1 Class Based Thresholds and Optimal Parameters

When selecting the optimal parameters different traffic patterns under CBT, we consider many factors. These factors are based on the goals of CBT. CBT attempts to isolate misbehaving traffic from both TCP and multimedia traffic. It also seeks to offer low latency and a low drop-rate for multimedia. CBT accomplishes this goal by limiting the average queue occupancy allowed for a given class. This results in an effective allocation of bandwidth on the outbound link. In our experiments we use three classes, TCP, multimedia, and "other". A weighted queue average is maintained to track the average queue occupancy by each class.

The key to obtaining good performance from CBT lies in accurately setting the threshold values for each class. These threshold settings vary with the expected traffic load. To determine the correct threshold settings we must consider the amount of bandwidth to allocate for each class. Naively, one may assume that setting the ratio of the thresholds between the classes equal to the desired ratio of their bandwidths would give the desired result. Unfortunately, the thresholds are allocated in terms of packets and the classes may have widely varying average packet sizes. Table 1-1 shows the packet sizes and maximum load generated for each class of traffic used in our experiments. For example, the BULK traffic has an average packet size of 1,440 bytes, while proshare has an average packet size of 700 bytes. We must also include the packet headers in the total packet size. In our experiments the TCP headers are 60 bytes and the UDP headers are 28 bytes. Setting a 1:1 ratio between the thresholds would result in a 2:1 ratio in bandwidth on the outbound link.

Traffic Type	**Packet Size**	**Maximum Load**
BULK	1439 + *60* = 1499	> 10 Mb/s
HTTP	1062 + *60* = 1122	1101 KB/s
UDP Blast	1047 + *28* = 1075	> 10 Mb/s
Proshare	700 + *28* = 728	163KB/s
MPEG	811 + *28* = 839	190KB/s (150KB/s)

Table 1-1 Packet Sizes and Desired Bandwidth by Traffic Type

So, to allocate bandwidth we must consider the ratio in the number of bytes represented by the threshold settings between the classes. Determining the correct threshold settings for a given traffic mix requires using a set of formulas based on the bandwidth allocations, the average packet size for each class, the capacity of the outbound link, and the desired total latency. Table 1-2 describes the symbols used to calculate the proper threshold settings.

Name	**Symbol**	**Value** (if constant)
Outbound Link Capacity	$C_{outbound}$	10Mb/s
Number of Classes	N	3
Queue induced latency	L	configuration parameter
Threshold for class i	Th_i	output
Bandwidth allocation for class i	B_i	configuration parameter
Average packet size for class i	P_i	configuration parameter

Table 1-2 Variables and Constants for setting CBT thresholds

After using the equations in sections 1.1 - 1.3 below to determine the threshold settings, we then empirically evaluate the performance of CBT with a range of traffic mixes and threshold settings to evaluate the accuracy of these initial calculations and to determine the optimal parameter settings for each traffic mix. These experiments are shown in section **Error! Reference source not found.**.

1.1 Equations for Bandwidth Allocation

The expected average worst-case latency can be defined as shown in Equation 1.1. This expression represents the sum of the worst-case average number of bytes enqueued for each class divided by the speed of the outbound link. Any packet placed in the outbound queue should, on average, spend no longer in the queue than the time it takes for the queue to drain. In the worst case, all classes are maintaining an average queue occupancy equal to their threshold value and the amount of time it takes for that queue to drain is *L*. Note that *L* is the same for all classes since all classes share the same queue.

$$L = \left(\sum_{i=1}^{N} Th_i P_i \right) \frac{1}{C_{outbound}} \tag{1.1}$$

The bandwidth allocated for each class can be expressed as shown in Equation 1.2. That is, each class's bandwidth is a share of the outbound link equal to its share of the total bytes enqueued. This is based on an assumption that the link is in a constant state of overload with the queue never draining.

$$B_i = \frac{P_i Th_i}{\sum_{i=1}^{N} \left(P_i Th_i \right)} C_{outbound} \tag{1.2}$$

Equation 1.2 is based on an assumption. The sum of all of the bandwidth allocations should equal the capacity of the link, as shown in Equation 1.3.

$$C_{outbound} = \sum_{i=1}^{N} B_i \tag{1.3}$$

Given the desired bandwidth allocations and average packet sizes for two classes on the outbound link, we can also determine the ratio between the threshold settings. This is the key to using allocation of queue space to manage allocation of bandwidth on the outbound link. The outbound queue can be thought of as an extension of the outbound link, with the ratios of bytes in the queue matching the ratios of bytes on the link. The relationship between the thresholds for two classes can be expressed as shown in Equation 1.4. The ratio between the bandwidth allocated to each class is also the ratio between the maximum average number of bytes that can be enqueued.

$$\frac{B_i}{B_j} = \frac{P_i Th_i}{P_j Th_j} \Rightarrow \frac{B_i P_j}{B_j P_i} = \frac{Th_i}{Th_j} \tag{1.4}$$

From this we can see that we can express the threshold requirement for a given class (Th_i) in the queue in terms of the threshold for another class (Th_j) as shown in Equation 1.5.

$$Th_i = \frac{B_i P_j Th_j}{B_j P_i} \tag{1.5}$$

Using substitution, we can express the latency in terms of the threshold of a single class, *j*, as shown in Equation 1.6.

$$L = \left(\sum_{i \neq j}^{N} \frac{B_i P_j Th_j}{B_j} + P_j Th_j \right) \frac{1}{C_{outbound}} \tag{1.6}$$

Using this equation, we can solve for Th_j, as shown in Equation 1.7 and then, using the relationship from Equation 1.5, we can solve for the other threshold values.

$$L = \left(\sum_{i \neq j}^{N} \frac{B_i P_j Th_j}{B_j} + P_j Th_j \right) \frac{1}{C_{outbound}} \Leftrightarrow \left(\sum_{i=0}^{N} \frac{B_i P_j Th_j}{B_j} \right) \frac{1}{C_{outbound}} \Leftrightarrow \frac{\left(\sum_{i=0}^{N} \frac{B_i}{B_j} \right) P_j Th_j}{C_{outbound}} \tag{1.7}$$

$$Th_j = \frac{L \cdot C_{outbound}}{P_j \left(\sum_{i=1}^{N} \frac{B_i}{B_j} \right)}$$

Since the bandwidth of all of the classes sums to the total outbound link capacity (Equation 1.3) we can simplify Equation 1.7 to that shown in Equation 1.8. This allows us to express the threshold for a given class in terms independent of the other classes.

$$Th_j = \frac{L \cdot C_{outbound}}{\frac{P_j}{B_j} \sum_{i=1}^{N} B_i} = \frac{B_j L \cdot C_{outbound}}{P_j C_{outbound}} = \frac{B_j L}{P_j} \tag{1.8}$$

1.2 Equations for Re-Allocation of Bandwidth

One may also consider the resource allocation in terms of shares for each class. The expected maximum average latency can still be defined as shown in Equation 1.1.
Starting with Equation 1.2, we can express the share assigned to given class, *i*, as shown in Equation 1.9.

$$s_i = \frac{B_i}{C_{outbound}} \Leftrightarrow \frac{P_i Th_i}{\sum_{i=1}^{N} (P_i Th_i)} \tag{1.9}$$

Of course, Equation 1.3 is simply an expression of the idea that the shares assigned to each class sum to 1, as shown in Equation 1.10.

$$1 = \sum_{i=1}^{N} s_i \tag{1.10}$$

A note on the shares summing to one: The requirement that the shares sum to one is merely to simplify the following equations and discussion. If the shares do not sum to one (or bandwidth allocations do not sum to the capacity of the link) one may still apply these equations by adding an additional phantom class, $N + 1$, with a bandwidth allocation that is the remaining unused capacity of the link. This phantom class will, of course, be over-provisioned since its average load will be zero. This unused capacity will simply be divided among the other classes as explained in the subsequent equations.

Figure 1-1, below, illustrates the relationship between *B*, *B¢*, and *Load*. B_i is the initial bandwidth allocation for a given class, *i*. $B¢_i$ is the bandwidth re-allocation that results when one class is under load. $Load_i$ is the actual load being generated by the class, *i*. In this example there are 3 classes, *TCP*, *multimedia*, and *other*. In the figure, the y-axis represents KB/s. The x-axis represents a range of bandwidth allocation combinations. The initial bandwidth allocations (*B*) are represented by blue lines. In this example the bandwidth allocation for other (B_{other}) stays constant throughout. Moving left to right, the allocation for TCP (B_{tcp}) increases as the allocation for multimedia (B_{mm}) decreases by a corresponding amount. This is a policy choice. Recall (Equation 1.3) that the sum of the allocations must always equal the capacity of the link. So, in order to decrease B_{mm} some other class must have its allocation increased. In this case we chose B_{tcp}. The load generated by each class is indicated by the black lines labeled $\text{Load}_{\text{class}}$. TCP and other are both under-provisioned across all of the allocations. (The blue line for the class is always below the black line). However in the case of multimedia, the bandwidth allocation ranges above and below the load. The dashed vertical line indicates the point at which the B_{mm} is equal to $Load_{mm}$. To the left of the dashed line multimedia is over-provisioned while to the right it is under-provisioned. In the under-provisioned case, no reprovisioning is necessary so *B¢* (indicated by a red line) is equal to *B* for all classes. However, whenever a class is over-provisioned, *B¢* diverges from *B* for those classes where *Load* is not equal to *B*. The first point to notice is that $B¢_{mm}$ is *less than* B_{mm}. The allocation for multimedia is reduced to the actual demand. The difference (shown in grey) between the original allocation, B_{mm}, and the recalculated allocation, $B¢_{mm}$, is available for the other classes to borrow. The excess for a given class is expressed as shown in Equation 1.11.

$$E_i = \min(B_i - Load_i, 0) \tag{1.11}$$

The excess across all of the classes (*E*, shown in Equation 1.12) is divided among the other under-provisioned classes according to their shares as calculated with Equation 1.9. The recalculated allocation (*B¢*) is simply the original allocation (*B*) plus the appropriate share of the over-provisioned classes' excess.

$$E = \sum_{i=0}^{N} E_i \tag{1.12}$$

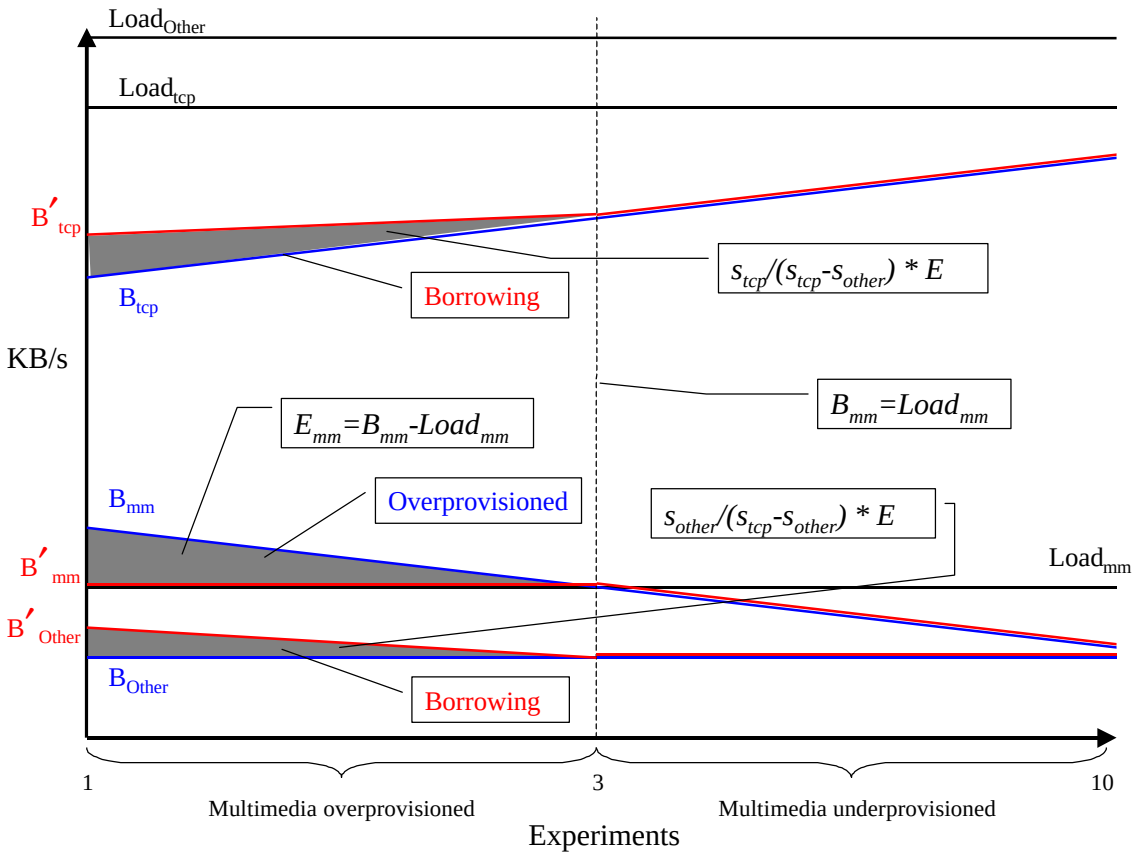

Figure 1-1 Relationship between B_i, $B\mathfrak{c}_i$, and $Load_i$

There are a number of ways to determine the value of *B-prime*. First, one must realize that the reallocation of excess capacity may, in fact be an iterative process. It is possible that when the excess from the initially over-provisioned class is divided among the other classes one of the other classes will have *B-prime* greater than the *Load*. As a result, this new excess will have to be divided among the other classes that are still under-provisioned. When the load generated by a particular traffic class, $Load_{i,}$ is less than the bandwidth. B_i, allocated to it, other classes are able to use more than the capacity originally allocated to them. The adjusted allocation for another class, *j*, is expressed as $B_j\mathfrak{c}$. To arrive at $B\mathfrak{c}_j$ we take an iterative approach. $B_{j,1}$, is defined in Equation 1.13.

$$B_{j,1} = B_j + \frac{s_j}{(1-s_i)}(B_i - Load_i) \tag{1.13}$$

When more than one class's load is less than its allocated bandwidth, all of the unused provisioned bandwidth can be expressed by the term shown in Equation 1.14. It is simply the sum of the difference between the provisioning and the actual load across all of the classes that are not fully using their allocation.

$$\sum_{\exists i, B_i > Load_i} (B_i - Load_i) \tag{1.14}$$

Each class that is exceeding its provisioned bandwidth will be able to borrow capacity from the unused provisioned bandwidth based on its share in ratio to the shares of the other classes that are exceeding their allocation. This ratio, for class *j*, is expressed by the term shown in Equation 1.15.

$$\frac{s_j}{\sum_{\exists i, B_i < Load_i} s_i} \tag{1.15}$$

(*Kevin says there are other ways to define the above equation. Particularly refers to max-min fair. What is he talking about?*) More generally, the expected bandwidth $B_{j,k}$ for a given class, including borrowing from classes that aren't using their full allocation, can be expressed as shown in Equation 1.16. $B_{j,k}$ indicates the k-th iteration of recomputing the expected bandwidth allocation.

$$B_{j,1} = B_j + \frac{s_j}{\sum_{\exists i, B_i < Load_i} s_i} \sum_{\exists i, B_i > Load_i} \left(B_i - Load_i\right) \tag{1.16}$$

While Equation 1.16 expresses the bandwidth available to each class it is possible that $B_{j,k} > Load_j$. As a result, the difference may, once again be divided among the classes where $B_{j,k} < Load_j$. So, ultimately, the approach is iterative until $B_{j,k}$ is less than or equal to $Load_j$ for all classes. Equation 1.17 illustrates the iterative solution.

$$B_{j,k} = \min\left(Load_j, B_{j,k-1} + \frac{s_j}{\sum_{\exists i, B_{i,k-1} < Load_i} s_i} \sum_{\exists i, B_{i,k-1} > Load_i} \left(B_{i,k-1} - Load_i\right) \right) \tag{1.17}$$

We define the final converged bandwidth allocation, B'_j, as the k-th iteration, $B_{j,k}$ where k is defined as shown in Equation 1.18.

$$k \middle| \forall j, B_{j,k} \le Load_j, k \le N \tag{1.18}$$

Note that this iteration is only necessary to predict and understand how the bandwidth is allocated between the classes when classes are overprovisioned. The actual queueing mechanism simply compares the average queue occupancy to the threshold for that class.

This calculation of B' points out an important aspect of CBT, *predictability*.

1.3 Equations for Predicting Latency

When one or more classes are overprovisioned the latency caused by queueing delays will decrease. That is because the average queue occupancy for those overprovisioned classes will decrease, resulting in a shorter total queue, and, as a result, less latency resulting from queueing delay. Recall that even though classes can exceed their initial bandwidth allocation by *borrowing* from.the classes that are overprovisioned they do *not* borrow queue capacity. Rather, since other classes are not maintaining a queue occupancy equal to the initial threshold settings those classes that do maintain a queue occupancy equal to their threshold settings represent a larger fraction of the total queue occupancy and, as a result, receive a larger fraction of the capacity of the outbound link.
To better understand this decrease in latency, recall that thresholds for each class place an upper limit on the queue occupancy, and as a result, the queueing delays induced by that class. This is bound on the occupancy is shown in Equation 1.19.

$$Th_i \geq q\_avg_i \tag{1.19}$$

From that statement it is straightforward to arrive at the relation shown in Equation 1.20. The sum of the thresholds is always greater than or equal to the average queue occupancy.

$$\sum_{i=0}^{N} Th_i \geq \sum_{i=0}^{N} q\_avg_i \tag{1.20}$$

The classes that are overprovisioned will not reach their threshold so their queue occupancy is strictly less than their threshold. This is expressed in Equation 1.21. Note that the load is compared to $B¢$, not B. If other classes are overprovisioned a given class may be able to exceed its original bandwidth allocation without reaching its threshold.

$$B_i^{'} > Load_i \Rightarrow Th_i > q\_avg_i \tag{1.21}$$

If we express this difference between the actual average queue occupancy and the threshold value with a delta as shown in Equation 1.22 we can express the decrease in latency as shown in Equation 1.23

$$\Delta_i = Th_i - q\_avg_i \tag{1.22}$$

$$L_\Delta = \frac{\Delta_i P_i}{C_{outbound}} \tag{1.23}$$

We can express the expected latency given initial threshold values and the actual average queue occupancy for each class as shown in Equation 1.24.

$$L^{'} = L - \frac{\sum_{i=0}^{N} \Delta_i P_i}{C_{outbound}} \tag{1.24}$$

While this is a very straight forward way to think about the average queue occupancy and its effect on latency is not a useful way to calculate the expected latency because it is not easy to acquire information about average queue occupancy for each class. However, we can extend the equations from section 1.2 and section 1.1 to derive an expression of expected latency based on expected bandwidth.
First, recall the equation to express a class's threshold in terms of the desired bandwidth and desired latency. This is shown in Equation 1.25.

$$Th_i = \frac{B_i L}{P_i} \tag{1.25}$$

We can reorganize that equation to express the desired latency in terms of the bandwidth, threshold and packet size as shown in Equation 1.26. This equation basically an expression of how long it takes a given number of bytes (Th_iP_i) to drain with a given drain-rate ($1/B_i$).

$$L = \frac{Th_i P_i}{B_i} \tag{1.26}$$

In section 1.2 we derived equations to express the revised bandwidth allocations, $B¢$, when some classes are overprovisioned. Since we know that the classes that are underprovisioned, those where $Load_i > B_i¢$, still

maintain an average queue occupancy of Th_i we can use a modified form of Equation 1.26, shown in Equation 1.27, to express the expected latency, $L¢$. For each class where the load is greater than the reallocated bandwidth we can express the expected latency as the amount of time it takes the class to drain its average queue occupancy at its expected average bandwidth reallocation. Also, the expected latency, $L¢$, is equal across all classes so once we derive it for a given class that is underprovisioned, we have derived it for all.

$$\exists i \left| Load_i \geq B_i^{'}, L_i^{'} = \frac{Th_i P_i}{B_i^{'}} \right.$$
$$L^{'} = L_i^{'}, \forall i \qquad (1.27)$$

Finally, note that if no class is underprovisioned after the bandwidth reallocations then the link is underutilized and no queue will be expected to form, resulting in an average queueing latency of approximately zero.