Aggregation of Guaranteed Service Flows

Jens Schmitt[1] , Martin Karsten[1] , Lars Wolf[1] , and Ralf Steinmetz[1,2]
[1] Darmstadt University of Technology *
[2] German National Research Center for Information Technology, GMD IPSI
{Jens.Schmitt, Martin.Karsten, Lars.Wolf, Ralf.Steinmetz}@KOM.tu-darmstadt.de

Abstract

It is common belief that the Integrated Services architecture (IntServ) is not scalable to large networks as, e.g. 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 IntServ traffic flows in the core of the network. 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, we show in this paper 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. Applications of these theoretical insights could be to use the derived formulas for resource allocation in either a hierarchical RSVP/IntServ, IntServ over DiffServ (Differentiated Services), or IntServ over ATM network.

Keywords: Integrated Services, Aggregation, Guaranteed Service, Network Calculus.

1 Introduction

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 (IntServ) and a resource reservation protocol (RSVP) to "signal" users' requirements with respect to service classes and their parameters (see [WC97] 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.

So, one obvious approach to this problem is the aggregation of IntServ 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 is necessary (as e.g. described in [GBH97], [BV98], or [TKWZ99]). 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 paper, we look at the static aspect of aggregation for the specific case of IntServ's Guaranteed Service flows. We regard the Guaranteed Service class as particularly interesting due to its comparably strong guarantees on rate, delay and loss. Furthermore, due to its mathematical description it allows for an exact analysis with regard to the problem of resource allocation for aggregated flows.

1.1 Assumptions and Terminology

The part of the network that only "sees" aggregated flows will further on be called "aggregation region". Flows that shall be aggregated must share the same path over the aggregation region. We therefore constrain 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 are considered to be 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 paper 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.

1.2 Outline

In the next section we give a brief review of the semantics and basic mathematical background of the IETF's Guaranteed Service class. 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 numerical examples on the trade-offs for the resource allocation inside and outside of the aggregation region, we briefly discuss some of the issues when applying the results on concrete candidates for the aggregation region, like

*. This work is sponsored in part by: Volkswagen-Stiftung, Hannover, Germany and by Deutsche Telekom AG, Darmstadt, Germany.

an IntServ, DiffServ, or ATM cloud. Before concluding the paper, we also give an overview of related work.

2 The IETF Guaranteed Service Class

Guaranteed Service (GS) as specified in [SPG97] 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).*

Due to its mathematically provable bounds on end-to-end queuing delay we consider GS to be of high importance for time-critical applications as, e.g., in the domain of telemedicine.

The mathematics of GS are originally based on the work of Cruz [Cru95] (refined by others, see e.g. [Bou98]) on arrival and service curves. In case of the IntServ specifications the arrival curve corresponding to the *TSpec(r,b,p,M)* is

$$a(t) = min(M + pt, b + rt) \qquad (1)$$

whereas the service curve for GS is

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

where $V = \frac{C}{R} + D$ and *R* is the service rate.

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 ([PG93], [PG94]).

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

$$d_{max} = \frac{b}{R} + \frac{C}{R} + D \qquad (3)$$

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

$$\begin{array}{ll} p \geq R \geq r & d_{max} = \frac{(b-M)(p-R)}{R(p-r)} + \frac{M+C}{R} + D \\ R \geq p \geq r & d_{max} = \frac{M+C}{R} + D \end{array} \qquad (4)$$

From the perspective of the receiver desiring a maximum queuing delay d_{max}, the rate *R* (in bytes/s) that has to be reserved at the routers on the path from the sender follows directly from (3) and (4):

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

$$R = \frac{b + C}{d_{max} - D} \qquad (5)$$

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

$$R = \begin{cases} \dfrac{p\frac{b-M}{p-r} + M + C}{d_{max} + \frac{b-M}{p-r} - D} & p \geq R \geq r \\ \dfrac{M+C}{d_{max} - D} & R \geq p \geq r \end{cases} \qquad (6)$$

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

$$B = \begin{cases} M + \frac{(p-R)(b-M)}{p-r} + C + RD & p \geq R \geq r, \frac{C}{R} + D \leq \frac{b-M}{p-r} \\ b + r\left(\frac{C}{R} + D\right) & \frac{C}{R} + D > \frac{b-M}{p-r} \\ M + p\left(\frac{C}{R} + D\right) & R \geq p \geq r \end{cases} \qquad (7)$$

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 [PG93], because they also apply to many other scheduling algorithms [Zha95]

$$C = M; \qquad D = \frac{M'}{c} \qquad (8)$$

where *M* is the maximum packet size of the flow, *M'* is the MTU 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≈95*p* 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.

3 The Mathematics of Flow Grouping

In this section we derive some fundamental formulas about flow grouping. We show how grouping of flows can save resources when compared to isolated flows.

3.1 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

*. For our discussions we can omit this parameter of the TSpec further on.

$$\sum_{i=1}^{n} TSpec(r_i, b_i, p_i, M_i) = TSpec\left(\sum_{i=1}^{n} r_i, \sum_{i=1}^{n} b_i, \sum_{i=1}^{n} p_i, max(M_i)\right) \tag{9}$$

In RFC 2216 [SW97], which gives the general requirements for specifying service classes, the summation of TSpecs is described 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

$$R^I(S) = \sum_{i=1}^{n} \frac{b_i + C}{d_{max,i} - D} \tag{10}$$

while for the grouped system of these *n* flows, with the sum of single token buckets defined analog to (9), it is

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

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.:

$$GE(S) = R^I(S) - R^G(S) \tag{12}$$

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 $R= \{R_1,...,R_k\}$

such that $\sum_{l=1}^{k} GE(R_l)$ and *k* are minimized.

It can be easily seen from (11) 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: Let $S=\{1,...,n\}$ be a set of reservations ($tb_i=(r_i,b_i)$ *and* $d_{max,i}$), $i=1,...,n$. 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 $R(P) = \sum_{i=1}^{n} R(P_i)$.

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 $d_{max,i}$).

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

$$\begin{aligned} R(Q) &= R(P) - \frac{b_h + C}{min(d_{max,i}, i \in P_u) - D} + \frac{b_h + C}{min(d_{max,i}, i \in P_v) - D} \\ &< R(P) \end{aligned} \tag{13}$$

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. ❐

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:

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

That means we obtain gains independent of the reserved rate for delay-homogeneous flows, i.e. these gains are relatively highest if the single 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 B*, 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 partial delay, and TSpec parameters), where we use the summed TSpec as arrival curve for the grouped flow:

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

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: For a set *S* of *n>1* delay- and TSpec-homogeneous flows *GE>0* if *C>Mr/(p-r)*. [a very weak condition taking into account that for many schedulers *M* is the rate-dependent error term and that there may be other rate-dependent deviations]

Proof: We have to distinguish two cases for isolated flows: *R≥p* (1) or *R<p* (2). Analogously, there are two cases for the grouped flow: *R≥np* (3) and *R<np* (4). The only possible com-

binations are (1)+(3), (1)+(4) and (2)+(3). (2)+(4) is impossible as can be verified easily.

“(1)+(3)”:

$GE(S) = R^I(S) - R^G(S) = n\frac{M+C}{d_{max}-D} - \frac{M+C}{d_{max}-D} = (n-1)\frac{M+C}{d_{max}-D} > 0$, for *n>1* (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)+(3)”:

$$\begin{aligned} GE(S) = R^I(S) - R^G(S) &= n\frac{p\frac{b-M}{p-r} + M + C}{d - D + \frac{b-M}{p-r}} - \frac{np\frac{nb-M}{np-nr} + M + C}{d - D + \frac{nb-M}{np-nr}} \\ &= \frac{np\frac{b-M}{p-r} + nM + nC}{d - D + \frac{b-M}{p-r}} - \frac{p\frac{nb-M}{p-r} + M + C}{d - D + \frac{nb-M}{np-nr}} \\ &> \frac{np\frac{b-M}{p-r} + nM + nC - \left(p\frac{nb-M}{p-r} + M + C\right)}{d - D + \frac{b-M}{p-r}} \\ &= \frac{n-1}{d - D + \frac{b-M}{p-r}}\left(C - M\frac{r}{p-r}\right) \end{aligned},$$

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*.

3.2 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.

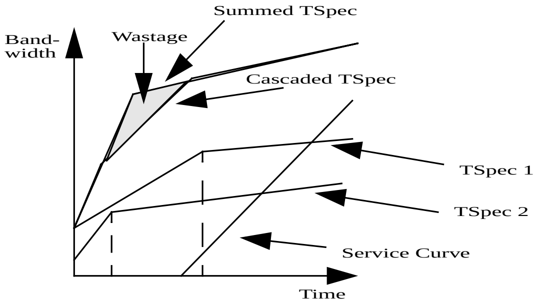

Figure 1: Summed vs. Cascaded TSpecs.

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 and which allow for 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 1. 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

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

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(M₁,...,Mₙ)*.

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

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 ⊗ as concatenation operator for token buckets)

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

If we apply the known results from network calculus [Bou98] on this tight arrival curve, assuming the GS service curve, we obtain the delay bound

$$\begin{aligned} d_{tac} \leq h(tac, c) &= sup_{s \geq 0}(inf\{T : T \geq 0 \wedge tac(s) \leq c(s+T)\}) \\ &= \frac{\sum_{l=1}^{k-1} (b_l - M_l) + M + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l\right) x_k}{R} - x_k + \frac{C}{R} + D \\ &= \frac{(p_k - r_k)\sum_{l=1}^{k-1} (b_l - M_l) + \left(\sum_{j=k}^{n} p_j + \sum_{l=1}^{k-1} r_l - R\right)(b_k - M_k)}{R(p_k - r_k)} + \frac{M+C}{R} + D \end{aligned} \quad (17)$$

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

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

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

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

It can be easily shown that, for a given rate *R*, d_{sum} is always greater than or equal to d_{tac} [Sch98], since the summed TSpec “contains” the cascaded TSpec.

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)

$$R = \begin{cases} \dfrac{\sum_{j=1}^{n} p_j \dfrac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j} + M + C}{d_{max} + \dfrac{\sum_{j=1}^{n} b_j - M}{\sum_{j=1}^{n} p_j - \sum_{j=1}^{n} r_j} - D} & \sum_{j=1}^{n} p_j > R \ge \sum_{j=1}^{n} r_j \\ \dfrac{M+C}{d_{max} - D} & R \ge \sum_{j=1}^{n} p_j \end{cases}, \qquad (21)$$

whereas for the cascaded TSpec we obtain for some
$k \in \{1,...,n\}$: (22)

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

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

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

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

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 however not very intuitive, we want to illustrate the effects of flow grouping on delay, rate and buffer requirements by presenting some numerical examples.

3.3 Numerical Examples of the Grouping Gains

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=155Mb/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}*=50ms*. The TSpecs of the flows are as given in the following table:

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

Let us first assume that we want to group 10 flows with TSpec# 1. Then we obtain:

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

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. We obtain

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

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 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, length *(b-M)/(p-r)* 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 theoretically does not violate the worst-case delay bound, is complicated to handle and might in reality add some delay after all. So, we address this issue in the next section.

3.4 Policing/Shaping the Grouped Flow

Once the service rate is calculated from (22), it is possible to achieve the desired delay bound with a much simpler arrival curve. It can be shown [Sch98] that the following arrival curve is sufficient for achieving the same delay bound for a given *R* as the tight arrival curve:

$$a(t) = \begin{cases} \sum_{l=1}^{k-1}(b_l - M_l) + M + \sum_{j=k}^{n} p_j t + \sum_{l=1}^{k-1} r_l t & t \le x_k \\ \sum_{l=1}^{k}(b_l - M_l) + M + \sum_{j=k+1}^{n} p_j t + \sum_{l=1}^{k} r_l t & t > x_k \end{cases} \tag{23}$$

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

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

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 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.

While the delay bound remains the same as for the cascaded TSpec, the buffer requirements depend on whether $V<=x_k+1$ or $V>x_k+1$. 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 $h \in \{1,...,n\text{-}k\}$, or more formally:

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

or, as token bucket concatenation:

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

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

4 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 give some numerical examples on how that scheme would perform. Afterwards we take a short look at the application of the model to emerging network technology supporting QoS.

4.1 Conceptual Model

We view 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.

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:

$$\frac{M + C_{sum}}{R} + D_{sum} \le d_p \le \frac{(b - M)(p - R)}{R(p - r)} + \frac{M + C_{sum}}{R} + D_{sum} \tag{24}$$

where C_{sum} and D_{sum} 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* error terms 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* and allocating more resources outside the AR. This trade-off should probably be weighted by how scarce the resources inside and outside the AR really are.

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

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 [Bou98].

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*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 3.4. 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 some numerical examples in the next section.

4.2 Numerical Examples

For the AR let us assume the same setting as in Section 3.3, i.e. we use the same 10 flows as specified there 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=1500bytes* and *c=100Mb/s* ("Fast Ethernet hops"). Furthermore, we assume that all flows have the same requirements for the end-to-end delay bound d_{max}*=100ms*.

In Figure 2, the accumulated rate, i.e. the rate over all hops and all flows is depicted, in relation to the delay inside the AR (note that the delay outside the AR=100-delay inside AR), i.e. depending on the delay partition. The dotted line represents the accumulated rate for the segregated system.

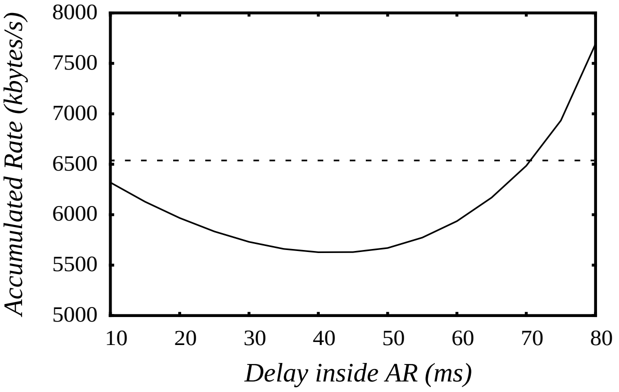

Figure 2: Segregated flows vs. Aggregated Flow.

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 is less than half (*~46.67%*) 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) is taken (from (24)), 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).

4.3 Application To Emerging Technology

While we have assumed 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 gives rate guarantees.

There are many issues to be dealt with when using aggregated RSVP-based requests over one of these technologies. These dynamic aspects of the aggregation are however not the focus of this paper and we refer to other work in this area (for hierarchical RSVP/IntServ see [GBH97], [BV98], [TKWZ99], for DiffServ see [BYF$^+$99], for ATM see [SDMT97]). 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 paper. This marking is required in order to not destroy the flow isolation stipulated by the GS specification. So, if the AR is a(n)

- 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 3.4 could be used as a profile.
- 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,
- Aggregated IntServ cloud there is a problem, since no marking mechanism is provided; while the individual flows could be policed strictly 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.

5 Related Work

The use of piecewise linear functions as traffic envelopes has been suggested before, e.g. in [KWLZ95], 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 [RG97]. 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).

6 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 GS despite aggregation in the core of the network. Furthermore, we found out that aggregation can offer interesting resource trade-offs 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. Though an example is not a proof, it is at least a hint that aggregation could offer more efficient network resource usage, a further argument for aggregation besides its main attraction of reducing state in the core of a large network.

For future work there is certainly the necessity of a more formal investigation under which circumstances aggregation offers more efficient resource usage in comparison to the segregated system. We derived the necessary formulas, but a detailed analysis of the parameter space of possible topologies, different flow mixes, different scheduling disciplines remains to be done. In addition, it has to be noted that aggregation is 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 could be good tools to aid such decisions, but how exactly is for further study.

References

[Bou98] J.-Y. Le Boudec. Application of Network Calculus To Guaranteed Service Networks. *IEEE Trans. on Information Theory*, 44(3), May 1998.

[BV98] S. Berson and S. Vincent. Aggregation of Internet Integrated Services State, August 1998. Internet Draft, work in progress.

[BYF$^+$99] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K Nichols, M. Speer and R. Braden. Interoperation of RSVP/Int-Serv and Diff-Serv Networks, February 1999. Internet Draft, work in progress.

[Cru95] Rene L. Cruz. Quality of Service Guarantees in Virtual Circuit Switched Networks. *IEEE Journal of Selected Areas in Communication*, 13(6), August 1995.

[EMW95] 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.

[GBH97] R. Guerin, S. Blake, and S. Herzog. Aggregating RSVP-based QoS Requests, November 1997. Internet Draft, work in progress.

[GBTZ97] 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.

[KWLZ95] 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*, 1995.

[LZTK97] 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.

[ORBG97] 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.

[PG93] 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.

[PG94] 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.

[Sch98] J. Schmitt. Aggregation of Guaranteed Service Flows. Technical Report TR-KOM-1998-06, Darmstadt University of Technology, November 1998.

[SDMT97] 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.

[SPG97] S. Shenker, C. Partridge, and R. Guerin. Specification of Guaranteed Quality of Service, September 1997. RFC 2212.

[SW97] S. Shenker and J. Wroczlawski. General Characterization Parameters for Integrated Service Network Elements, September 1997. RFC 2216.

[TKWZ99] A. Terzis, J. Krawczyk, J. Wroczlawski, and L. Zhang. RSVP Operation over IP Tunnels, February 1999. Internet Draft, work in progress.

[WC97] Paul White and Jon Crowcroft. Integrated Services in the Internet: State of the Art. *Proceedings of IEEE*, 85(12), December 1997.

[Zha95] 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 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 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: $R \geq \sum_{j=1}^{n} p_j$

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

Appendix B - Accumulated Rate and Buffer

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 minimum available delay inside AR as obtained from (24), which is for the given example 22.949 ms.

x	aR_x	aB_x
SEGGR	**6524362**	**587925**
AGGR,MIN	5884343	271761
AGGR,10	6319383	257940
AGGR,15	6128250	264860
AGGR,20	5967073	269729
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