Bandwidth allocation and pricing in multimode network

Jyrki Joutsensalo, Ari Viinikainen, Mika Wikström and Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
P.O. Box 35
40014 University of Jyväskylä, FINLAND
jyrkij@mit.jyu.fi, arjuvi@mit.jyu.fi, wikstrom@mit.jyu.fi and timoh@mit.jyu.fi

Abstract

This paper presents adaptive resource sharing model that uses a revenue criterion to allocate network resources in an optimal way. The model ensures QoS requirements of data flows and, at the same time, maximizes the total revenue by adjusting parameters of the underlying scheduler. Besides, the adaptive model eliminate the need to find the optimal static weight values because they are calculated dynamically. The simulation consists of several cases that analyse the model and the way it provides the required QoS guarantees. The simulation reveals that the installation of the adaptive model increases the total revenue and ensures the QoS requirements for all service classes.

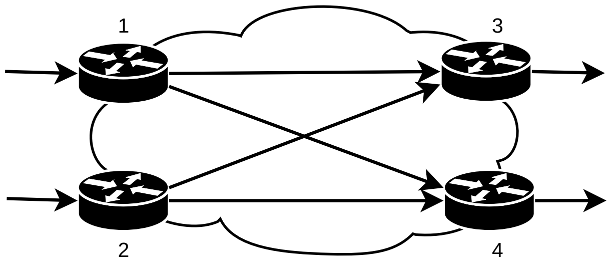

Figure 1. Four edge node system, with two ingress and egress nodes.

1. Introduction

Today's Internet supports QoS for the flows that are sensitive to effects like delay and jitter. It must ensure that all the flows receive their reserved resources while QoS is also maintained. To ensure this, there must be mechanisms to give guaranteed bandwidth and computational resources to incoming flows. However, allocation of bandwidth and CPU resources are interdependent and maintaining fairness in one resource allocation does not necessarily entail fairness in another resource allocation. Therefore, for better maintenance of QoS guarantees and overall fairness in resource allocations for the contending flows, the processor and bandwidth scheduling schemes should be integrated. A significant amount of work has been done in resource scheduling for traditional networks.

The fair allocation of bandwidth is typically achieved by using per-flow queueing mechanisms that are complex to implement such as Fair Queueing [3], [10] and its many variants [4], [1], [5]. However, these mechanisms require that each arriving packet has to be classified into a flow and the router must perform several operations based on the updated per-flow state variables. Several methods have been presented to reduce the complexity of the per packet operations, such as [12], [13], [6], [15], [9], [11] and [17]. In [14] the complexity is localized in the edge routers, where the per-flow information is computed, while the core routers just use first-in-first-out (FIFO) queueing and keep no per-flow state. However, it still remains unclear if these algorithms can be cost-effectively implemented.

In this paper we present a low complexity packet scheduling algorithm (to be used e.g. in edge routers) for allocation a fair share of bandwidth to different service classes while providing optimal revenue to the service provider. We extend our previous studies [8, 7], where delay was considered as QoS parameter.

The rest of the paper is organized as follows. Section 2 discusses network model and bandwidth formulation. In Section 3 the pricing function and the optimal weights for revenue maximization is presented. The next section 5 considers implementation issues and computational complexity. In Section 6 the operation of the algorithm is simulated. Finally, the conclusions are discussed in Section 7.

2. Multinode Network and Bandwidth

In this section, we formulate expression for bandwidth (bit rate) of the data traffic in the multinode network. Although the algorithm operates in the general multinode system with arbitrary number of nodes, we present it by using the four node case to avoid complicated notation. Consider a network with four edge nodes (Fig. 1). Data are transmitted from nodes 1 and 2 to either nodes 3 and 4. The customers of the gold class pay more than the customers of the silver and bronze classes - in our case for the available bandwidth - but on the other hand, they get more bandwidth.

There are three service classes, namely gold, silver, and bronze.

There are two types of connections in the jth queue in the node 1. They are denoted by $N_j^{1\to 3}$ and $N_j^{1\to 4}$. Notation $N_j^{i\to p}$ means that there are $N_j^{i\to p}$ such connections (customers) in the jth queue which transfers data through nodes i and p. Total number of connections in the node i and queue j is denoted by N_{ij}, and it obeys the condition

$$N_{ij} = \sum_{p=1}^{n} N_j^{i\to p}, \tag{1}$$

where n denotes number of the nodes. In our example case, $n = 4$, and

$$N_{1j} = N_j^{1\to 3} + N_j^{1\to 4}, \tag{2}$$

$$N_{2j} = N_j^{2\to 3} + N_j^{2\to 4}, \tag{3}$$

$$N_{3j} = N_j^{1\to 3} + N_j^{2\to 3}, \tag{4}$$

$$N_{4j} = N_j^{1\to 4} + N_j^{2\to 4}. \tag{5}$$

Here e.g. $N_1^{1\to 2} = 0$.

Let us consider the bandwidth in the node i. Let the processing time of the data be T [seconds/bit] in the packet scheduler. There are N_{ij} connections or packets in the class j. Let us denote the packet size b_{ijk} [bits] or [kbytes] in the node i, $i = 1, \ldots, n$, class $j = 1, \ldots, m$ and the connection $k = 1, \ldots, N_{ij}$. Variable w_{ij} is the weight allocated for class j in the node i. Constraint for weights w_{ij} is

$$\sum_{j=1}^{m} w_{ij} = 1, \quad w_{ij} > 0. \tag{6}$$

Variables w_{ij} give weights, how long time queues (i, j) are served per total time. It is easy to see that bandwidth of the packet (i, j, k) is

- linearly proportional to the packet size b_{ijk},
- linearly proportional to the weight w_{ij},
- inversely proportional to the processing time T, and
- inversely proportional to the total sum of the packet lengths b_{ijk}, $k = 1, \ldots, N_{ij}$, because other packets occupy the same band in a time-divided manner.

Therefore, the expression for the bandwidth of the packet (i, j, k) is

$$B_{ijk}[bits/s] = \frac{b_{ijk} w_{ij}}{T \sum_{l=1}^{N_{ij}} b_{ijl}} = \frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \tag{7}$$

where the processing time T in the denominator can be scaled $T = 1$, without loss of generality. If processing times differ from node to node, notation is more complicated, but there is no critical difference on our formalism.

3. Pricing Model and Revenue Optimization

We concentrate on the pricing and fair resource allocation from the point of view of the customers. On the other hand, from the point of view of the service provider, we try to maximize revenue. First, we introduce the concept of *pricing function*. Naturally, the price is usually concave with respect to the bandwidth; for example, when images are transferred, the price may be twice compared to the price when voice is transferred, while the bandwidth ratio - image bandwidth/voice bandwidth - is much more than two.

Consider the price paid by customers in class j to the service provider. It depends on the bit rate. The price $r_j(B)$ is increasing with respect to the bit rate B, and it is concave. We use the *polynomial* pricing model, because using that model, one can get a closed form approximating solution and the model is increasing and concave.

Definition. *The pricing model*

$$r_j(B) = r_j B^p, \tag{8}$$

where B is the bandwidth,

$$r_j > 0, \tag{9}$$

$$p \in (0, 1) \quad (\textit{concavity}) \tag{10}$$

is polynomial.

When connection (i, j, k) in the node i in class j obtains the bandwidth B_{ijk}, he/she is paying

$$r_j(B_{ijk}) = r_j \left(\frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \right)^p \tag{11}$$

units of money per second to the service provider.

Bit rate in the multinode system is the same as the bit rate at the weakest point. Therefore, revenue obtained from the connection (i, j, k) in our four node system is

$$F_{jk}^{i\to q} = r_j \min \left\{ \frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}}, \frac{b_{qjk} w_{qj}}{\sum_{l=1}^{N_{qj}} b_{qjl}} \right\}^p . \tag{12}$$

Here, $F_{jk}^{i\to q}$ denotes revenue for jth class and kth packet, which passes through nodes i and q. Using Lagrangian constraint, total revenue is

$$\begin{aligned} F &= F^{1\to 3} + F^{1\to 4} + F^{2\to 3} + F^{2\to 4} \\ &+ \sum_{i=1}^{4} \lambda_i (1 - \sum_{j=1}^{3} w_{ij}) \\ &= \sum_{j=1}^{3} \sum_{k=1}^{N_j^{1\to 3}} F_{jk}^{1\to 3} \\ &+ \sum_{j=1}^{3} \sum_{k=N_j^{1\to 3}+1}^{N_{1j}} F_{jk}^{1\to 4} \\ &+ \sum_{j=1}^{3} \sum_{k=1}^{N_j^{2\to 3}} F_{jk}^{2\to 3} \\ &+ \sum_{j=1}^{3} \sum_{k=N_j^{2\to 3}+1}^{N_{2j}} F_{jk}^{2\to 4} \\ &+ \sum_{i=1}^{4} \lambda_i (1 - \sum_{j=1}^{3} w_{ij}). \end{aligned} \tag{13}$$

When minimum operators are taken from (13), revenue reduces to the general form

$$F = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{N_{ij}} a_{ijk} w_{ij}^{p} + \sum_{i=1}^{n} \lambda_i (1 - \sum_{j=1}^{m} w_{ij}), \tag{14}$$

where *general* parameters are as follows: n is the number of nodes, m is the number of service classes, and $a_{ijk} > 0$ is known parameter, being a function of r_j, b_{ijk}, and $\sum_{l=1}^{N_{ij}} b_{ijl}$.

4. Bandwidth Broker Algorithm

Here we present the bandwidth broker algorithm for approximating optimal solutions for maximizing revenue still allocating bandwidth in a fair way. In the simplest approach, called BB1, it is assumed that the bit rates in the minimum operation brackets are roughly the same, i.e.

$$\begin{aligned} \min & \left\{ \frac{b_{111} w_{11}}{\sum_{l=1}^{N_{11}} b_{11l}}, \frac{b_{311} w_{31}}{\sum_{l=1}^{N_{31}} b_{31l}} \right\} \\ \approx & \frac{1}{2} \left(\frac{b_{111} w_{11}}{\sum_{l=1}^{N_{11}} b_{11l}} + \frac{b_{311} w_{31}}{\sum_{l=1}^{N_{31}} b_{31l}} \right) \end{aligned} \tag{15}$$

etc. for all operations in (13). Then the revenue can be linearized, and approximated as follows:

$$\begin{aligned} F &= r_1 \sum_{k=1}^{N_1^{1\to 3}} \min\{B_{11k}, B_{31k}\}^p \\ &+ r_1 \sum_{k=N_1^{1\to 3}+1}^{N_{11}} \min\{B_{11k}, B_{41k}\}^p \\ &+ \ldots \\ &+ \sum_{i=1}^{4} \lambda_i (1 - \sum_{j=1}^{3} w_{ij}) \\ &\approx \frac{1}{2} r_1 \sum_{k=1}^{N_1 1} \left(\frac{b_{11k} w_{11}}{\sum_{l=1}^{N_{11}} b_{11l}} \right)^p \\ &+ \ldots \\ &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} r_j \sum_{k=1}^{N_{ij}} \left(\frac{b_{ijk} w_{ij}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \right)^p \\ &+ \sum_{i=1}^{n} \lambda_i (1 - \sum_{j=1}^{m} w_{ij}), \end{aligned} \tag{16}$$

where last expression is obtained by taking account of equalities (2)-(5). In that case, w_{ij}:s can be directly evaluated by treating F as a simple linear constrained optimization problem as in the previous section. Weights have the closed form solution

$$w_{ij} = \frac{r_j^{-\frac{1}{p-1}} \left[\sum_{k=1}^{N_{ij}} \left(\frac{b_{ijk}}{\sum_{l=1}^{N_{ij}} b_{ijl}} \right)^p \right]^{-\frac{1}{p-1}}}{\sum_{q=1}^{m} r_q^{-\frac{1}{p-1}} \left[\sum_{s=1}^{N_{iq}} \left(\frac{b_{iqs}}{\sum_{h=1}^{N_{iq}} b_{iqh}} \right)^p \right]^{-\frac{1}{p-1}}} \tag{17}$$

$$\frac{\partial^2 F}{\partial w_{ij}^2} < 0, \quad p \in (0,1) \quad \text{(global optimum for (16))} \tag{18}$$

As a special case, we consider square root pricing scenario, where $p = \frac{1}{2}$. Then

$$w_{ij} = \frac{r_j^2 \left(\sum_{k=1}^{N_{ij}} \sqrt{\frac{b_{ijk}}{\sum_{l=1}^{N_{ij}} b_{ijl}}} \right)^2}{\sum_{q=1}^{m} r_q^2 \left(\sum_{s=1}^{N_{iq}} \sqrt{\frac{b_{iqs}}{\sum_{h=1}^{N_{iq}} b_{iqh}}} \right)^2} \tag{19}$$

BB1 algorithm has an important advantage that it allows *local updating in the nodes*, because in calculation of w_{ij} only the parameters of the node i are used:

- Gain factors $r_1, \ldots, r_m$ are same for all nodes,
- Packet lengths b_{ijk} are defined for node i.

In the previous derivation, every minimum operation included exactly two arguments. In the general case, there are different number of arguments in the minimum operation. This changes the form of the approximated revenue formula in such a way that there are other scaling factors than $\frac{1}{2}$; however, it is still a linearly constrained optimization formula yielding a closed form solution.

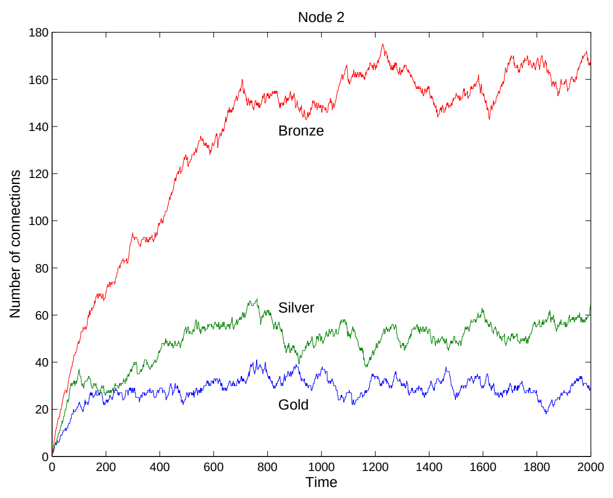

Figure 2. Evolution of number of connections in node 2.

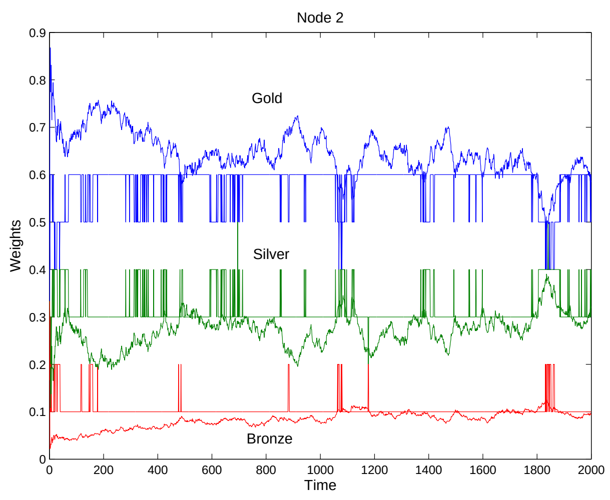

Figure 3. Weights - Evolution of the weights as a function of time for both the optimal weights (smoother lines) and "brute-force" weights (angled lines).

5. Implementation Issues and Computational Complexity

We have selected edge routers of the DiffServ architecture [2] for implementation target to our adaptive model. In this case, all major adaptive issues will be implemented at the edge of a domain. Unlike the core routers, the edge routers have the per-flow information that enables them to perform classification and policing. Since data regarding each traffic flow is available, it is much more convenient to perform the adaptive resource allocation in this part of a DiffServ domain. In this framework, the core routers could remain intact and are not overburdened with additional adaptive software that can slow the packet forwarding process. Moreover, such an approach fits into the original idea of the DiffServ technology which states that the edge routers perform sophisticated functions while the core routers perform only the simple forwarding.

In the proposed adaptive framework, the key role of the adaptive egress routers is to control the amount of traffic injected into a DiffServ domain. By tracking the number of active data flows and their Quality of Service (QoS) parameters, the adaptive edge routers can allocate optimally the output bandwidth between different traffic aggregates. Of course, this solution does not diminish the use of other adaptive solutions that could be implemented in the core routers to provide finer resource allocation and to achieve better utilization [16]. Assume that there are n nodes, and that the user's data packets go on the average through q nodes. There are m service classes. Consider the computation in the BB1 algorithm. When weight w_{ij} is updated according to the rule (17), no iterations are needed. To calculate numerator, four operations are needed. To calculate denominator, $O(m)$ operations are needed. Thus, about 10 operations are needed, when m is small.

6. *

Simulations

In this section we demonstrate by simulation the operation of the square root pricing function (8) with $m = 3$ service classes (gold, silver and bronze) and four nodes (two ingress and two egress). We compare the optimal weights (17) with a version of the algorithm that obtains the weights by "brute-force" method, for which the possible values for the weights are $w_{ij} = 0.1, \ldots, 0.8$ (i.e. the step size is 0.1). We present the weights for all the nodes using both optimal weights and "brute force" weights. The connections in the different service classes have different average data packet sizes $E(b_1) = 50$, $E(b_2) = 25$ and $E(b_3) = 10$, with a standard deviation of 1 (i.e. in a specific service class the connections have similar demand for bandwidth). The processing time of each packet scheduler is chosen as $T = 1/10000$ s/kbyte. The arrival rates of the connections to the nodes 1 and 2 are Poisson distributed and they are $\alpha_1 = 0.30$, $\alpha_2 = 0.40$ and $\alpha_3 = 0.50$ per unit time for the gold, silver, and bronze classes, respectively. Each connection has equal probability of being routed to egress node 3 or 4.

The life time of a connection (i.e. the time the connection is served and has packets in the queue) is exponentially distributed. The duration parameters (i.e. "decay rates") for

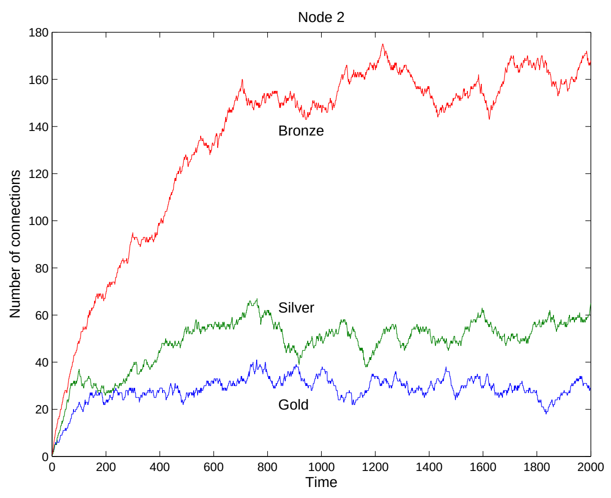

Figure 4. Evolution of number of connections in node 2.

the connections are $\beta_1 = 0.01$, $\beta_2 = 0.007$ and $\beta_3 = 0.003$, where the probability density functions for the durations are

$$p_i(t) = \beta_i e^{-\beta_i t}, \quad ,i = 1,2,3 \quad t \geq 0. \tag{20}$$

We run both the "brute-force" simulation and the optimal weights simulation with the same parameters so that gives us a good possibility to compare the results. In both cases the number of connections are shown in Fig. 4. For the rest of the nodes and for all the "brute force" nodes, the results concerning the number of connections are quite similar to the results in Fig 4. In Fig. 5 we have mean bandwidth for the classes in node 2 for the optimal weights and the mean bandwidth for "brute-force" is shown in Fig. 6. The slight difference between the mean bandwidth for optimal weights and the "brute-force" is derived straight from the weights which are calculated for the classes (Fig. 7). The behavior is similar in all the nodes when we study the mean bandwidth for the classes.

The difference between the two methods is clearly seen when we look at Fig. 7 which shows the optimal weights and "brute force" weights in node 2. The weights in nodes 1, 3 and 4 are again behaving quite similar with the weights in node 2. From Fig. 7 we can also see that the optimal weights behave smoothly while the "brute-force" weights make quite sudden changes. The behavior of the "brute-force" weights comes straight from the definition where we specify the step size. By reducing the step size we would get smoother behavior for the "brute-force" but at the same time we would increase the computation time dramatically. If we have in "brute-force" method weights $w_{ij} = 0.1, \ldots, 0.8$ for all the four nodes we have to make $36^4 = 1.679.616$ loops to find out the best weights. We can limit the possible weights to a certain range in order to lower the computation time. For example limiting the lowest weight for the gold to be 0.4 and the lowest weight for the silver class to be 0.2, we can decrease the number of loops to $15^4 = 50.625$. This needs still a lot of computation time compared to optimal weights. In the simulation environment the "brute-force" method with limited range required 1000 times more computation time compared to the optimal weights method.

In Fig. 8 we have the revenues for both methods. The optimal weighs revenue has the upper one or green line and the "brute-force" has the lower, blue line. The mean revenues for 7757.9 and 7724.2 for optimal and "brute-force" weights, respectively. This slight 0.5% difference can be seen in Fig. 9 but the result has been gained with a significantly lower computational time.

7. Discussion and Concluding Remarks

We have derived an analytic form to the revenue algorithm for updating the weights w_{ij} which allocate data traffic to the connections of different service classes in a multinode network. The weight updating procedure of the packet schedulers has low computational complexity and is robust against errors that may occur from inaccurate models as it is deterministic and nonparametric. Also, the weight updating rule has an important advantage that it allows local updating in the nodes. We have shown by simulation that our algorithm gives larger revenue and allocates bandwidth in a fairer way than the brute-force algorithm.

References

[1] J. C. R. Bennett and H. Zhang. WF^{2}Q: worst-case fair weighted fair queueing. In *Proc. IEEE INFOCOM*, volume 1, pages 120 – 128, 1996.

[2] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. IETF RFC 1633, June 1994.

[3] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queuing algorithm. *ACM SIGCOMM Comput. Commun. Rev.*, 19(4):1–12, Sept. 1989.

[4] S. J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proc. IEEE INFOCOM*, volume 2, pages 636 – 646, 1994.

[5] P. Goyal, H. M. Vin, and H. Cheng. Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks. *IEEE/ACM Trans. Networking*, 5(5):690–704, Oct. 1997.

[6] C. Guo. SRR: an $O(1)$ time complexity packet scheduler for flows in multi-service packet networks. *ACM SIGCOMM Comput. Commun. Rev.*, 31(4):211–222, 2001.

[7] J. Joutsensalo, T. Hämäläinen, K. Luostarinen, and J. Siltanen. Adaptive scheduling method for maximizing revenue in flat pricing scenario. International Journal of Electronics and Communications, 2005. in press.

[8] J. Joutsensalo, T. Hämäläinen, M. Pääkkönen, and A. Sayenko. QoS- and revenue aware adaptive scheduling algorithm. *Journal of Communications and Networks*, 6(1):68–77, Mar. 2004.

[9] S. S. Kanhere, H. Sethu, and A. B. Parekh. Fair and efficient packet scheduling using elastic round robin. *IEEE Trans. Parallel Distrib. Syst.*, 13(3):324–336, Mar. 2002.

[10] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Trans. Networking*, 1(3):344–357, June 1993.

[11] S. Ramabhadran and J. Pasquale. Stratified round robin: A low complexity packet scheduler with bandwidth fairness and bounded delay. In *Proc. 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 239 – 250. ACM Press, 2003.

[12] M. Shreedhar and G. Varghese. Efficient fair queuing using deficit round-robin. *IEEE/ACM Trans. Networking*, 4(3):375–385, June 1996.

[13] D. Stiliadis and A. Varma. Efficient fair queueing algorithms for packet-switched networks. *IEEE/ACM Trans. Networking*, 6(2):175–185, Apr. 1998.

[14] I. Stoica, S. Shenker, and H. Zhang. Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high-speed networks. *IEEE/ACM Trans. Networking*, 11(1):33–46, Feb. 2003.

[15] S.-C. Tsao and Y.-D. Lin. Pre-order deficit round robin: a new scheduling algorithm for packet-switched networks. *Comput. Networks*, 35(2-3):287–305, Feb. 2001.

[16] S. Yi, X. Deng, G. Kesidis, and C. R. Das. Providing fairness in diffserv architecture. In *Proc. IEEE Global Telecommunications Conference (GLOBECOM'02)*, volume 2, pages 1435 – 1439, 2002.

[17] X. Yuan and Z. Duan. FRR: a proportional and worst-case fair round robin scheduler. Technical Report TR-040201, Department of Computer Science, Florida State University, Tallahasse, FL 32306, 2004.

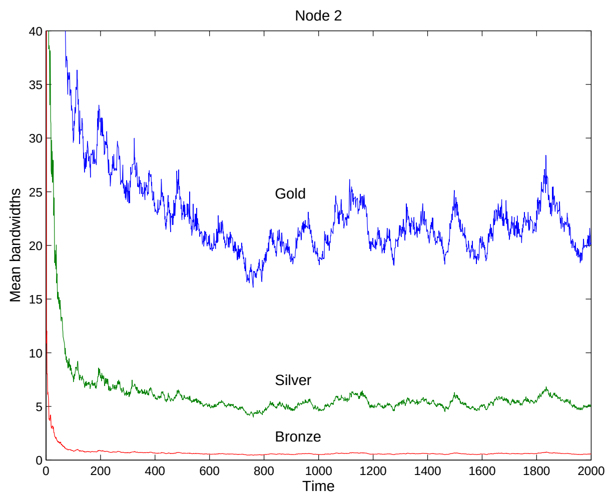

Figure 5. Optimal weights - Evolution of mean bandwidth for all the three classes in node 2.

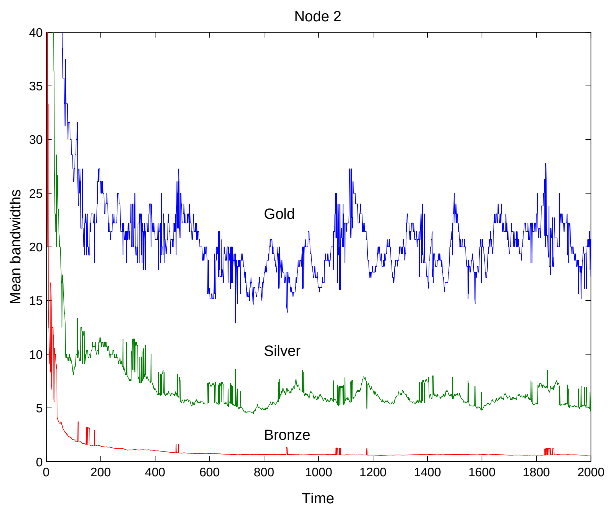

Figure 6. "Brute-force" - Evolution of mean bandwidth for all the three classes in node 2.

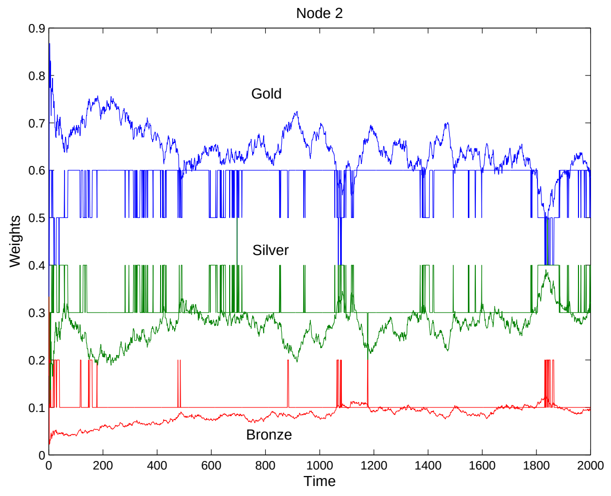

Figure 7. Weights - Evolution of the weights as a function of time for both the optimal weights (smoother lines) and ”brute-force” weights (angled lines).

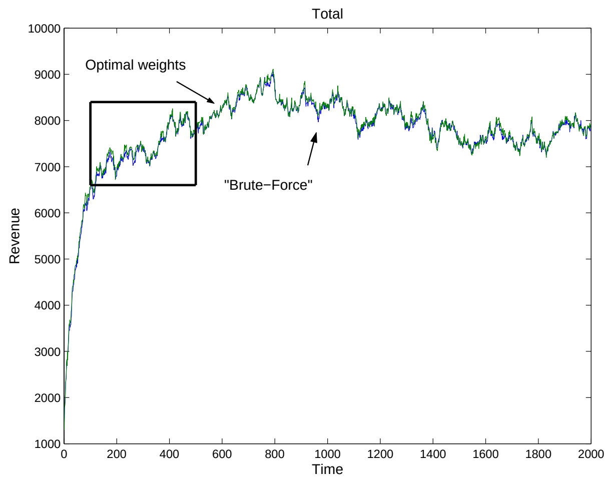

Figure 8. Revenue - Evolution of the revenue as a function of time for both the optimal weights (upper, green line) and ”brute-force” weights (lower, blue line).

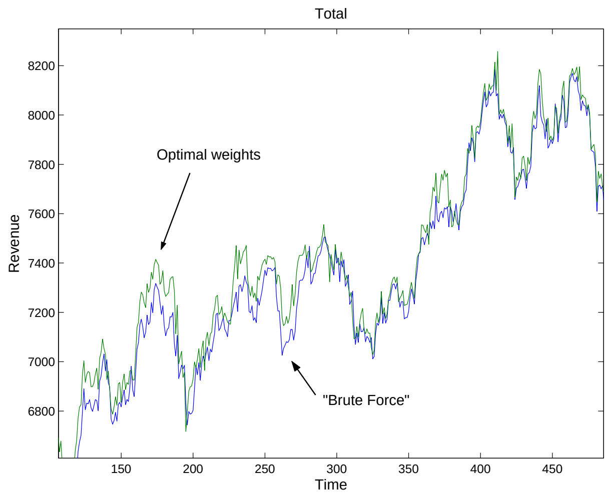

Figure 9. Close-up picture of revenue from Fig. 8.