Service Differentiation and Guarantees for TCP-based Elastic Traffic

Nidhi Hegde and Konstantin E. Avrachenkov

INRIA, 2004 route des Lucioles, B.P.93
06902 Sophia Antipolis Cedex, France
{Nidhi.Hegde, K.Avrachenkov}@sophia.inria.fr,

Abstract. We compare policies that offer differentiated services to TCP traffic, with the goal of providing some sort of performance guarantees to a premium class of traffic. Specifically, we study the effectiveness of a scheduling policy combined with various buffer management policies on the performance of competing TCP connections. In this work we consider a stochastic model for a class-based weighted-fair-queueing scheduling policy where packets of different classes are scheduled according to their pre-assigned static weight. We consider two buffer management policies : complete partitioning, and complete sharing with pushout at various thresholds. We consider two classes of TCP traffic in our model. Our goal is to propose mechanisms with easily adjustable parameters to achieve service differentiation as required. In the numerical results we show how the scheduling and buffer policies can be used to provide some sort of performance guarantee to the higher class.

1 Introduction

It is expected that the next generation IP networks will be able to provide service differentiation [1, 2]. With many different types of applications on such networks, users may also expect a performance guarantee. Our goal is to study the possibility of time priority and space priority schemes to provide such service differentiation, and some sort of performance guarantees. The DiffServ Architecture[1] proposes packet-level mechanisms to achieve service differentiation. Traffic is classified upon entry into the network and packets are forwarded in the core network based on such pre-classified per-hop behaviours. Two main classes of per-hop behaviour are the Expedited Forwarding(EF) class for stream traffic and the Assured Forwarding(AF) class for elastic traffic. Here we assume that for stream traffic, the EF class receives a guaranteed fixed bandwidth[3]. Thus, we concentrate on the remaining bandwidth being shared by elastic traffic. Users might want to have the possibility of further traffic differentiation for the elastic traffic. This can be achieved by using Assured Forwarding (AF) scheme [4]. In principle, the Assured Forwarding scheme might have up to four classes of traffic and up to three drop precedence levels within each traffic class. We restrict ourselves to the simplest case of two AF classes in our model, which is still tractable for analysis. Our model can also be viewed as a model for two different drop precedence levels within one AF class.

We consider mechanisms that do not include adjustments to the underlying architecture and transport protocols such as TCP. More desirable are simple mechanisms that can be easily implemented and easily tunable as required. To differentiate between classes of traffic we propose to use either time priority (scheduling) or space priority (buffer management) schemes or both. We will show that the very same policies may also be used towards the goal of performance guarantees. As the time priority scheme, we use stochastic Weighted Fair Queueing(WFQ) which is a packet level approximation of Generalized Processor Sharing(GPS) [5, 6]. For buffer management we consider both a partitioned buffer and a shared buffer space. For the shared buffer space, we propose a Pushout with Threshold (POT) policy to achieve further service differentiation. We suppose that the traffic is sent by greedy TCP sources [7–9]. Therefore the TCP loss-based control mechanism has to be taken into account.

To the best of our knowledge, the DiffServ scheme with stochastic weighted fair queueing and the buffer management schemes described above along with the TCP congestion control mechanism is analyzed for the first time here. Most of the related work concentrates on either scheduling priority or buffer priority, not both. In [10, 11], the authors considered RED with colours as a tool for differentiation. Such a scheme uses packet marking and various drop thresholds at the buffer. This may correspond to different drop precedence levels within an AF class. RED with colours can attain various levels of differentiation by adjusting several parameters. The advantage of the schemes that we propose in comparison to RED with colours is that our schemes have only two tunable parameters, allowing enough freedom to alter differentiation levels without being too complex. In [12], pushout without threshold and threshold-based drop mechanisms have been proposed as means to provide priority. A pushout scheme is proposed where a class 1 packet that arrives to a full buffer may still enter if there is at least one class 2 packet. This corresponds to a special case of our pushout with threshold scheme where the threshold is set to 0. We will show in Section 4 that setting a pushout threshold of at least one still achieves service differentiation without starvation of class 2 traffic. In addition, this threshold parameter may be adjusted to achieve varying levels of differentiation. For the threshold mechanism in [12], arriving lower priority packets are dropped if they number more than some given threshold. In [13], an alternate threshold policy is proposed where the drop threshold for any particular class of packets is based on the total number of all packets in the queue. Both such threshold-based drop schemes may be easy to implement, but suffer from wasted resources if the arrival rate of higher priority packets is not high and possible starvation of lower priority packets.

In this paper we consider a network where a bandwidth guarantee for a premium class of traffic may be required. For instance, consider an enterprise with multiple sites that may be connected through an IP network managed by some Internet Service Provider (ISP). Such a company might expect its traffic to receive service similar to that of a leased line[14]. In order to achieve some desired level of service for adaptive connections (such as TCP traffic), it is important that scheduling be combined with a proper buffer management scheme[14]. In this work, we first consider a partitioned buffer space with priority given at the scheduler. However, because such a buffer allocation wastes resources, we recommend a shared buffer space with the POT policy to provide service differentiation in addition to that achieved through scheduling. We use our numerical results to show how an ISP might adjust the scheduling and buffer parameters to achieve service differentiation as well as some sort of service guarantees.

The paper is organized as follows. Section 2 outlines the priority policies we consider and how the TCP performance measures are obtained. Section 3 details the performance analysis for each priority scheme proposed. Numerical results from our analysis are shown in Section 4, and we conclude with Section 5.

2 Model

Consider a bottleneck link with fixed capacity, that which is available after bandwidth has been allocated for the EF class. We study the case where greedy TCP connections of two AF classes with different service level requirements compete for this capacity. Specifically, our goal is to study how some sort of bandwidth guarantee may be offered to a high priority class of TCP traffic. One such scenario is an ISP providing connectivity between one company's multiple sites, as suggested in [14]. Such a company would want its intranet traffic between multiple sites to receive service similar to that of a leased line. Specifically, it might request that some portion of the total bandwidth be reserved for its traffic. The ISP would then classify traffic between this company's sites as high priority. We suggest guidelines on how some sort of bandwidth guarantee can be given to this class of traffic using simple scheduling and buffer management techniques. We combine a queueing system that models these priority policies with a TCP model in order to calculate performance measures of the types of traffic in the system. The "guarantees" are therefore on the average TCP throughput achieved.

We model the bottleneck link under consideration as a single server queue with two priority classes. Differentiation between the two service classes can be realized through priority at the server and/or priority in the buffer space. The priority policies we consider each contain a parameter which allows us to easily adjust the relative priority of the two classes. In what follows, we describe the policies we consider for each of these types of priority.

2.1 Scheduling Priority

We provide priority in scheduling through Stochastic WFQ which can be seen as a practical packet-level implementation of GPS. Each class i is given a weight of α_i where $\sum_i \alpha_i = 1$. At the end of each service time, a packet of class i is served with probability $\alpha_i / \sum_{j:N_j>0} \alpha_j$, where N_i is the number of packets of class i in the buffer. In our case of two classes, class 1 packets have a weight of $\alpha_1 = \alpha$ and class 2 packets a weight of $\alpha_2 = 1 - \alpha$. We may adjust α to give one class priority over the other. For instance, we may impose a strict priority for class 1 by setting α to 1. Such a priority policy may be combined with either a partitioned or shared buffer space. In what follows we explain buffer management schemes for both types of buffers, that allow for greater service differentiation or better resource utilization.

2.2 Buffer Priority

Complete Partitioning(CP). This system consists of a single server that serves two separate buffers. Each class i of packets has a buffer of size K_i. Our model is general enough to offer priority to a particular class by increasing its buffer size relative to the other. However, in Section 4 we present results for the case where both types have equal buffer size. We assume that in a partitioned buffer space, size of each buffer is not easily tunable. We consider only mechanisms that allow the network operator to easily adjust parameters to suit the requested performance guarantee.

Complete Sharing(CS). Packets of both type share a single buffer of size K. Neither class has priority at the buffer and may be differentiated only at the server. For such a buffer space, we can offer priority to the higher class with the Pushout with Threshold(POT) scheme. In such a policy, priority is given to type 1 traffic in the following way. A type 1 arriving packet that finds the shared buffer full may enter the buffer by pushing out a type 2 packet if the number of type 2 packets in the buffer is above a given threshold, K_2. Priority to type 1 class may be changed by adjusting the threshold K_2. For instance when $K_2 = K$, a type 2 packet is never pushed out by a type 1 packet arriving to a full buffer, and no priority is given to class 1. When K_2 is decreased to 0, a type 1 packet arriving to a full buffer always pushes out a type 2 packet if any, and type 1 has maximal priority over type 2 at the buffer. Note that for $K_2 = 0$, the lower priority traffic may experience starvation. We do not consider the drop threshold policy where incoming packets of a particular type are dropped if they number greater than a preset threshold, regardless of whether or not the buffer is full. Such a policy wastes resources and causes starvation of type 2 traffic if the arrival rate of type 1 traffic is not high.

CP is the simplest policy for implementation. However, it leads to under-utilization of resources. The CS buffer without a threshold policy utilizes the buffer space well, but does not provide any differentiation. The POT policy on the other hand, not only utilizes the resources well, but also provides differentiation in buffer management.

3 Performance Analysis

3.1 TCP Model

Let m_i be the number of class i connections at the link, S_i be the average sending rate of each TCP source corresponding to class i, L_i be the loss probability of class i packets, and $\overline{W}_i$ be the average waiting time for class i packets. We use the standard square-root equation for calculating the TCP sending rate of each source [8]:

$$S_i(L_i, \overline{W}_i) = \mathrm{MSS} \cdot \min\left(\frac{1}{d_i + \overline{W}_i}\sqrt{\frac{k}{L_i}}, \frac{M}{d_i + \overline{W}_i}\right), \tag{1}$$

where MSS is the maximum segment size, d_i is the two-way propagation delay, and M is the maximum window size of the receiver. The constant k represents the TCP model considered. For instance, in our results in Section 4, we consider $k = 2$ where the inter-loss times are assumed to be exponentially distributed and the delay ACK mechanism is disabled[7]. We assume fair link sharing within each class, and define the load for each class of traffic by:

$$\rho_i = m_i \frac{S_i}{C}$$

where C is the capacity of the link.

As a measure of quality of service, we use the throughput of each connection,

$$T_i = S_i(1 - L_i) \tag{2}$$

In the next subsection, we show how the packet loss probabilities and the average waiting time can be calculated as functions of ρ_1 and ρ_2. Namely,

$$L_i = L_i\left(\rho_1, \rho_2\right), \qquad \overline{W}_i = \overline{W}_i\left(\rho_1, \rho_2\right) \tag{3}$$

One can see that Equations (1) and (3) form a system of four equations in four unknowns. This system can be solved using a fixed-point approach.

3.2 Performance Measures

The loss probabilities and waiting time averages for each priority policy are computed by solving the respective queueing system. We assume that packets of class i arrive at the link according to an independent Poisson process with rate $\lambda_i = m_i S_i$. The service time for each packet is assumed to be exponential with rate $\mu_i = C$. Let N_i be the number of class i packets in the buffer. For each of the priority policies, the vector (N_1, N_2) will be a Markov Chain. As shown in the Appendix, we solve the respective Markov chain of each system using numerical techniques in order to obtain the steady-state probabilities $P(n_1, n_2) = \Pr[N_1 = n_1, N_2 = n_2]$, and $\bar{N}_i$, the average number of class i packets in the buffer. As shown below, we use these values to calculate the loss probability and average waiting time of each class.

Loss Probability. For the system with partitioned buffers (CP), the loss probability for each class is the probability that its buffer is full:

$$L_1 = \sum_{n_2=0}^{K_2} P(K_1, n_2), \qquad L_2 = \sum_{n_1=0}^{K_1} P(n_1, K_2) \tag{4}$$

With the shared buffer policy (CS), the loss probabilities of both classes are equal. This can be determined from the solution to a M/M/1/K+1 system with an arrival rate of $\sum_i S_i$ and a service rate of C. Namely, we have:

$$L_i = \frac{(1-\rho)\rho^{K+1}}{1-\rho^{K+2}},$$

$$\rho = \sum_i \rho_i$$

For the POT policy, the loss probability is calculated as follows. Let P_K denote the probability that the shared buffer is full, φ denote the probability that a class 1 arrival pushes out a class 2 packet, and ϕ denote the probability that a packet in the buffer is pushed out by a class 1 arrival. An arriving class 1 packet is lost only when $N_2 \leq K_2$:

$$L_1 = P_K - \varphi,$$

$$P_K = \frac{(1-\rho)\rho^{K+1}}{1-\rho^{K+2}}, \quad \varphi = \sum_{n_2=K_2+1}^{K} P(K - n_2, n_2)$$

A packet of class 2 is lost either when the buffer is full upon its arrival, or when it enters the buffer and then is pushed out by a class 1 arrival.

$$L_2 = P_K + (1 - P_K)\phi$$

We solve for ϕ by equating the rates of pushout loss: the rate of class 1 packets that push out a class 2 packet must equal the rate of accepted class 2 packets that are subsequently pushed by a class 1 arrival, as follows:

$$S_2(1 - P_K)\phi = S_1\varphi$$

Average Delay. For both CP and CS without POT, the average delay is determined by invoking Little's Law:

$$\overline{W}_i = \frac{\overline{N}_i}{T_i}$$

We use Little's Law also for the POT scheme, noting that only those arrivals that are eventually served are considered in the determination of average waiting time. Therefore, for both the average waiting time and loss probability calculations, it is irrelevant which class 2 packet is pushed out: the first in queue, the last in queue, or a random class 2 packet. The waiting time averages are as follows:

$$\overline{W}_1 = \frac{\overline{N}_1}{T_1}$$

$$\overline{W}_2 = \frac{\overline{N}_2}{S_2(1 - P_K)(1 - \phi)} = \frac{\overline{N}_2}{T_2}$$

4 Numerical Results

In this section we will show how the network operator may use our priority schemes to provide type 1 connections with a throughput guarantee. We consider two cases of core routers, those with a partitioned buffer space and those with a common buffer for both types of traffic. For the first type with partitioned buffer space, we use only stochastic WFQ to provide priority. For the case of a shared buffer, we use the pushout with threshold buffer management scheme in addition to stochastic WFQ at the server.

We consider a 100Mbps bottleneck link. For all our numerical results we have assumed the maximum window size at the sender $M = 20$, the average maximum segment size MSS of 512 Bytes, and a buffer space of 40 segments in total for both type 1 and type 2 packets. We assume that the high priority class is a premium class and therefore the number of this type of connections is very small relative to the number of low priority connections. We vary the number of high priority connections through this link, m_1, from two to five, and the number of low priority connections $m_2 = 20, \ldots, 50$. Our model is general to allow any values of m_1 and m_2, however we use these values for illustration purposes.

4.1 Partitioned Buffer

Let us consider the case where the total throughput of all high priority connections require some portion of the total bandwidth, and a partitioned buffer space. We consider this case as a form of bandwidth reservation, but only loosely so because since all connections are of greedy TCP sources, if the "reserved bandwidth" is not utilized, it is shared among the remaining low priority connections. We use Fig. 1 and 2 to show how the network operator may choose the scheduling parameter α to provide type 1 connections at least 20% of the total bandwidth. We note that in all cases, such performance *guarantees* are for the average TCP throughput T_1. For example, let us assume that type 1 traffic request 20% of total bandwidth and there will be three type 1 connections. Then, given the number of type connections, Fig. 1 can be used to set α. We note that in general, because of the stochastic nature of the arrival traffic, in order to guarantee some portion

x of the total bandwidth to type 1 traffic, it is not sufficient to set $\alpha = x$. The value of α depends not only on the arrival rates, or number of connections of both types of traffic, but also on the type of buffer management scheme.

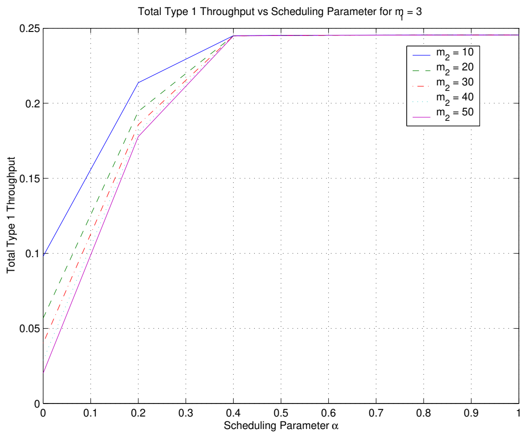

Fig. 1. Providing Bandwidth Guarantee with WFQ, $m_1 = 3$

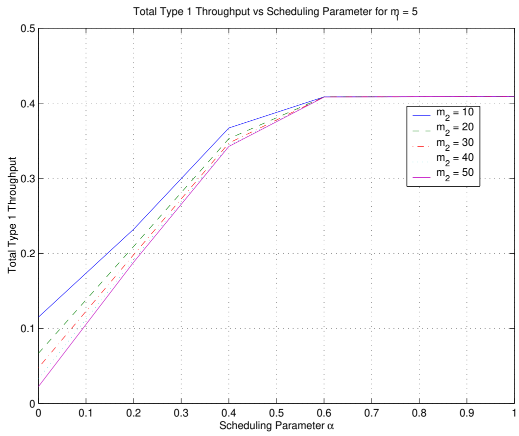

Fig. 2. Providing Bandwidth Guarantee with WFQ, $m_1 = 5$

We note that ideally, a TCP connection with no queueing delay and no packet losses will have a maximum sending rate of $S_i^{\max} = \frac{M \cdot \text{MSS}}{d_i} = 8.19$Mbps. Note also that in Figs. 1 and 2 as α is increased to 1, the total bandwidth occupied by type 1 traffic does not go beyond $m_1 S^{\max_1}$.

The remaining bandwidth which is not used by type 1 connections is shared equally by type 2 connections. For instance, when $m_1 = 2$, $m_2 = 40$, and $\alpha = 1$, type 1 connections get a total throughput of about 16.4 Mbps. Type 2 connections share the rest of the total overall throughput, and get a total throughput of 80.4 Mbps.

4.2 Shared Buffer

We now consider a shared buffer space with the pushout with threshold policy. We first offer priority only at the buffer, by setting $\alpha = 0.5$. Figure 3 shows how the pushout threshold K_2 may be set to achieve some guarantee for type 1 connections, when $m_1 = 5$. For instance, for a 20% total bandwidth guarantee for type 1 connections, may be set to a value between 40 and 37, depending on the number of type 2 connections. A relatively small change in K_2 is required to achieve this performance guarantee.

We may offer priority both at the buffer and at the scheduler. The network operator may find it more convenient to change one paramater at smaller time scales and the other at longer time scales. We show examples of when one parameter is set to a fixed value, the other can be changed to achieve the performance guarantee. In Fig. 4 we set $\alpha = 0.2$ for $m_1 = 5$, and show the change in total type 1 bandwidth as K_2 is varied, for various values of m_2. For instance, if type 1 connections require a total throughput of 0.2, K_2 can be set to a value between 37 and 28 depending on the number of type 2 connections.

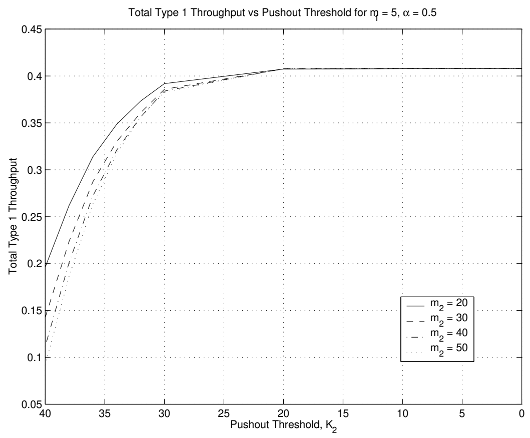

Fig. 3. Providing Bandwidth Guarantee with POT, $m_1 = 5$

In Fig. 5 we set $K_2 = 32$ and vary α for $m_1 = 5$. $K_2 = 32$ corresponds to 20% of the buffer space "reserved" for type 1 packets. If we now want to give type 1 connections a total throughput of 0.2, α can be set to a value between 0.14 and 0.22 depending on m_2.

5 Conclusion

We have proposed scheduling and buffer management schemes in order to achieve performance guarantees in IP networks in the form of average TCP throughput. For a partitioned buffer space, we have used a stochastic weighted-fair queueing policy at the scheduler. For a shared buffer, we have used a pushout with threshold policy at the buffer along with stochastic weighted-fair queueing at the server. We have combined analysis of such priority policies with the TCP model to study guarantees in the achieved throughputs of the TCP sources.

References

1. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service", RFC 2475, December 1998.
2. Kakevi Kilkki, *Differentiated Services for the Internet*, Macmillan Technical Publishing, 1999.
3. V. Jacobson, K. Nichols, and K. Poduri, "An expedited forwarding PHB group", RFC 2598, June 1999.
4. J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured forwarding PHB group", RFC 2597, June 1999.
5. Srinivasan Keshav, *An Engineering Approach to Computer Networking, ATM Networks, the Internet, and the Telephone Networks*, Addison-Wesley, 1997.
6. Abhay Parekh and Robert Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case", *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–57, June 1993.

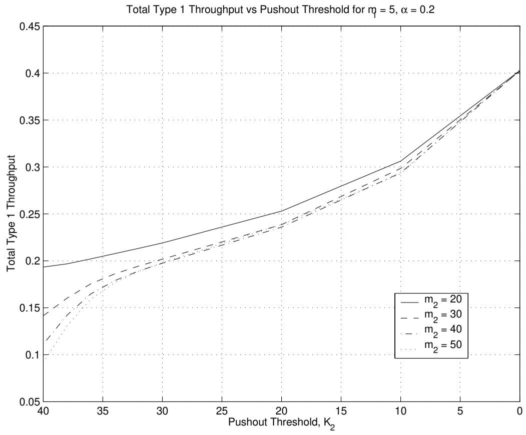

Fig. 4. Providing Bandwidth Guarantee with WFQ and POT, $m_1 = 5$

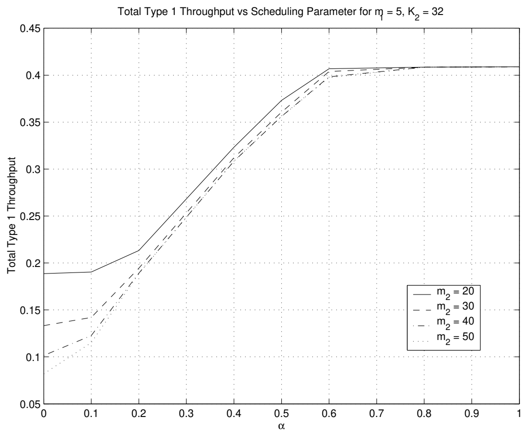

Fig. 5. Providing Bandwidth Guarantee with WFQ and POT, $m_1 = 5$

7. Eitan Altman, Konstantin E. Avrachenkov, and Chadi Barakat, "A stochastic model of TCP/IP with stationary random losses", in *Proceedings of ACM SIGCOMM'00*, Stockholm, Sweden, Aug. 28 - Sept. 1 2000.
8. Mathew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm", *Computer Communications Review*, vol. 27, no. 3, July 1997.
9. Jitendra Padhye, Victor Firiou, Don Towsley, and Jim Kurose, "Modeling TCP throughput: A simple model and its empirical validation", in *Proceedings of ACM SIGCOMM'98*, August 1998.
10. Chadi Barakat and Eitan Altman, "A markovian model for TCP analysis in a differentiated services network", in *Proceedings of the First International Workshop on Quality of future Internet Services (QofIS)*, Berlin, Germany, September 2000.
11. Sambit Sahu, Philippe Nain, Don Towsley, Christophe Diot, and Victor Firiou, "On achievable service differentiation with token bucket marking for TCP", in *Proceedings of ACM SIGMETRICS'00*, Santa Clara, CA, USA, June 18-21 2000.
12. Martin May, , Jean-Chrysostome Bolot, Alain Jean-Marie, and Christophe Diot, "Simple performance models of differentiated services schemes for the internet", in *INFOCOM (3)*, 1999, pp. 1385–94.
13. Eeva Nyberg, Samuli Aalto, and Jorma Virtamo, "Relating flow level requirements to diffserv packet level mechanisms", Tech. Rep., COST279, 2001, available at http://tct.hut.fi/tutkimus/cost279/.
14. Vijay P. Kumar, T. V. Lakshman, and Dimitrios Stiliadis, "Beyond best effort: Router architectures for the differentiated services of tomorrow's internet", *IEEE Communications Magazine*, pp. 152–64, May 1998.
15. Marcel Neuts, *Matrix-geometric solutions in stochastic models : an algorithmic approach*, John Hopkins University Press, 1981.

Appendix A

Complete Partitioning

The vector (N_1, N_2), representing the number of packets of each class in the respective buffer, is a Markov chain with state space $\{(n_1, n_2) : n_1 = 0 \ldots K_1, n_2 = 0 \ldots K_2\}$. The generator of this chain is a square matrix of size $(2(K_1+1)(K_2+1)+1)$ as follows:

$$Q = \begin{pmatrix} -B_1 & B_1 & & & \\ B_2 & E_0 & A & & \\ & D & E & \ddots & \\ & & \ddots & \ddots & A \\ & & & D & E+A \end{pmatrix} \tag{5}$$

$$\text{where} \quad B_1 = [\lambda_1 \lambda_2], \quad B_2 = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, A = \mathrm{diag}\{\lambda_2\},$$

$$E_0 = \begin{pmatrix} \mathbf{a} & \Lambda_1 & & & \\ \mathbf{d} & \mathbf{a} & \Lambda_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{d} & \mathbf{a} & \Lambda_1 \\ & & & \mathbf{d} & \mathbf{a}+\Lambda_1 \end{pmatrix}, \; E = \begin{pmatrix} \mathbf{a} & \Lambda_1 & & \\ \mathbf{b}_1 & \ddots & \ddots & \\ & \ddots & \mathbf{a} & \Lambda_1 \\ & & \mathbf{b}_1 & \mathbf{a}+\Lambda_1 \end{pmatrix}, \; D = \begin{pmatrix} 0 \; \mu_1 & & & \\ 0 \; \mu_2 & & & \\ & \mathbf{b}_2 & & \\ & & \ddots & \\ & & & \mathbf{b}_2 \end{pmatrix},$$

$\Lambda_i = \mathrm{diag}\{\lambda_i\}_{2\times 2}$, $\mathbf{a} = \mathrm{diag}\{-(\lambda+\mu_i)\}_{i=1,2}$,

$$\mathbf{d} = \begin{bmatrix} \mu_1 & 0 \\ \mu_2 & 0 \end{bmatrix}, \quad \mathbf{b}_1 = \alpha_1 \mathbf{d}, \quad \mathbf{b}_2 = \begin{bmatrix} 0 & \alpha_2\mu_1 \\ 0 & \alpha_2\mu_2 \end{bmatrix}$$

B_1 and B_2 are of size (1×2) and (2×1) respectively, and D, E, E_0, and A are of size $(2(K_1+1)\times 2(K_1+1))$. The generator matrix Q is of QBD-type with complex boundary behaviour. We use numerical techniques [15] to solve for the steady-state probabilities $P(n_1,n_2) = \Pr[N_1 = n_1, N_2 = n_2]$.

Pushout with Threshold

The vector (N_1, N_2) representing the number of packets of each type in the shared buffer is a Markov chain with state space $\{(n_1,n_2) : n_1+n_2 \le K\}$. The generator for this chain is as follows:

$$Q = \begin{pmatrix} -B_1 & B_1 & & & & \\ B_2 & E_0 & A_0 & & & \\ & D_1 & E_1 & A_1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & D_{K-1} & E_{K-1} & A_{K-1} \\ & & & & D_K & E_K+\Lambda_2 \end{pmatrix} \tag{6}$$

$$\text{where} \quad A_j = \begin{pmatrix} \Lambda_2 & & \\ & \ddots & \\ & & \Lambda_2 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{pmatrix}, \; E_j = \begin{pmatrix} \mathbf{a} & \Lambda_1 & & \\ \mathbf{b}_1 & \ddots & \ddots & \\ & \ddots & \mathbf{a} & \Lambda_1 \\ & & \mathbf{b}_1 & \tilde{\mathbf{a}}_j \end{pmatrix}, \; D_j = \begin{pmatrix} 0 \; \mu_1 & & & \\ 0 \; \mu_2 & & & \\ & \mathbf{b}_2 & & \\ & & \ddots & \\ & & & \mathbf{b}_2 \; \mathbf{v}_j \end{pmatrix}$$

$\tilde{\mathbf{a}}_k = \mathbf{a} + \Lambda_1 + \Lambda_2$ and $\mathbf{v}_k = \mathbf{0}$ for $k \le K_2$, $\tilde{\mathbf{a}}_k = \mathbf{a} + \Lambda_2$ and $\mathbf{v}_k = \Lambda_1$ for $k > K_2$. The matrices A_{K-j}, E_{K-j}, and D_{K-j} are of sizes $2j \times 2(j+1)$, $2(j+1)\times 2(j+1)$, and $2(j+1)\times 2(j+2)$ respectively

This generator is of type level-dependent QBD. We use numerical techniques to solve for the steady-state probabilities.