet is rejected is $\alpha(k,k)$. Observe that $\alpha(N,N) = 1$.

Let $p(k)$ be the probability that there are k packets in the equivalent network, for all $k \in E_1$. With each point k, $k \in E_1$, are associated the variables $x = (x(k,j))$, $(k,j) \in E_1 \times E_1$, defined by

$$x(k,k) \overset{\text{def}}{=} p(k)\alpha(k,k) \quad , \tag{7.1}$$

$$x(k,k+1) \overset{\text{def}}{=} p(k)\alpha(k,k+1) \quad , \tag{7.2}$$

for all k, $k = 0, 1, \cdots, N-1$, and

$$\alpha(k,k) + \alpha(k,k+1) = 1 \quad , \tag{7.3}$$

for all k, $k = 0, 1, \cdots, N-1$.

Proposition 7.1 *The optimal flow control parameters λ^*_k, $k \in E_1$, are given by the equations*

$$\lambda^*_k = c_k \frac{x(k,k+1)}{x(k,k) + x(k,k+1)} \quad , \tag{7.4}$$

where $x = (x(k,j))$, $(k,j) \in E_1 \times E_1$, is the solution of the following iterative algorithm:

Step 0 : *N=1*, $E\gamma_{N-1} = 0$.

Iteration:

Step 1: *For the current value of N, solve the following linear optimization problem:*

$$\max \sum_{k=0}^{N-1} c_k x(k,k+1) \quad , \tag{7.5}$$

subject to the linear constraints

$$x(k,k+1)c_k = (x(k,k) + x(k,k+1))\,\nu_{k+1} \quad , \tag{7.6}$$

for all k, $k = 0, 1, \cdots, N-1$,

$$\sum_{k=0}^{N-1} (x(k,k) + x(k,k+1)) + x(N,N) = 1 \quad , \tag{7.7}$$

$$\sum_{k=0}^{N-1} (x(k,k) + x(k,k+1))\,k + Nx(N,N)$$

$$- T \sum_{k=0}^{N-1} c_k x(k, k+1) \leq 0 \quad , \qquad (7.8)$$

and

$$x(i,j) \geq 0 \quad , \qquad (7.9)$$

for $i,j = 0,1,2,\cdots,N$.

Step 2: *If* $E\gamma_N = E\gamma_{N-1}$, *stop. Else,* $N := N + 1$, *and repeat all the steps of the iteration, using the optimal solution of the linear program as the initial feasible point of the next iteration.*

Proof : The proof can be found in [BOV88b], and [BOV87]. The computation of the Norton equivalents of the equivalent forward and acknowledgement networks (Fig. 3.2) can be done efficiently using Mean Value Analysis algorithm [REI80].

References

[BCMP75] Basket, F. and Chandy, M.M. and Muntz, R.R. and Palacios, F.G., "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," *Journal of the ACM,*Vol. 22, No. 2, April 1975, pp.248-260.

[BOV87] Bovopoulos, A.D. and Lazar, A.A., "Optimal Load Balancing for Markovian Queueing Networks," *Proceedings of the* 30th *Midwest Symposium on Circuits and Systems,* Syracuse University, August 17-18,1987, pp. 1428-1436.

[BOV88a] Bovopoulos, A.D. and Lazar, A.A., "Asynchronous Iterative Algorithms for Optimal Load Balancing," Proceedings of the Twenty-Second Annual Conference on Information Sciences and Systems, Princeton University, Princeton, New Jersey, March 16-18, 1988, pp. 1051-1057.

[BOV88b] Bovopoulos, A.D., "Resource Allocation Algorithms in Packet Switched Networks," Ph.D. Dissertation, Columbia University, October 1988.

[KOB83] Kobayashi, H. and Gerla, M., "Optimal Routing in Closed Queueing Networks," *ACM Transactions on Computer Systems*, Vol. 1, No. 4, Nov. 1983, pp. 294-310.

[LAZ83] Lazar, A. A., "Optimal Flow Control of a Class of Queueing Networks in Equilibrium," *IEEE Transactions on Automatic Control,* Vol. AC-28, No. 11, November 1983, pp. 1001-1007.

[LUE84] Luenberger, D.G., *Linear and Nonlinear Programming,* 2nd edition, Addison-Wesley, 1984.

[REI80] Reiser, M. and Lavenberg, S.S., " Mean Value Analysis of Closed Multi-chain Queueing Networks," *JACM,* Vol. 27, No. 3, July 1980.

[SHA86] Shanthikumar, J. G. and Yao, D., "Closure Properties of Equilibrium Rates Under Convolution," preprint, 1986.

[WAL88] Walrand, J., *An Introduction to Queueing Networks,* Prentice Hall, New Jersey, 1988.

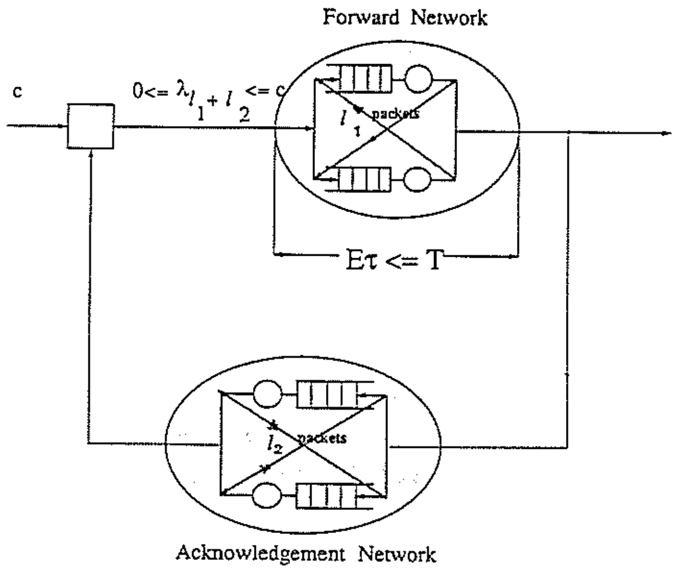


FIGURE 2.1. A Jacksonian network with Acknowledgement delays.

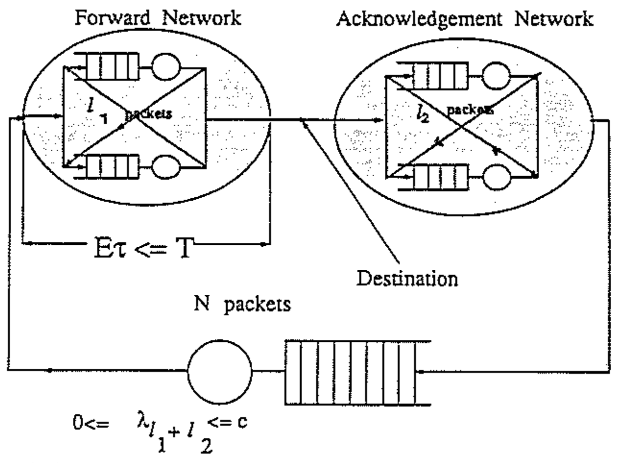


FIGURE 3.1. A Jacksonian network with acknowledgement delays and maximum capacity of N packets.

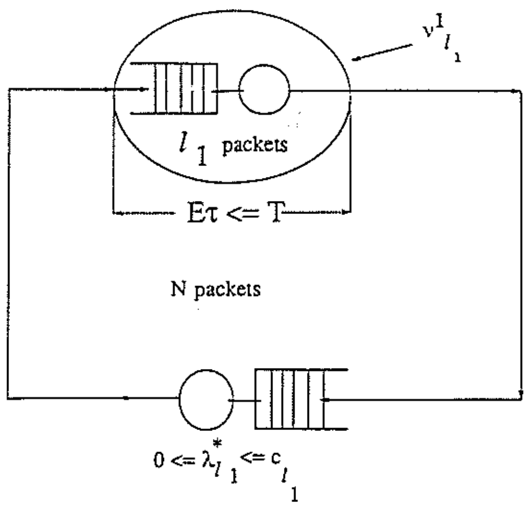


FIGURE 3.2. The equivalent network.

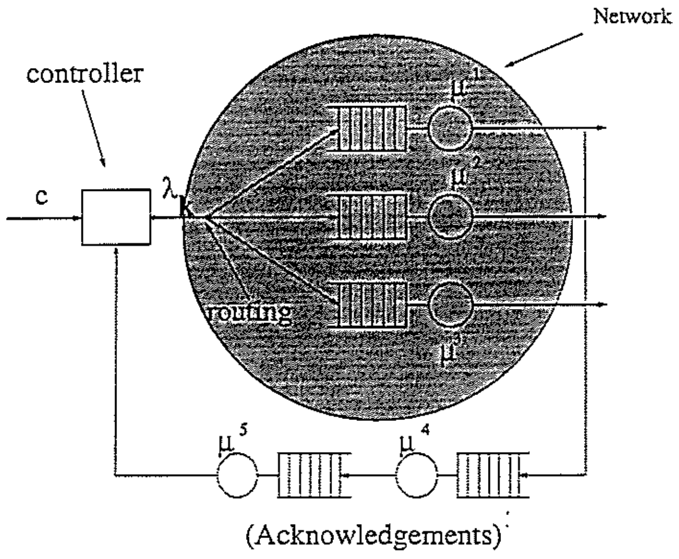
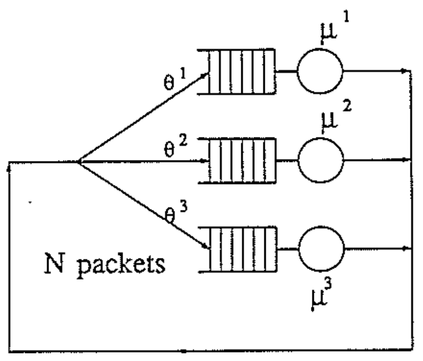


FIGURE 5.1. A network of three parallel processors with acknowledgement delays subject to state dependent flow control.

N	θ^1	θ^2	θ^3
1	1.000	0.000	0.000
2	0.873	0.127	0.000
3	0.767	0.202	0.031
4	0.712	0.229	0.059
5	0.679	0.244	0.077
6	0.659	0.253	0.088
7	0.645	0.258	0.097
8	0.635	0.262	0.103
9	0.627	0.265	0.108
10	0.621	0.268	0.111
11	0.616	0.270	0.114
12	0.612	0.271	0.117
13	0.609	0.272	0.119
14	0.606	0.273	0.121
15	0.603	0.274	0.123

FIGURE 5.2. The optimal values of the parameters θ^k for all k, $k = 1, 2, 3$, as a function of the circulating packets.



N	θ^1	θ^2	θ^3
1	1.000	0.000	0.000
2	0.843	0.150	0.007
3	0.746	0.210	0.044
4	0.698	0.235	0.068
5	0.670	0.247	0.083
6	0.652	0.255	0.093
7	0.640	0.260	0.100
8	0.630	0.264	0.106
9	0.624	0.266	0.110
10	0.618	0.269	0.113
11	0.614	0.270	0.116
12	0.610	0.272	0.118
13	0.608	0.273	0.120
14	0.604	0.274	0.121
15	0.602	0.275	0.123
16	0.560	0.276	0.124
17	0.598	0.276	0.126
18	0.597	0.277	0.127
19	0.595	0.277	0.128

FIGURE 5.3. The traffic flow parameters which maximize the Norton equivalent of the forward network, for different values of N.

N	c^0	c^1	c^2	c^3	c^4
1	1.6	0.0	0.0	0.0	0.0
2	2.35	1.6	0.0	0.0	0.0
3	2.77	2.35	1.6	0.0	0.0
4	3.04	2.77	2.35	1.6	0.0
5	3.21	3.04	2.77	2.35	1.6

The equivalent arrival rates given that in the network there are at most N packets.

Iteration	λ^0	λ^1	λ^2	λ^3	λ^4	$E\gamma$	$E\tau$
1	1.6	0.0	0.0	0.0	0.0	0.89	0.5
2	2.35	1.6	0.0	0.0	0.0	1.35	0.74
3	2.77	2.35	1.6	0.0	0.0	1.66	0.96
4	3.04	2.77	2.35	1.6	0.0	1.91	1.14
5	3.21	3.04	2.77	2.35	1.6	2.10	1.34
6	3.34	3.21	3.04	2.77	1.86	2.18	1.40
7	3.34	3.21	3.04	2.77	1.87	2.18	1.40

FIGURE 5.4. The derivation of the state dependent flow control of the equivalent network, for T = 1.4 *sec*.

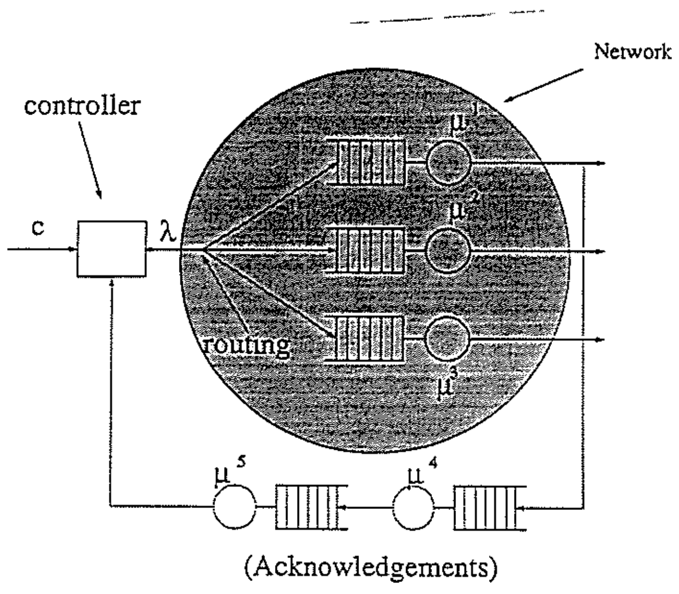


FIGURE 5.5. A network of three parallel processors with acknowledgement delays subject to state independent flow control.

Iteration	y^1	y^2	y^3	$E\tau$	$E\gamma$
0	1.00.0	1.000	1.000	----	----
1	0.062	0.062	0.062	0.730	0.186
2	0.247	0.187	0.127	0.837	0.561
3	0.492	0.312	0.132	0.972	0.936
4	0.747	0.437	0.127	1.188	1.311
5	0.930	0.500	0.070	1.250	1.400
6	1.111	0.381	0.101	1.325	1.593
7	1.157	0.397	0.087	1.365	1.641
8	1.165	0.415	0.085	1.386	1.665
9	1.169	0.416	0.089	1.397	1.674
10	1.170	0.420	0.090	1.400	1.680

FIGURE 5.6. The derivation of optimal state independent path flows under state independent flow control for $T = 1.4$ *sec*.

Expected Throughput (packets/sec)				Expected Time Delay (sec)
A	B1	B2	B3	
0.33	1.08	1.57	1.60	0.5
0.57	1.19	1.64	1.67	0.6
0.75	1.32	1.71	1.74	0.7
0.78	1.52	1.82	1.84	0.8
0.97	1.62	1.98	1.99	0.9
1.15	1.80	2.05	2.06	1.0
1.30	1.87	2.14	2.15	1.1
1.44	2.00	2.25	2.25	1.2
1.56	2.07	2.31	2.31	1.3
1.68	2.17	2.38	2.38	1.4
1.78	2.23	2.45	2.46	1.5
1.88	2.32	2.50	2.51	1.6
1.96	2.36	2.56	2.56	1.7
2.04	2.43	2.61	2.61	1.8
2.11	2.47	2.65	2.65	1.9
2.17	2.53	2.69	2.69	2.0
2.23	2.57	2.72	2.73	2.1
2.29	2.62	2.76	2.76	2.2
2.34	2.65	2.79	2.79	2.3

FIGURE 5.7. Parameterized performance of a network of three parallel processors with respect to the controller's information.