Decoupling Different Time Scales of Network QoS Systems

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

Keywords: Network QoS, Edge Device,Time Scales.

Abstract

Providing quality of service (QoS) in large-scale networks like the Internet inherently needs to deal with heterogeneous network QoS systems. Therefore, the interworking between different network QoS systems is of high importance. In this paper, the interworking with respect to a basic characteristic of network QoS systems, the time scale of the system, is under investigation. The time scale of a network QoS system is its speed of reaction to individual requests for differentiated treatment of units of service. A slow time scale system will prefer requests to arrive with a low frequency and persist unaltered for a substantial period of time while a fast one is able to support much higher arrival rates of requests and is thus more amenable for short-lived units of service. Obviously, when overlaying a slow time scale QoS system over a faster one, there is no problem. However, and that is a more likely case, for the overlay of a fast time scale system on a slow one, there is a mismatch to be mediated at the edge between the two. The technique that is applied at an edge device for this mediation is called *decoupling* of time scales. Decoupling can also be viewed as aggregation of requests in time in contrast to spatial aggregation on the data path. In the paper we develop an adaptive heuristic scheme to deal with decoupling and evaluate this scheme by extensive simulations.

1 INTRODUCTION

1.1 Motivation

Different time scales of QoS systems may arise due to different QoS architectures like RSVP/IntServ (Resource reSerVation Protocol/ Integrated Services) [1], DiffServ (Differentiated Services) [2], or ATM (Asynchronous transfer Mode) [3] being used but may also be due to different QoS strategies followed by providers even if they employ the same QoS architecture. Choosing different QoS architectures as well as different strategies results from serving different needs, e.g., for an access and backbone provider. An access provider that has a comparatively moderate load and directly connects to end-systems may favor a fast time scale system responding immediately to the end-systems requests. A backbone provider that connects access providers respectively offers transit services is generally faced with a drastically higher load of individual transmissions, so that reaction on the time scale of individual requests is usually not possible and a slower time scale system needs to be enforced.

When different time scales are in operation in heterogeneous network QoS systems, it is simply not possible to query the underlying QoS system each time an overlaid system is altering its state. Here, the system operating on a faster time scale needs to be smoothed when overlaying it onto a system that operates only on slow time scales. A realistic configuration for access and backbone providers may be, e.g., that access providers use RSVP/IntServ to suit their customers' needs while a backbone provider uses DiffServ with a Bandwidth Broker (DiffServ/BB) to allow for some dynamics but on a slower time scale. This scenario is shown in Figure 1.

Here it is also very obvious why a BB is generally not able to react to individual RSVP requests that are arriving at edge devices between access and backbone provider. Because if it did, the BB would need to operate at a throughput of requests that is proportional to the square of the number of access providers it serves - that is not scalable. To see this, assume each of N edge devices would have M (new or modified)

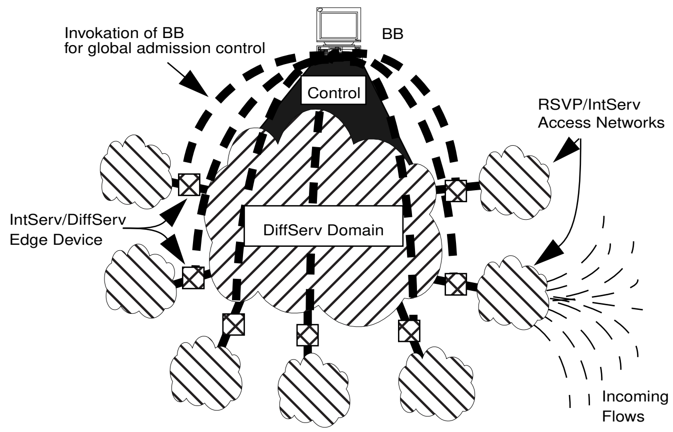

Figure 1: Combined local and global admission control.

RESV messages for each other edge device in a given time period and would query the BB for each of these requests. Then the BB would have to deal with $N \times (N\text{-}1) \times M$ requests in the same period. Note that the problem is not solved by spatial aggregation approaches like, e.g., [4] or [5] since for each of the M RESV messages the aggregate would have to be rearranged. Here a decoupling of the different time scales is necessary. The decoupling can be achieved by building "depots" of capacity which stabilize the fluctuations of the "nervous" demand curve for backbone capacity by individual requests. From another perspective, the decoupling technique can also be viewed as introducing a combined local and global admission control for the DiffServ/BB network. Global admission control is only invoked whenever local admission control at an edge device runs out of resources in its capacity depot. This scheme allows to trade off resource efficiency for a more stable and long-term capacity demand presented to the BB.

While the problem of different time scales is very obvious for the described interoperation of RSVP/IntServ over DiffServ/BB, it also occurs in other scenarios. For example, even in a homogeneous RSVP/IntServ case where both, access and backbone providers, use RSVP/IntServ, the backbone provider may decide to build up so-called RSVP tunnels [6] in which the individual requests from the access regions are fed. Again, the backbone provider can try to remain scalable on the control path by decoupling the different time scales and not rearranging the reservations for tunnels whenever an individual request is received by an edge device. The same applies to a backbone provider that operates an ATM network where several individual requests are collected together in a single virtual circuit.

Note that the slow time scale of an underlying QoS system may not express itself in being unable to process requests for QoS at short time scales but by the fact that significant setup costs are incurred for QoS requests between different administrative domains. Such a scheme of QoS tariffing is an instance where a QoS strategy of a network provider restricts the capabilities of the employed QoS architecture. A possible reason for this may be, e.g., that the charging and accounting system is not able to deal with a large number of individual requests since this involves a lot of operational costs.

1.2 Outline

In the next section, a closer and more formal look at the generic problem of decoupling time scales for heterogeneous network QoS systems is undertaken. Then solution techniques based on a heuristic adaptation scheme are devised and evaluated by simulations, before, at the end, related work is reviewed and some conclusions are drawn.

2 Decoupling Time Scales - The Problem and Its Complexity

2.1 Problem Statement

In order to assess the complexity of the decoupling problem, we first try to state the problem in a more formal manner. We model capacity as one-dimensional here, e.g., a rate resource that may be requested from a BB for a certain path across a DiffServ domain. This is certainly simplifying as more capacity dimensions like, e.g., a buffer resource may be involved. However, the resulting problem can be generalized albeit at the cost of a higher complexity (see [7] for a discussion of this). Hence, we can model the *capacity demand curve R* for the overlaid QoS system as a step function

$$R(t) = \sum_{i=1}^{n^R} f_i^R(t) \text{ where } f_i^R(t) = \begin{cases} h_i^R & t \in [s_i^R, e_i^R] \\ 0 & \text{otherwise} \end{cases} \text{ and}$$

$$e_i^R = s_{i+1}^R \quad \forall i \tag{1}$$

So, n^R is the number of steps here; h_i^R, s_i^R and e_i^R are the height, the start and the end of step i. Furthermore, we denote $l_i^R = e_i^R - s_i^R \quad \forall i$ as the length of step i.

From the capacity demand curve (CDC) for the overlaid system, the CDC of the underlying system is derived. A necessary condition on the CDC of the underlying QoS system is that it *covers* the CDC of the overlaid system. A cover of a CDC R is simply defined to be a CDC $\bar{R}$ for which $\bar{R}(t) \geq R(t) \quad \forall t$. An illustrative example of a CDC and a cover for it is shown in Figure 2.

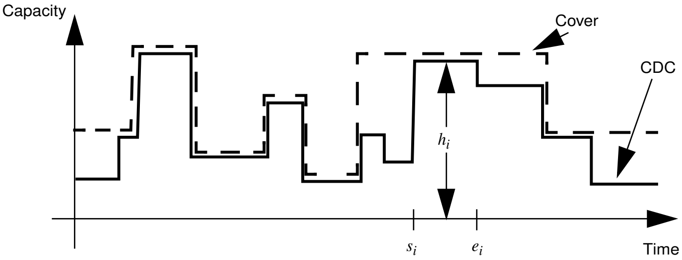

Figure 2: Example CDC with a cover.

For the underlying QoS system, it is assumed that a slow time scale is enforced by the introduction of setup costs for requests from the overlaid system. An alternative for enforcing a slow time-scale would be to only allow for a certain number of setups in a given period of time. The latter, however, is less flexible and can usually be achieved by choosing setup costs adequately.

The cost of a CDC R for an underlying QoS system in a given time period $[t_0, t_1]$ is defined as

$$c(R|F, U) = F \times n^R + U \times \int_{t_0}^{t_1} R(t)dt \tag{2}$$

where F are fixed setup costs involved for changing the requested capacity level and U are variable costs per capacity unit. We assume these parameters do not change in the planning period, although again this is easy to generalize [7].

Under these prerequisites, decoupling of QoS systems with different time scales can be formulated as a *minimal-cost CDC covering problem*, i.e.:

Find a CDC $\bar{R}$ for R such that $c(\bar{R}|F,U)$ is minimal.

The cost-minimal cover of a CDC R is denoted by R^{opt}.

2.2 Some Observations about Complexity

The possible set of covers for a CDC is, of course, unlimited without further restrictions being made. One observation is, however, that R^{opt} always is a *tight* cover. A cover C of a CDC R is called tight iff

$$\left\{ h_i^C \middle| i \in \{1, n^C\} \right\} \subseteq \left\{ h_i^R \middle| i \in \{1, n^R\} \right\}, \tag{3}$$

i.e., the step heights of the cover are a subset of the step heights of the CDC that is to be covered. The simple fact that R^{opt} is necessarily tight can be seen if one assumes that it is not. In that case it would be possible to lower R^{opt} for a step where it is not tight to the nearest h_i^R and it would still be a cover of R but with lower costs (at least if $U > 0$), which, of course, contradicts the cost-minimality. The space of tight covers is restricted as the following theorem states.

Theorem 1: The state space complexity for tight covers of a CDC with n steps is $O(2^{n-1})$.

Proof: We show this by giving a worst-case example of a CDC with n steps where the number of strict covers is indeed 2^{n-1}. Such a CDC is either monotonically increasing or decreasing; an example of a monotonically decreasing CDC (for $n = 6$) is depicted in Figure 3.

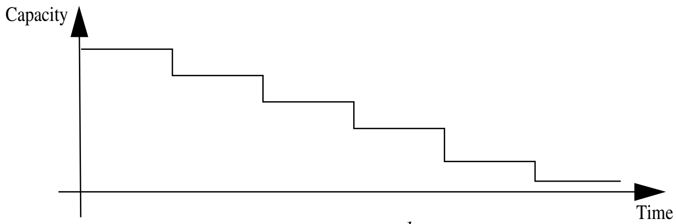

Figure 3: Example CDC yielding 2^{n-1} tight covers ($n = 6$ here).

We further on restrict without loss of generality on monotonically decreasing CDCs. Let $T(n)$ be the number of possible tight covers for a decreasing CDC with n steps as in Figure 3. We show the statement of the theorem by induction on the number of steps:

$n = 1$:

$$T(1) = 1 = 2^0 = 2^{1-1} \tag{4}$$

$n \rightarrow n+1$:

$$\begin{aligned} T(n+1) &= 1 + \sum_{i=1}^{n} T(n+1-i) = 1 + \sum_{i=1}^{n} 2^{n-i} \\ &= 1 + \sum_{j=0}^{n-1} 2^j = 1 + \frac{2^n - 1}{2 - 1} = 2^n \end{aligned} \tag{5}$$

If we assume that step n+1 of the CDC that was added for the induction step is the first, i.e., the highest one, then the first equation is due to the fact that we have n+1 possibilities for the length of this step in a strict cover of this CDC. One is just to prolong it to the end of the CDC. This alternative is represented by the first addend (the 1), the other alternatives are captured by the sum term, and correspond to prolonging the first step to an increasing number of steps of the CDC to be covered, for which the rest can then be covered by the same procedure but lesser steps. This allows then to apply the induction assumption, i.e., the theorem's statement, which eventually confirms the theorem. ∎

So, we see that while strict covers are limiting the space of possible covers, there is still a huge search space in which the cost-minimal cover, for which we are naturally striving, may be located.

All of the discussions so far have silently assumed that the search of the cost-minimal cover of a CDC could take place under certainty about this CDC. That is, of course, not the case in general. It would be the case if the overlaid system used only advance reservations (see, e.g., [8] or [9] for this concept). However, for immediate requests which we are focussing on here, the CDC that is to be covered is not known beforehand and for every step of the CDC, a decision has to be made whether the cover should follow this step or not. In fact, due to little experience with real network QoS systems, there is not even an established theory for statistical models on how a CDC could look like, although one could argue that some of the models known from telephony could be applicable to some parameters of the CDC. The parameters in question of the CDC are

- the step length l_i^R, which is a product of the interarrival times of the individual requests at an edge device and the duration of such requests, and
- the step height h_i^R, which corresponds to the aggregate capacity required to serve the requests.

Especially, the latter parameter is extremely difficult to model as there is no practical experience with it. It depends upon which applications are actually using reservations and how widely resource requirements are differing for actual reservation-based applications. The first parameter, the step length, might be modeled by markovian models known from teletraffic theory [10] as the characteristics might be similar (at least as long as the individual requests correspond to personal communications). However, also for this parameter, there is a certain degree of uncertainty whether traditional models fit.

3 Adaptation Scheme for Decoupling

From the observations of the preceding section, the need for adaptive heuristic techniques when tackling the decoupling problem under uncertainty about the CDC can be derived. The use of heuristic techniques is necessary since the involved problem is fairly complex even under certainty as discussed in the preceding section. Furthermore, as statistical models for CDCs are generally not available, we argue for the use of adaptation as a way to learn the statistical properties of the system in an on-line fashion. This is also highly useful in an environment where there are unpredictable, but rather long-term fluctuations in the demand for capacity. In general, the adaptation to behavior that would have been “good” in the past is the best a heuristic technique can do under complete uncertainty about a CDC.

The question what is “good” behavior can be assessed by comparing the outcome of an on-line heuristic with the results of applying a technique to solve the cost-minimal covering problem for the known CDC from the past. In the next section, such a technique as well as an inexpensive approximation is introduced. Hence, let us assume that we have a technique to solve the cost-minimal covering problem for the CDC of past system behavior. If we further on assume that a parametrized heuristic $h(\theta)$ is applied to the on-line cost-minimal CDC covering problem, there are essentially two different modes of adaptation that can be directed by good behavior as achieved by the cost-minimal cover of the past CDC:

Adaptation in Action Space. In this mode, the heuristic’s parameter (vector) θ is adapted such that the behavior of the CDC cover produced by applying the heuristic deviates as little as possible from the optimal cover with respect to some characteristic as, e.g., the number of steps of the optimal covers. More formally, if we define the similarity characteristic of two covers R and S as $s(R,S)$ (with higher values of $s(.)$ representing higher similarity), this means the adaptation problem is

$$\max.\ s(H(\theta), O)$$

where $H(\theta)$ and O represent the covers produced by applying heuristic $h(\theta)$ and the optimum technique.

Adaptation in Performance Space. In this mode the heuristics parameter (vector) θ is adapted such that the cost of the cover produced by applying the heuristic deviates as little as possible from the optimal cover’s cost. Again, this can be stated formally as

$$\min.\ c(H(\theta)) - c(O)$$

Discussions on which mode is better suited to our decoupling problem are postponed until Section 6 when the individual building blocks of the scheme like the employed heuristic and the technique for computing optimal covers have been investigated in more detail.

Both adaptation modes have three parameters with which a flexible trade-off between adaptation complexity and the cost performance of the optimum-directed adaptation can be achieved:

1. The *frequency of adaptation* determines how often the adaptation of the heuristics parameter is carried out.
2. The *time window of adaptation* determines the length of the past period that is taken into account for the adaptation.
3. The *accuracy of adaptation* determines how thoroughly the parameter space is searched during the optimization problem for the adaptation.

It might seem that the adaptation in performance space does not depend on the optimum cover to be computed as it is only a constant in the objective function. However, if one takes into account the accuracy of adaptation parameters, it is obvious that without the notion of a target cost to strive for the heuristic, this parameter cannot be set reasonably. Thus, in both modes of adaptation the optimal cover for the past CDC directs the adaptation. Therefore the whole scheme is called ODAH (Optimum-Directed Adaptive Heuristic).

4 Searching for the Minimal Cover under Certainty

As the ODAH scheme depends heavily on being able to compute the cost-minimal cover for past CDCs, the problem of finding such a cover for a CDC under certainty is investigated in this section. First, an exhaustive search technique to deterministically find the cost-minimal cover is presented. This approach, however, is computationally very expensive for CDCs with a considerable number of steps. Therefore an inexpensive approximation technique based on the optimal algorithm is devised.

4.1 Finding the Optimal Cover

Simply searching the space of strict curves is prohibitively expensive as Theorem 1 states. An observation that can be made for R^{opt} is that for the peak step of the regarded CDC R it takes the same value for the period of this step, i.e.,

$$R^{opt}(t) = R(t) \text{ for } t \in [s_k^R, e_k^R] \text{ with}$$

$$k = \left\{ j \in \{1, \dots, n^R\} \,\middle|\, h_j^R = max\left\{ h_i^R \,\middle|\, i \in \{1, \dots, n^R\} \right\} \right\} \tag{6}$$

Furthermore, it applies that the shapes of the right and left side from the peak, i.e., $[s_1^R, s_k^R]$ and $[e_1^R, e_{n^R}^R]$ do not influence each other. So, the question for R^{opt} is how far to prolong the peak step to the left and to the right. These observations can be combined into a divide-and-conquer strategy to recursively search the space of strict covers by the algorithm given in Figure 4, which is denoted OPT.

OPT finds the cost-minimal cover of a CDC under all circumstances, yet it is less expensive than a total enumeration of the space of strict covers by using the observation from (6) and by pruning the search space using a lower bound on the costs for further prolongations.

OPT has been implemented in a discrete-event simulation environment which simulates the overlaying of a fast time scale reservation system onto a slow one. The environment allows to generate CDCs with different statistical properties and to apply decoupling techniques on these CDCs in an on-line as well as an off-line manner. Using this simulation environment, OPT has been tested on a number of CDCs in order to obtain a feeling how complex it would be to compute a cost-minimal cover. An example CDC and the cost-minimal cover that has been computed by OPT is given in Figure 5.

For the simulated CDC R, we have had the following (arbitrary) settings: $n^R = 40$; $h_i^R \in [1,10]$ and $l_i^R \in [1,6]$ drawn from uniform random distributions; $F = 25$ and $U = 1$. This yields a cost of $c(R) = 1809$. Under these settings the cost-minimal cover O as computed by OPT has the following characteristics $n^O = 6$ and $c(O) = 1353$. Hence, had the optimal cover been used for decoupling the two QoS systems as simulated here, about 25% costs could have been saved. The saving in

```
OPT(R, a, b) // R is the CDC, a and b are the start and
         // end times for which to find an optimal cover
   if (a != b)
     find k // as defined in Equation (6)
     for l = k-1 downto a
       prolong R to the left till step l
       leftCost = OPT(R, a, e[l]) + cost for prolongation
       if (leftCost < minLeftCost)
         minLeftCost = leftCost
         left = l
       LB = sum of variable costs for steps from a to l
         + cost for prolongation
       if (LB > minLeftCost)
         break
     for r = k+1 to b
       prolong R to the right till step l
       rightCost = OPT(R, a, e[l]) + cost for prolongation
       if (rightCost < minRightCost)
         minRightCost = rightCost
         right = r
       LB = sum of variable costs for steps from r to b
         + cost for prolongation
       if (LB > minRightCost)
         break
     return minLeftCost + minRightCost +
          (s[right]-e[left])*h[k]*U + F
   else
     return 0;
```

Figure 4: Algorithm to find cost-minimal cover of a CDC (OPT).

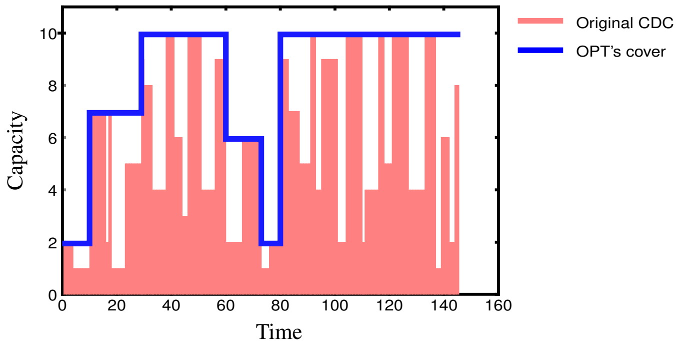

Figure 5: Cost-minimal cover computed by OPT.

cost, however, is, of course, totally dependent on the cost parameters F and U. If F is very high compared to U, then the cost savings can be considerably higher.

Larger values for n^R are generally not possible as even $n^R = 40$ already took up to a few seconds on average for the computations from OPT (on a 400 MHz Pentium-II processor). That OPT is increasingly expensive to compute can be seen when observing that the average size of the space of covers $S_O(n)$ searched by OPT for a CDC with n steps is recursively defined as (corresponding to the operation of the algorithm)

$$\begin{aligned} \bar{S}_O(n) &= 1 + \frac{1}{n}\sum_{k=1}^{n}\left(\sum_{i=1}^{k-1}\bar{S}_O(i) - \sum_{j=k}^{n-1}\bar{S}_O(n-j)\right) \\ &= 1 + \frac{1}{n}\sum_{k=1}^{n}\left(\sum_{i=1}^{k-1}\bar{S}_O(i) - \sum_{j=1}^{n-k}\bar{S}_O(j)\right) \end{aligned} \quad (7)$$

A comparison of $\bar{S}_O(n)$ with $S_T(n)$, the size of the space of tight covers, for some example values of n is given in Table 1. This is intended to give an illustration of how much is saved by OPT when compared to a total enumeration of tight covers. At the same time, of course, it also illustrates that even $S_O(n)$ is too large to be searched exhaustively (although the pruning is quite effective on average so that only a small part of the search space needs to be traversed). So, while the search space is diminished by the recursive operation of OPT, it is still too large if the number of steps of the CDC is becoming larger.

Table 1: Growth of search spaces.

n	10	50	100	200	500	1000
$\bar{S}_O(n)$	85.6	1.61e+6	3.78e+9	2.75e+14	1.72e+24	2.43e+35
$S_T(n)$	512	1.13e+15	1.27e+30	1.61e+60	3.27e+150	1.07e+301

An alternative formulation of the problem as an integer program is given in [7]. This opens up a standard set of operations research techniques to deal with the problem, however, all of these are computationally very expensive in the worst-case. That means, even if they produce the cost-optimal cover in a reasonable time on average, the execution times might vary considerably. This is something the ODAH scheme cannot deal with as it requires the optimum to be computed fast for recent past behavior to be able to adapt heuristics best. Therefore, we go a different way and try to find a good approximation of OPT that is computationally inexpensive.

4.2 Finding Near-Optimal Covers

In the ODAH scheme the optimum is required to adapt parameters from heuristics to "good" past behavior. In the preceding section, it has been shown that the determination of the optimum for a past CDC is very compute-intensive if the CDC becomes too large in terms of steps. That means if the time window of adaptation becomes moderately large, and that is generally desirable in order to take more past behavior into account, then it is more suitable to compute an approximation of the cost-minimal cover for the adaptation of the heuristic instead of the absolute optimum.

So, in this section, an approximation approach to compute the cost-minimal cover is introduced. It is based on the strategy followed by OPT but instead of trying all prolongations from a peak step for a certain part of the regarded CDC, it only compares the cost for prolonging the level of the peak until the next peak (in both directions certainly) with the sum of a further setup cost F and the cost of the subsection between the peaks being calculated by this strategy itself. We call this algorithm NEAROPT. It uses the notion of OPT to cut the problem into halves wherever possible and always tries only two different choices for prolongation. To compare exactly those two cases for each step of the algorithm is motivated by observations of the covers that were produced by OPT and which mostly used just either of these extremes. The detailed working of NEAROPT is given in Figure 6.

```
NEAROPT(R, a, b) // R is the CDC, a and b are the start and
         // end times for which to find an optimal cover
   if (a != b)
     find k // as defined in Equation (6)
     leftCost = NEAROPT(R, a, e[k-1]) + F
     if (leftCost < (e[k-1]-a)*h[k]*U)
       prolong R to the left till a
     else
       leftCost = (e[k-1]-a)*h[k]*U
     rightCost = NEAROPT(R, s[k+1], b) + F
     if (rightCost < (b-s[k+1])*h[k]*U)
       prolong R to the right till b
     else
       rightCost = (b-s[k+1])*h[k]*U
     return leftCost + rightCost + (s[k]-e[k])*h[k]*U + F
   else
     return 0;
```

Figure 6: NEAROPT algorithm.

The following theorem shows that NEAROPT is indeed substantially less expensive than OPT and should be easy to compute for all reasonable time windows of adaptation.

Theorem 2: NEAROPT has a time complexity of $O(n)$.

Proof: Let C be the time consumed for the operations in a single NEAROPT call without the time consumed by the recursive calls to the NEAROPT subroutine. Then the total time $T(n)$ for the execution of NEAROPT on a CDC with n steps is given by

$$T(n) = T(k-1) + T(n-k) + C \text{ for some } k \in \{1, n\} \quad (8)$$

The statement of the theorem is shown by induction on the number of steps of the CDC. The induction statement is

$$T(n) = nC \tag{9}$$

Then the induction works as follows

$n = 1$:

$$T(1) = C \tag{10}$$

$n \rightarrow n+1$: for some $k \in \{1, n+1\}$

$$\begin{aligned} T(n+1) &= T(k-1) + T(n+1-k) + C \\ &= (k-1)C + (n+1-k)C + C = (n+1)C \end{aligned} \tag{11}$$

Of course, $T(n) = nC = O(n)$ and thus the theorem holds. ■

So, NEAROPT has linear time complexity and is thus inexpensive to compute. The question certainly is how good the results are that can be achieved with NEAROPT. Therefore, a simulative comparison of NEAROPT with OPT is done. As a metric for this comparison, we use the achieved cost saving, denoted by ACS(NEAROPT) and defined as

$$ACS(NEAROPT) = \frac{c(R) - c(R^{nopt})}{c(R) - c(R^{opt})} \in [-\infty, 1] \tag{12}$$

where R^{nopt} is the cover as computed by the NEAROPT algorithm (later on the ACS(.) metric will also be used for other decoupling heuristics). As above, we use $F = 25$ and $U = 1$ for the fixed respectively variable cost of capacity from the underlying QoS system and draw the capacity demands h_i of the overlaid QoS system from a uniform random distribution over [1, 10]. For the step length l_i of the generated CDCs, we use 3 different scenarios, called F, M, and S, for which l_i is drawn from [1, 3], [1, 6] and [1, 10]. This corresponds to fast, medium, and slow fluctuations of the CDC. For all scenarios, we repeat the simulations 100 times. The sample means and standard deviations of the ACS(NEAROPT), μ^{ACS} and σ^{ACS}, and the average number of steps for the covers produced by OPT and NEAROPT, n^{OPT} and $n^{NEAROPT}$ are given in Table 2.

Table 2: ACS(NEAROPT) for different scenarios.

	Scenario F	Scenario M	Scenario S
μ^{ACS}	0.97	0.91	0.87
σ^{ACS}	0.002	0.008	0.01
n^{OPT}	3.4	6.1	12.6
$n^{NEAROPT}$	3.7	8.3	15.1

From these experiments, it can be seen that NEAROPT performs very well if the CDC is fluctuating very fast, and behaves worse if the fluctuations become slower, though still doing pretty well. This is due to the fact that for fast fluctuations as in F the problem actually becomes simpler because it rarely makes sense to change the capacity level as it is expensive compared to the time period for which the fixed setup costs can be amortized. Furthermore, it can be noticed that NEAROPT has a tendency to change capacity levels too often for all scenarios. Actually, when taking a closer look at the covers produced by NEAROPT and OPT, it was observed that NEAROPT often did not prolong capacity levels for peaks long enough. That often produced situations as depicted in Figure 7 (indicated by an **X**), where a small prolongation to the left or right from a peak would have yielded the optimum behavior (Figure 7 is one particular simulation outcome of type scenario S). This observation led to a simple improvement technique for the cover computed by NEAROPT: try to prolong each peak of R^{nopt} up to K steps (of the CDC) to the left and to the right and see if an improvement can be achieved. We call this improvement technique K-REPAIR and the combination of NEAROPT and K-REPAIR is denoted as NEAROPT-K. Of course, K-repair is an $O(n)$ algorithm for a fixed K and as NEAROPT and K-repair are performed sequentially, NEAROPT-K's time complexity is still $O(n)$.

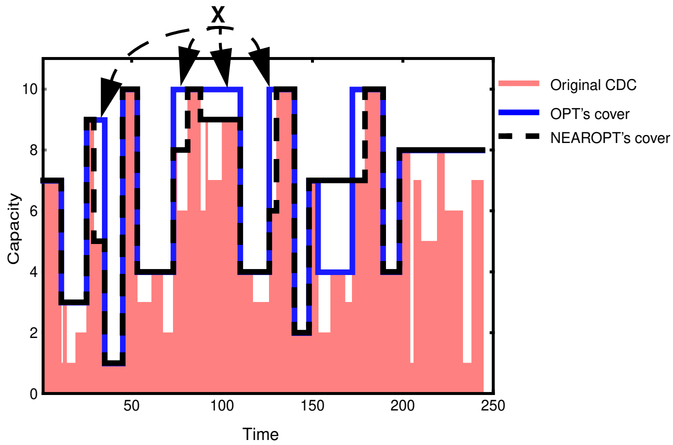

Figure 7: Covers computed by OPT and NEAROPT.

To investigate NEAROPT-K's potential and to see how different values for K perform, another set of simulations with the same parameters as above has been performed. The resulting means of the achieved cost saving for different values of K, denoted by $\mu^{ACS(K)}$, in the different scenarios are given in Table 3 (note that NEAROPT-0 is equal to NEAROPT).

Table 3: ACS(NEAROPT-K) for different scenarios.

	Scenario F	Scenario M	Scenario S
$\mu^{ACS(0)}$	0.97	0.92	0.86
$\mu^{ACS(1)}$	0.97	0.95	0.92
$\mu^{ACS(2)}$	0.98	0.96	0.95
$\mu^{ACS(3)}$	0.98	0.96	0.95
$\mu^{ACS(4)}$	0.98	0.97	0.96
$\mu^{ACS(5)}$	0.98	0.97	0.96
$\mu^{ACS(10)}$	0.98	0.97	0.96

It can be seen that the use of K-repair actually pays off, especially for scenario S. The experiments also exhibit that small values of K, e.g., 3 or 4, are performing well. Considerably larger values for K, e.g., 10, do not achieve better ACS values. This is, of course, beneficial with respect to the efficiency of NEAROPT-K.

In conclusion, the experiments indicate that NEAROPT-K with small values for K is a good approximation technique for finding near-optimal covers of a known CDC. It is very fast as its time complexity is linear in the number of steps of a CDC (for the rather small CDCs in the simulations above it could be computed in the order of µsecs on a 400-MHz Pentium-II processor). So, it can serve as substitute for the optimum calculation within the ODAH scheme if larger time windows of adaptation are to be used than the exact technique for determination of the cost-minimal cover can accommodate.

5 Simple Heuristic for Decoupling - Thresholded Depot Excess

So far, it has been investigated how to compute covers under certainty about the CDC that is to be covered, yet the decoupling problem needs to compute covers under uncertainty. In this section, a very simple, yet reasonable heuristic is introduced that deals with the problem under uncertainty at each single step in time. It is called thresholded depot excess (TDE) as it ensures that the capacity depot held for decoupling is never above a certain threshold.

Note that TDE is to be regarded as an illustrative example for how parametrized heuristics can be integrated into the ODAH scheme and how they can be improved by this integration. There are certainly "smarter" heuristics than TDE. However, the emphasis is on investigating what results can be achieved by adaptation of a simple heuristic like TDE (see Section 6).

The exact working of the TDE algorithm is given in Figure 8. A slotted time is assumed for TDE and the algorithm is applied in every time slot. If the CDC rises above the current capacity depot, i.e., $D(t\text{-}1) < R(t)$, this change is always followed (assuming that there is enough capacity at the underlying QoS system). Whenever the CDC takes a step downward, i.e., if $R(t) < R(t\text{-}1)$, TDE checks whether the step is smaller than a certain fraction $\alpha \in [0,1]$ of the old level of the capacity depot ($D(t\text{-}1)$) and if that is the case, TDE follows this step.

```
TDE(t, alpha) // R is the CDC and alpha is the
        // the relative threshold parameter
    if (R[t] < alpha*D[t-1] || R[t] > D[t-1])
      D[t] = R(t)
    else
      D[t] = D[t-1]
```

Figure 8: TDE algorithm.

An obvious refinement of TDE could be to always leave a certain safety margin between the depot level and the CDC when taking a step downward. Another would be to integrate some memory about past steps into the decision to follow a step or not, which would be particularly suited to non-stationary CDCs. However, as noted above, here we do not want to pursue such refinements any further but stay with the simple TDE as it is. Despite its simplicity, TDE does some reasonable things: It does not increase the capacity depot if there is no need, which is correct since for time-invariant setup costs, as we assume here, there is no reason to increase a depot without absolute necessity. Furthermore, it gives downward steps a higher probability if the depot is comparably high to the average level of the CDC. This is intuitively the right thing to do since for high levels of the depot there are higher chances of wasting capacity and consequently incurring higher costs. Of course, the value of parameter α is crucial for the success of TDE. If α is set too high, then TDE is too "nervous", and will produce too many changes in the level of the depot and if it is set too low, TDE is too "lazy", and will waste a lot of capacity.

Again, simulations are used to evaluate the potential of TDE for decoupling of QoS systems that operate on different time scales. Yet, this time an attempt is made to use more realistic and significantly longer CDCs. The CDCs are produced by simulating individual requests with poisson arrival and exponential holding times. They are thus based on markovian models as known from teletraffic theory. The capacity demand for each individual request is still drawn from random distributions as for this quantity there are no known statistical models. In order to assess the covers generated sequentially by TDE, we also apply NEAROPT-4 to the CDCs in an off-line manner once these are known (to use OPT as a reference value is computationally infeasible for the large CDCs we used). The cover produced by NEAROPT-4 is then used to get an approximate ACS for TDE's covers that is, of course, a little bit higher than the correct ACS value. The aggregated results of 100 simulations are shown in Table 4.

Here μ_α^{ACS} and σ_α^{ACS} denote again the sample means and standard deviations from the simulations for different values of α. $n^{TDE,\alpha}$ *and* n^{NOPT4} are the average number of steps produced by TDE (with parameter α) respectively NEAROPT-4. For each simulation 5000 individual requests were generated with poisson arrival ($\lambda = 6$) and capacity demands drawn randomly from the uniform distribution over [1,30].* For the lifetime of a request, we simulated three different scenarios with short-, medium and long-lived flows by drawing from an exponential distribution with parameter μ = 40, 100 and 400. In order to model very different time scales for the two QoS systems, we set F = 2000 and U = 1 for all simulations (note that under these settings the absolute cost saving was extremely high (one order of magnitude on average)).

Table 4: ACS(TDE) for requests with different lifetimes.

	short-lived requests	medium-lived request	long-lived requests
$\mu_{0.1}^{ACS}$	*0.84*	*0.65*	*0.36*
$\sigma_{0.1}^{ACS}$	*0.002*	*0.003*	*0.005*
$n^{TDE,0.1}/n^{NOPT4}$	*1.8*	*0.2*	*0.2*
$\mu_{0.5}^{ACS}$	*0.71*	*0.84*	*0.51*
$\sigma_{0.5}^{ACS}$	*0.004*	*0.001*	*0.004*
$n^{TDE,0.5}/n^{NOPT4}$	*9.4*	*2.3*	*0.8*
$\mu_{0.9}^{ACS}$	*0.19*	*0.42*	*0.81*
$\sigma_{0.9}^{ACS}$	*0.003*	*0.008*	*0.002*
$n^{TDE,0.9}/n^{NOPT4}$	*27.8*	*16.8*	*2.2*

It can be seen from the results that for different lifetimes of requests TDE performs best with different values of α. For short-lived flows, it is good to set α rather low, for medium-lived requests it is good to set it at an intermediate level and for long-lived requests, it is best chosen very high. Furthermore, it can be perceived that wrong values for α can have a devastating effect for the performance of TDE. For example, in the short-lived requests case setting α = 0.9 only yields about 20% of the cost saving potential. So, TDE cannot be considered to deliver a robust behavior if the lifetimes of requests vary. It is interesting to note that for all kinds of flows the value of α that yields the best results for TDE exhibits for the ratio of steps for its cover and for the cover of NEAROPT-4 values fairly close to 2 while the worse α produce ratios far apart from this. In particular, a ratio close to one as is the case for TDE with α = 0.5 and long-lived requests did not achieve a good result. The likely reason for this is that TDE cannot cope with the same number of steps as NEAROPT-4 to produce a good cover.

The overall result from these discussions is, not surprisingly, that TDE alone cannot guarantee to deliver good covers for decoupling QoS systems with different time scales but an integration in the ODAH scheme to self-control the setting of α instead of setting it manually to some arbitrary value may be a promising direction.

6 TDE IN THE ODAH SCHEME

In this section, the integration of TDE into the ODAH scheme is described and the resulting heuristic, called ODAH-TDE, is evaluated again by simulations. This integration is motivated by the previous discussions on TDE's sensitivity to the parameter α.

6.1 Embedding TDE in ODAH

As discussed in Section 3, there are two modes of adaptation in the ODAH scheme: adaptation in performance space and in action space. In principle, both kinds of adaptation are possible for ODAH-TDE. In both cases, we use NEAROPT-K instead of OPT if the time window of adaptation is too large for OPT to compute the minimal cover in a reasonable time (which is the case in most circumstances).

The adaptation in performance space works by simply adjusting TDE's parameter α such that

$$c(R^{TDE,\alpha}) - c(R^{opt}) \qquad (13)$$

is minimized. This minimization is done by a simple recursive grid search [11] through the interval [0,1] for parameter α as there is no

*. These parameter settings were taken arbitrarily due to a lack of empirical data, however, simulations indicated that the results are not very sensitive to these parameters.

simple relationship between α and c for a more intelligent search to exploit.

For the adaptation in action space, it was decided to use the number of steps as basis for the similarity relation between covers, so that in this case

$$\left|n^{TDE,\alpha} - RF \times n^{OPT}\right| \quad (14)$$

is to be minimized. *RF* is a relaxation factor that compensates for the observation that TDE cannot produce good covers with the same number of steps as OPT respectively NEAROPT-K. In the simulations below, we always set $RF = 2$ since experiments showed good results for that value (see also Section 5 for a discussion of this). For the minimization in this case we can use an interpolation search [11] since α and n have a simple relationship: $n^{TDE,\alpha}$ is monotonically increasing in α. This is, of course, much more efficient than the recursive grid search for the adaptation in performance space mode.

The adaptation parameters for both modes are more or less the same, so we discuss them together:

Frequency of Adaptation. This parameter determines partially how expensive the technique is in terms of computational effort because the computation of the optimal or even the near-optimal cover is certainly much more compute-intensive than the simple TDE algorithm on its own. So, if the frequency of adaptation is very high, e.g., every time period, ODAH-TDE can become a very expensive technique while little new data collected about the CDC may not change the adaptation process significantly and hence not justify the effort. On the other hand, if the adaptation frequency is too low, then ODAH-TDE may be too slow to react on changes in the CDC. Hence, a good trade-off between computational effort and responsiveness to changes is the target here.

Time Window of Adaptation. As well as the frequency of adaptation, this parameter is jointly responsible for the computational effort invested in the adaptation in ODAH as it controls how expensive it is to compute the "optimal" cover for a certain past period. Moreover, it controls how much past behavior is taken into account for the adaptation process. The larger the time window of adaptation the more past information is included. However, including too much "old" behavior is not necessarily helpful because recent behavior might be more relevant for the decision on future behavior. On the other hand, if not enough past behavior is captured, some important information from the past may be lost. For efficiency reasons of the ODAH-TDE algorithm it is, of course, beneficial to use smaller windows.

Accuracy of Adaptation. This parameter deals with the exactness of each adaptation step, i.e., how thoroughly the parameter space for α is searched during the minimization problems solved at each adaptation step. Extreme accuracy should not be required since a "perfect" fitting to past behavior does not necessarily yield better results since ODAH-TDE is still only a heuristic (in particular as it is based on NEAROPT-K for larger *K*). Furthermore, less accuracy certainly improves the efficiency of ODAH-TDE.

For ODAH-TDE, we use the number of steps of the CDC to be covered as units for the frequency as well as for the time window of adaptation. This means these parameters are not specified in absolute time but adapt themselves to the rate of changes of the offered CDC, i.e., adaptation takes place often in times of many changes and less often in more quiet periods. That is a desirable behavior from our point of view.

The accuracy of adaptation in ODAH-TDE is measured by the granularity of the parameter space for α at which the minimization procedures terminate to search any further (in case they do not succeed before). In all of the simulations of ODAH-TDE that are discussed in the next subsection, this accuracy was set to 10^{-4}.

6.2 Simulations for ODAH-TDE

Using again the simulation environment for QoS systems with different time scales, this subsection evaluates ODAH-TDE's performance for the on-line sequential determination of a cover for a CDC. The same kind of CDCs generated from different types of requests as in Section 5 is used in order to allow for a comparison of ODAH-TDE with the values for plain TDE given in Table 4. Again, NEAROPT-4 is applied to the off-line problem under certainty about the generated CDC in order to be able to compute the approximate ACS metric for ODAH-TDE.

Although, we have experimented with both adaptation modes we concentrate on adaptation in action space for the simulations here, since both modes performed very similar and, as we argued in the preceding section, adaptation in action space is more efficient due to the less compute-intensive adaptation step.

The simulations are targeted at evaluating different adaptation parameters for ODAH-TDE, in particular different time windows and adaptation frequencies. To limit the possible number of alternatives for the adaptation parameters, it has been decided to investigate ODAH-TDE algorithms for cases where the time window of adaptation equals the reciprocal value of the frequency of adaptation. In these cases, all past information about the CDC is used exactly once for an *adaptation epoch* as we call it. So, at the end of an adaptation epoch, the adaptation step is carried out using only the data collected about the CDC within this epoch. In the simulations ODAH-TDE works with adaptation epochs of 20, 100, 200, 500, and 1000. For an adaptation epoch of 20 ODAH-TDE uses OPT to compute the cost-optimal cover against which the adaptation is performed whereas for the larger epochs NEAROPT-4 is applied since OPT is computationally infeasible for these. In all cases ODAH-TDE starts with $\alpha = 0.5$, and adapts itself in the course of time.

As in all preceding experiments, 100 simulations each for the different adaptation epochs and requests with different lifetimes have been performed, the results of which are given in Table 5.

Table 5: ACS(ODAH-TDE) for requests with different lifetimes.

	short-lived requests	medium-lived request	long-lived requests
μ_{20}^{ACS}	*0.83*	*0.84*	*0.85*
σ_{20}^{ACS}	*0.003*	*0.004*	*0.003*
μ_{100}^{ACS}	*0.93*	*0.92*	*0.91*
σ_{100}^{ACS}	*0.002*	*0.001*	*0.001*
μ_{200}^{ACS}	*0.93*	*0.92*	*0.91*
σ_{200}^{ACS}	*0.001*	*0.001*	*0.002*
μ_{500}^{ACS}	*0.92*	*0.92*	*0.90*
σ_{500}^{ACS}	*0.004*	*0.003*	*0.001*
μ_{1000}^{ACS}	*0.91*	*0.91*	*0.88*
σ_{1000}^{ACS}	*0.002*	*0.002*	*0.001*

Here, μ_{AE}^{ACS} and σ_{AE}^{ACS} denote the sample mean and standard deviation of an approximate ACS (based on NEAROPT-4) in the simulations for different adaptation epochs $AE \in \{20, 100, 200, 500, 1000\}$.

As the results indicate, ODAH-TDE generally achieves a good and robust performance over all types of requests especially for medium-size adaptation epochs. For the smallest adaptation epoch of 20, the performance is considerably worse although it is the only one based on OPT. However, the adaptation epoch apparently is too short so that the adaptation is too sensitive to short-term random effects. This emphasizes the necessity of an approximation technique like NEAROPT-K as a substitute for OPT in ODAH-TDE since OPT is computationally infeasible for suitable adaptation epoch sizes.

The slight deterioration for large adaptation epochs may be explained by the rather slow responsiveness of ODAH-TDE for these. So, if an unfortunate adaptation of α is done, it has a long lasting impact on the performance of ODAH-TDE as the next adaptation step is far away.

Anyway, in conclusion the simulation results give evidence that ODAH represents a robust scheme for heuristically dealing with the sequential decoupling problem under uncertainty about a CDC. In particular, it should work well even if flow characteristics as the lifetime of requests change since it shows good performance for all types of requests in the simulations.

7 Related Work

The problem of different time-scales of network QoS systems has been largely neglected in the literature so far. There is some work that deals with RSVP/IntServ over DiffServ, probably the most important scenario for decoupling to be applied. For instance, in the IETF, there is work within the ISSLL working group that gives a very comprehensive framework for RSVP/IntServ over DiffServ and the issues involved [12]. However, how to deal with different time scales of QoS systems based on the two architectures is not considered. In fact, a scenario is depicted in which both systems are assumed to operate on the same time scale, i.e., each RSVP request results in a query to the BB. From our point of view, this is not desirable as it would destroy the scalability of the DiffServ/BB approach. In [13], the design of an IntServ/DiffServ edge device is described. But again the focus of this work is more on the interworking of mechanisms like mapping of IntServ classes onto DSCPs and so on whereas decoupling is not studied. A last example in that area of work is [14] which gives very detailed treatment of the protocol between an IntServ/DiffServ edge device and a BB based on the COPS protocol. It is, however, not the target of that work to treat strategic decisions of an edge device as represented by decoupling.

One piece of work that explicitly deals with different time scales of access and backbone networks on the control paths is [15]. Here a backbone QoS signalling is proposed which integrates mechanisms in order to dampen the faster time scales of access networks. This mechanism is based on hysteresis and quantization for traffic aggregates which are based on sink trees towards destinations. The applied algorithm is to always reserve capacity in multiples of a certain quantity Q. Whenever the reserved capacity level of $k \times Q$ is no more sufficient, it is increased to $(k+1) \times Q$ and the new quantum is only relinquished when the reserved capacity falls below $(k-1) \times Q$. This is very comparable to the simple strategy of the TDE algorithm, and uses no adaptation. Moreover, we think that the integration of such a mechanism into a signalling protocol represents an unfortunate mixing of strategy and mechanisms since decisions on decoupling of time scales should be subject to the strategy of an edge device irrespective of the utilized signalling protocol.

Interestingly, the decoupling problem may also be applied to the situation where a QoS system supporting dynamic QoS is mapped onto a system that only allows for static QoS as is the case when mapping an RSVP/IntServ- onto an ATM-based QoS system. A further instance of the decoupling problem is the computation of renegotiation schedules for a non-stationary variable rate source which uses a renegotiated service class as, e.g., described in [16]. The algorithms presented in [16] are pretty similar to the ones derived in this paper, especially for the case where the source's rate process is known beforehand. However, the algorithm proposed for interactive sources, which is comparable to the covering under uncertainty about the CDC, is not based on adaptation directed via the optimum cover calculation.

8 Conclusions

This paper has dealt with a largely neglected problem when interworking heterogeneous QoS systems - the accommodation of different time scales for QoS systems by decoupling. The decoupling problem has been formalized in order to analyze its complexity and derive solution approaches. These approaches are based on the ODAH adaptation framework which we devised for that purpose. The ODAH framework makes use of past knowledge about capacity demands by adapting parametrized heuristics with the aid of optimal techniques which, however, require perfect knowledge about CDCs. Throughout this paper, we have used simulations to verify the performance of our solution approaches to the decoupling problem. In particular, it has been demonstrated that a very simple heuristic like TDE could be integrated into the ODAH scheme resulting in a very robust and still computationally feasible solution to the decoupling problem at an edge device between a fast and a slow time scale QoS system. While the heuristics developed in this paper may be enhanced by introducing more empirical data into the heuristics (once this data is available), we believe that an adaptive scheme as presented here (based on (near-) optimal decisions for the past) may continue to play an important role for the decoupling problem.

References

[1] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. Informational RFC 1633, June 1994.

[2] D. Black, S. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. Informational RFC 2475, December 1998.

[3] U. Black. ATM: Foundation for Broadband Networks, 1995. Prentice Hall, Englewood Cliffs.

[4] J. Schmitt, M. Karsten, and R. Steinmetz. On the Aggregation of Deterministic Service Flows. *Computer Communications*, 24(1):2–18, January 2001.

[5] J. A. Cobb. Preserving Quality of Service Guarantees in spite of Flow Aggregation. In *Proceedings of the 6th International Conference on Network Protocols (ICNP'98)*, pages 90–97. IEEE, November 1998.

[6] A. Terzis, J. Krawczyk, J. Wroczlawski, and L. Zhang. RSVP Operation over IP Tunnels. Proposed Standard RFC 2746, January 2000.

[7] O. Heckmann and J. Schmitt. Multi-Period Resource Allocation at System Edges. Technical Report TR-KOM-2000-05, University of Technology Darmstadt, October 2000. ftp://ftp.kom.e-technik.tu-darmstadt.de/pub/TR/TR-KOM-2000-05.pdf.

[8] L. C. Wolf and R. Steinmetz. Concepts for Resource Reservation in Advance. *Multimedia Tools and Applications*, 4(3):255–278, May 1997. Special Issue on State of the Art in Multimedia Computing.

[9] M. Karsten, N. Berier, L. C. Wolf, and R. Steinmetz. A Policy-Based Service Specification for Resource Reservation in Advance. In *Proceedings of the International Conference on Computer Communications (ICCC'99), Tokyo, Japan*, pages 82–88, September 1999. ISBN 1-891365-05-3.

[10] L. Kleinrock. *Queueing Systems – Theory*. Wiley-Interscience, New York, vol.1, 1975.

[11] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press Inc., New York, USA, 1981. ISBN 0-12-283952-8.

[12] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine. A Framework for Integrated Services Operation over DiffServ Networks. Informational RFC 2998, November 2000.

[13] G. Eichler, H. Hussmann, G. Mamais, C. Prehofer, and S. Salsano. Implementing Integrated Services and Differentiated Services for the Internet with ATM Networks: A Practical Approach. *IEEE Communications Magazine*, 38(1):132–141, January 2000.

[14] S. Salsano. COPS Usage for Outsourcing Diffserv Resource Allocation. Internet Draft, February 2000. Work in progress.

[15] P. Pan, E. Hahne, and H. Schulzrinne. BGRP: A Tree-Based Aggregation Protocol for Inter-domain Reservations. *Journal of Communications and Networks*, 2(2):157–167, June 2000.

[16] M. Grossglauser and S. Keshav. RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic. *IEEE/ACM Transactions on Networking*, 5(6):741–755, December 1997.