# Per-Node Delay Assignment Strategies For Real-Time High Speed Networks

Abhishek Vagish, Taieb Znati and Rami Melhem
Computer Science Department
University of Pittsburgh
Pittsburgh, PA 15260
(avagish,znati,melhem)@cs.pitt.edu

## Abstract

*A variety of packet scheduling strategies have been proposed to guarantee the application's end-to-end delay requirements in real-time networks. The schemes, however, do not address specifically the mechanisms used to assign a per-node delay or service rate in order to meet the required end-to-end delay. In this paper, using a methodology for computing feasible per-node delays along a routing path, we describe different strategies to assign a delay value for each node so that the end-to-end delay requirement is satisfied. The performance of these schemes for different network environments is discussed.*

## 1 Introduction

Support of multimedia applications often requires that their performance requirements, in terms of delay and jitter, be *guaranteed.* Several scheduling strategies have been proposed to provide service guarantees for real-time applications [6]. These schemes differ from each other in the strategies they employ to enforce rate control and in the policy they use to service packets. More recent schemes seek to maintain a high level of fairness while reducing the complexity of emulating General Processor Sharing (GPS) scheduling policy [4, 1, 3]. Very few of these schemes, however, address the question of how to assign per-node delay values, or equivalently service rates, to a new session to meet its end-to-end delay requirements. An approach, proposed by Parekh [5], computes the largest and smallest values of $\phi$ that would ensure the worst case delay for the new session and selects the midpoint of the interval resulting from these extreme points. It is clear, however, that this strategy may not be inefficient.

This paper addresses the above issue and describes several strategies to assign a specific delay value for each node along the routing path so that the end-to-end delay requirement of the connection is satisfied. The rest of the paper is organized as follows: In section 2, a methodology for computing per-node delays is presented [2]. In section 3, three per-delay assignment strategies are described. In section 4, the performance comparison of these schemes is presented. The last section provides the conclusion of this work.

## 2 Per-Node Delay Computation Methodology

Consider a real-time channel characterized by its traffic rate specification vector, $(m, r, n)$, its maximum end-to-end delay value, $\Delta_{max}$, and a level of traffic guarantee $\alpha$. The channel's rate is specified based on a Linear Bounded Arrival Process (LBAP) so that the application's long-term packet rate is $\frac{n}{r}$, where $n$ is the number of packets generated over $r$ and $m$ is the maximum burst size over any time interval. The value $\alpha$ specifies the percentage of high priority traffic, relative to the total amount of traffic, the application expects to generate over a time interval of size $r$. This high priority traffic provides the *basic* information to be delivered on-time for the application to operate properly. Of the $n$ packets which can be generated over $r$, at most $\alpha \cdot n$ are considered basic, while the remaining $(1 - \alpha) \cdot n$ packets are considered *enhancement* traffic. Notice that $\lceil \alpha \cdot n \rceil$ packets may be required to guarantee that all basic traffic is transmitted reliably on-time [1].

[1] The methodology presented in this paper does not specifically depend on LBAP; other traffic specifiers can still apply. Furthermore, the parameter $\alpha$ may not be required by some applications in which case it can be set to 1.

Based on the traffic specification, the maximum number of packets, $\sigma_i(t)$, generated by $i$ over an interval of size $t$ can be expressed as:

$$\sigma_i(t) = m_i + \left\lceil \frac{t}{r_i} \right\rceil \cdot n_i. \qquad (1)$$

Let $\mu_k = \frac{P_i}{R_k}$, where $P_i$ is the packet size of channel $i$ and $R_k$ the service at node $k$, represent the maximum amount of service time required by a packet from $i$ at node $k$. The maximum amount of service time required by $i$ over $t$ is $\sigma_i(t) \cdot \mu_k$. Furthermore, the amount of time required to service all $BT$ packets generated over an interval of size $t$ can be expressed as:

$$C_{i,k}(t) = \left( m_i + \left\lceil \frac{t}{r_i} \right\rceil \cdot n_i\alpha_i \right) \cdot \mu_k. \qquad (2)$$

Notice equation (2) does not lead to the exact value of $C_{i,k}(t)$ when $t$ is not an integral multiple of $r_i$, but rather provides an *upper bound* on the amount of $BT$ traffic generated by channel $i$ over an interval $t$.

## 2.1 Processing and Buffer Capacity Model

The *total processing capacity*, $P_k^t$, defines the maximum percentage of the processing capacity at node $k$ which can be used to provide guaranteed service to real-time channels. At any time, an amount $P_k^a$ of the total capacity is allocated to support the real-time requirements of the currently supported channels, while an amount $P_k^v$ is reserved to account for the non-preemptive aspect of packet transmission in communication networks. The excess capacity, $P_k^x$, is the percentage of the node's total processing capacity which can be allocated to support new network channels. At any instant, $P_k^x$ satisfies:

$$P_k^x = P_k^t - P_k^a - P_k^v. \qquad (3)$$

Each node is also configured with memory to buffer packets as they are queued for service. The *total buffering capacity*, $B_k^t$, determines the total number of packets which can be queued at node $k$. At any time, $B_k^a$ of this total capacity is allocated to accept channels such that no $BT$ packets are dropped due to lack of buffer space, while $B_k^x$ represents excess buffers which can be allocated to handle $BT$ packets from future channels.

## 2.2 Scheduling Feasibility Tests

Given a set of $N$ channels, with per-node delays, $\delta_{1,k} \leq \delta_{2,k} \cdots \leq \delta_{N,k}$, and processing requirements, $C_{1,k}(\delta_{1,k}), \cdots, C_{N,k}(\delta_{N,k})$, respectively, the feasibility test derived for non-preemptive EDF and RM, FIFO and GPS can be expressed in a general form as:

$$\frac{C_{1,k}(\delta_{1,k})}{\delta_{1,k}} + \cdots + \frac{C_{N,k}(\delta_{N,k})}{\delta_{N,k}} \leq P_k^t - \frac{\mu_k}{\delta_{1,k}}. \qquad (4)$$

The term $\frac{1}{\delta_{1,k}}$ represents the amount of service rate required to account for non-preemption. For an EDF scheduler, the value of $P_k^t = 1$, for a RM scheduler $P_k^t = N \cdot (2^{\frac{1}{N}} - 1)$, and for GPS the upper bound reduces to $R_k$.

A closer look at the delay bound characteristics reveals that, the characterization of the *smallest feasible delay bound*, $d_{i,k}$, of channel $i$ at node $k$, for a fixed amount of buffers, becomes a factor of node $k$'s *excess processing capacity*. Similarly, the characterization of the *largest feasible delay bound*, $D_{i,k}$, of channel $i$ at node $k$, is directly correlated to the buffering capacity of the node.

### Smallest Feasible Delay Value Computation

Based on the maximum workload, $C_{i,k}(t)$, required by a new channel $i$ over a potential delay bound $t$, the exact criterion for the new channel to be accepted at node $k$, without violating the delay requirements of currently supported channels, can be expressed as:

$$\frac{C_{1,k}(\delta_{1,k})}{\delta_{1,k}} + \cdots + \frac{C_{N,k}(\delta_{N,k})}{\delta_{N,k}} \leq P_k^t - \frac{\mu_k}{min\ (\delta_{1,k},\ t)}. \qquad (5)$$

Substituting $\frac{C_{i,k}(t)}{t}$ by its packet workload based value, results in :

$$\frac{m_i + \lceil \frac{t}{r_i} \rceil \cdot n_i\alpha_i}{t} \leq P_k^x - \underbrace{\frac{\mu_k}{min\ (\delta_{1,k},\ t)}}_{P_k^v}. \qquad (6)$$

where, $P_k^x = P_k^t - \left( \frac{C_{1,k}(\delta_{1,k})}{\delta_{1,k}} + \cdots + \frac{C_{N,k}(\delta_{N,k})}{\delta_{N,k}} \right)$, denotes node $k$'s excess processing capacity, while $\frac{m_i + \lceil \frac{t}{r_i} \rceil \cdot n_i\alpha_i}{t}$ represents the processing requirements of channel $i$ over a delay $t$. The value $t = d_{i,k}$, for which the equality holds, specifies a lower bound on delay values node $k$ can offer to $i$, based on $k$'s current excess capacity and $i$'s requirements.

### Largest Feasible Delay Value Computation

The upper bound delay, $D_{i,k}$, represents the maximum *total* delay a $BT$ packet from $i$ can be delayed at node $k$ such that $k$ provides loss-free $BT$ service to $i$, without violating the buffer requirements of previously established channels. This delay value can be computed based on the current excess buffer capacity of node $k$ which verifies: can be expressed as:

$$B_{1,k} + B_{2,k} + \cdots + B_{N,k} \leq B_k^t, \qquad (7)$$

where $B_{j,k}$ denotes the number of buffers allocated to channels $1 \leq j \leq N$ to guarantee none of its $BT$ packets are dropped. Since "early packets" from $i$ may be queued for up to $\delta_{i,k-1}$ units, before they become eligible for service, and for no more than $\delta_{i,k}$ units of time in the $BT$ service queue, the maximum amount of time a packet from $i$ can be queued at $k$ is $(\delta_{i,k-1} + \delta_{i,k})$. Consequently, the number of buffers, $B_{i,k}$, required to ensure a $BT$ loss-free service to $i$ can be expressed as:

$$B_{i,k} = m_i + \left\lceil \frac{\delta_{i,k-1} + \delta_{i,k}}{r_i} \right\rceil n_i \alpha_i. \qquad (8)$$

Combining equations (7) and (8) results in:

$$m_i + \left\lceil \frac{\delta_{i,k-1} + \delta_{i,k}}{r_i} \right\rceil n_i \alpha_i \leq \underbrace{B_k^t - (B_{1,k} + \cdots + B_{N,k})}_{B_k^x}. \qquad (9)$$

A legitimate value for $D_{i,k}$ can then be expressed as:

$$D_{i,k} = \frac{r_i}{n_i \alpha_i} \left[ B_k^x - m_i - n_i \alpha_i \right]. \qquad (10)$$

## 3 Flexible Delay Assignment Strategies

The methodology described above provides a lower and upper bounds on the feasible delays of a given channel at a given node. In the following, schemes for a per-node delay assignment are discussed.

### 3.1 Optimal Delay Assignment

Consider a new connection request, characterized by its end-to-end delay requirement, $D_e$, over a routing path of length $N$. Let $\rho_i$ be the load at node $i$, along the path, just before the new connection request arrives. Furthermore, let $\omega$ be the amount of work requested by the new connection and $L_i \leq \Delta_i \leq U_i$, the delay that node $i$ can provide to the new connection; $L_i$ and $U_i$ represent the minimum and maximum delays currently feasible at node $i$, respectively. In order to achieve a balance between the amount of resources used at each network, the optimal delay assignment problem can be formalized as follows:

$$\textbf{Minimize: } \sum_{i=1}^{N} (\rho_i + 1/\delta_i), \qquad (11)$$

$$\textbf{Subject to: } \sum_{i=1}^{N} \delta_i = D, \text{ and} l_i \leq \delta_i \leq u_i, \qquad (12)$$

where $\delta_i = \Delta_i/\omega$, $D = D_e/\omega$, $l_i = L_i/\omega$, and $u_i = U_i/\omega$, for $i = 1, \cdots, N$.

### 3.2 Optimal Algorithm (Opt) Basic Steps

The Opt algorithm ensures that when accepted, a new connection induces the smallest possible increase in the network load. The input parameters of the algorithm are the load, $\rho_i$, the delay bounds, $l_i$ and $u_i$, for each node $i$, along the routing path, and the end-to-end delay requirement, $D$, of the new connection. The basic steps of the algorithm can be described as follows:

1. If $\sum_{i=1}^{N} l_i > D$ then reject the new connection request and exit.
2. Set LF=UF=$\emptyset$ and calculate $\xi = D/N$.
3. Compute LBV=$\{i : l_i > \xi$ & $i \notin$ LF $\cup$ UF$\}$ and UBV=$\{i : u_i < \xi$ & $i \notin$ LF $\cup$ UF$\}$.
4. While LBV $\cup$ UBV $\neq \emptyset$ do {
   - From the sets LBV and UBV, choose the set of nodes, MAXBV whose violated bounds (lower or upper) have the maximum deviation from $\xi$.
   - For each node belonging to MAXBV, fix the delay at that node to its $l_i$ or to its $u_i$, depending on whether the node belongs to LBV or UBV, respectively. Add nodes from MAXBV to LF and to UF such that LF=LF $\cup$ $\{i : i \in$ LBV and $i \in$ MAXBV$\}$ and UF=UF $\cup$ $\{i : i \in$ UBV and $i \in$ MAXBV$\}$.
   - Recalculate $\xi$ = (D - $\sum_{i \in LF} l_i$ - $\sum_{i \in UF} u_i$)/(N - $| LF | - | UF |$).
   - Compute LOV=$\{i : l_i < \xi$ & $i \in$ LF$\}$ and UOV=$\{i : u_i > \xi$ & $i \in$ UF$\}$.
   - While LOV $\cup$ UOV $\neq \emptyset$ do {
     - From the sets LOV or UOV (only one of them is nonempty at a time), choose the set of nodes, MAXOV whose violated fixed bounds (lower or upper) have the maximum deviation from $\xi$.

```
    – For each node belonging to MAXOV, unfix the delay at that node from its $l_i$ or
      from its $u_i$, depending on whether the node belongs to LOV or UOV, respectively.
      Accordingly, update LF or UF so that LF=LF - $\{i : i \in$ LOV and $i \in$ MAXOV$\}$
      or UF=UF - $\{i : i \in$ UOV and $i \in$ MAXOV$\}$.
    – Recalculate $\xi = (D - \sum_{i \in LF} l_i - \sum_{i \in UF} u_i)/(N - | LF | - | UF |)$.
    – Recompute LOV and UOV.}
  • Recompute LBV and UBV.}
5. The final delay distribution is as follows: For $\forall$ $i \in$ LF, $\delta_i = l_i$, for $\forall$ $i \in$ UF, $\delta_i = u_i$ and for $\forall$ $i \in$ NBV, $\delta_i = \xi$.   ◇
```

## 3.3 Load Balancing (Heu-LB) and Equi-Partitioning (Heu-D/N) Heuristics

In the following section, we discuss two heuristics which can be used to assign per-node delays to new connection requests along a routing path. The first heuristic, Heu-LB, attempts to balance the load along the routing path. It achieves this goal by computing an initial delay values $\delta_i$'s which is proportional to the respective loads of the nodes along the path and then adjusting these values such that they lie within the lower and upper bounds of the feasible delays at each node, while satisfying the end to end delay requirement of the new connection.

The second heuristic, Heu-D/N, uses an equipartitioning based approach, similar to the Opt algorithm, to compute initial per-node delay values and then adjust these values to meet the lower and upper bound constraints, as well as the end-to-end delay constraint of the current connection request. The basic steps of the algorithms can be described as follows:

**Heu-LB Heuristic:**

```
1. for $i$=1 to N
   • $x_i$=Max( $(l_i * D)/ \sum_{i=1}^{N} l_i), l_i)$ $(l_0 = 0.0)$
   • $y_i$=$l_i$+ abs($u_i - (l_i + l_{i-1})$
   • $\delta_i$=min( $x_i, y_i$ )
2. for $i$=1 to N
   • while $\delta_i > u_i$
     – $\delta_i$=$l_i$ + $(\delta_i - l_i)/2.0$
3. while $\sum_{i=1}^{N} \delta_i > D$
   • for $i$=1 to N
     – $\delta_i$=$l_i$ + $(\delta_i - l_i)/2.0$   ◇
```

**Heu-D/N Heuristic:**

```
1. for $i$=1 to N
   • $x_i$=max(D/N, $l_i$) $(l_0 = 0.0)$
   • $y_i$=$l_i$+ abs($u_i - (l_i + l_{i-1})$)
   • $\delta_i$=min($x_i, y_i$)
2. for $i$=1 to N
   • while $\delta_i > u_i$
     – $\delta_i$=$l_i$ + $(\delta_i - l_i)/2.0$
3. while $\sum_{i=1}^{N} \delta_i > D$
   • for $i$=1 to N
     – $\delta_i$=$l_i$ + $(\delta_i - l_i)/2.0$   ◇
```

## 4 Simulation Results and Analysis

A simulation experiment was carried to compare the performance of the proposed delay assignment strategies, namely Opt, Heu-LB and Heu-D/N. The simulation results were obtained for the *static* and *dynamic* model. In the Static model, the connections last for the lifetime of the simulation experiment, while in the dynamic model, the connections are characterized by their average interarrival rate and accepted connections last only for an average service time.

Figure 1 shows the average number of connections accepted against the number of connections generated for the Opt algorithm and the two heuristics, assuming a static model. As expected, at low network load, the number of accepted connections increases rapidly with the number of connections generated for the network resources are mainly unused.

However, since the accepted connections last for the duration of the simulation the amount of free resources is reduced considerably over time. This in turn reduces the rate of acceptance of new connections and causes the curves to converge when the network load becomes high.

Figure 2 depicts the average percentage increase in the number of channels accepted by Opt and Heu-D/N over Heu-LB, as the average arrival rate of the connections varies, assuming a dynamic model. The results show that Heu-LB performs better than Opt (by less

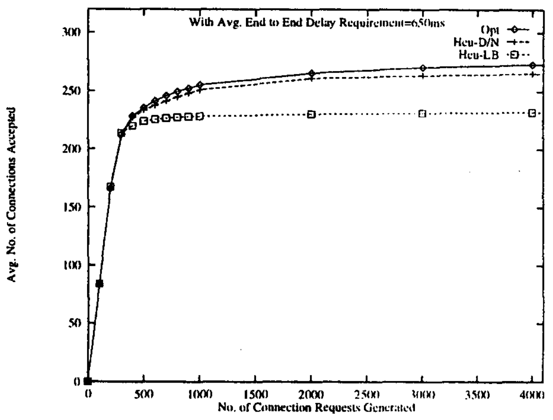

Figure 1: Average Ratio of Successful Connections

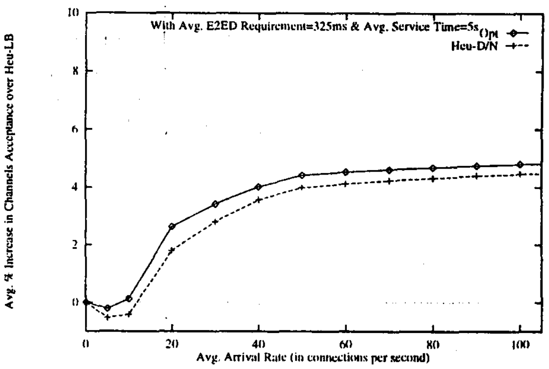

Figure 2: Average Percentage Acceptance Increase

than 0.5%) and Heu-D/N (by less than 1%) at low arrival rates. As the network load increases, the performances of Opt and Heu-D/N exceed those of Heu-LB by 5% and 4.5%, respectively.

The same reasoning used for the analysis of the static model behavior can be applied to explain the behavior observed in in the case of the dynamic model. The increase in the average throughput with the average arrival rate is more pronounced in Heu-LB than in Opt and Heu-D/N at low arrival rate (low network load). Opt and Heu-D/N, however, perform better than Heu-LB at higher arrival rate (high network load).

## 5 Conclusion

In this paper, we proposed a methodology to compute feasible delay values for different classes of scheduling strategies. We also described an optimal algorithm and two heuristics which can be used to assign feasible delay values so that a specific objective is achieved. A set of simulation experiments were developed and used to compare the performance of each scheme. The results show that the optimal algorithm result in higher connection acceptance ratios than the two other schemes. For lightly loaded networks, however, the results show that the computational complexity of the optimal algorithm may not be warranted and simple heuristics usually lead to highly acceptable results.

## References

[1] R. L. Cruz. Service Burstiness and Dynamic Burstiness Measures: A Framework . *Journal of High Speed Networks*, 2:105–127, 1992.

[2] B. Field, T. Znati, and D. Mossé. V-net: A Framework for a Versatile Network Architecture to Support Real-Time Communication Performance Guarantees. In *Infocom 95*, pages 1188–1197, Boston, April 1995.

[3] P. Goyal and H. M. Vin. Fair airport Scheduling Algorithms. In *Proceedings of NOSSDAV '97*, St. Louis, Missouri, May 1997.

[4] Edward W. Knightly, Robert F. Mines, and Hui Zhang. Deterministic Characterization and Network Utilizations for Several Distributed Real-time Applications. In *Proceedings of IEEE WORDS*, Dana Point, CA, October 1994.

[5] A. Parekh. *A Generalized Processor Sharing Approach ti Flow Control in Integrated Services Networks.* PhD thesis, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, February 1992.

[6] H. Zhang and S. Keshav. Comparison of Rate-Based Service Disciplines. In *SIGCOMM '91*, pages 113–121, 1991.