2002-03-21

Scheduling Flows with Unknown Sizes: Approximate Analysis

Scheduling Flows with Unknown Sizes: Approximate Analysis*

Liang Guo Ibrahim Matta
Computer Science Department
Boston University
Boston, MA 02215
{guol, matta}@cs.bu.edu

Technical Report BUCS TR-2002-009
March 21, 2002

Abstract

Previous studies have shown that giving preferential treatment to short jobs helps reduce the average system response time, especially when the job size distribution possesses the heavy-tailed property. Since it has been shown that the TCP flow length distribution also has the same property, it is natural to let short TCP flows enjoy better service inside the network. Analyzing such discriminatory system requires modification to traditional job scheduling models since usually network traffic managers do not have detailed knowledge about individual flows such as their lengths. The Multi-Level (ML) queue, proposed by Kleinrock, can be used to characterize such system. In an ML queueing system, the priority of a flow is reduced as the flow stays longer. We present an approximate analysis of the ML queueing system to obtain a closed-form solution of the average system response time function for *general* flow size distributions. We show that the response time of short flows can be significantly reduced without penalizing long flows.

1 Introduction

Previous job scheduling studies indicate that providing rapid response to interactive jobs which place frequent but small demands, can reduce the overall system average response time [1]. Such size-aware discriminatory scheduling algorithms have been shown, both experimentally and analytically (see [2] and references therein), to work extremely well when the job size distribution possesses the heavy-tailed (HT) property[1]. Since data transfer in a network can be modeled as a flow scheduling problem, and the HT property has been observed in the length of Internet transactions, especially Web file transfers, it is natural to design a network system that favors short file transfers.

*This work was supported in part by NSF grants CAREER ANI-0096045 and ANI-0095988.

[1]Note that the HT property only requires that the largest small number (say 10%) of jobs contribute most (say 90%) of the load to the system. This is different from the classical definition of heavy-tailed distribution, which requires the tail of the distribution to be of the power-law form.

As a result, before starting transmission of a flow[2], the network has to know the length of each flow in advance. Such information may not be readily available (e.g. as for dynamic web pages). Even if flow lengths are available, it may not be desirable to propagate this information to the network to keep it independent of application semantics [3]. Instead, one can let the network implicitly identify short flows by "testing" their status. Specifically, we assume some traffic controller inside the network could measure how much traffic a flow had inserted into the network. Henceforth, we refer to such measure as the "age" of a flow. Young (new) flows are always assigned to the highest priority. Once a flow's age exceeds a certain threshold, its priority is reduced and the data transfer rate becomes slower than that allocated to other "younger" flows.

Since the Internet is a packet-forwarding system, it can be well-approximated by a processor sharing queueing system, where jobs represent network flows. Discriminatory scheduling in the processor sharing queue has been explored by Kleinrock and Muntz in [4]. In fact, when the number of job classes is finite, the "testing" method above has been called the MultiLevel (ML) queueing system in that study, and the average response time function of jobs in such system has been derived for arbitrary job length distribution and Poisson arrivals.

However, the solution is given in the form of an integral equation and to date the equation has been solved only for job size distribution that has the form of mixed exponential functions. As mentioned earlier, the distribution of Internet flow lengths (sizes) obeys the HT property and can hardly be characterized in the form of a parsimonious combination of exponential distributions [5].

In this paper, we use a different approach, namely a conservation law by Kleinrock [1], to solve for the average response time in such system. To that end, we approximate the average response time of jobs by a linear function in the job size and solve for the stretch factors. We show by simulation that such approximation works well for HT job size distributions, although it does not work so well for exponential distributions.

In the next section, we introduce some recent work related to the design and analysis of such discriminatory system. In Section 3 we describe the queueing model and summarize the general approach to solving for the average response time of such time-shared system. The solution for the M/G/1 processor sharing system under the assumption that the response time of a job is linear in its size is given in Section 4. We conclude our paper in Section 5 with possible ways of extending our analytical model to more complicated systems.

[2]A flow is generally defined as a sequence of packets that share certain common properties. Here a flow refers to data packets belonging to the same transaction.

2 Related Work

The advantage of giving high priority to short jobs has been studied thoroughly in the past decades. Most of the research has been focused on optimal scheduling policies like Shortest-Remaining-Processing-Time first (SRPT) or Shortest-Job-First (SJF) (see [2] and references therein). To implement these optimal policies, information such as the remaining processing time of a job at any time is needed. It may be practically impossible to maintain such information in a large scale environment like the Internet, for which we seek a sub-optimal alternative scheduling algorithm. In his book [1], Kleinrock gives the queueing analysis of some sub-optimal heuristics such as the Foreground-Background (FB) scheduling and the MultiLayer (ML) scheduling, which are designed specifically for time-shared systems. Explicit-form results for M/G/1 queues are given for specific job size distributions (e.g., hyper-exponential). For more general distributions, the solution is written in the form of integral equations and may only be obtained through numerical methods. In this paper, we are able to obtain a closed-form solution using approximations that work well for job sizes possessing the heavy-tailed property, which is commonly observed in the Internet measurements.

Job-size aware scheduling has been implemented in end-systems. Recently, in [6], Harchol-Balter *et al.* implement the SRPT scheduling algorithm in a Linux web server. They then argue that since the file distributions in most web servers possess the HT property, such implementation can significantly enhance the system performance. However, to observe such enhancement, they have to assume that the bottleneck be at the outgoing interface of the web server, which is not always true in a wide-area network environment. In other words, preferential treatment to short jobs has to be provided inside the network as well for the sake of *end-to-end* response time. Moreover, since the number of priority levels in a machine is *finite*, their implementation is only an approximation of the real SRPT algorithm. They do not analyze this approximation scheme. In [7], Yang and de Veciana propose an optimal size-aware bandwidth allocation scheme which gives higher transmission rate to short TCP transactions. In their scheme, end-system TCP slows down its window increase rate once the session size exceeds a certain threshold. Therefore, without modification to TCP at *every* end-user, clients who are equipped with the new congestion control algorithm may lose bandwidth to those who are not. In their analysis, they formalize the bandwidth sharing problem as an optimization problem. The objective is to minimize the average response time for the entire system. They did not give in closed-form the response time for different sizes.

On the contrary, our objective in this paper is to compute the average response time for each individual flow size under general flow size distributions. Moreover, our numerical analysis considers the effect of TCP congestion control on flow scheduling. Therefore our results closely match those obtained from *ns*-2 [8], a packet-level simulator.

3 The M/G/1/ML Queueing System

We use the MultiLevel (ML) queueing model to describe a discriminatory service system made of a finite number of classes, without knowledge on input job sizes. The structure of the queueing model is illustrated in Figure 1.

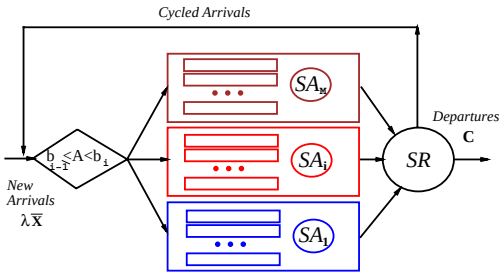


Figure 1: The ML queueing system

In such a system, whenever the server becomes idle, some customer[3] is allowed to be served for some amount of time, g_iQ, where g_i is the weight factor associated with i, the class that the customer belongs to, and Q is referred to as a quantum of service, which denotes the smallest unit of service time in the system. For ease of analysis, we assume $Q \to 0$. The customer departs the system if A, the total accumulated service she receives (denoting the "age" of this customer), equals her required service time; otherwise, the customer cycles back to reenter the system of queues at the end of her quantum. The class that a customer belongs to is determined by her age. Class i customers are those whose ages are between b_{i-1} and b_i, where b_i's are some predefined cutoff thresholds such that $0 = b_0 < b_1 < ... < b_M = \infty$, where M is the number of customer classes. Each class of customers have their own queue(s). Different scheduling algorithms can be used within each class (SA's in Figure 1). Another inter-class scheduler, denoted by SR, can be used to schedule customers from different classes.

Moreover, in this paper we use the following notations:

- λ (λ_i): the arrival rate of all (class i) customers.
- $B(x)$: the cumulative distribution of required service time for all customers.
- $\overline{X}$: the average service time over all customers.
- $\overline{X_i}$: the average amount of work served at class i.
- C: the service rate of the server.

[3]We use the terms customer and job interchangeably in this paper.

- $\rho = \frac{\lambda \overline{X}}{C}$: the normalized system load.
- $S_i = \lambda_i \overline{X_i}$: the amount of service at class i.
- $\overline{N_i}$: the average number of class i customers in the system.

.

We are more interested in the processor-sharing (PS) family of algorithms. Thus, we assume all intra-class SA_i algorithms are PS. In addition, for the inter-class SR algorithm, we consider two cases. The first is the priority queueing (PRIO) algorithm. That is, $g_i = \infty$ as long as there are no higher priority (at classes indexed lower than i) customers in the system. The second is called the Discriminatory Processor Sharing (DPS) algorithm. That is, SR is processor sharing but with $g_i > g_j$ for $i < j$. Following the argument in [9], for the DPS scheduling algorithm, if there are N_j jobs of class j present, $j = 1, 2, ..., M$, then the service rate that class j jobs receive is:

$$r_i = \frac{g_i N_i}{\sum_{j=1}^{M} g_j N_j} C \qquad i = 1, 2, ..., M \tag{1}$$

We refer to the first system as the ML-PRIO queue, and the second as the ML-DPS queue. In the special case of 2 job classes ($M = 2$) and $g_1 = \infty$, the two systems are equivalent.

We assume customers arrive according to a Poisson process, and the service distribution is arbitrary. Thus, the queueing system is refered to as M/G/1/ML-SR system, where SR can be either PRIO or DPS.

Our goal is to solve for the *average (expected) response time* for customers who require x total service time, denoted by $T(x)$. For convenience, we define $T_i(x_i)$ as the expected time to serve x_i unit of service while customer is at class i. Thus, we have, for $b_{i-1} \leq x < b_i$,

$$T(x) = \sum_{j=1}^{i-1} T_j(b_j - b_{j-1}) + T_i(x - b_{i-1}) \tag{2}$$

3.1 One Approach to Solve for $T(x)$

The main idea of our analysis is to apply the *Conservation Law for Work-conserving Time-Shared Systems*, proposed and proven by Kleinrock ([1], pages 197-199). Formally, it states the following:

Theorem 1 Kleinrock's Conservation Law for Time-Shared Systems. *For any M/G/1 system and any work-conserving queueing discipline, the average response time $T(x)$ for jobs of length x, satisfies the following equation:*

$$\int_{0-}^{\infty} T(x)[1 - B(x)]dx \quad = \quad \frac{\overline{X^2}}{2(1-\rho)} \tag{3}$$

where ρ, $B(x)$ and $\overline{X^2}$ represent the average load of the system, the cumulative distribution of job service times and the second moment of the job service time distribution, respectively.

In summary, the Conservation Law states that, no matter how the jobs are scheduled, the average unfinished work in the system, which is written as a function of the average response time for jobs of different sizes, must be invariant.

Therefore, if $T(x)$ has only one free variable, we can utilize the Conservation Law to solve for the closed-form of $T(x)$, given the job service time distribution and average system load.

4 Solving M/G/1/ML with Service Times possessing HT Property

In [1], Kleinrock gives a general solution for the M/G/1/ML-PRIO queue in the form of integral equations. Therefore, to obtain a closed-form solution, the service time distribution $B_i(x)$ for the portion of service classified as class i customers is required to take the form of $1 - q(x)e^{-\beta x}$, where $q(x)$ is a polynomial function of x. Using a similar approach, in [9], Fayolle *et al.* attack the problem of deriving M/G/1/DPS response functions. The solutions of [1] and [9] can be combined to obtain M/G/1/ML-DPS response time functions. However, so far we are only able to attack the special case of $M = 2$ and the solution is again written in the form of integral equations and a closed-form solution may only exist for specific distributions, as shown in Appendix A. The major difficulty is that, the instantaneous service rate each job gets not only depends on the number of concurrent jobs of the same class, but jobs from other classes. The aftermath of such dependence is that the average response time function for a job of specific size generally takes a non-linear form and the shape strongly depends on the underlying job size distribution. This can be seen from Equations (18) and (19) in Appendix A where the average number of class i jobs when a tagged job has received service t is a function of s.

However, if we assume that such dependence is small so that the average response time function is approximately linear in the job size, we can use the Conservation Law to solve for the average response time function for arbitrary job size distributions with finite mean and second moment, as we show in this section.

Approximation 1 *It has been shown that the average response time in an M/G/1 Processor Sharing queueing system is proportional to the job's required service time ([1], page 169). Intuitively, each arriving unit of job has to wait the same amount of time before it gets served. In other words, assuming each class of customers still arrives in a Poisson process, the response time function is a strictly linear function written as:*

$$T_i(x_i) = \theta_i x_i \tag{4}$$

where θ_i denotes the penalty factor for class i.

It has been shown in [1] that the penalty factor for an M/G/1/PS system equals

$$\theta = \frac{1}{1-\rho}$$

where ρ represents the normalized system load. Therefore, if we assume the arrivals of jobs of different classes in a DPS system are independent, then the penalty factor for class i can be written as:

$$\theta_i = \frac{1}{1-\rho_i} = \frac{1}{1-\frac{\lambda_i \overline{X_i}}{C_i}} \tag{5}$$

where C_i denotes the average service rate for class i jobs.

We now make the second approximation in order to obtain C_i.

Approximation 2 *We replace all N_i's in Equation (1) by their expectations $\overline{N_i}$'s and compute C_i as:*

$$C_i = \frac{g_i \overline{N_i}}{\sum_{j=1}^{M} g_j \overline{N_j}} C \qquad\qquad i = 1, 2, ..., M \tag{6}$$

From (4),(5) and (6), we are able to compute the average response time *if we know* $\overline{N_i}$, the average number of jobs of class i. Fortunately, since we are dealing with a work-conserving system, by Little's Law, we have the following relationship between $\overline{N_i}$ and $\overline{T_i}$:

$$\overline{N_i} = \lambda_i \overline{T_i(x_i)}$$

Now we need to know the average time customers spend in class i, i.e., during the interval $[b_{i-1}, b_i)$, for all job sizes. This is very easy to obtain for processor sharing systems since $T_i(x_i)$ takes the form of a linear function. That is, we have:

$$\overline{T_i(x_i)} = T_i(\overline{X_i}) = \theta_i \overline{X_i}$$

Plugging these equations into (6), we now have:

$$C_i = \frac{g_i \lambda_i \theta_i \overline{X_i}}{\sum_{j=1}^{M} g_j \lambda_j \theta_j \overline{X_j}} C \tag{7}$$

From (5) and (7) we now get our first set of equations:

$$\theta_i = (1 - \frac{\lambda_i \overline{X_i} \sum_{j=1}^{M} g_j \theta_j \lambda_j \overline{X_j}}{g_i \theta_i \lambda_i \overline{X_i} C})^{-1} = (1 - \frac{S_i \sum_{j=1}^{M} g_j \theta_j S_j}{g_i \theta_i S_i C})^{-1} \tag{8}$$

where S_i can be computed as:

$$S_i = \lambda\{\int_{b_{i-1}}^{b_i^-} (x - b_{i-1})dB(x) + (b_i - b_{i-1})[1 - B(b_i)]\} \tag{9}$$

Also, from Kleinrock's Conservation Law (cf. Equation (3)) and Equations (2) and (4) we have:

$$\int_{0-}^{\infty} T(x)[1 - B(x)]dx = \sum_{i=1}^{M} \int_{b_{i-1}^-}^{b_i} [\sum_{j=1}^{i-1} \theta_j(b_j - b_{j-1}) + \theta_i(x - b_{i-1})][1 - B(x)]dx = \frac{\overline{X^2}}{2(1-\rho)} \tag{10}$$

The set of equations have degree of freedom of M (θ_i's) and we have M independent linear equations, we can thus solve for θ_i's.

4.1 Validation by Simulation

Notice that the solutions above apply to general distributions which have finite first and second moments. However, we do need the above two approximations to obtain the response time function $T(x)$. More specifically, we assume:

- Arrivals at each level (class) are Poisson.
- The average service rate each class receives is independent of each other and can be approximated by Equation (6).

We now study how these approximations affect the accuracy of the analysis for different service time distributions. As an example, we show here the cases where jobs follow the Bounded Pareto distribution $B_{BP}(x, \alpha, k, p)$ and generalized Exponential distribution $B_{EXP}(x, \mu, k)$ [10], i.e.,

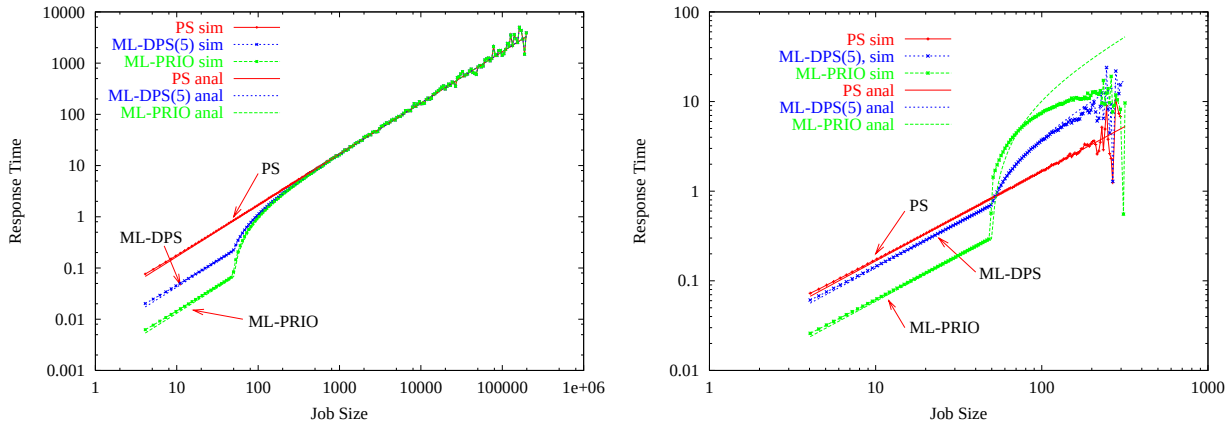
$$B_{BP}(x, \alpha, k, p) = \begin{cases} 0 & x < k \\ \frac{1-(k/x)^\alpha}{1-(k/p)^\alpha} & k \le x \le p \\ 1 & x > p \end{cases} \tag{11}$$

$$B_{EXP}(x, \mu, k) = \begin{cases} 0 & x < k \\ 1 - \mu e^{-\mu(x-k)} & x \ge k \end{cases} \tag{12}$$

The Bounded Pareto distributions have finite first and second moments, but they do possess the *HT property*, as defined in [2]. On the contrary, the generalized Exponential distribution does not have such property since the probability of having large jobs is very small.

Given the job size distributions, we can now compute the penalty factors θ_i's as from Equations (8)–(10). We also use simulation to validate our analysis. We study the case of a two-level system and the cutoff size is set to $b_1 = 50$ (about two times the average size). For the ML-DPS system, we set the weight factor to be $\mathbf{g} = (5, 1)$, i.e., each class 1 job gets 5 times unit of service as each

class 2 job. We also let $g_1 = \infty$ to obtain results for the ML-PRIO scheme. Figures 2(a) and 2(b) show the simulation as well as analytical results for cases in which job sizes follow Bounded Pareto distribution $B_{BP}(x, 1.2, 4, 200000)$ and the generalized Exponential distribution $B_{EXP}(x, 17.243, 4)$, respectively. The two distributions have the same mean and minimum value but only the former possesses the HT property. For all the simulations, we assume $C = 2000$ and $\lambda = 91.32$. Thus, the total load on the system is approximately 0.97. In the figures, PS denotes the nominal processor sharing scheme (where g_i's are all equal to 1).



(a) Bounded Pareto input

(b) Exponential input

Figure 2: Performance Comparison of PS, ML-PRIO and ML-DPS

The following observations can be made:

- For bounded Pareto input, the response time function at the second level queue ((i.e., job sizes greater than 50) can be well-approxiimated by a linear function in size, thus our analysis gives very accurate prediction. On the contrary, our analysis is not accurate for Exponential input. The actual response function penalizes more jobs whose sizes are just above the cutoff threshold. We also notice that our analysis still gives relatively good approximation under ML-DPS where the relative weight of the low priority jobs is not too small.

- When the job size distribution has the HT property, size-aware scheduling significantly reduces the response time of small jobs but only slightly increases the response time of long jobs. The ML-PRIO scheme, which gives absolute priority to short jobs, can reduce small job response time by a factor of 15, while only increase long job response time by a factor of 1.008. The ML-DPS scheme performs between ML-PRIO and the PS scheme.

- In case of exponential distributions which have very light tails, the benefit of giving preferential

treatment to short jobs (e.g. in ML-PRIO, a factor of 4.02) is achieved at the expense of significantly sacrificing long jobs' performance (by a factor of 8.29 in ML-PRIO). Moreover, although a very large weight (a factor of 5) is given to short jobs, the overall performance enhancement by employing the DPS scheduling algorithm is limited (a factor of 1.323). This is vastly different from what we observed in the previous scenario where file sizes possess the HT property, in which case the same DPS scheduling algorithm can reduce short job response times by a factor of 3.25.

We now give an intuitive explanation of why a linear response time function is a good approximation for job size distributions with a HT property but not for those without. We notice that the asymptotic behavior of $T(x)$ as $x \to \infty$ is always $T(x) \sim \frac{x}{1-\rho}$ [9]. With our analysis, when the job size distribution has the HT property, the worst case penalty factor for large jobs (generating most of the load of the system) is already very close to $\frac{1}{1-\rho}$, thus the approximation is good.

5 Discussion and Future Work

In this paper, we assume a fluid model and ignore the effect of TCP congestion control algorithm on bandwidth sharing, especially for small flows. Previous studies (see [11] and references therein) reveal that TCP can only statistically achieve idealized bandwidth sharing when the size of flows is long enough. On the contrary, due to the conservative nature of TCP congestion control, short TCP flows usually receive less share than long flows when competing for bandwidth under the same conditions [12].

In a separate technical report [13], we introduce a numerical approach to take into account TCP congestion control. The main idea is to derive the response time function of a typical Web transaction through the techniques introduced in [14], and apply Kleinrock's conservation law to numerically obtain the fixed-point solution of the system. This numerical solution can be used to engineer TCP traffic under proportional dropping queue management, which gives higher packet dropping rate to long TCP flows. We are currently investigating other more advanced active queue management policies (e.g., Virtual Queue based management algorithms [15]) to exercise better control over different classes of flows. We are also investigating the system behavior under overload conditions [11].

References

[1] L. Kleinrock, *Queueing Systems, Volume II: Computer Applications*, ISBN 0-471-49111-1. John Wiley & Sons Inc., 1976.

[2] N. Bansal and M. Harchol-Balter, "Analysis of SRPT Scheduling: Investigating Unfairness," in *Proc. of ACM SIGMETRICS 2001*, Boston, MA., June 2001.

[3] J.H. Saltzer, D.P. Reed, and D.D. Clark, "End-to-End Arguments in System Design," in *Proc. of Second International Conference on Distributed Computing Systems (ICDCS'81)*, April 1981, pp. 509–512.

[4] L. Kleinrock and R.R. Muntz, "Processor Sharing Queueing Models of Mixed Scheduling Disciplines for Time Shared Systems," *Journal of the ACM*, vol. 19(3), pp. 464–482, July 1972.

[5] D. Starobinski and M. Sidi, "Modeling and Analysis of Power-Tail Distributions via Classical Teletraffic Methods," *Queueing Systems (QUESTA)*, vol. 36 (1-3), pp. 243–267, November 2000.

[6] M. Harchol-Balter, N. Bansal, B. Schroeder, and M. Agrawal, "Implementation of SRPT Scheduling in Web Servers," Tech. Rep. CMU-CS-00-170, School of Computer Science, Carnegie Mellon Univ., Pittsburgh, PA, 2001, Submitted for Publication.

[7] S.-C. Yang and G. de Veciana, "Size-based Adaptive Bandwidth Allocation: Optimizing the Average QoS for Elastic Flows," Tech. Rep., Dept. of Electrical and Computer Engineering, Univ. of Texas at Austin, Jan. 2001, http://www.ece.utexas.edu/~scyang/research.html.

[8] E. Amir et al., "UCB/LBNL/VINT Network Simulator - ns (version 2)," Available at http://http://www.isi.edu/nsnam/ns/.

[9] G. Fayolle, I. Mitrani, and R.Iasnogorodski, "Sharing a Processor among Many Job Classes," *Journal of the ACM*, vol. 27(3), pp. 519–532, July 1980.

[10] N.L. Johnson and S. Kotz, *Continuous Univariate Distributions-1*, Houghton Mifflin Co., Boston, 1970.

[11] S. Ben Fredj, T. Bonald, A. Proutiere, G. Régnié, and J.W. Roberts, "Statistical Bandwidth Sharing: A Study of Congestion at Flow Level," in *Proc. ACM SIGCOMM 2001*, San Francisco, CA, September 2001.

[12] L. Guo and I. Matta, "The War Between Mice and Elephants," in *Proc. ICNP 2001*, Riverside, CA, November 2001.

[13] L. Guo and I. Matta, "Differentiated Control of Skewed Web Traffic: A Numerical Analysis," Tech. Report, Dept. of Computer Science, Boston Univ., 2002, http://www.cs.bu.edu/techreports/.

[14] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP Latency," in *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[15] S. Kunniyur and R. Srikant, "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management," in *Proceedings of ACM SIGCOMM'01*, San Diego, CA, August 2001.

[16] R.W. Wolff, "Poisson Arrivals See Time Averages," *Operations Research*, vol. 20, pp. 223–231, 1982.

A Solving M/G/1/ML-DPS with Two Priority Levels

We show in this section that the exact solution to M/G/1/ML-DPS system may be very complicated and a closed-form may only exist for specific distributions. We solve an M/G/1/ML-DPS system with two levels, i.e., $M = 2$. For ease of presentation, we use the following notation: $g = \frac{g1}{g2}$, the ratio of service weights between that of the high priority class (short customers) and the low priority class (long customers); and b, the cutoff threshold.

We use a similar approach as in [9]. Note that $T(s)$, the response time for a job of size s, can also be interpreted as the average time necessary for a job whose required service time is greater than s to attain service s. Thus $\Delta T(s) = T(s + \Delta s) - T(s) = T'(s)\Delta s$ is the expected time for a job to increase its attained service from s to $s + \Delta s$, given that the job size is larger than $s + \Delta s$.

We now tag a job of size greater than s, and study the average time it needs to increase its attained service time from s to $s + \Delta s$. As stated in [9], if there are N_1 short jobs and N_2 long jobs present in the system, $\Delta T(s) = E[\Delta s + \sum_{i=1}^{2} \frac{g_i}{g_k} N_i \Delta s]$, where k is the class index of the tagged job. Thus, we can compute $T'(s)$ as:

$$T'(s) = \begin{cases} 1 + \overline{N_1} + \frac{1}{g}\overline{N_2} & s \leq b \\ 1 + g\overline{N_1} + \overline{N_2} & s > b \end{cases} \tag{13}$$

We now need to compute $\overline{N_i}$, the expected number of jobs of class i, conditioned on the attained service time of the tagged job. For convenience, we write $\overline{N_i}$'s in two parts, that is:

$$\overline{N_i} = n_i^- + n_i^+$$

where n_i^- and n_i^+ denote the number of class i jobs which arrive before and after the tagged job, respectively.

Before computing $\overline{N_i}$'s, let us look at some general results:

Theorem 2 Average Number of Jobs. *If jobs arrive according to a Poisson process with rate λ, denote by $n_{u,v}^-$, the average number of jobs which satisfy the following two conditions:*

(1) the job arrives before the tagged job does, and has received u unit of service when the tagged job arrives, and,

(2) stays in the system and receives another v unit of service when the tagged job receives some s unit of service,

and denote by $n_{u,v}^+$, the average number of jobs which satisfy the following two conditions:

(1) the job arrives after the tagged job has received u unit of service, and,

(2) stays in the system when the tagged job attains s unit of service and receives another v unit of service,

then $n_{u,v}^-$ and $n_{u,v}^+$ can be computed as:

$$n_{u,v}^- = \lambda T'(u)[1 - B(u+v)]du \tag{14}$$

$$n_{u,v}^+ = \lambda T'(u)[1 - B(v)]du \tag{15}$$

Proof: The proof is similar to that in [9]. To compute $n_{u,v}^-$, note that the probability that a job received u unit of service at the arrival of the tagged job and then attains v is:

$$P[A > u + v | A > u] = \frac{1 - B(u+v)}{1 - B(u)} \tag{16}$$

On the other hand, from Little's Law and the PASTA property [16], the expected number of jobs whose attained service time is u when the tagged job arrives is given by:

$$\overline{N}[A > u] = \lambda[1 - B(u)]\Delta T(u) = \lambda[1 - B(u)]T'(u)du \tag{17}$$

Multiplying (16) and (17) yields (14).

To compute $n^+_{u,v}$, recall that the expected time taken by the tagged job to increase its attained service from u to $u + \Delta u$ is $T'(u)du$, thus by Little's Law, on average $\lambda T'(u)du$ jobs arrive during that time. The probability that one of these arrivals is still in the system and attains service v while the tagged job attains s is equal to $[1 - B(v)]$. We thus get Equation (15). □

We now present how to get $\overline{N_i}$'s. We first write down the final results, and then only gives the derivation for $\overline{N_2}$ given $s > b$ (i.e. the tagged job is classified as long) due to limited space. The other cases can be obtained similarly.

$$\overline{N_1} = \begin{cases} \lambda\{\int_0^{b-s} T'(u)[1 - B(s+u)]du + \int_0^s T'(u)[1 - B(s-u)]du\} & s \le b \\ \lambda\{\int_{g(s-b)}^{b} T'(u)[1 - B(b-u+g(s-b))]du + \int_b^s T'(u)[1 - B(g(s-u))]du\} & s > b \end{cases} \quad (18)$$

$$\overline{N_2} = \begin{cases} \lambda\{\int_b^{\infty} T'(u)[1 - B(u+gs)]du + \int_{b-s}^{b} T'(u)[1 - B(u+g(s-b+u))]du\} & s \le b \\ \lambda\{\int_0^b T'(u)[1 - B(s+\frac{u}{g})]du + \int_b^{\infty} T'(u)[1 - B(u+s-b+\frac{u}{g})]du + \\ \qquad \int_0^b T'(u)[1 - B(s-\frac{u}{g})]du + \int_b^s T'(u)[1 - B(s-(u-b)-\frac{b}{g})]\} & s > b \end{cases} \quad (19)$$

We now show how to get Equation (19) given $s > b$. From Theorem 2, given u, the relative position of a specific job j to the tagged job k, and the target service s, we only need to compute the additional service v that job j could attain if its total required service time is larger than $u + v$.

When $s > b$, that is, the tagged job k is at the low priority class, we can use Figure 3 to illustrate how to compute v for job j that now belongs to the low priority class (i.e., one of those N_2 jobs).

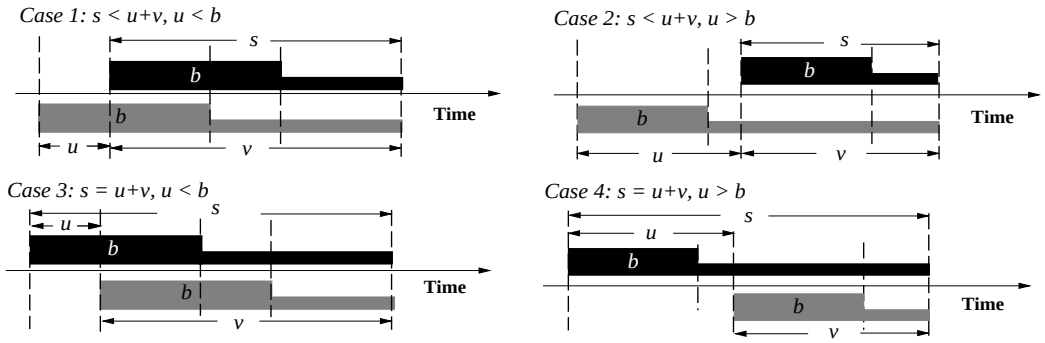


Figure 3: Cases for computing $\overline{N_2}$ with $t > b$

As shown in the Figure, there are 4 cases, the first two for n_2^- and the other two for n_2^+. The tagged job k is shown in black boxes and job j is shown in grey boxes. The height of the boxes represents the service rate assigned to the job. Thus, according to the ML-DPS rule, the rate is reduced after b units have been serviced. Given the parameters u, t and b, and the relative weight ratio g, it is straightforward to obtain v. For example, for case 1, $v = s - u + \frac{1}{g}u$, since when job k receives service from $b - u$ to b (totally u unit), job j can receive $\frac{1}{g}u$ service, and job k and j are served at the same rate otherwise. Applying Equation (14) in Theorem 2 and integrating over all valid u (u must be

less than b in this case), we get the first term in the right-hand side of Equation (19) for $s > b$. The remaining terms can be derived similarly.

Since now we obtain $\overline{N_i}$'s conditioned on the attained service s, we can plug them in Equation (13) to obtain the solution for $T'(s)$ in the form of integrodifferential equations. Note that by letting $g \to \infty$, we also obtain solution for the M/G/1/ML-PRIO system, which is consistent with that given in [1]. Our analysis here illustrates how the models become intractable for general job size distributions $B(.)$. In Section 3, our approximation allowed us to overcome such difficulties.