[SKSt01] *Jens Schmitt, Martin Karsten, Ralf Steinmetz;* **On the aggregation of guaranteed service flows;** Computer Communications, 24(1):2-18, January 2001.

On the Aggregation of Deterministic Service Flows

Jens Schmitt[1], Martin Karsten[1], and Ralf Steinmetz[1,2]
[1] Industrial Process and System Communications, Darmstadt University of Technology, Germany
[2] German National Research Center for Information Technology, GMD IPSI, Darmstadt, Germany
Email: {Jens.Schmitt,Martin.Karsten,Ralf.Steinmetz}@KOM.tu-darmstadt.de

Abstract -- **It is common belief that "flat" Quality of Service (QoS) architectures, as e.g. the IETF's Integrated Services architecture (IntServ), are not scalable to large networks as for example the global Internet. This is due to the ambitious goal of providing per-flow QoS and the resulting complexity of fine-grained traffic management. One solution to this problem is the aggregation of traffic flows in the core of the network, thus creating a hierarchical resource allocation system. While one might suspect that aggregation leads to allocating more resources for the aggregated flow than for the sum of the separated flows if flow isolation shall be guaranteed *deterministically*, we show in this article that for IntServ's Guaranteed Service flows this is not necessarily the case even if flow isolation is retained. We compare different approaches to describe the aggregated traffic and analyze their impact on bandwidth consumption and ease of flow management. Furthermore, we perform a thorough numerical evaluation of the derived results with respect to their behavior in response to changes in exogenous parameters like the traffic specifications of the flows or the configuration of the network.**
Applications of these theoretical insights and numerical evidence could be to use the derived formulas for resource allocation in either a hierarchical IntServ, IntServ over DiffServ (Differentiated Services), or IntServ over ATM (Asynchronous Transfer Mode) network.

Keywords: Network QoS, Scalability, Aggregation, Deterministic Multiplexing, Network Calculus.

I. Introduction

A. Motivation

The provision of integrated services over a shared infrastructure is often seen as the "holy grail" of networking. It would allow to save resources on a large scale and be more flexible when the total traffic distribution varies as it, e.g., seems to do right now. The IETF therefore developed the so-called Internet Integrated Services architecture which proposes a set of service classes (defined by the IntServ working group) and a resource reservation protocol (RSVP) to "signal" users' requirements with respect to service classes and their parameters (see [24] for an overview). This architecture is designed very general (though sometimes also considered complex), so that all sorts of applications shall be able to benefit from the QoS offered by the network. However, due to the provision of QoS on the level of application flows it is considered not to be scalable to large networks like the Internet. The scalability problem is mainly due to the potentially large number of flows in the core of the network and the corresponding complexity of classifying and scheduling these flows at interior nodes. An obvious approach to this problem is the aggregation of application flows in the core of the network, so that interior routers only need to exert their traffic management on aggregated flows. This approach has a dynamic and a static aspect. The dynamic aspect is how the routers can coordinate themselves to allow for the aggregation and segregation of flows. Here an extension of RSVP or an alternative protocol is necessary (as e.g. described in [9], [3], [23], or [1]). The static aspect refers on the one hand to the necessary resource allocations for an aggregated flow and on the other hand to the question of which flows should be grouped together.

In this article, we focus on the static aspect of aggregation for the case of regulated traffic requiring deterministic service guarantees, mainly drawing upon the specific example of IntServ's Guaranteed Service flows. We regard the Guaranteed Service class as particularly interesting due to its comparably strong, deterministic guarantees on rate, delay and loss. Especially for future hard, and possibly even critical real-time applications, as e.g. in the field of telemedicine, such services will play a crucial role as they are able to isolate the operation of such applications from other less critical applications despite running on the same shared network infrastructure. Furthermore, due to its mathematical description it allows for an exact analysis with regard to the problem of resource allocation for aggregated flows.

B. Assumptions

The part of the network that only handles/"sees" aggregated flows will further on be called *aggregation region*. We take a topological approach towards the aggregation of flows, which means that flows that shall be aggregated must share the same path over the aggregation region. An alternative would be an approach based on traffic classes, where flows would be assigned to traffic classes and all packets belonging to the same class would receive the same treatment within the aggregation region. This approach would result in a constant amount of state within the aggregation region corresponding to the number of traffic classes supplied. On the other hand, topological aggregation requires state within the aggregation region of $O(n^2)$, where n is the number of peering edge devices located around the aggregation region, which perform the aggregation of flows before these enter the aggregation region. While this is a clear advantage of class-based aggregation over topological aggregation, it has been shown in [12], [5], and [22] that for deterministic service guarantees the resource utilization within the aggregation region would be extremely low. As it is primarily deterministic services we are interested in, we are following the topological approach towards aggregation, also arguing that by a sensible network design of the aggregation region, which would introduce enough redundancy into the network to avoid bottlenecks under "normal" traffic patterns, the worst case state complexity will be a very loose bound on the actual state complexity experienced.

Due to selecting the topological aggregation approach, we constrain our investigations on unicast flows, since multicast flows are unlikely to share the same partial multicast tree over the aggregation region. However, if they did, e.g. because the partial multicast tree is the same tandem of nodes through the aggregation region, the results derived below would still apply. Note that anyway unicast flows can be considered more "evil" with respect to scalability since they are expected to be much more numerous than multicast flows.

An important distinction for the line of argument of our article is how we use the terms *aggregation* and *grouping* of flows. By aggregation we mean the general problem of merging different flows over an aggregation region inside the network. By grouping of flows we refer to the restricted problem of the whole network being the aggregation region, i.e. flows are aggregated end-to-end. So, in our terminology grouping is a special case of aggregation. In Figure 1 these different concepts are schematically depicted.

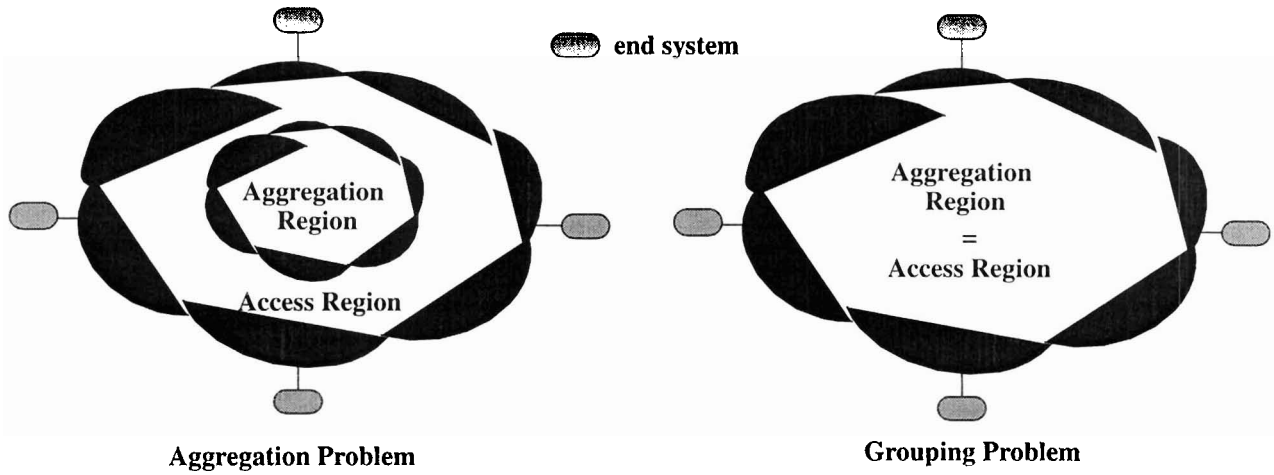

Figure 1: Aggregation vs. Grouping.

C. Outline

In the next section we give a brief review of the semantics and basic mathematical background of the IETF's Guaranteed Service class, as a special instance of a deterministic service. Then we derive some fundamental formulas for the problem of grouping flows as defined above. Here we first quantify the effect of grouping flows onto resource allocation. Next we suggest a way to characterize the grouped flow which allows for more efficient resource utilization, followed by some numerical examples to illustrate these results. The results for flow grouping are then applied to the more general problem of aggregating flows. To do so we introduce a conceptual model of the aggregation problem and show what has to be done to make it conform to the prerequisites of flow grouping. After giving again some simple numerical examples on the trade-offs for the resource allocation inside and outside of the aggregation region, we perform a more in-depth numerical evaluation of the proposed mechanisms in the form of a "sensitivity analysis" of

the derived formulas on exogenous parameters like traffic specifications and network configuration parameters. This analysis is based upon experiments we performed using a simulator we developed for that purpose. Before concluding the article and having an outlook to future work, we also give an overview of related work with respect to both the static and the dynamic aspect. In particular for the dynamic aspect we briefly discuss the issue of how to deal with excess traffic when applying the results on existing candidates for the aggregation region, like an IntServ, DiffServ, or ATM cloud.

II. The IETF Guaranteed Service Class

Guaranteed Service (GS) as specified in [20] provides an assured level of bandwidth, a firm end-to-end delay bound and no queuing loss for data flows that conform to a given traffic specification (TSpec). The TSpec, which is essentially a double token bucket, i.e. two token buckets in series, is characterized by the following parameters:

- the token bucket rate r (in bytes/s),
- the token bucket depth b (in bytes),
- the peak rate p (in bytes/s),
- the maximum packet size M (in bytes), and
- the minimum policed unit m (in bytes).[1]

The mathematics of GS are originally based on the work of Cruz [6] (refined by others, see e.g. [4]) on arrival and service curves. While arrival curves describe the worst-case behavior of a source within given time intervals, service curves specify the minimal service that is provided by a queue server. By combining these two concepts it is possible to derive deterministic guarantees on loss and delay under the worst-case scenario of a greedy source and a fully loaded server.

In case of the IntServ specifications the arrival curve corresponding to the *TSpec(r,b,p,M)* is

$$a(t) = \begin{cases} M + pt & t < \dfrac{b-M}{p-r} \\ b + rt & t \geq \dfrac{b-M}{p-r} \end{cases}, \tag{1}$$

whereas the service curve for GS is

$$c(t) = R(t-V)^{+}\ ,^{2} \tag{2}$$

$$\text{where } V = \frac{C}{R} + D \text{ and } R \text{ is the service rate,} \tag{3}$$

assuming that the stability condition $R \geq r$ holds. Here, the C and D terms represent the rate-dependent respectively rate-independent deviations of a packet-based scheduler from the perfect fluid model as introduced by ([16], [17]). These *error terms* are summed up along the data transmission path for each server/router during the advertisement phase [25].

While the TSpec is a double token bucket it is sometimes more intuitive to regard the mathematical derivations for a simple token bucket *tb=(r,b)* (which is equivalent to assuming an infinite peak rate). In this simplified case we obtain for the end-to-end delay bound

$$d_{max} = \frac{b}{R} + \frac{C}{R} + D\ . \tag{4}$$

While for the more complex TSpec as arrival curve it applies that

$$\begin{aligned} p \geq R \geq r \qquad & d_{max} = \frac{(b-M)(p-R)}{R(p-r)} + \frac{M+C}{R} + D \\ R \geq p \geq r \qquad & d_{max} = \frac{M+C}{R} + D \end{aligned} \tag{5}$$

From the perspective of the receiver desiring a maximum queuing delay d_{max}, the service rate R (in bytes/s) that has

1. For our discussions we can omit this parameter of the TSpec further on.
2. $(x)^{+}$ is defined as 0 if $x < 0$ and x otherwise.

to be reserved at the routers on the path from the sender follows directly from (4) and (5):

For the simple token bucket *tb(r,b)*

$$R = \frac{b + C}{d_{max} - D}, \tag{6}$$

for the complete *TSpec(r,b,p,M)*

$$R = \begin{cases} \dfrac{p\dfrac{b-M}{p-r} + M + C}{d_{max} + \dfrac{b-M}{p-r} - D} & p \geq R \geq r \\ \dfrac{M + C}{d_{max} - D} & R \geq p \geq r \end{cases} \tag{7}$$

While the buffer B to guarantee a lossless service for the single token bucket simply equals b, the buffer formula for the TSpec's double token bucket is more complicated:

$$\begin{cases} M + \dfrac{(p-R)(b-M)}{p-r} + C + RD & p \geq R \geq r, \dfrac{C}{R} + D \leq \dfrac{b-M}{p-r} \\ b + r\left(\dfrac{C}{R} + D\right) & \dfrac{C}{R} + D > \dfrac{b-M}{p-r} \\ M + p\left(\dfrac{C}{R} + D\right) & R \geq p \geq r \end{cases} \tag{8}$$

To illustrate the meaning of the C and D terms we refer to their values in case of a PGPS (Packetised General Processor Sharing) scheduler [16], because they also apply to many other packet scheduling algorithms [26]

$$C = M \text{ and } D = \frac{M'}{c}, \tag{9}$$

where M is the maximum packet size of the flow, M' is the MTU (Maximum Transfer Unit) and c is the speed of the link. In real routers, there are potentially many other contributions to these error terms as, e.g., link layer overhead for segmentation and reassembly in the case of ATM or token rotation times for FDDI or token ring.

There are two related problems with GS:

1. It may not be scalable enough to be used in the backbone of the Internet since no aggregation mechanisms were provided (due to the stipulation of per-flow QoS and flow isolation). Thus, the number of queues is proportional to the number of flows.
2. It wastes a lot of resources, especially for "low bandwidth, short delay"-type of flows. As an example consider a data flow with *TSpec=(1000, 2000, 2000, 1500)*, let us assume 5 hops (all with *MTU=9188 bytes* and link speed *c=155 Mb/s*) all doing PGPS. Then we have *C=7500 bytes*, *D=2.371 ms*. Let us further assume the receiver desires a maximum queueing delay of d_{max}*=50 ms*. Then we obtain from the formulas given above that *R=191489 bytes/s* ≈ *95p* and *B=1578 bytes*.

By aggregating/grouping GS flows we address both problems, because less state has to be managed by routers and the resulting aggregated flows are of higher bandwidth.

III. The Mathematics of Flow Grouping

In this section we derive a set of fundamental formulas about flow grouping. We show how grouping of flows can save resources when compared to isolated flows. Recall that by grouping we refer to the restricted problem where the aggregation takes place end-to-end between sender and receiver.

A. *Grouping Gains from Sharing Error Terms*

For the grouping of flows we need a concept of how to characterize the traffic of the grouped flow. In RFC 2212, the sum over n TSpecs is defined as

$$\sum_{i=1}^{n} TSpec(r_i, b_i, p_i, M_i) = TSpec\left(\sum_{i=1}^{n} r_i, \sum_{i=1}^{n} b_i, \sum_{i=1}^{n} p_i, max(M_i)\right) \quad (10)$$

In RFC 2216 [21], which gives the general requirements for specifying service classes, the summation of TSpecs as defined in (10) is motivated as follows:

> *This function computes an invocation request which represents the sum of N input invocation requests. Typically this function is used to compute the size of a service request adequate for a shared reservation for N different flows. It is desirable but not required that this function compute the "least possible sum".*

So, as a starting point we use the "summed TSpec" as arrival curve for the grouped flow. We want to compare the rates for grouped flows with the sum of the rates of the isolated flows.

Let us start by looking at the simplified model of using single token buckets for the characterization of the isolated flows:

Let S be a set of n receivers with $tb_i=(r_i,b_i)$ and $d_{max,i}$, then the rate for the isolated system of these n flows is

$$R^I(S) = \sum_{i=1}^{n} \frac{b_i + C}{d_{max,i} - D} \quad (11)$$

while for the grouped system of these n flows, with the sum of single token buckets defined analog to (10), it is

$$R^G(S) = \frac{\sum_{i=1}^{n} b_i + C}{min(d_{max,i}) - D} \quad (12)$$

Now let us define the difference between the isolated and the grouped system with respect to the allocated accumulated service rate over flows 1 to n as "Grouping Efficiency" (GE), i.e.:

$$GE(S) = R^I(S) - R^G(S) \quad (13)$$

Thus, we can state the problem of which flows to group together as:

For a set of n reservations ($tb_i=(r_i,b_i)$ or $TSpec(r_i,b_i,p_i,M_i)$ and $d_{max,i}$), find a partition $P=\{P_1,...,P_k\}$

such that $\sum_{l=1} GE(P_l)$ is maximized and k is minimized.

It can be easily seen from (12) that it is advantageous if those flows to be grouped together have equal or at least similar delay requirements. Thus, we can order the flows by their delay requirements and restrict the search to the space of ordered partitions for the optimal flow to group assignment since it can be proven that the optimum must be an ordered partition:

Theorem 1: Let $S=\{1,...,n\}$ be an ordered set of reservations ($tb_i=(r_i,b_i)$ *and* $d_{max,i}$), $i=1,...,n$. The ordering criterion is $d_{max,i}$. Then the rate-optimal partition is ordered after $d_{max,i}$. Here, the rate of a partition $P=\{P_1,...,P_k\}$ is defined as

$$R(P) = \sum_{i=1} R(P_i)$$

Proof: Assume $P=\{P_1,...,P_k\}$ is rate-optimal, but unordered, i.e. we have at least two reservations $h, l \in \{1,...,n\}$ with $h \geq l$ and $h \in P_u$, $l \in P_v$ where $u<v$ (we assume the P_i to be ordered ascendingly in $d_{max,i}$).[3]

Then for $Q=P\setminus(P_u \cup P_v) \cup (P_u \setminus \{h\}) \cup (P_v \cup \{h\})$ we obtain

$$R(Q) = R(P) - \frac{b_h + C}{min(d_{max,i}, i \in P_u) - D} + \frac{b_h + C}{min(d_{max,i}, i \in P_v) - D}$$
$$< R(P)$$

where the inequality holds due to the proposition that $u<v$. This however is a contradiction to the assumption that P is rate-optimal and thus the theorem holds. □

---

[3] Note that if $k = 1$ then the statement of the theorem trivially follows, as there is only one group of flows which can thus not be unordered with respect to other groups.

From now on let us suppose that there are enough flows to assume that those flows grouped together have *equal* delay. For n such delay-homogeneous flows we obtain the following for the simplified model:

$$GE(S) = \sum_{i=1}^{n} \frac{b_i + C}{d_{max} - D} - \frac{\sum_{i=1}^{n} b_i + C}{d_{max} - D} = \frac{(n-1)C}{d_{max} - D} > 0 \text{ where } d_{max,i} = d_{max} \forall i . \quad (14)$$

That means we obtain gains independent of the reserved rate for delay-homogeneous flows, i.e. these gains are relatively highest if the separate flows have low bandwidth requirements. It can also be seen that GE increases with n, C and D and decreases with d_{max}. To illustrate how large the grouping gains can be, let us look at an example:

We assume again 5 hops in the aggregation region, all using PGPS as a service discipline, with an *MTU=9188 bytes* and *c=155 Mb/s*. We have 10 flows with *M=500 bytes*, and d_{max}*=50 ms* for all of them. Then we obtain: *GE(S)≈3.7 Mb/s*, irrespective of the actual token buckets of the flows.

This effect of saving resources due to grouping of flows is a result of "sharing the error terms" for the group of flows, while for the isolated flows these error terms must be accounted for separately. Therefore we call this concept "Pay scheduling errors only once" in analogy to the "Pay bursts only once" principle.

For the actual IntServ model with double token bucket TSpecs we obtain a more complex formula for the grouping efficiency of n arbitrary flows (arbitrary with respect to delay requirements and TSpec parameters), where we use the summed TSpec as arrival curve for the grouped flow:

$$GE(S) = \sum_i \frac{p_i \frac{b_i - M_i}{p_i - r_i} + M_i + C}{d_{max,i} + \frac{b_i - M_i}{p_i - r_i} - D} - \frac{\sum_i p_i \frac{\sum_i b_i - max(M_i)}{\sum_i p_i - r_i} + max(M_i) + C}{min(d_{max,i}) + \frac{\sum_i b_i - max(M_i)}{\sum_i p_i - r_i} - D} \quad (15)$$

The first term represents $R^I(S)$ and the second $R^G(S)$, both for the "usual" case that the reserved rate R is smaller than the peak rate of the corresponding flow. While it is still true that equal delay requirements of the grouped flows are favorable for gaining resources by grouping, they are no longer a sufficient condition to actually achieve a gain. However, for delay-homogeneous flows with the same TSpec (TSpec-homogeneous flows) it can be shown that always *GE>0* under weak conditions:

Theorem 2: For a set S of $n>1$ delay- and TSpec-homogeneous flows *GE>0* if $C > M\frac{r}{p-r}$. [This is a very weak condition taking into account that on the one hand for many schedulers M is the rate-dependent error term and on the other hand that r will often be much smaller than p so that $\frac{r}{p-r}$ is (much) smaller than 1. Furthermore there may be other rate-dependent deviations besides M.]

Proof:

We have to distinguish two cases for isolated flows: $R \geq p$ (1) or $R < p$ (2).

Analogously, there are two cases for the grouped flow: $R \geq np$ (3) and $R < np$ (4).

The only possible combinations are (1)+(3), (1)+(4) and (2)+(4).

(2)+(3) is impossible as can be verified easily.

"(1)+(3)":

$$GE(S) = R^I(S) - R^G(S) = n\frac{M+C}{d_{max} - D} - \frac{M+C}{d_{max} - D} = (n-1)\frac{M+C}{d_{max} - D} > 0 \text{, for } n>1 \text{ (as assumed).}$$

"(1)+(4)":

$GE(s) = R^I(S) - R^G(S) \geq np - R^G(S) > 0$, simply as a result of conditions (1) and (4).

"(2)+(4)":

$$GE(S) = R^I(S) - R^G(S) = n\frac{p\frac{b-M}{p-r} + M + C}{d - D + \frac{b-M}{p-r}} - \frac{np\frac{nb-M}{np-nr} + M + C}{d - D + \frac{nb-M}{np-nr}}$$

$$= \frac{np\frac{b-M}{p-r} + nM + nC}{d - D + \frac{b-M}{p-r}} - \frac{p\frac{nb-M}{p-r} + M + C}{d - D + \frac{nb-M}{np-nr}}$$

$$> \frac{np\frac{b-M}{p-r} + nM + nC - \left(p\frac{nb-M}{p-r} + M + C\right)}{d - D + \frac{b-M}{p-r}}$$

$$= \frac{n-1}{d - D + \frac{b-M}{p-r}}\left(C - M\frac{r}{p-r}\right)$$

which implies that $GE(S) > 0 \Leftrightarrow C > M\frac{r}{p-r} \wedge n > 1$. □

For TSpec-heterogeneous flows the summed TSpec may incur a higher rate because it overestimates the arrival curve for the group of flows. How to circumvent this effect will be discussed in the next section.

Anyway, *GE* can be used as a hint towards the decision whether a set of flows should be grouped together respectively whether a new flow should be added to an existing group of flows, simply by the fact whether *GE>0* or *<0*.

B. Tight Arrival Curves for Grouped GS Flows

We have shown in the previous section how grouping of flows can reduce resource requirements. However, the flows had to be homogeneous with respect to their TSpec and their delay requirements to achieve a guaranteed reduction. Taking into account that additionally the flows have to share the same path through the aggregation region, these can be very restricting prerequisites to the grouping of flows. Therefore, we now try to relax the first prerequisite of TSpec-homogeneity by using a tighter arrival curve than the summed TSpec for the characterization of the grouped flow. Instead of the summed TSpec we use a series of token buckets which can be shown to be an arrival curve for the grouped flow. This allows a lower resource reservation for the grouped flow when compared to the summed TSpec as arrival curve. We call this arrival curve "cascaded TSpec".

This discussion is illustrated by the simple example in Figure 2.

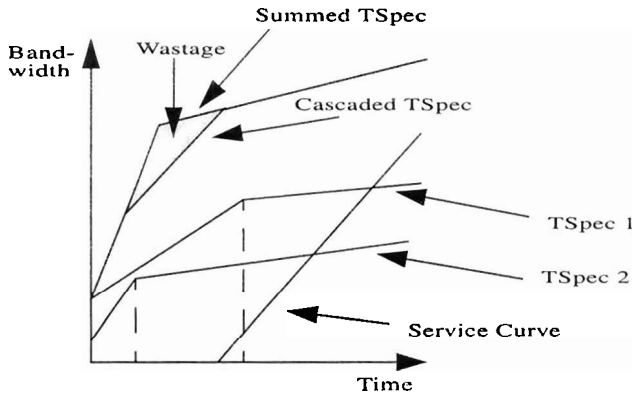

Figure 2: Summed vs. Cascaded TSpecs.

Here we have two flows with differing TSpecs. It can be seen that by using the summed TSpec we may give away some bandwidth we "know" of that it will never be used. Therefore, we would like to use the exact sum of the arrival curves, the cascaded TSpec.

Let us now take a more formal look at the problem. In general the tight arrival curve *tac(t)* for *n* TSpecs has the following form

$$tac(t) = \sum_{j=1}^{n} a_j(t) = \begin{cases} M + \sum_{j=1}^{n} p_j t & t \le x_1 \\ b_1 - M_1 + M + \sum_{j=2}^{n} p_j t + r_1 t & x_1 < t \le x_2 \\ \cdots & \\ \sum_{l=1}^{k-1} (b_l - M_l) + M + \sum_{j=k}^{n} p_j t + \sum_{l=1}^{k-1} r_l t & x_{k-1} < t \le x_k \\ \cdots & \\ \sum_{l=1}^{n} (b_l - M_l) + M + \sum_{l=1}^{n} r_l t & t > x_n \end{cases} \quad (16)$$

where x_j, the burst duration for flow j, is defined as: $x_j = \frac{b_j - M_j}{p_j - r_j}$ and $M=max(M_1,...,M_n)$.

Here we have assumed without loss of generality that $x_1 \le \ldots \le x_n$

This tight arrival curve for the grouping of n GS flows is equivalent to the concatenation of $(n+1)$ token buckets (the cascaded TSpec), i.e. (with $\otimes$ as concatenation operator for token buckets)

$$tac(t) = tb\left(M, \sum_{j=1}^{n} p_j\right) \otimes tb\left(b_1 - M_1 + M, \sum_{j=2}^{n} p_j + r_1\right) \otimes \ldots \otimes tb\left(\sum_{l=1}^{k-1} (b_l - M_l) + M, \sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right) \otimes \ldots \otimes tb\left(\sum_{l=1}^{n} (b_l - M_l) + M, \sum_{l=1}^{n} r_l\right)$$

If we apply the known results from network calculus [4] on this tight arrival curve, assuming the GS service curve, we obtain the delay bound

$$d_{tac} \le h(tac, c) = sup_{s \ge 0}(inf\{T : T \ge 0 \wedge tac(s) \le c(s+T)\})$$

$$= \frac{\sum_{l=1}^{k-1} (b_l - M_l) + M + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right) x_k}{R} - x_k + \frac{C}{R} + D \quad (17)$$

$$= \frac{(p_k - r_k) \sum_{l=1}^{k-1} (b_l - M_l) + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l - R\right)(b_k - M_k)}{R(p_k - r_k)} + \frac{M + C}{R} + D$$

where $k \in \{1,...,n\}$ is such that: $\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l > R \ge \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l$ (18)

If $R > \sum_{j=1}^{n} p_j$ (i.e. there is no such k), then $d \le \frac{M + C}{R} + D$. (19)

In contrast, the delay bound for the summed TSpec of n flows is:

$$d_{sum} \leq \begin{cases} \dfrac{\left(\sum_{j=1}^{n} b_j - M\right)\left(\sum_{j=1}^{n} p_j - R\right)}{\left(R\sum_{j=1}^{n}(p_j - r_j)\right)} + \dfrac{M+C}{R} + D & \sum_{j=1}^{n} p_j > R \geq \sum_{j=1}^{n} r_j \\ \dfrac{M+C}{R} + D & R \geq \sum_{j=1}^{n} p_j \end{cases} \tag{20}$$

It can be easily shown that, for a given rate R, d_{sum} is always greater than or equal to d_{tac}, since the summed TSpec is an envelope for the cascaded TSpec. We do so by presenting a more general result:

Theorem 3: Let a_1, a_2 be arrival curves with $a_1 \geq a_2$ and let c be a monotonically increasing service curve. Then it applies for the delay bounds d_1, d_2 corresponding to the arrival curves that $d_1 \geq d_2$.

Proof:

$$\begin{aligned} d_1 &= h(a_1, c) = sup_{s \geq 0}(inf\{T : T \geq 0 \wedge a_1(s) \leq c(s+T))\}) \\ &\geq sup_{s \geq 0}(inf\{T : T \geq 0 \wedge a_2(s) \leq c(s+T))\}) \\ &= h(a_2, c) = d_2 \end{aligned} \tag{21}$$

The inequality holds due to the prerequisite of $a1 \geq a2$ and the monotonically increasing service curve c. ❐

Let us now look at the formulas for the service rate when given a certain delay. For the summed TSpec we obtain: (where $M = max(M_1,...,M_n)$ again)

$$R = \begin{cases} \dfrac{\sum_{j=1}^{n} p_j \dfrac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j} + M + C}{d_{max} + \dfrac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j} - D} & \sum_{j=1}^{n} p_j > R \geq \sum_{j=1}^{n} r_j \\ \dfrac{M+C}{d_{max} - D} & R \geq \sum_{j=1}^{n} p_j \end{cases}, \tag{22}$$

whereas for the cascaded TSpec we obtain for some $k \in \{1,...,n\}$:

case 1: $\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l > R \geq \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l$

$$R = \frac{\sum_{l=1}^{k-1}(b_l - M_l) + M + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right)\left(\frac{b_k - M_k}{p_k - r_k}\right) + C}{d_{max} + \frac{b_k - M_k}{p_k - r_k} - D} \tag{23}$$

case 2: $R \geq \sum_{j=1}^{n} p_j$

$$R = \frac{M+C}{d_{max} - D}.$$

For the sake of completeness, we also give the buffer requirements for both arrival curves in Appendix A.

With these formulas it is now possible to compare the different resource allocation schemes for the isolated flows and for the group of flows characterized by either the summed or cascaded TSpec. Since the formulas are not very intuitive, we illustrate the effects of flow grouping on delay, rate and buffer requirements by presenting a numerical example.

C. A Simple Numerical Example

We want to contrast the different resource allocations with regard to rate and buffer for the isolated flows (R_{ISO}, B_{ISO}) against the grouped flow with either summed TSpec (R_{SUM}, B_{SUM}) or cascaded TSpec (R_{CAS}, B_{CAS}). We assume an aggregation region of 5 hops with *MTU=9188 bytes*, and *c=155 Mb/s* ("ATM hops"). Furthermore, it is assumed that 10 flows are to be grouped together, with all of them having a delay bound d_{max}=*50 ms*. The TSpecs of the flows were chosen arbitrarily, besides the fact that rather "narrow" flows were selected. They are as given in Table 1.

TSpec#	r	b	p	M
1	10000	15000	20000	500
2	20000	40000	130000	500
3	10000	10000	40000	500
4	20000	20000	125000	500
5	40000	30000	60000	500
6	8000	8000	100000	500
7	15000	50000	33000	500
8	20000	12000	40000	500
9	30000	30000	45000	500
10	10000	15000	220000	500

TABLE 1 TSpecs of the sample flows.

Let us first assume that we want to group 10 identical flows with TSpec# 1. The accumulated rate allocations inside the aggregation region for the different schemes are shown in Table 2.

x	R_x	B_x
ISO	629868	13410
SUM	195769	9788
CAS	195769	9788

TABLE 2 Accumulated rate allocations for homogeneous TSpecs.

So we can see that the gains from sharing the error terms can be substantial. Since we have a case of delay- and TSpec-homogeneous flows, the summed and the cascaded TSpec achieve the same values because for that case they are actually the same arrival curves. Now we relax the assumption of TSpec-homogeneous flows and group all the different flows from the table above. The results are shown in Table 3.

x	R_x	B_x
ISO	615311	60209
SUM	642307	64230
CAS	419884	41988

TABLE 3 Accumulated rate allocations for heterogeneous TSpecs.

In conclusion, what we gain from grouping flows is the sharing of error terms, so we know that for delay- and

TSpec-homogeneous flows grouping almost always leads to a gain. For TSpec-heterogeneous flows however there is also a negative contribution of grouping due to overestimating the arrival curve when adhering to the summed TSpec characterization for the grouped flow, an effect that depends upon how heterogeneous the isolated flows really are (heterogeneity here is mainly captured by two characteristics of bursts, size b and intensity p/r). This effect can "mask" the positive effect of sharing the error terms as shown in the last example. To avoid this negative effect, the exact arrival curve of the grouped flows, the cascaded TSpec, can be used for the calculations of rate and buffer and thus we have again only the positive effect. The downside of this is that the traffic specification is often used for purposes like reshaping or policing, and with many heterogeneous flows being grouped together this can lead to a very complicated arrival curve which, while it does not violate the worst-case delay bound, is complicated to handle and increases the average delay. So, we address this issue in the next section.

D. Policing/Shaping the Grouped Flow

Once the service rate is calculated from the formulas above, it is possible to achieve the desired delay bound with a much simpler arrival curve. It can be shown (see Theorem 4 below) that the following arrival curve is sufficient for achieving the same delay bound for a given R as the tight arrival curve:

$$a(t) = \begin{cases} \sum_{l=1}^{k-1}(b_l - M_l) + M + \sum_{j=k}^{n} p_j t + \sum_{l=1}^{k-1} r_l t & t \le x_k \\ \sum_{l=1}^{k}(b_l - M_l) + M + \sum_{j=k+1}^{n} p_j t + \sum_{l=1}^{k} r_l t & t > x_k \end{cases} \tag{24}$$

or, as token bucket concatenation:

$$a(t) = tb\left(\sum_{l=1}^{k-1}(b_l - M_l) + M, \sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right) \otimes tb\left(\sum_{l=1}^{k}(b_l - M_l) + M, \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l\right).$$

That means $a(t)$ can also be described as $TSpec\left(\sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l, \sum_{l=1}^{k}(b_l - M_l) + M, \sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l, \sum_{l=1}^{k-1}(b_l - M_l) + M\right)$

Theorem 4: The above arrival curve a has the same delay bound d_{max} as the tight arrival curve tac for the given R as calculated from the formulas in (17)-(19).

Proof: We know from (17)-(19) that if a delay bound d_{max} is desired then it applies that for some fixed $k \in \{1,...,n\}$:

$\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l > R \ge \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l$, therefore we obtain

$$\begin{aligned} d_{max,a} &= h(a,c) = \frac{a(x_k)}{R} - x_k + \frac{C}{R} + D \\ &= \frac{\sum_{l=1}^{k-1}(b_l - M_l) + M + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right)x_k}{R} - x_k + \frac{C}{R} + D \\ &= \frac{(p_k - r_k)\sum_{l=1}^{k-1}(b_l - M_l) + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l - R\right)(b_k - M_k)}{R(p_k - r_k)} + \frac{M + C}{R} + D \\ &= h(tac,c) = d_{max,tac} \end{aligned} \tag{25}$$

□

Hence, we can reduce policing/shaping complexity dramatically without compromising resource allocation efficiency. The idea is, not to take the complete piecewise linear arrival curve of the cascaded TSpec, but only those two adjacent segments at which angular point (x_k) the delay bound, i.e. the supremum of the horizontal deviation between

arrival and service curve, is actually taken on. This can be done after the service rate is calculated from the cascaded TSpec and it is thus known that those two segments are "responsible" for the delay bound. An actual algorithm to determine k would have to sort the linear segments by their slopes and find those two adjacent segments for which one is smaller than R and the other one is larger than R (neglecting the case that R is equal to one of the slopes).

While the delay bound remains the same as for the cascaded TSpec, the buffer requirements depend on whether $V \leq x_{k+1}$ or $V > x_{k+1}$ (where $V = C/R + D$, see (3)). For the first case they are the same, while in the second case the buffer requirements of $a(t)$ are higher. If the buffer requirements shall also be kept equal for the latter case this "costs" another token bucket for the linear segment of the cascaded TSpec for which applies that $x_{k+h} < V < x_{k+h+1}$, where $h \in \{1,...,n-k\}$. More formally:

$$a(t) = \begin{cases} \sum_{l=1}^{k-1}(b_l - M_l) + M + \sum_{j=k}^{n} p_j t + \sum_{l=1}^{k-1} r_l t & t \leq x_k \\ \sum_{l=1}^{k}(b_l - M_l) + M + \sum_{j=k+1}^{n} p_j t + \sum_{l=1}^{k} r_l t & x_k < t \leq \frac{\sum_{l=k+1}^{k+h}(b_l - M_l)}{\sum_{l=k+1}^{k+h}(p_l - r_l)} \\ \sum_{l=1}^{k+h}(b_l - M_l) + M + \sum_{j=k+h+1}^{n} p_j t + \sum_{l=1}^{k+h} r_l t & t > \frac{\sum_{l=k+1}^{k+h}(b_l - M_l)}{\sum_{l=k+1}^{k+h}(p_l - r_l)} \end{cases}$$

or, as token bucket concatenation:

$$a(t) = tb\left(\sum_{l=1}^{k-1}(b_l - M_l) + M, \sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right) \otimes tb\left(\sum_{l=1}^{k}(b_l - M_l) + M, \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l\right) \otimes tb\left(\sum_{l=1}^{k+h}(b_l - M_l) + M, \sum_{j=k+h+1}^{n} p_j + \sum_{l=1}^{k+h} r_l\right).$$

While requiring some more work on policing/shaping, this triple token bucket offers the same delay bound and buffer requirements at a given service rate as the exact arrival curve, the cascaded TSpec, which is composed of $n+1$ token buckets.

IV. Application of Grouping to Aggregation

After having established some results on the problem of grouping flows, we now apply these results to the more general problem of aggregating flows. We first present a conceptual model of how aggregation could be achieved and then give a simple numerical examples on how such a scheme would perform. Afterwards we take a detailed look at the aggregated system and compare it to the segregated system. We use numerical simulations to investigate the effect of various parameters like flow specifications and network configurations on this comparison.

A. Conceptual Model

We consider the conceptual model for aggregation as a two-level resource allocation system, corresponding to inside and outside the aggregation region (AR). Outside the AR resource allocations are done for individual flows, while inside the AR it is done for aggregated flows. Flows that shall be aggregated must share the same path over the AR, but can follow different routes outside the AR.

When we want to apply the results for grouping to that general model of aggregation we face three problems:

1. A fixed delay over the AR is required, i.e. a portion of the end-to-end queuing delay bound of each flow must be devoted to the AR.

2. There are possibly distorted (with respect to their TSpec), i.e. non-conforming, incoming flows at the ingress to the AR. These could occupy the shared buffer of their group and destroy the guarantees on rate, delay and lossless service for other flows of that group.
3. A possible distortion of the grouped flow might lead to overflows in the routers behind the egress of the AR.

1) Delay Partition

Our approach to the first problem is the partitioning of the delay into two parts, delay inside and outside the AR. The question however is how to assign these two parts of the overall delay. While it is not possible to determine exactly the partial delay d_p of a flow which is available for the subpath over the AR, we have the following relationship:

$$\frac{M + C_{AR}}{R} + D_{AR} \le d_p \le \frac{(b - M)(p - R)}{R(p - r)} + \frac{M + C_{AR}}{R} + D_{AR} \quad , \tag{26}$$

where C_{AR} and D_{AR} are the accumulated error terms of the subpath over the AR. The lower bound corresponds to the pessimistic assumption that packets "pay their burst" outside the AR, while the upper bound represents the case where a burst is paid inside the AR. Due to the worst-case nature of the guarantees given by GS we must however assume the lower bound as the available partial delay. The partial delay may thus become very small if the error terms are comparably small to the first term ("the burst term") of the upper bound. This would lead to a relatively high allocation of resources in the AR. A protocol mechanism to circumvent this is to advertise a high D_{AR} error term for the AR. From the perspective outside the AR, the AR could thus be regarded as a fixed delay element on the path from the sender to the receiver. The drawback of this approach is that the routers outside the AR would need to reserve more resources than in the case of non-aggregated flows. There is obviously a trade-off between saving resources inside the AR by advertising a higher D_{AR} and allocating more resources outside the AR. This trade-off should be weighted by how scarce the resources inside and outside the AR really are (more on this in the Section IV.C).

Alternatively to increasing D_{AR}, the slack term could be used by the AR to increase its "delay budget". This would however require the receiver to be aware of its resource requests being possibly aggregated.

2) Ingress Reshaping

The solution to the second problem is to reshape the individual flows to their original TSpec at the ingress to the AR. While this may increase the average delay of the packets of a GS flow, it has been shown that the delay bound is not violated by reshaping (see e.g. [Bou98]).

3) Egress Reshaping

The third problem can be solved by reshaping the aggregate against the cascaded TSpec of the grouped flows. Alternatively, the reshaping at the egress could be executed on the individual flows. This would however be more costly since for a group of n flows $2 \times n$ token buckets have to be passed, whereas for the first alternative it is only $n+1$ token buckets. Note that the reshaping cannot be done using the simplified arrival curves introduced in Section III.D. These are only for use inside the AR.

Under these prerequisites it is now possible to utilize the formulas derived for the grouping of flows for resource allocation inside the AR. To illustrate how the aggregation model compares to the model of resource allocation for individual flows we give a numerical example in the next section.

B. A Simple Numerical Example

For the AR let us assume the same setting as in Section III.C, i.e. we use the same 10 flows as specified in Table 1 and 5 "ATM hops" inside the AR. For outside the AR we assume 2 hops in front and 2 hops behind the AR, all of them with *MTU=1500 bytes* and *c=100 Mb/s* ("Fast Ethernet hops"). Furthermore, we assume that all flows have the

same requirements for the end-to-end delay bound d_{max}=*100 ms*. This scenario is depicted in Figure 3

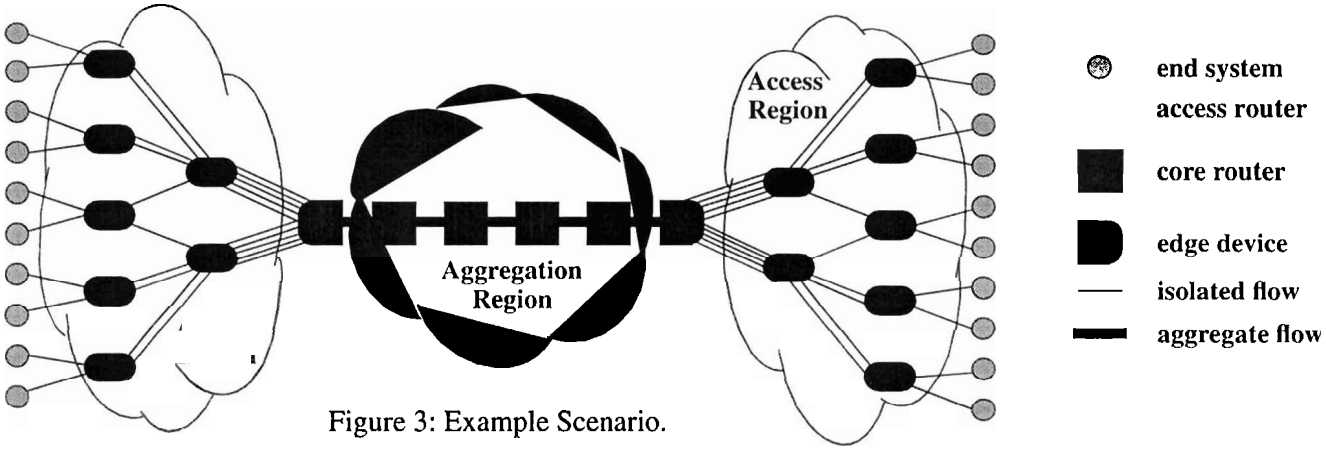

Figure 3: Example Scenario.

In Figure 4, the accumulated rate $\bar{R}_{aggr}$ for the aggregated system, i.e. the accumulated rate over all hops and all flows is depicted, in relation to the delay assigned inside the AR (note that the delay outside the AR is 100ms-delay inside AR), i.e. depending on the delay partition. The dotted line represents the accumulated rate for the segregated system $\bar{R}_{segr}$.

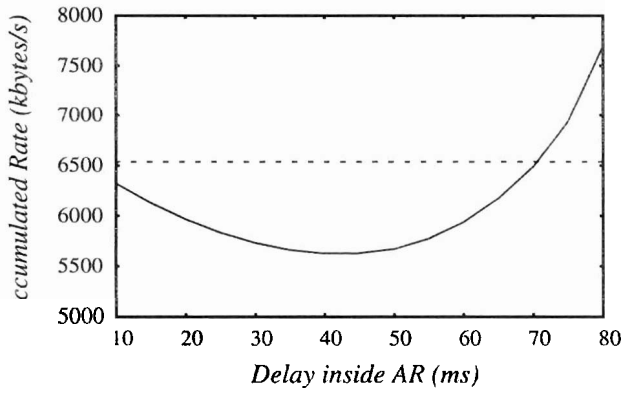

Figure 4: Segregated flows vs. aggregated flow.

Here we can see that aggregation can be beneficial in terms of resource usage if the delay partitioning is done carefully. The exact values for the accumulated rate and buffer consumption of the segregated and the aggregated system can be found in Appendix B. From those it can be seen that a delay bound of 40 ms inside the AR is optimal with respect to the accumulated rate. It gives a reduction of *13.74%* with respect to the accumulated rate while for the accumulated buffer it produces less than half (*46.67%*) of what is required for the segregated system (with respect to the accumulated buffer this delay partition is not optimal, however the buffer variations between different delay partitions are not very significant). Even if the simple approach of using the lower bound of the delay inside the AR (in our setting this is 22,949 ms, see again Appendix B) is taken (from (26)), maybe because it might be considered too time-consuming to search for the optimal delay partition or because not all the relevant information is available, a significantly better accumulated rate and buffer can be achieved than for the segregated system (*9.81%* for the accumulated rate and *53.78%* for the accumulated buffer).

C. Simulations for Parameter Sensitivity Analysis

The above example might seem ad hoc respectively arbitrary and it actually is. Therefore, in this section we conduct a more thorough numerical analysis of the derived concepts/mathematical tools by using randomly generated groups of flows. For this purpose we developed a small simulation environment which allows us to analyse and visualize the sensitivity of the aggregated system on exogenous variables like traffic specifications and network configuration parameters. We concentrate on the service rate as resource parameter here as it seems more sensitive on parameter changes, and furthermore one could argue that it is the economically more interesting parameter than buffer space

[10]. In order to have a better metric than accumulated rate allocations for the comparison of different experiments we define the *Aggregation Efficiency* (*AE*) as follows:

$$AE = \frac{\bar{R}_{segr} - \bar{R}_{aggr}}{\bar{R}_{segr}} \in (-\infty, 1) \tag{27}$$

An $AE > 0$ means the aggregated system performs better than the segregated system with respect to rate allocations, while a negative *AE* indicates the opposite.

In all of the experiments we have usually kept all but one parameter fixed and investigate how variations of the regarded parameter affect the *AE*. As in the simple example above we choose again to treat the delay partition as the design variable of the aggregated system and therefore show the achieved *AE* within our experiments in relation to the delay allocated for the network hops inside the AR. In most experiments the same network configuration as in the simple example has been assumed (see Figure 3): 5 "ATM hops" inside the AR and 2 "Fast Ethernet" hops in front as well as behind the AR (with the error terms as defined in the preceding sections). Furthermore, we assume that all flows have the same requirements for the end-to-end delay bound $d_{max}=100$ *ms*. All experiments have been repeated until they reached a confidence interval size < 0.01 at all the measurement points, which required between 20-50 runs of each experiment in our simulator. The values from which the curves are drawn are the sample means. The confidence intervals are not shown for reasons of better legibility of the graphs. Let us now look at the parameter sensitivity experiments.

1) Different Number of Flows

At first we examine the influence of the number of flows (*N*) that are to be aggregated. For the traffic specifications it has been assumed that p is chosen randomly from a uniform distribution over [5000,10000] and r is chosen from the uniform distribution over $[0.3p, 0.5p]$. Similarly, M is taken from the uniform distribution over [50,200] and b from $[M, 0.1r]$. Obviously we have taken rather "narrow" flows with moderate burstiness, since it is them who are most needy with regard to aggregation. They are taken from the range where one would expect IP telephony flows would fall in. The randomness introduced should make it harder for the aggregated system to deal with the resulting heterogeneity of the flows. In Figure 5 the *AE* for $N = 10, 100, 1000$ is depicted in relation to the delay available inside the AR.

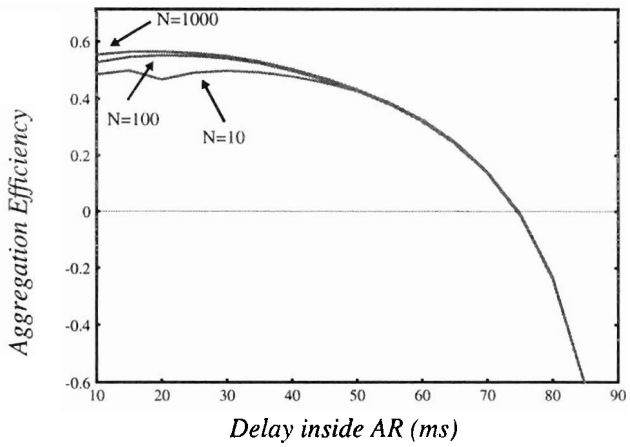

Figure 5: *AE* for different numbers of flows.

As can be easily seen the influence of the number of flows on the *AE* is not dramatic and only exists for a low delay inside the AR. This essentially means that the number of flows may be regarded as a simple "scaling factor" for the performance of the aggregated system, although some stochastic effects as can be seen for N=10 are avoided for larger numbers of flows. Also note here that the aggregated system with low allocation of delay inside the AR performs extremely well in comparison to the segregated by saving more than half of its rate allocations. The opposite is however true if the delay partition is chosen the other way around.

2) Different Burst Sizes

Let us now look at the sensitivity of the aggregated system with respect to burst sizes (b) of the flows. The same set-

tings for the randomly generated flows as above are used, besides the fact that we now choose $N = 1000$ fixed and vary the uniform distribution from which b is chosen over $[M,0.1r]$, $[0.3r,0.5r]$ and $[1.5r,2.0r]$. These three alternatives represent flow groups with low, medium and high burst sizes. The results of this experiment are shown in Figure 6.

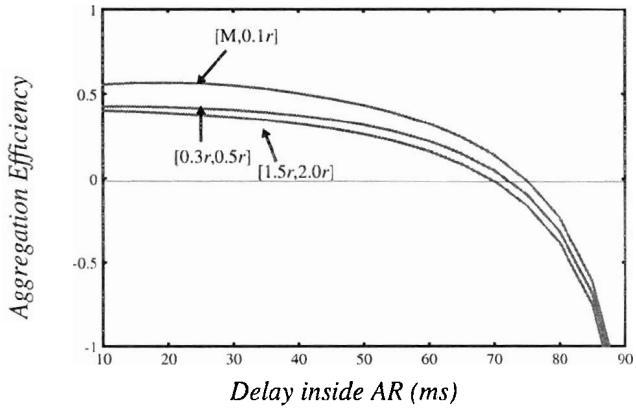

Figure 6: *AE* for different burst sizes.

While it can be observed that the flows with smaller bursts achieve a better *AE*, it has to be noted that the *AE* is not very sensitive on the burst sizes, since all three flow group curves are located pretty close together and have the same shape.

3) Different Burst Intensities

After having examined the effect of different burst sizes, let us now see how the intensity of a burst influences the *AE*. We therefore define the *burst intensity* as the ratio p/r. We use the same setting as in the preceding experiment, except that b is fixed again now (chosen from $[M,0.1r]$) and r is taken randomly from $[0.1p,0.3p]$, $[0.4p,0.6p]$, respectively $[0.7p,0.9p]$, thus varying the burst intensity from high to low. In Figure 7 these three alternatives are compared.

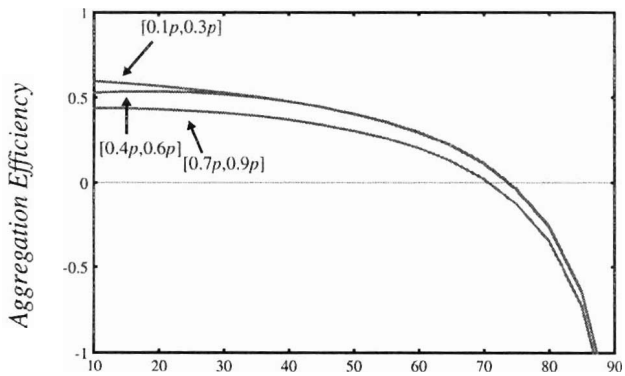

Figure 7: *AE* for different burst intensities.

Again the *AE* is not very sensitive on variations of the burst intensity, solely a small decrease in *AE* can be noticed when the burst intensity is chosen higher. Together with the last experiment this tends to imply that the burst characteristics do not very much influence the performance of the aggregated system. This is good news, since it means that one does not have to pay much attention to different burst characteristics when aggregating flows.

4) Different Maximum Packet Sizes

The next experiment is concerned with the effect of different maximum packet sizes of the flows on the *AE*. Again the same settings are used, except that r is fixed again (chosen from $[0.1p,0.3p]$) and the uniform distribution from which M is taken is now varied between [50,200], [300,500], and [1200,1500].

The results of this experiment are shown in Figure 8.

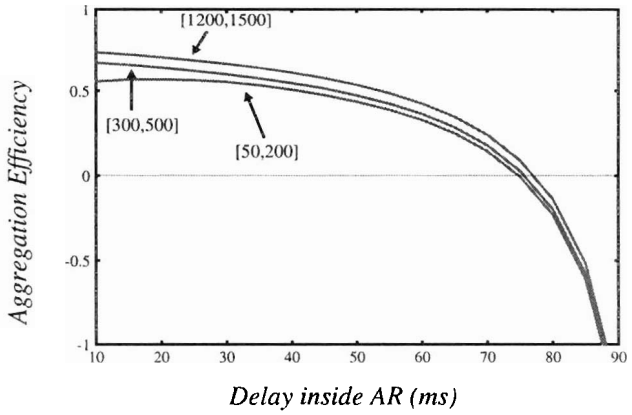

Figure 8: *AE* for different maximum packet sizes

We observe that higher M lead to higher *AE*, which is the expected result since higher M imply higher error terms and thus more gains from "paying scheduling errors only once" can be achieved. Yet, again it has to be noted that *AE* is not highly sensitive on this parameter.

5) Different Flow Sizes

As mentioned above we have so far assumed rather "small" flows. Now we want to investigate how larger flow sizes affect the aggregation. Therefore we do three different simulation runs with flows chosen from different "populations" representing small, medium and large flows. For the small flows we use the flow setting as above. For the medium flow category p is chosen from [30000,70000], r is chosen from [0.1p,0.7p], M is taken from [100,500] and b from [0.1r,0.5r]. The kind of flows one could think of here may be video-conferencing flows with several participants but only moderate video quality. For the large flow category p is chosen from [100000,250000], r is chosen from [0.1p,0.7p], M is taken from the uniform distribution over [200,1500] and b from [0.1r,0.5r]. This flow category should represent characteristics as they are typical for streamed video transmissions using e.g. MPEG-1 encoding. For each of the flow categories the simulation was run with N = 100 flows[4] and the usual network configuration. The results of this experiment are depicted in Figure 9.

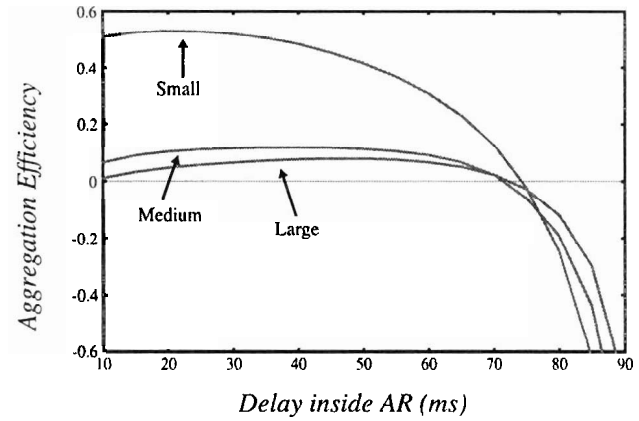

Figure 9: *AE* for different flow sizes.

Obviously there is a strong sensitivity of the *AE* for different flow sizes. Small flows yield much better *AE* than larger flows. This means that one should always first try to aggregate smaller flows. This also makes much sense from the state complexity perspective since small flows contribute most to the state scalability problem inside core networks. While this result might have been intuitively obvious it is nice to find some numerical evidence for it as well.

4. N = 1000 flows could not be used, since then the large flows would (correctly) not all be admitted by our simulator.

6) Different Traffic Mixes

Having seen the strong influence of flow sizes in the preceding experiment, it may be interesting to see how different traffic mixes consisting of different compositions of the three flow categories as defined above behave when being aggregated. We therefore tested 4 different compositions of the three flow categories as given in Table 4.

traffic mix	# small flows	# medium flows	# large flows
A	1000	0	0
B	800	100	100
C	500	300	200
D	300	400	300

TABLE 4 Different Traffic Mixes.

Traffic mix A corresponds to our default setting with flows taken only from the small flow category. From the compositions B to D there is a trend to have more and more larger flows.

The results of this experiment are shown in Figure 10.

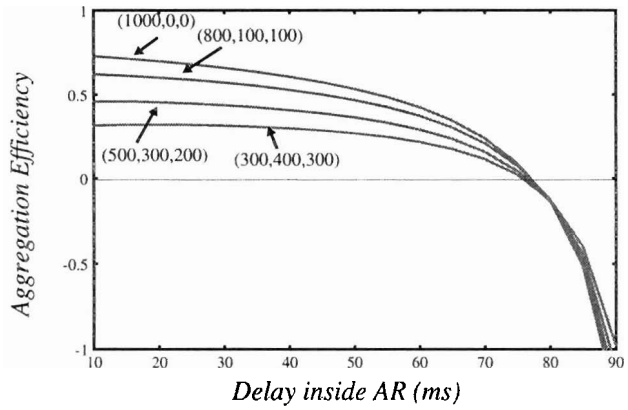

Figure 10: *AE* for different traffic mixes.

The observation that can be made is that the *AE* becomes less for those traffic mixes where the share of larger flows is higher - the expected result when taking into account the preceding experiment. The good news is that it seems that *AE* tends to be "additive" with regard to the different components of a traffic mix, which essentially means that smaller flows when being aggregated with larger flows do not suffer from the latter's worse "aggregatability". Note that a presumably typical traffic mix as B, with many small flows and some medium and large flows still performs very well if the delay partition is done appropriately.

7) Different Cost Tradeoffs

Up to now it has been assumed that the rate resource inside the AR and outside the AR is of equal value to the operator(s) of the overall network. Yet, this may seem a simplistic assumption (and it certainly is), as it may very well be that a network operator values resources outside the AR (the access area) and inside the AR (the core area) very differently. Often, there will be the situation that the resources in the core network are assessed more precious since they are shared by all access areas and thus competed for by these. However, the other way around, where access network resources are scarce, is also imaginable.

Consequently, we tried to capture that discussion by doing simulations for different cost tradeoffs between rate resources inside and outside AR. We used traffic mix B from the preceding section and applied the following cost tradeoffs between rate inside and outside AR: 1:10, 1:2, 1:1, 2:1, and 10:1. For example a cost tradeoff of 1:10 means access bandwidth from outside the AR is 10 times as precious as core bandwidth from inside AR, whereas 10:1 implies exactly the opposite.

The results of this are depicted in Figure 11.

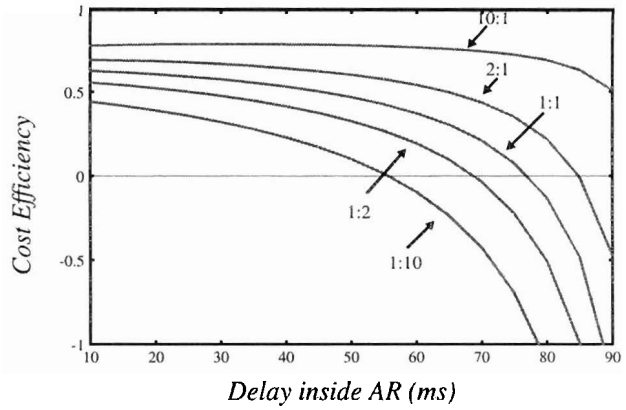

Figure 11: *AE* for different cost tradeoffs

Here we introduced instead of *AE* the *cost efficiency* (*CE*), which is a weighted version of the *AE* taking into account the cost relation between rate inside and outside AR. It can be seen that the more precious the rate inside the AR (backbone bandwidth) is assessed, the higher the *CE*. This means aggregation pays off especially if core bandwidth is a scarce resource. This can be seen best for the cost tradeoff 10:1, where no matter how the delay partition is done almost the same (high) *CE* is achieved.

Note that it is a general advantage of the two-level resource allocation system that it is possible to reflect such cost tradeoffs in different rate allocations inside and outside the AR.

8) Different AR sizes

So far we have only investigated flow related parameters (although the different cost tradeoffs could be viewed as a network configuration parameter). Let us now draw our attention to an important network configuration parameter: the size of the AR. We assume the same setting as in the preceding experiment with different cost tradeoff, but leave the cost tradeoff now at 1:1 again and instead vary the size of the AR as 1, 3, 5, and 7 hops (recall that we assume 9 hops in total). The results of this are shown in Figure 12.

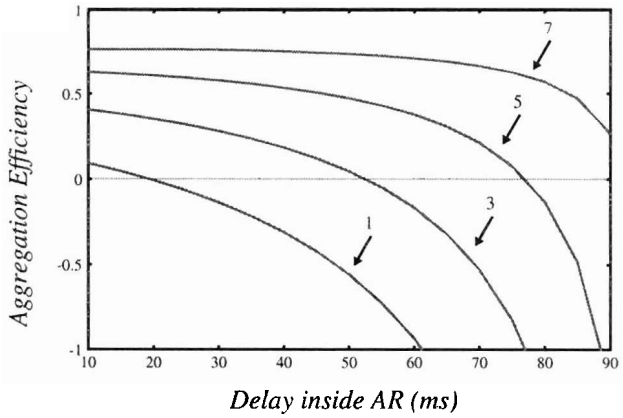

Figure 12: *AE* for different AR sizes.

We see the expected result that the *AE* depends very much upon the size of the AR. If the AR is too small then it is difficult to achieve a good *AE*, yet if it is large a high *AE* is almost granted. This is due to the effect that "paying the burst twice" by the two-level resource allocation can only be compensated by "paying scheduling errors only once" if the AR and therefore the error terms for the AR are large enough.

V. Related Work

In this section we discuss related work separated along the dynamic and static aspect of the aggregation problem

since these usually are very different areas of work.

A. Dynamic Aspect - Handling of Excess Traffic

Although we have not dealt with the static aspect of the aggregation problem, i.e. the way routers coordinate themselves to allow for aggregation and segregation of flows, we now want to briefly discuss a rather practical issue that arises when our results on the static aspect shall be applied to emerging technology in the field of QoS. While we have assumed a flow-based QoS technology like RSVP/IntServ as the technology being used outside the AR, we could in principle utilize the results for any of the following technologies inside the AR:

- ATM,
- Differentiated Services,
- RSVP/IntServ (Hierarchical RSVP/IntServ), or
- any connection-oriented technology that is able to give rate guarantees.

There are many issues to be dealt with when using aggregated RSVP-based requests over one of these technologies. This dynamic aspect of the aggregation problem is however not the focus of this article and we refer to other work in this area (for hierarchical RSVP/IntServ see [9], [3], [23], [1] for DiffServ see [2], for ATM see [19]). However, one of these issues, the "marking" of excess packets at the ingress into the AR, is related to the static aspects of aggregation we looked at in this article. This marking is required in order to not destroy the flow isolation stipulated by deterministic services like GS. So, if the AR is

- a DiffServ cloud then the DS byte could be used, e.g. by marking conformant traffic with the EF PHB and excess traffic with the DE PHB, furthermore the simplified arrival curves of Section III.D could be used as a profile.
- an ATM cloud then a separate VC for the conformant part of the aggregated flow should be used, while the best-effort VC (setup by e.g. Classical IP over ATM) could be used for excess traffic,
- an aggregated IntServ cloud there is a problem, since no marking mechanism is provided; while the individual flows could be strictly policed at their entrance to the AR and be forced to conform, this would disobey the GS specification's recommendation of sending excess traffic as best-effort.

In the case of a DiffServ cloud note that while DiffServ uses a class-based aggregation approach it may offer a "virtual leased line" (VLL) service as described in [14], though. Hence, our results based on the topological aggregation approach can be applied in order to dimension such a VLL.

B. Static Aspect

The use of piecewise linear functions as traffic envelopes has been suggested before, e.g. in [11], to give a better utilization of network resources for bursty sources like compressed video than the use of simple token buckets. While in these cases empirical evidence showed the utility of piecewise linear arrival curves with multiple segments, we looked at the case of a group of regulated flows were the gain can be shown analytically.

There is also some work on the generic problem of multiplexing regulated traffic onto shared resources (see e.g. [EMW95], [LZTK97], [GBTZ97]). However, all of these do not treat the case of delay-constrained flows and are thus not directly applicable to GS flows.

The problem of resource allocation for the grouping of GS flows has also been addressed by [18]. The discussion there is however restricted to the case of the simple token bucket model and homogeneous flows. We go one step further with our analysis for the model of TSpec-characterized flows and the inclusion of TSpec-heterogeneous flows. Furthermore, we do not restrict to grouping but also discuss how aggregation can be achieved (in terms of our terminology) and show by simulation how aggregation may affect resource usage and how exogenous parameters like traffic specifications and network configuration parameters influence this.

VI. Conclusion and Future Work

We believe that aggregation of stateful application flows inside the network is a necessary mechanism to retain scalability for large networks, as e.g. the Internet. We have looked at the static aspects of aggregation, i.e. which flows to aggregate and how much resources to allocate for the aggregated flow, for the specific case of IntServ's GS class. We have shown how it is possible to ensure the strong per-flow guarantees given by deterministic services despite aggregation in the core of the network. Furthermore, we found out that aggregation can offer interesting resource tradeoffs

between the AR and the non-AR part of the network if flow grouping and resource allocation is done carefully. We have given an example where the aggregated system even performed superior to the segregated system, whereas intuitively one might have thought that aggregation would only come at a price of more resources being required. Since an example is not a proof, we conducted a number of simulation experiments to investigate under which circumstances aggregation may pay off with regard to resource usage. While an aggregated system does not perform superior to a segregated system with regard to resource usage under all circumstances, we have given numerical evidence that there are many situations under which it does. This is a further argument for aggregation besides its main attraction of reducing state in the core of a large-scale network.

Left for future work is mainly the integration of the achieved results with the protocol-related aspects of the aggregation problem. It has to be noted that aggregation is inherently a dynamic problem, i.e. in general there are some already established groups of flows, so if new ones arrive, they must be assigned to these groups or groups must be reorganized. The derived formulas are good tools to aid such decisions, but how exactly is for further study. Furthermore, while we have made a serious approach towards investigating the parameter space of the aggregation problem, we were probably not able to cover all interesting "areas" of it. There are certainly more insights to be gained by further simulation experiments. Another area of interest could be how the average delay is affected by the presented mechanisms. In our simulation environment we assumed greedy sources which always led to worst-case behaviour. Extending the simulation environment by other traffic source models which are only bounded by linear traffic envelopes but do not necessarily meet them at all times should be straightforward and could shed some light on this issue, although it might be argued that average delay is of no interest for applications using deterministic services. Moreover, we have started to look at the aggregation of statistically guaranteed services. The problem here is that approaches from statistical multiplexing cannot be applied directly as most of them assume statistically independent sources. Yet, this can hardly be claimed for sources that have used a shared access region. How to resolve this open issue is left for future work as well.

VII. References

[1] F. Baker, C. Itturalde, F. Le Faucheur, and B. Davie. Aggregation of RSVP for IPv4 and IPv6 Reservations, March 2000. Internet Draft, work in progress.

[2] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E Felstaine. A Framework for Use of RSVP with DiffServ Networks, March 2000. Internet Draft, work in progress.

[3] S. Berson and S. Vincent. Aggregation of Internet Integrated Services State. In *Proceedings of 6th IEEE/IFIP International Workshop on Quality of Service, Napa, CA, USA*. IEEE/IFIP, May 18–20 1998.

[4] J.-Y. Le Boudec. Application of Network Calculus To Guaranteed Service Networks. *IEEE Trans. on Information Theory*, 44(3), May 1998.

[5] A. Charny. Delay bounds in a network with aggregate scheduling, February 2000. Cisco.

[6] Rene L. Cruz. Quality of Service Guarantees in Virtual Circuit Switched Networks. *IEEE Journal of Selected Areas in Communication*, 13(6), August 1995.

[7] A. Elwalid, D. Mitra, and R.H. Wentworth. A New Approach for Allocating Buffers and Bandwidth to Heterogeneous, Regulated Traffic. *IEEE Journal of Selected Areas in Communication*, 13(6), August 1995.

[8] S. Giordano, J. Y. Le Boudec, P. Thiran, and A. Ziedins. Multiplexing of Heterogeneous VBR Connections over a VBR Trunk. Technical report, EPFL, May 1997.

[9] R. Guerin, S. Blake, and S. Herzog. Aggregating RSVP-based QoS Requests, November 1997. Internet Draft, work in progress.

[10] Martin Karsten, Jens Schmitt, Lars Wolf, and Ralf Steinmetz. Provider-Oriented Linear Price Calculation for Integrated Services. In *Proceedings of the Seventh IEEE/IFIP International Workshop on Quality of Service (IWQoS'99), London, UK*. IEEE/IFIP, June 1999.

[11] E. Knightly, D. Wrege, J. Liebeherr, and H. Zhang. Fundamental Limits and Trade-offs of Providing Deterministic Guarantees to VBR Video Traffic. In *Proc. of ACM SIGMETRICS'95*, August 1996.

[12] J.Y. LeBoudec. A proven delay bound for a network with aggregate scheduling. Technical Report DSC2000/002, EPFL-DSC, January 2000.

[13] F. LoPresti, Z.-L. Zhang, D. Towsley, and J. Kurose. Source Time-Scale and Optimal Buffer/Bandwidth Tradeoff for Regulated Traffic. In *Proceedings of IEEE Infocom*, January 1997.

[14] Kathleen Nichols, Van Jacobson, and Lixia Zhang. RFC 2638 - A Two-bit Differentiated Services Architecture for the Internet. Informational RFC, July 1999.

[15] P. Oechslin, S. Robert, J.-Y. Le Boudec, and S. Giordano. VBR over VBR: The Homogeneous, Loss-free Case. In *Proceedings of IEEE Infocom*, January 1997.
[16] Abhay K. Parekh and Robert G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case. *IEEE/ACM Transactions on Networking*, 1(3), June 1993.
[17] Abhay K. Parekh and Robert G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case. *IEEE/ACM Transactions on Networking*, 2(2), April 1994.
[18] S. Rampal and R. Guerin. Flow Grouping for Reducing Reservation Requirements for Guaranteed Delay Service, July 1997. Internet Draft, work in progress.
[19] L. Salgarelli, M. DeMarco, G. Meroni, and V. Trecordi. Efficient Transport of IP Flows Across ATM Networks. In *IEEE ATM '97 Workshop Proceedings*, May 1997.
[20] S. Shenker, C. Partridge, and R. Guerin. Specification of Guaranteed Quality of Service, September 1997. RFC 2212.
[21] S. Shenker and J. Wroczlawski. General Characterization Parameters for Integrated Service Network Elements, September 1997. RFC 2216.
[22] I. Stoica and H. Zhang. Providing guaranteed services without per-flow management. Technical Report CMU-CS-99-133, Carnegie-Mellon University, May 1999.
[23] A. Terzis, J. Krawczyk, J. Wroczlawski, and L. Zhang. RSVP Operation over IP Tunnels, January 2000. RFC 2746.
[24] Paul White and Jon Crowcroft. Integrated Services in the Internet: State of the Art. *Proceedings of IEEE*, 85(12), December 1997.
[25] John Wroclawski. RFC 2210 - The Use of RSVP with IETF Integrated Services. Informational RFC, September 1997.
[26] Hui Zhang. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE*, 83(10), October 1995.

APPENDIX A: Buffer for Summed and Cascaded TSpec

For the buffer requirements B of the *summed TSpec* we obtain:

case 1: $\sum_{j=1}^{n} p_j > R \geq \sum_{j=1}^{n} r_j, \frac{C}{R} + D \leq \frac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j}$

$$B = M + \frac{\left(\sum_{j=1}^{n} p_j - R\right)\left(\sum_{j=1}^{n} b_j - M\right)}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j} + C + RD$$

case 2: $\frac{C}{R} + D > \frac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j}$

$$B = \sum_{j=1}^{n} b_j + \sum_{j=1}^{n} r_j\left(\frac{C}{R} + D\right)$$

case 3: $R \geq \sum_{j=1}^{n} p_j$

$$B = M + \sum_{j=1}^{n} p_j\left(\frac{C}{R} + D\right)$$

For the buffer requirements B of the *cascaded TSpec* we obtain ($k \in \{1,...,n\}$):

case 1: $\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l > R \geq \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l, \frac{C}{R} + D \leq \frac{b_k - M_k}{p_k - r_k}$

$$B = \sum_{l=1}^{k-1} (b_l - M_l) + M + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l - R \right) \left(\frac{b_k - M_k}{p_k - r_k} \right) + C + RD$$

case 2: $\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l > R \geq \sum_{j=k+1}^{n} p_j + \sum_{l=1}^{k} r_l,$

$$\exists h \in \{1, \ldots, n-k-1\} \frac{b_{k+h} - M_{k+h}}{p_{k+h} - r_{k+h}} < \frac{C}{R} + D \leq \frac{b_{h+k+1} - M_{k+h+1}}{p_{k+h+1} - r_{k+h+1}}$$

$$B = \sum_{l=1}^{k+h} (b_l - M_l) + M + \left(\sum_{j=k+h+1}^{n} p_j + \sum_{l=1}^{k+h} r_l - R \right) \left(\frac{C}{R} + D \right)$$

case 3: $\frac{C}{R} + D > \frac{b_n - M_n}{p_n - r_n}$

$$B = \sum_{l=1}^{n} (b_l - M_l) + M + \left(\sum_{l=1}^{n} r_l \right) \left(\frac{C}{R} + D \right)$$

case 4: $R \geq \sum_{j=1} p_j$

$$B = M + \sum^{n} p_j \left(\frac{C}{R} + D \right)$$

APPENDIX B: Accumulated Rate and Buffer for the Simple Numerical Example of Section IV.B

We denote the accumulated rate and buffer as aR_x and aB_x (in bytes/s respectively bytes), where x ∈ {SEGGR, AGGR,y}, i.e. the segregated and aggregated system, and y stands for the delay inside AR (in ms). MIN denotes the lower bound on the minimum available delay inside AR as obtained from (26), which is for the given example 22.949 ms.

x	aR_x	aB_x
SEGGR	*6524362*	*587925*
AGGR,10	6319383	257940
AGGR,15	6128250	264860
AGGR,20	5967073	269729
AGGR,MIN	*5884343*	*271761*
AGGR,25	5833865	272862
AGGR,30	5730647	274542
AGGR,35	5660979	275250
AGGR,40	5627958	274973
AGGR,45	5629268	273696
AGGR,50	5669737	271530
AGGR,55	5773221	270084
AGGR,60	5935809	268507
AGGR,65	6169384	266233
AGGR,70	6484611	263128
AGGR,75	6933713	259144
AGGR,80	7693418	254275