MŰEGYETEM 1782

REDUCED-STATE RESOURCE RESERVATION

Collection of Ph.D. Theses
by

András Császár

Research Supervisors:

Róbert Szabó, Ph.D.   Tamás Henk, Ph.D.

High Speed Networks Laboratory
Department of Telecommunications and Media-Informatics
Budapest University of Technology and Economics

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
BUDAPEST, HUNGARY
NOVEMBER 2007

1 Introduction

In recent years, the volume of streaming media transported over IP networks has increased noticeably [1]. This is not only due to emerging bandwidth intensive video applications but also a consequence of old services appearing over IP networks, like telephony (e.g., via Skype). Moreover, traditional telecommunication services, like circuit-switched voice, converge to be carried over a common IP transport network. In a carrier-grade network, such applications require stringent bandwidth guarantees to fulfil their throughput and delay requirements.

In my dissertation, I focus on on-path resource reservation protocols as means to assure bandwidth for bandwidth sensitive applications. On-path resource reservation, where per-flow admission control is performed hop-by-hop, is an alternative to over-provisioning [1, 2, 3, 4] and to off-path solutions, e.g., Bandwidth Brokers [5, 6]. On-path resource reservation better suits the distributed nature of IP networks (unlike off-path solutions) and it also works during busy hours or special occasions of very high traffic load or in scarce-resource radio networks (unlike over-provisioning). Moreover, as high quality video on demand services are getting deployed, bandwidth assurance may become an issue more frequently even in backbone networks.

Due to the scalability limitations of the standard Resource Reservation Protocol (RSVP) [7], in the Internet Engineering Task Force (IETF) the Next-Steps in Signaling (NSIS) working group [8] is standardising the next-generation on-path resource reservation protocols. The working group considers two variants, a *stateful* and a *reduced-state* [9] quality of service (QoS) model. These terms reflect the implementation complexity of the protocol in interior nodes. The stateful solution stores per-flow state information in all network nodes suiting the Integrated Services/RSVP QoS model. On the other hand, in the newer Differentiated Services (DiffServ) architecture [10], which defines the concept of network domain edge router and interior router, only edge nodes are permitted to dispose of per-flow states. Interior nodes should be as simple as possible and should, therefore, work on aggregates. Therefore, the reduced-state solution relies on per-traffic-class aggregation in interior nodes using the DiffServ/RMD model, where RMD stands for Resource Management in Diffserv [O1, C6].

In my dissertation I show that under normal circumstances in a high-speed network with many flows the reduced-state mode is preferable over stateful operation. The reason is that a reduced-state protocol is simpler, poses less processing capacity overhead on core routers and so it is more scalable even in high-speed networks, while the achievable performance in most of the time is similar.

However, besides normal circumstances, reservation protocols should also handle the exceptional situations of node or link failures. Markopoulou et al. [11] showed that link failures are part of everyday operation in a big IP network. They found that on the average link failures occur in every 30 minutes in the Sprint backbone. Failures are due to a number of causes, like optical fibre cut and other environmental effects, router hardware/software failures, or operator errors. Watson et al. also observed in [12] that route changes are frequent in operational networks.

The routing protocols of IP are capable of re-directing flows from their original paths to alternative paths after failures. Although routing protocols are an integral part of a robust IP network, they were invented when IP networks did not offer QoS. In case of re-routing, however, QoS solutions face the problem that there may not be enough bandwidth to accommodate all re-routed flows. That is, congestion may occur albeit the previous positive admission decision. In fact, it has been shown by Iyer et al. [13] that most occurrences of link overloads in the Internet are caused by re-routing after link failures despite of the heavily over-provisioned networks. Thus quick and efficient re-route handling might be the most important aspect of a bandwidth reservation protocol. Therefore, a major part of my dissertation focuses on re-route handling with RMD, which did not have the necessary functionality at the time of my research effort.

2 Research Goals

Throughout my research I have performed a thorough evaluation of reduced-state reservation protocols, and to apply the results to RMD to improve its features. During my work, an important goal was to contribute to the standardisation of RMD with my results and solutions. Although reduced-state protocols have clear advantages, simplicity and scalability, over stateful protocols, there could be drawbacks or problems to be solved.

I investigated those aspects of the protocol which were not part of its original design. I was curious how the fact that the protocol can rely only on aggregated instead of per flow information constrains the applicability of admission control algorithms (Thesis Group 1). I also wished to know if there are disadvantages or other advantages in the performance compared to the yet only standardised protocol, the stateful RSVP (Thesis Group 2).

While I have found that during normal operation RMD is at least as good as its stateful competitor RSVP, or in some cases even better, I have demonstrated the negative consequences of re-routing and I have found that per-flow states are really useful for a fast and precise resolution of such situation and that the existing mechanisms of RMD are not able to properly cope with re-routing (Thesis Group 3). Therefore, my next objectives were to increase the robustness of RMD by giving solutions to the problems occurring after re-routing (Thesis Group 4).

3 Methodology

During the analysis of the reduced-state constraint on admission control I relied on mathematical formalisation. The primary tools of my evaluations have been, however, protocol analysis and packet level simulations in the network simulator ns-2 [14]. For these simulations I have used realistic voice and video traffic models. Investigated networking scenarios run from the simplest delta network topology, which is the smallest network to show re-routing, to more complex European and US backbone networks and some special radio access network topologies too. For the investigations of the congestion handling time-line I have created a worst case model. For the performance evaluations in Thesis Group 2 besides protocol analysis I also relied on architectural observations and deductions.

4 New Contributions

4.1 Reduced-State Admission Control

In a reduced-state protocol interior nodes may only store aggregated information, e.g., in a per class manner. I have, therefore, investigated how much this property narrows the set of feasible admission control formulas in interior nodes. Finding formulas that are reduced-state compatible is important as these formulas could be implemented in reduced-state protocols, like RMD, without requiring approximation.

THESIS GROUP 1 – Reduced-State Admission Control

> *After defining reduced-state admission control formulas, I have shown that such formulas build a* real *subset of stateful admission control formulas, and I have given conditions when an admission control formula is reduced-state compatible. I have concluded that even though not all but a lot of formulas can be implemented in reduced-state mode. I have proposed a generalisation of the existing soft-state mechanism of RMD to suit the formulas covered by the sufficient condition.*

In general, when a new flow request is received by an interior node, an admission control algorithm evaluates a formula. If the formula evaluates to true, the new flow is admitted and the information about the admitted flow is merged into the database. The scope of admission control formulas in my investigations is limited to cases, where the formula is based on one or more of the following information:

- The traffic descriptors of the new flow (like peak rate, average rate, activity factor)
- Stored data about the existing flows (which have been admitted previously)
- Measured aggregated data (like measured utilisation of the link, actual queue size)
- Pre-configured parameters (like link capacity, maximum utilisation threshold of the link, and overflow probability)

Definition 1.1 (Admission Control Formula)**.** If a system has n already admitted flows and must decide about the admission of a new flow, then an admission control formula is defined as

$$F_n(\bar{\bar{\mathbf{P}}}_n,\, \bar{p}_{\text{new}},\, \bar{M},\, \bar{C}) \leq 0$$

where

$$\bar{\bar{\mathbf{P}}}_n \triangleq \begin{pmatrix} p_1^1 & p_1^2 & \dots & p_1^k \\ \vdots & \vdots & \vdots & \vdots \\ p_n^1 & p_n^2 & \dots & p_n^k \end{pmatrix} \;,\;\; \bar{p}_{\text{new}} \triangleq \{p_{\text{new}}^1, p_{\text{new}}^2, \dots, p_{\text{new}}^k\} \;,$$

$$\bar{M} \triangleq \{M^1, M^2, \dots, M^m\} \;\; and \;\; \bar{C} \triangleq \{C^1, C^2, \dots, C^l\} \;.$$

$p_i^j \in \mathbb{R}$ denotes the j^{th} $(j = 1..k)$ traffic descriptor for flow i, i.e. a flow is characterised by k descriptors. The formula contains $m \in \mathbb{N}$ measured values (M^i, $i = 1..m$) and $l \in \mathbb{N}$ pieces of pre-configured parameters (C^i, $i = 1..l$) like thresholds, limits, etc.

In the following I introduce a shorthand form for the k pieces of traffic descriptors of a flow by introducing k long vectors:

$$\bar{p}_i \triangleq \{p_i^1, p_i^2, \ldots, p_i^k\}$$

A reduced-state resource reservation protocol does not allow storing per-flow information in interior nodes. Therefore, the admission control algorithm itself must also have a *reduced state* implementation. Algorithms with only a stateful implementation are not considered scalable for high-speed networks, where an interior router needs to manage many thousands or even hundreds of thousands of flows.

Remark 1.1. The advantage of reduced-state protocols over stateful protocols in terms of memory size or processing capacity can be a question to discuss. Stateful formulas require a number of variables that is a monotony increasing function of the number of flows. In reduced-state implementations the amount of registers is independent of the amount of previously admitted flows. The number of entries in the memory is often correlated to the processing time or capacity required to lookup, add or delete an entry. That is, storing more states generally not only requires bigger memory but a faster processor. An exception is when the memory entries are stored in a hardware associative memory, in which case the processing time is constant and independent of the number of entries. However, in this case the memory size is even more vital since such content-addressable memories are expensive. Another exception would be if the memory was so big that it contains a register for each possible entry. In this case, instead of lookup the memory cell could be directly addressed. In the case of admission control flows are typically identified by a five-tuple of the source/destination IP address, the source/destination port and the protocol identifier. In IPv4 this would require a memory holding in worst case as many as $2^{(2\cdot 32+2\cdot 16+8)}$ registers, which is infeasible.

Definition 1.2 (Reduced-State Compatible Admission Control)**.** An admission control formula is reduced-state compatible requiring a constant amount of $d \in \mathbb{N}$ variables, if for any n already admitted flows there exists a $S_n : \mathbb{R}^{n\times k} \to \mathbb{R}^d$ transformation function such that a

1. $G : \mathbb{R}^d \times \mathbb{R}^k \times \mathbb{R}^m \times \mathbb{R}^l \to \mathbb{R}$ transformed admission control function can be found so that $G(S_n(\bar{\bar{\mathbf{P}}}_n), \bar{p}_{\text{new}}, \bar{M}, \bar{C}) = F_n(\bar{\bar{\mathbf{P}}}_n, \bar{p}_{\text{new}}, \bar{M}, \bar{C})$, and a

2. $\Phi : \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}^d$ admission update function can be found so that $\Phi(S_n(\bar{\bar{\mathbf{P}}}_n), \bar{p}_{\text{new}}) = S_{n+1}(\bar{\bar{\mathbf{P}}}_{n+1})$, and a

3. $\Psi : \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}^d$ release update function can be found so that $\Psi(S_{n+1}(\bar{\bar{\mathbf{P}}}_{n+1}), \bar{p}_j) = S_n(\bar{p}_1, ..., \bar{p}_{i\neq j}, \ldots, \bar{p}_{n+1}) \quad \forall\, j = 1..(n+1)$.

The $S_n(\cdot)$ functions ensure that it is possible to compress the information about any number of flows to a constant number of variables. That is, the information from a variable length $(n \times k)$ matrix can be reduced to a d length vector (state space).

Condition 1 of Def. 1.2 means that there exists a *single* function that can make the exact same admission decisions based on a constant size vector holding aggregated information as the different original F_n functions would make based on a variable size matrix holding per-flow information.

Condition 2 ensures that if a flow is admitted, the system is able to incorporate the data about the new flow into the finite number of variables. While in a stateful protocol, all the flow parameters would be simply added to the database, in a reduced-state protocol the flow parameters are merged into aggregated states using *the same* function in each step.

Condition 3 ensures that when a flow leaves the system, using *the same* function in each step it is possible to derive the state that would have been calculated if the flow had never been there.

THESIS 1.1 – Real Subset [D]

> *I have proven that the set of reduced-state compatible admission control formulas (as defined in Def. 1.2) is a* real *subset of all admission control formulas.*

Example 1.1. The next formula is not reduced-state compatible:

$$n \cdot \max(p_1, \ldots, p_n) \leq C \ .$$

Although I could not give a condition for reduced-state compatibility that is sufficient and necessary at the same time, I have found conditions that are either necessary or sufficient.

THESIS 1.2 – Necessary condition: arrival order independence [D]

> *I have shown that an admission control algorithm may be reduced-state compatible* only if *it is not dependent on the arrival order of the flows.*

The above condition is "only" necessary, but not sufficient. For instance, the admission control formula of Example 1.1 is also independent of the arrival order.

By looking at Condition 2 of Def. 1.2 it can be seen that this condition resembles recursion.

Definition 1.3 (Function Series)**.** For my work I define a function series as a series of functions, in which the number of arguments depends on the running index, i.e., $F_n(\bar{p}_1, ..., \bar{p}_n)$.

Definition 1.4 (Recursive Function Series)**.** A function series $R_n(\bar{p}_1, ..., \bar{p}_n)$ is recursive if for any n and $\bar{p}_n$ there exists a single f function so that

$$R_n(\bar{p}_1, ..., \bar{p}_n) = f\big(R_{n-1}(\bar{p}_1, ..., \bar{p}_{n-1}),\, \bar{p}_n\big) \ .$$

THESIS 1.3 – Recursion as the basis for reduced-state compatibility [D]

> *For a theoretical admission control module, which has to serve only flow arrivals (i.e., flows never leave the system), I have shown that if the admission control formula*

can be expressed as a function of a fixed *($d \in \mathbb{N}$) number of independent recursive functions ($R_n()$) as follows:*

$$F_n(\bar{p}_1, ..., \bar{p}_n, \bar{p}_{new}) = G\big(R^1_{n+1}(\bar{p}_1, ..., \bar{p}_n, \bar{p}_{new}),\ \ldots\ ;\ R^d_{n+1}(\bar{p}_1, ..., \bar{p}_n, \bar{p}_{new})\big)\ ,$$

then the given admission control can be implemented with constant amount of variables $O(1)$, meaning that conditions (1) and (2) of Definition 1.2 hold.

After the arrival of a flow the value of a recursive function series is calculated from the previous value of the function series and the new flow parameters. Nothing guarantees, however, that leaving flows can be removed from the aggregated states since it is not certain that there exists a reverse function which can calculate the previous value of the function (as in Condition 3 of Definition 1.2).

Finding such reverse functions is an interesting task. The following thesis, which is the basis of an international patent application [P9], gives a sufficient condition for reduced-state compatibility. The reverse function exists because of the commutative, associative and invertible property of the “add” and “multiply” operators in the sum and product constructs.

As protocols are considered to be more robust if reservations cannot be stuck in the network forever, most protocols rely on periodical refreshments of the reservations. The stored aggregated states have to timeout in the sense that the absence of the explicit refresh of the parameters of a flow must be equivalent to the explicit release of the flow. This is the so called “soft-state” approach. As RMD does not have per-flow states, running a timer for each flow, like with RSVP, is not an option. Instead, the aggregated states are periodically re-built. Previously, RMD was expected to run only the *sum of rates* (see [D]/Sec. 3.2.1) admission control method, for which Karagiannis et al. gave a soft-state implementation in [15], where flows have to refresh their reservations in periodical R intervals. I have been able to generalise that idea to suit all formulas covered by the template in the following thesis:

THESIS 1.4 – Sum and Product Constructs and their Soft-State Implementation [P9, D]

I have shown that if an admission control formula is constructed suiting the following template of sum and product constructs, then it is reduced-state compatible:

Template

$$G\left(\left\{\Big(\sum \mid \prod\Big)_{i=1}^{n} g_1(\bar{p}_i),\ \ldots\ ,\ \Big(\sum \mid \prod\Big)_{i=1}^{n} g_d(\bar{p}_i)\right\},\ \bar{p}_{\texttt{new}},\ \bar{M},\ \bar{C}\right) \leq 0\ ,$$

where $d \in \mathbb{N}$, $d < \infty$ and $g_j : \mathbb{R}^k \to \mathbb{R}\ \ j = 1..d$.

Note that

- *The template contains a constant amount (d) of sum or product constructs, denoted by $(\sum \mid \prod)$.*

- *Inside the sum or product constructs, each term can be a function ($g_i(\cdot)$) but only the function of the descriptors of a single flow.*
- *The outer function (G) does not depend on the flow descriptors directly, but only on the sum and product constructs.*

I have also created an algorithm (Alg. 1.1) to implement formulas suiting this template in a soft-state *manner.*

Algorithm 1.1. Let us suppose that the algorithm has d sum or product constructs from which the first s are sum constructs and the rest are product constructs, i.e.,

$$G\left(\left\{\sum_{i=1}^{n} g_1(\bar{p}_i),\ \ldots,\sum_{i=1}^{n} g_s(\bar{p}_i),\ \prod_{i=1}^{n} g_{s+1}(\bar{p}_i),\ \ldots,\prod_{i=1}^{n} g_d(\bar{p}_i)\right\},\ \bar{p}_{\texttt{new}},\ \bar{M},\ \bar{C}\right) \le 0\,.$$

Let us assume that $last_i$ ($i = 1..d$) represents the current value of the sum or product constructs. Therefore, the admission condition at a new reservation request is the following:

$G\left(last_1, \ldots, last_d;\ \bar{p}_{\texttt{new}},\ \bar{M},\ \bar{C}\right) \le 0$

In each router, the R long time windows are divided to N time intervals, called cells, and the system individually count and remembers –besides the active cell– N past cells using counters C_i^j ($j = 1..N$, $i = 1..d$). The active cell's refreshes and reservations are collected in $count_i$ ($i = 1..d$).

If a reservation request is admitted, then:

```
for $i = 1..s$ : { $last_i := last_i + g_i(\bar{p}_{\texttt{new}})$ ;  $count_i := count_i + g_i(\bar{p}_{\texttt{new}})$ ; }
for $i = (s+1)..d$ : { $last_i := last_i \cdot g_i(\bar{p}_{\texttt{new}})$ ;  $count_i := count_i \cdot g_i(\bar{p}_{\texttt{new}})$ }
```

When a refresh message arrives:

```
for $i = 1..s$: $count_i := count_i + g_i(\bar{p}_{\texttt{new}})$
for $i = s + 1..d$: $count_i := count_i \cdot g_i(\bar{p}_{\texttt{new}})$
```

After each R/N long cell the states are updated by sliding the window with one cell:

```
for $i = 1..d$ : {
  for $j = N..2$ : $C_i^j := C_i^{j-1}$
  $C_i^1 := count_i$
  if $i \le s$ { $last_i := \sum_{j=1}^{N} C_i^j$ ; $count_i := 0$ }
  else { $last_i := \prod_{j=1}^{N} C_i^j$ ; $count_i := 1$ }
}
```

Corollary 1.1 (Reduced state feasibility)**.**

By showing example admission control formulas (as shown in [D]/Sec. 3.2), which fit the template of Thesis 1.4, like the very simple sum of rates method, the Hoeffding-Bounds *[16] method, the* Heavy Traffic Approximation *[17] algorithm or the* Tangent at Peak *[18] admission control algorithm, I can state that many existing admission control solutions are feasible in a reduced-state protocol, and furthermore, they can also be used in combination with a soft-state protocol.*

4.2 Performance of RMD

In the previous section I showed that in a lot of cases it is possible to use the same local admission control algorithms in RMD with aggregated states as in RSVP with per-flow states. I have also investigated the protocol related performance of RMD compared to RSVP when the admission control algorithm is identical. In this case, the protocol mechanisms can be responsible for possible differences of performance metrics like notification time; signalling bandwidth overhead; or reservation blocking ratio. The original design of RMD and its ancestor protocol (Load Control [19]) did not focus on these performance metrics, although the evaluation was important to support standardisation.

THESIS GROUP 2 – Performance of RMD

While the reduced memory requirement and processing overhead (i.e., scalability) are obvious advantages of reduced-state protocols, I have shown that notification times and signalling overhead of RMD are at least as good as those of the stateful standard protocol RSVP. Despite some differences in the reservation transient, I have shown that when applying the same admission control algorithm, the blocking ratio and the achievable utilisation of the two protocols are similar. Though, if neither protocols applied explicit release, RMD would have an advantage.

By investigating these aspects I have concluded that in fault-free conditions the simpler RMD protocol provides similar or in some cases better performance than RSVP.

The first version of RMD –or Load Control as it was called at that time– did not have means for explicit release of the reservations, it was just relying on soft-state refreshments. If explicit release signals are not used (or simply lost), then longer soft-state timeout could lead to inefficient bandwidth usage due to the superfluous reservation of flows which already left the network. Another cause of superfluous reservation, and so unutilised capacity, could be when a flow request is admitted on the first part of its path but refused later. In such a situation, some time will elapse until the reservations on the first part will be released.

My first thesis deals with superfluous reservations and notification times:

THESIS 2.1 – Excess reservations and notification times [C8, O6, D]

I have proven that if $E(H)$ denotes the expected session holding time and R denotes the refresh interval, then the superfluous reservation of soft states with RMD is less than with RSVP. More precisely, superfluous reservation is $\frac{0.5(R+\frac{R}{N})}{E(H)}$ with RMD and $\frac{0.75R}{E(H)}$ with RSVP, as an average.

I have shown that RMD keeps superfluous reservations of refused sessions for a bit longer *period (1 RTT) than RSVP (avg. $\frac{1}{2}$ RTT, max. 1 RTT).*

I have also shown that in case of uni-directional traffic streams RMD notifies the sender about successful reservation establishment in the same time *as RSVP if we neglect the hop-by-hop processing time of signalling messages. If processing time of signalling messages is included, RMD is clearly* faster. *In any case, the receiver* *is notified a round-trip time later with RSVP than with RMD. Moreover, in case of bidirectional reservations RMD notifies both edge nodes about the success or failure of the reservation* faster *than RSVP.*

Example 2.1. Assuming the default $N = 10$ and applying the thesis to call holding times widely used to model speech telephony (e.g., VoIP) calls, we get that with the identical default R=30 seconds refresh intervals and without explicit release signals, RMD has 18.3% average excess reservation resulting from the delayed soft-release of flows while RSVP has 25% if we assume 90 seconds mean call holding time. By assuming 120 seconds as the mean holding time, the excess reservation for RMD decreases to 13.75% and for RSVP to 18.8%.

Remark 2.1. Note that theoretically the timeout and refresh procedure of RSVP could be modified to have a similar superfluous reservation as RMD. However, it is not so specified in the standard documents of RSVP because longer timeouts allow tolerance against random loss of refresh messages.

Keeping superfluous reservations a little longer could lead to inefficient bandwidth usage, hence I have also investigated the network wide admission performance of the protocols.

THESIS 2.2 – Admission Performance [C8, C3, C6, C11, J4]

I have found and confirmed by simulation results that RMD can achieve the same reservation blocking ratio and the same network utilisation *while providing the same bandwidth assurance for flows.*

Protocol overhead can be classified to computational, storage and signalling bandwidth overhead. Computational and storage overheads are strongly correlated through the complexity of searching algorithms[1]. The memory needs are shown in Table 1, where f means the number of flows. In contrast to RMD, RSVP does not scale well in a high speed network with many flows.

	Edge node	**Core node**
RMD	$O(f)$	$O(1)$
RSVP	$O(f)$	$O(f)$

Table 1: Memory requirements

Signalling bandwidth overhead means the bandwidth consumption of signalling messages. As both protocols rely on per-flow signalling as well as on regular refreshments of soft states, signalling overhead was a relevant performance metric to investigate.

THESIS 2.3 – Signalling Bandwidth Overhead [C8, C6, D]

I have found that in case of uni-directional reservation *if links are not overloaded* signalling bandwidth overhead of RSVP is marginally higher than that of RMD. *If network load is high, resulting in refused flows, RSVP has noticeably higher signalling overhead. In case of bidirectional sessions, RMD always produces less signalling overhead than RSVP.*

Corollary 2.1 (RMD vs. RSVP performance)**.**

Even though RMD is much simpler than RSVP, normally, when there are no faults present in the network, it achieves the same or even better performance.

[1] See, however, Remark 1.1 on page 5

4.3 Resource Reservation and Network Failures

IP networks have inherent support for re-routing of flows after network failures due to their dynamic routing. Today, faster and faster re-routing solutions appear to provide highly robust and reliable services over the Internet. A carrier-grade IP network, however, not only has to be robust but must provide QoS guarantees at the same time. Therefore, I have investigated how RMD and RSVP behaves and should behave during network failure transients.

The standardised RMD and many times in practice RSVP, too, applies a simple admission control formula, the *sum of rates* formula. If p denotes the peak rate of a flow, a new flow is admitted if

$$\sum_{i=1}^{n} p_i + p_{\texttt{new}} \leq A \quad ,$$

where A is the admission threshold for the link.

Definition 2.1 (Congestion, overload)**.** If the QoS traffic volume on a link is denoted V, congestion occurs if $V > A$.

Although the resource reservation protocol assured congestion free delivery for flows, this assurance may be violated. Today's routing protocols like OSPF, RIP, IS-IS and BGP, however, act independently flows may get re-routed to paths that do not have enough free capacity to accommodate all. Therefore, resource reservation independent re-routing may cause *instant congestion*, although sessions have been assured with congestion-free delivery. This congestion degrades the performance of re-routed flows and may as well degrade the traffic that was originally admitted to the (newly) overloaded link.

Formally, if V_{after} denotes the volume of arriving QoS traffic on a link after re-routing then the volume of *overload* (V_{overload}) after re-routing is $V_{\text{overload}} = V_{\text{after}} - A$.

A re-routing event may not only degrade the performance of ongoing flows, but newly arriving flows may as well develop congestion in spite of using admission control. When the reservation states along the path do not immediately reflect the presence of the re-routed flows and the re-routed flows are forwarded with the same precedence as the original flows but admission control is unaware of them, then *flows newly arriving into the network may be admitted incorrectly above the target threshold.* Ultimately, if the reservation protocol is not prepared for such a situation by some means, admission control may fail to achieve its primary goal to prevent congestion.

THESIS GROUP 3 – Resource Reservation and Network Failures

I have derived a set of properties of the efficient handling of resource reservation independent re-routing events by resource reservation protocols. I have shown that although having per-flow states in these protocols is an advantage when reacting to re-routing, none of the investigated protocols has all the properties.

THESIS 3.1 – Properties of Efficient Handling of Re-Routing [J4, D]

I have given the following list of properties that should be met when handling re-routing with resource reservation protocols in order to minimise degradation and *to enable operator control over the situation, like precedence of ongoing flows over new flows or preference of emergency flows over other flows:*

Property 1: In order to cease congestion, *the V_{overload} volume of flows must be terminated*. The word "terminate" may mean any of the following:

- Do not let any more packets of the flows into the network, e.g., by filtering their packets. This approach practically ends the corresponding flows.
- Re-map the packets of the flows to a lower priority class, e.g., to the best-effort class. The sessions are not ended, although the previously agreed bandwidth is not guaranteed any longer in the original traffic class.

Property 2: In order to minimise the violation of service agreements, *more flows should not be terminated than it is necessary to cease congestion, and the edge nodes of terminated flows should be informed about termination*.

The flows are terminated in order to ensure the required bandwidth for the remaining flows. If the protocol did not bring this sacrifice, all flows would experience congestion for a prolonged time resulting in degraded end-user quality. If congestion lasts too long, the perceived quality may not be tolerable. For example, the American National Standards Institute reports [20] that for voice services recovery times below 200 msec do not likely have impact on the service, and restoration times between 200 msec and 2 sec have only a minimal impact on voice applications. If, however, the degradation is longer than 2 sec, users start to hang-up their calls. Similarly, audio/video streaming application software usually abandon the sessions when experiencing bad quality and cause noticeable interruption for the end user.

Property 3: In order for the total recovery time to remain below 2 sec *excess flows should be terminated as quickly as possible*, preferably within a second after re-routing.

Property 4: In order to make proper admission decisions after re-routing, *admission control operation should be aware of re-routed flows*.

Property 5: In order to be able to preserve re-routed flows even above the regular admission threshold, *it should be possible to define a* ***severe*** *congestion detection threshold $A_{\text{sev}} > A$ that defines a load level below which ongoing flows are not terminated after re-routing*.

Formally, instead of Def. 2.1 *severe* congestion should be concluded if: $V_{\text{after}} > A_{\text{sev}}$, where $A_{\text{sev}} > A$. This way, the amount of overload decreases to $V_{\text{overload}} = V_{\text{after}} - A_{\text{sev}}$.

Example 3.1. Let us suppose that the operator wants to achieve that its voice traffic class, which is normally allocated 40% of the link capacity, should be able to use up to 60% after a possible re-routing even if its other traffic classes are degraded. If C denotes the link capacity, the operator would set $A = 0.4C$ and $A_{\text{sev}} = 0.6C$.

Property 6: For better operator control *policies should be able to select the flows to terminate* during severe congestion handling.

Example 3.2. Emergency voice calls may be preferred over other sessions, or the operator may opt to select high- or low-bandwidth flows to terminate based on pricing aspects.

In Thesis Group 1 I have shown that per-flow states theoretically make a difference in the capability of resource reservation protocols to implement different admission control algorithms. Here I will show that the presence or absence of per-flow states in interior nodes impacts the reactions after re-routing.

Per-flow state protocols, like RSVP, are often bound to per-flow policers, shapers and schedulers. Re-routed flows, however, are forwarded in the best-effort queue without any priority on the new path, because in these routers they did not reserve resources and so the schedulers have not been configured for them. Unfortunately, neither the ingress point nor the source of the flow is informed about this change.

In case of DiffServ networks, where RMD would be used, per-flow policing, shaping and classifying only happens at network ingress nodes. Interior nodes handle traffic classes, so called behaviour aggregates, based on the DiffServ code-point in the packet header. As a result, re-routed flows will be treated according to their original traffic class and may receive the same high priority on the new path without having reserved resources. The re-routed flows might, this way, disturb the original flows in the same priority class.

In both cases, network wide admission control fails as it cannot guarantee congestion-free delivery for flows. With RSVP, this impacts re-routed flows (since these are forwarded in the best-effort queues on the new path), while with RMD this may impact even more flows since re-routed flows may cause congestion on the new path involving other flows, too.

The situation after re-routing will normalise with RMD and RSVP thanks to their soft-state protocol mechanisms. The periodic reinforcement of reservations can be used to correct the amount of reserved resources along the new paths and to terminate any excess allocations. However, regular soft-states are inadequate for carrier-grade services as shown in the following thesis.

By utilising the per-flow state database that can be found in every node, stateful protocols may offer a much faster and more precise solution. In case of RSVP, this enhanced method is called *local repair*. By receiving a trigger signal in the router that changes its forwarding tables, a stateful protocol can initiate the re-reservation procedure immediately, so re-routed flows do not need to wait for the regular refresh messages to be re-admitted onto the new path. Per-flow states make such an action feasible in two ways. First, since the re-routing node has a per-flow database (which stores, e.g., the destination address for all flows) and the node knows which destination addresses have been given a new next-hop, it can select the flows that need to re-reserve their resources. Secondly, each router receiving a `Resv` message knows whether that particular flow has installed a state previously. If not, admission control is performed. However, not even local repair is perfect:

THESIS 3.2 – Solutions based on soft-state refreshes [J4]

> *I have shown that the reaction based on the periodical soft-state refreshes does not fulfil all above conditions. The soft-state refreshes of RSVP do not meet Properties 2, 5 and 6. The refreshes of RMD do not fulfil Properties 4, 5 and 6. With practical refresh periods (10–30 seconds), neither solution meets Property 3.*

The local repair feature of RSVP relying on instant refreshes is quick and fulfils Properties 1–4, however even this solution does not achieve Properties 5 and 6.

Property 5 is not achieved by either protocol. However, I have been able to give a simple solution to preserve ongoing flows after re-routing even above the normal admission threshold.

THESIS 3.3 – Preference of re-routed flows [J4]

In the form of Algorithm 3.1, I gave a solution to fulfil Property 5 with the soft-state refreshes of both RSVP and RMD. That is after exceptional situations, like re-routing, ongoing flows can temporarily occupy more bandwidth, but during normal operation new flows are still not admitted above the original threshold.

Algorithm 3.1.

Step 1 Ensure that reservation messages of new flows and reservation (or rather refresh) messages for already admitted flows can be distinguished inside the network.[2]

Step 2 Besides the regular admission control threshold A, configure in nodes a new A_{sev} *severe congestion threshold*, where $A_{\text{sev}} > A$

Step 3 For refresh messages signalling the refresh of rate p, apply the following condition (instead of the normal admission condition):

$$\text{administered reservation} + p > A_{\text{sev}}$$

If the condition is to true, indicate in the message for the next hops that it was refused.

Step 4 When edge nodes receive marked or refused refresh messages, release them.

4.4 Handling Re-Routing in Reduced-State Protocols

In the previous section I showed that the re-route handling capability of RMD with the built-in mechanisms is inadequate as most conditions are not fulfilled, while RSVP offers faster solution thanks to its per-flow states. Therefore, I researched extensions to RMD to overcome the congestion problem after re-routing. But as I have shown before, instant congestion is not the only problem. Admission control after re-routing could also be incorrect with reduced-state protocols, like RMD, due to the outdated reservation states. Therefore, I have researched methods to correct admission control during the transient until re-routed flows refresh their edge-to-edge reservation states and so re-establish their reservations along their new path.

[2] RMD by default distinguishes the two messages (reservation and refresh). For RSVP I propose either a separate message type or a one bit information field in the Resv message to be used for this purpose.

THESIS GROUP 4 – Handling Re-Routing

I have given a severe congestion handling algorithm, which quickly terminates the precise amount of flows, it protects re-routed flows even above the admission threshold and it enables implementing policies to select which flows to terminate.

I have shown that by overriding the normal admission control operation during the transient after a re-routing event, it is possible to reject flows that would be admitted incorrectly by normal admission control. This is done by readjusting the admission thresholds during these transients either by pure network load measurements or with a smoothing technique or approximating load from the rate of refresh messages.

The combination of these solutions meets all conditions of Thesis 3.1.

An RMD core node does not have per-flow identification, therefore it cannot initiate flow termination. However, RMD edge nodes have per-flow states, hence they can act to terminate flows in congestion. Therefore, congestion along the communication path has to be notified to the edge nodes.

THESIS 4.1 – Severe Congestion Handling [O1, J4, C10, C8, C2, C3]

I have $i)$ extended the RMD mechanisms to signal severe congestion to egress nodes in an in-band fashion and $ii)$ I have created an algorithm (Algorithm 4.1) to resolve severe congestion by terminating only as many flows as correspond to the severe congestion. This way, the excess traffic above the admission threshold is terminated as described in Property 1. Property 5 and 6 is also fulfilled.

Algorithm 4.1 (Severe Congestion Handling – see Fig. 1)**.**

Step 1. Re-routed packets arrive at a core router.

Step 2. The core router detects and estimates the volume of the overload ($\hat{V}_{\text{overload}}$) on the link in S long measurement periods (S is a domain-wide constant).

Step 3. If $\hat{V}_{\text{overload}} > 0$ at the end of a measurement period, the congested core node calculates the number of excess bytes causing congestion as $B_{\text{overload}} = \hat{V}_{\text{overload}} \cdot S$.

Step 4. During the next measurement period, the congested core router marks transmitted user data packets in order $i)$ *to identify* the packets traversing through a congested node, and $ii)$ *to inform* the edge nodes about the *amount* of the estimated overload.

A packet can have three marking states: *none*, *encoded*, and *affected*. With the *encoded* mark, the value of overload is encoded and signaled in an in-band fashion to the egress nodes. The core router will mark leaving packets as *encoded*, so that the total size of *encoded* packets is B_{overload}. Other packets, as long as severe congestion persists, will be marked as *affected*. As a result, if a packet is received that is marked as *affected* or *encoded*, the egress edge node knows that the corresponding flow has passed a congested node.

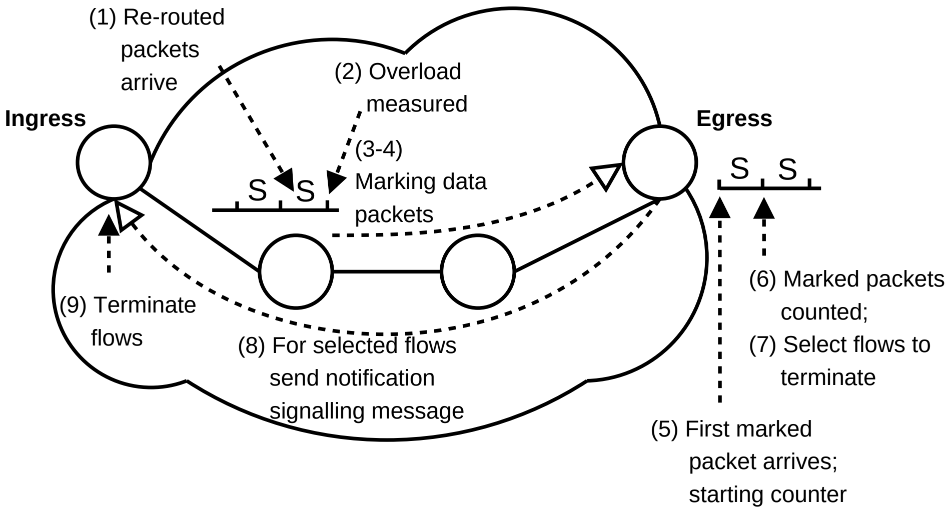
(1) Re-routed packets arrive
(2) Overload measured
(3-4) Marking data packets
Ingress
Egress
S
S
S
S
(6) Marked packets counted;
(7) Select flows to terminate
(9) Terminate flows
(8) For selected flows send notification signalling message
(5) First marked packet arrives; starting counter

Figure 1: Basic severe congestion handling mechanisms of RMD

Step 5. After having received the first marked packet, every egress node $e \in E$ counts the number of received *encoded* bytes B_e with counting interval S, and maintains a list of affected flows.

Step 6. When the counting interval is over, the egress node e transforms the counter back to an overload bandwidth value as B_e/S.

Step 7. If $B_e/S > 0$, any egress router e chooses the set of flows to terminate, which are altogether responsible for generating B_e/S bandwidth.

Step 8. For all flows to be terminated the egress edge sends a notification signaling message to the corresponding ingress edge node to terminate the session.

Step 9. The flows that receive notification messages will be terminated by the ingress nodes.

Remark 4.1. Marking in Step 4 can be done either by setting two unused bits in the IP packet header or potentially the ECN bits [21] (3 states need at least two bits to encode); or having assigned two other DSCPs for every traffic class and changing the DSCP of the packet. Though, in practice it is hard to find free bits in the IP header and it is also hard to allocate many DSCPs. My algorithm can operate with two marking states as well (*none* and *encoded*). In this case, only 1 bit or 1 extra DSCP is required, and the egress nodes are still informed about the precise volume of overload. Though, without knowledge about *affected* flows, egress nodes can choose from a smaller set of flows for termination in each round. As a result, the severe congestion handling may be slower if the flows with *encoded* packets are not enough to cease congestion.

Remark 4.2. The marking procedure essentially piggybacks signalling information onto regular user packets, so it is a good idea to ensure that marked packets are not dropped. This can be achieved, e.g., by proper queue setup. If an *encoded* packet is dropped, less flows will terminate in the given round. That is, dropped marked packets also slow down congestion handling.

Remark 4.3. My algorithm does not influence the validity of Thesis 3.3: by using a separate threshold for refresh messages, re-routed flows may be preserved up to a higher threshold than to which newly arriving flows are compared. Moreover, if the interior node uses packet drop count to detect overload, the scheduler can be set to rate-limit the traffic class to A_{sev} instead of A. With bandwidth measurement based overload detection, overload can be noticed above A_{sev}. *That is, Property 5 is fulfilled.*

Remark 4.4. *Property 6 is fulfilled* because flow selection in Step 7 may be based on arbitrary policies, e.g.: random selection; bit-rate (e.g., lower bit-rate flows are terminated first) or administrative priorities (e.g., within the same class some flows may receive higher precedence for higher price of service).

THESIS 4.2 – Response time of the algorithm [C2, C3, J4]

> *I have shown that if my Algorithm 4.1 is used without the loss of marked packets then congestion is resolved within $3S + \mathrm{RTT}_{max}$ time after the appearance of the first re-routed packet, where RTT_{max} denotes the maximal round-trip time in the domain. Hence, by choosing S appropriately my Algorithm 4.1 fulfills Property 3.*

Congestion handling is known to be a complex control theory problem [22]. Algorithm 4.1 is based on delayed feedback: marking is done in core nodes, the decisions are made at egress nodes, and flow termination is actually done in ingress nodes. From the marking of the first packet, the delay consists of the trip time of data packets from the congested core node to the egress, the counting interval in the egress (S), and the trip time of the notification signaling messages from egress to ingress (see Fig. 1). Moreover, until the overload decreases at the congested core node, an additional trip-time from the ingress node to the core node must elapse. Considering all the delays involved, one gets that signaling the congestion has no influence on the overload for as long as $S + \mathrm{RTT}$, where RTT is the round-trip time.

In classic control theory it is easy to demonstrate over-reactions of control algorithms due to a delayed feedback. In Algorithm 4.1, this over-control appears as well unless further mechanisms are introduced. It is natural to happen as the necessary number of flows are already being terminated but the core node still signals overload to the egress.Therefore, more flows than necessary will be terminated, which appears as an undershoot in the link utilisation figures.

THESIS 4.3 – Avoiding Overreaction [C2]

> *By applying a finite set of memory registers in the core nodes to keep track of previously signaled overloads related to recent measurement periods I have given a solution to estimate the* not yet signaled *portion of the current overload toward the egress nodes. This is done by taking into account the delays of the control loop. I have shown that with the minimum number of memory cells $\left(1 + \lceil\frac{\mathrm{RTT}}{S}\rceil\right)$ undershoot can be eliminated.*

Algorithm 4.2 (Overreaction-free Severe Congestion Handling)**.**

Compared to the Algorithm 4.1, let us use a set of registers in the core nodes to keep track of the signaled overload for a couple of recent measurement intervals. At the end of a measurement period before encoding and signaling the overload as marked packets, the actual measured overload is decreased with the sum of already signaled overloads stored in the memory, since that overload is already being handled in the control loop.[3]

With the above solution, instant severe congestion can be resolved. The problem of incorrect admission control still needs to be taken care of.

THESIS 4.4 – Override with Pure MBAC and Smooth Moving Average [J4, C3, P6]

I have extended RMD's signaling based admission control with a measurement based method, which temporarily overtakes control during the transients of severe congestion in order to avoid incorrect admission of flows into the network. In order to quickly identify the situations when the measurement based admission control shall switch on, I have used a combination of a smoothed and a fast moving average for bandwidth measurement (see Alg. 4.3). This ignores regular traffic fluctuations but reacts fast to sudden load shifts.

One of the grounds behind the usage of reservation based admission control instead of pure measurement based ones is that for the signaled reservations resources can be maintained even if no traffic is using them.

Remark 4.5. Using traditional sliding window or exponentially weighted moving averages the measured bandwidth values can be averaged and smoothed at different timescales. Depending on the parameter settings, however, either fast reaction is achieved, which is sensible to small changes in the measurements; or good smoothing is achieved though for the price of slow detection of sudden bigger changes in the load.

Algorithm 4.3 ([P6]).

1. Let us use an exponentially weighted moving average as a smooth average of the measured parameters with weight w.

2. At each period, after a new sample is available (s), it is checked whether the sample is significantly higher or lower than the previous average. If yes, the new measured value should receive a higher weight ($w_{\text{adaptation}} > w$, $w_{\text{adaptation}} \approx 1$) than normally, in order to quickly adapt to the new level:

```
If $|s - avg_{i-1}|$ significant Then
  $\text{avg}_i = \text{avg}_{i-1} \cdot (1.0 - w_{\text{adaptation}}) + m_i * w_{\text{adaptation}}$
Else
  $\text{avg}_i = \text{avg}_{i-1} \cdot (1.0 - w) + m_i \cdot w$
EndIf
```

Significant means that the difference between the new measurement and the previous average is higher than a relative or absolute threshold.

[3] Note that edge nodes are *not* good candidates to implement such a memory as the core node may mark packets of different flows as *encoded* in different measurement intervals. For instance, in the second round an egress node may also be notified about a certain overload, which egress node has received no *encoded* packets in the first round.

I have given alternatives that are able to remedy the problem of incorrect admission controls based only on the RMD signalling messages, which can be used when the operator cannot or does not want to run a parallel measurement-based admission control (MBAC) algorithm.

THESIS 4.5 – Override based on Refresh Estimation [J4, C3]

> *I have proposed a method to detect the presence of re-routed flows by comparing the number of refresh massages to the period a refresh interval earlier. Furthermore, $i)$ I have given a greedy solution, which, after the detection, refuses all incoming flows for a complete refresh interval; and $ii)$ I have also given an algorithm (Alg. 4.4) that estimates the volume of re-routed flows from the refresh messages.*

The greedy solution rejects all new flows for a complete refresh interval as –in worst case– some of the re-routed flows might just had refreshes their resources prior to the re-routing.

Algorithm 4.4 (Refresh Estimation [J4, C3])**.**

RMD edge nodes refresh flow reservations in periodic R interval (30 seconds by default). RMD core nodes have a timer which expires in periodic R/N interval, where N equals to 10 by default. This way, a complete R refresh interval is split to N smaller time interval, so called cells. When looking at a time window of length R in the core node, each flows reservation or refresh can be found in exactly one of the cells. Each cell is represented by a counter C^i, $i = 1$ to N, recording the reserved and refreshed bandwidth in the corresponding cell. At the end of each cell, the current counter is copied to C^0 and the window steps ahead one cell so that C^N contains the data of the oldest cell.

Opposed to the original RMD mechanism, I proposed that during operation, in the currently active cell, the refreshed bandwidth is counted separately from newly reserved bandwidth. Let us denote its variable $refcount$. If ϵ denotes a tolerance factor for late or early refresh caused by network jitter, then

$$\text{if } (C^N + \epsilon < refcount) \Rightarrow \text{re-route occurred.}$$

If at the end of a cell, the refreshed bandwidth is higher than the value a complete window earlier, then the difference (denoted by d_i) equals to the *volume of re-routed traffic* that refreshed in the given cell, i.e., $d_i = refcount - C^N$. Let us denote the cumulated excess traffic volume after cell i with A_i. At the end of each cell i:

$$A_i \leftarrow A_{i-1} + d_i$$

At the end of cell i the flows belonging to the remaining $(N - i)$ cells have not refreshed since the re-routing. By assuming a uniform distribution for the arrival of refresh messages over time, their volume can be estimated by $\frac{N-i}{i} \cdot A_i$. Therefore, the total volume of re-routed traffic can be estimated as $A_i + \frac{N-i}{i} A_i$. The estimate is more and more precise cell by cell.

Corollary 4.1.

By applying the proposed solutions, all expectations have been achieved: immediate severe congestion is quickly resolved without terminating more flows than necessary (Properties 1-3); incorrect admission of new flows can be prevented; it is possible to temporarily provide *preference of previously admitted flows (Prop. 5) in case of re-routing, and edge nodes are able to select which flows to terminate based on arbitrary policies (Prop. 6).*

5 Conclusion

In my dissertation I have shown that the reduced-state property of some resource reservation protocols like RMD theoretically constrains the set of feasible admission control formulas. However, I have found that many practically relevant admission formulas are feasible to implement in a reduced-state fashion, and I have given some necessary and some sufficient conditions to verify if a formula is reduced-state compatible.

I have shown as an important result, that the lightweight RMD protocol performs as good as RSVP with regard to reservation blocking, notification times (even better) and signaling overhead, while it is much simpler.

I have identified the inadequateness and incorrect operation of reduced-state admission controls in the case of severe congestion after a re-routing event. I have introduced methods to overcome these limitations. First I have given methods to regulate the excess traffic and second I have given methods to correct admission control during the transients of reservation re-establishments.

With my proposals, a reduced state resource reservation protocol, like the RMD protocol, can be made resilient to the negative consequences of re-routing, as it can restore QoS and to return to correct admission control in a short notice.

6 Application of My Results

My proposed methods have been adopted in the RMD QoS model of the NSIS Signalling Layer Protocol [O1]. The protocol is being standardised in the NSIS working group of the IETF. My RMD related work was part of seven international patent applications. My RMD simulator prototype has been used and developed for related work by University of Cyprus [23], University of Twente [24] and Ericsson Research Hungary.

Even though I have created most of my results for RMD, they are applicable in a more general scope. My investigation of Thesis Group 1 about the feasibility of admission control algorithms is valid for any environment where per-flow states cannot be used. The severe congestion handling algorithms described in Thesis Group 4 rely only on the facts that $i)$ network interior routers cannot identify individual flows, and $ii)$ edge routers possess per-flow states, so these can manage individual flows. These congestion handling algorithms work with any load control protocol fulfilling these properties, i.e., with DiffServ conform protocols.

In the RMD/NSIS protocol, severe congestion handling is only one, albeit important, task. A recently formed working group of IETF –Pre-Congestion Notification (PCN)–, however, specifically aims at this problem, which might by a good application of my results. The problem statement draft [25] of the PCN WG writes:

> “...While admission control will protect the QoS under normal operating conditions, an additional flow pre-emption mechanism is necessary in the times of heavy

> congestion (e.g. caused by route changes due to link or node failure).”

The description of the PCN working group is as follows [4]:

> “The Congestion and Pre-Congestion Notification (PCN) working group develops mechanisms to protect the quality-of-service of established inelastic flows within a DiffServ domain when congestion is imminent or existing. These mechanisms operate at the domain boundary, based on aggregated congestion and pre-congestion information from within the domain.”

That is, the WG aims to solve the congestion phenomenon in a DiffServ environment with an aggregation based network interior.

> “The focus of the WG is on developing standards for the marking behavior of the interior nodes and the encoding and transport of the congestion information. . . , metering of congestion information at the egress, and transport of congestion information back to the controlling ingress.”

That is, the WG considers a solution conceptually similar to my Algorithm 4.1, with encoding and transport of overload information to egress nodes after which the egress edges try to determine the value of overload from the encoded congestion information and where this congestion information is transported back to the ingress to make appropriate reactions.

My research work could be a basis for this working group and my given algorithms might be part of a solution.

Acknowledgements

I would like to thank Attila Takács for the joint work in the past years and Róbert Szabó, my supervisor, for his useful directions. I owe special gratitide to those who directly contributed to this thesis work: Felicián Németh for his help and review of Thesis Group 1; Csaba Antal, János Harmatos and Zalán Heszberger for the multiple reviews of the theses drafts; Attila Báder for his comments and also for keeping me up to date with IETF relations; and Péter Barta for the review and comments to [J4], the basis for my dissertation.

Thanks also to those people who had a positive indirect influence on my work through their individual professional and scientific views: Attila Mihály, Oktávián Papp, and Gábor Rétvári.

I also have to say that without the inspiration, love and endless support from Kati and my family, I would not have come this far.

This research work was supported by TrafficLab, Ericsson Research and the High Speed Networks Laboratory of the Department of Telecommunication and Media-Informatics at Budapest University of Technology and Economics.

[4] Citation from http://www.ietf.org/html.charters/pcn-charter.html

References

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurement from the Sprint IP backbone," *IEEE Network Magazine*, Nov 2003.

[2] Lee W. McKnight and Joseph P. Bailey, Eds., *Internet Economics*, The MIT Press, 1998.

[3] Clarence Filsfils and John Evans, "Engineering a multiservice IP backbone to support tight SLAs," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 40 Special issue: Towards a new internet architecture, pp. 131–148, Sept. 2002.

[4] Michael Menth, Rüdiger Martin, and Joachim Charzinski, "Capacity overprovisioning for networks with resilience requirements," in *Proc. of SigComm'06*, Pisa, Italy, Sept. 2006.

[5] "Internet 2 QBone Bandwidth Broker Advisory Council," homepage: `http://qos.internet2.edu/qbone/QBBAC.shtml`.

[6] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," RFC 2638 (Informational), July 1999.

[7] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (RSVP) – version 1 functional specification," RFC 2205, IETF, Sept. 1997.

[8] "Next Steps in Signaling (nsis)," IETF Working Group web page, `http://www.ietf.org/html.charters/nsis-charter.html`.

[9] Robert Hancock, Georgios Karagiannis, John Loughney, and Sven Van den Bosch, "Next steps in signaling (NSIS): Framework," RFC 4080, IETF, June 2005.

[10] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC 2475, IETF, Network WG, Dec. 1998.

[11] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, and Christophe Diot, "Characterization of failures in an IP backbone," in *Proceedings of IEEE InfoCom*, Hong Kong, Mar. 2004, IEEE.

[12] D. Watson, C. Labovitz, and F. Jahanian, "Experiences with monitoring OSPF on a regional service provider network," in *Proc. International Conference on Distributed Computing Systems*, 2003, pp. 204–213.

[13] Sundar Iyer, Supratik Bhattacharyya, Nina Taft, and Christophe Diot, "An approach to alleviate link overload as observed on an IP backbone," in *Proceedings of IEEE InfoCom*, San Fransisco, CA, USA, Mar. 2003, IEEE.

[14] Kevin Fall and Kannan Varadhan, *ns Manual*, UC Berkeley.

[15] Goergios Karagiannis, Simon Oosthoek, and Martin Jacobson, "Maintenance of sliding window aggregated state using combination of soft state and resource release principles," International Patent Application, WO 02/076035 A1, PCT/EP02/01856, Sept. 2002.

[16] Sally Floyd, "Comments on measurement-based admissions control for controlled-load services," Tech. Rep., Lawrence Berkeley National Laboratory, July 1996.

[17] Harry G. Perros and Khaled M. Elsayed, "Call admission control schemes: A review," *IEEE Communications Magazine*, vol. 34, no. 11, pp. 82–91, Nov. 1996.

[18] Lee Breslau, Sugih Jamin, and Scott Shenker, "Comments on the performance of measurement-based admission control algorithms," in *Proceedings of the IEEE Conference on Computer Communications (InfoCom)*, Tel-Aviv, Israel, Mar. 2000.

[19] Zoltán R. Turányi and Lars Westberg, "Load control: Congestion notifications for real-time traffic," in *9th IFIP Working Conference on Performance Modelling and Evaluation of ATM and IP Networks*, Budapest, Hungary, July 2001.

[20] "Technical report on enhanced network survivability performance," Tech. Rep. T1.TR.68, ANSI, Feb. 2001.

[21] S. Floyd, "Specifying alternate semantics for the explicit congestion notification (ECN) field," RFC 4774, IETF, Network WG, Nov. 2006.

[22] Andreas Pitsillides, Petros Ioannou, Marios Lestas, and Loukas Rossides, "Adaptive non-linear congestion controller for a differentiated-services framework," *IEEE/ACM Transactions on Networking*, vol. 13, no. 1, pp. 94–107, Feb. 2005.

[23] Costas Djouvas, "Extending diffserv architecture: Integration of idcc and rmd framework," M.S. thesis, University Of Cyprus, Department Of Computer Science, May 2003.

[24] Gerjan Stokkink, "Performance evaluation of severe congestion handling solutions for multilevel service in rmd domains," in *Proceedings of he 4th Twente Student Conference on Information Technology (TSC on IT)*, Twente, The Netherlands, Jan. 2006, pp. 89–98.

[25] K. Chan, A. Charny, and P. Eardley, "Pre-congestion notification problem statement," Internet Draft draft-chan-pcn-problem-statement-01, IETF, Oct. 2006, Work in progress!

Publication of New Results

Journal Articles

[J1] András Császár, Gábor Enyedi, Markus Hidell, Gábor Rétvári, and Peter Sjödin, "Converging the evolution of router architectures and IP networks," *IEEE Network Special Issue on Advances in Network Systems Architecture*, July-August 2007.

[J2] András Zahemszky, András Császár, Gábor Tóth, and Attila Takács, “Átjáró szerver választás a GMPLS PCE architektúrában,” *Híradástechnika*, Apr. 2007, in Hungarian.

[J3] Attila Takács, András Császár, Róbert Szabó, and Tamás Henk, “Generic multipath routing concept for dynamic traffic engineering,” *IEEE Communications Letters*, vol. 10, no. 2, pp. 126–128, Feb. 2006.

[J4] András Császár, Attila Takács, Róbert Szabó, and Tamás Henk, “Resilient reduced-state resource reservation,” *Journal of Communications and Networks*, vol. 7, no. 4, pp. 509–524, Dec. 2005.

[J5] Attila Takács, András Császár, Róbert Szabó, and Tamás Henk, “Forgalommenedzsment többszörös kapcsolatú tartományoknál,” *Híradástechnika*, vol. LIX, no. 2004/9, pp. 19–25, Sept. 2004, in Hungarian.

[J6] András Császár, Csaba Lukovszki, and Róbert Szabó, “CBQ alkalmazása differenciált szolgáltatásokhoz 3-ik generációs mobil rendszerekben,” *Híradástechnika*, vol. 11, pp. 27–34, Dec. 2001, in Hungarian.

[J7] András Császár, Attila Takács, and Róbert Szabó, “VoIP szolgálatok minőségbiztosítása,” *Magyar Távközlés*, vol. 5, pp. 14–17, May 2000, in Hungarian.

Conference Papers

[C1] András Zahemszky, András Császár, Gábor Tóth, Attila Takács, and Tibor Cinkler, “Dual purpose gateway selection in the GMPLS PCE architecture,” in *Proceedings of Transcom2007*, Zilina, Slovak Republic, June 2007.

[C2] András Császár, Attila Takács, and Attila Báder, “A practical method for the efficient resolution of congestion in an on-path reduced-state signalling environment,” in *Proceedings of the Thirteenth International Workshop on Quality of Service (IWQoS 2005)*, Passau, Germany, June 2005, number 3552 in Lecture Notes in Computer Science, pp. 282–293.

[C3] András Császár, Attila Takács, Róbert Szabó, and Tamás Henk, “State correction after re-routing with reduced state resource reservation protocols,” in *Proceedings of IEEE Global Telecommunications Conference (Globecom2004)*, Dallas, TX, USA, Dec. 2004, IEEE.

[C4] Attila Takács, András Császár, József Bíró, Róbert Szabó, and Tamás Henk, “Path integrity aware traffic engineering,” in *Proceedings of IEEE Global Telecommunications Conference (Globecom2004)*, Dallas, TX, USA, Dec. 2004, IEEE.

[C5] Róbert Szabó, Attila Takács, and András Császár, “Optimised multi homing – an approach for inter-domain traffic engineering,” in *Proceedings of the 2nd International*

Workshop on Inter-Domain Performance and Simulation (IPS2004), Budapest, Hungary, Mar. 2004, pp. 48–57.

[C6] Georgios Karagiannis, Attila Báder, Gergely Pongrácz, András Császár, Attila Takács, Róbert Szabó, and Lars Westberg, “RMD – a lightweight application of NSIS,” in *Proceedings of the 11th International Telecommunications Network Strategy and Planning Symposium (Networks2004)*, Vienna, Austria, June 2004, pp. 211–216.

[C7] Attila Takács, András Császár, Róbert Szabó, and Tamás Henk, “Examination of free capacity based load sharing,” in *Proceedings of the 2nd IASTED International Conference on Communications, Internet, & Information Technology (CIIT2003)*, Scottsdale, AZ, USA, Nov. 2003.

[C8] András Császár and Attila Takács, “Comparative performance analysis of RSVP and RMD,” in *Proceedings of the Fourth COST 263 International Workshop on Quality of Future Internet Servies QoFIS2003*, Stockholm, Sweden, Oct. 2003, vol. 2811 of *Lecture Notes in Computer Science*, pp. 41–51, Springer Verlag.

[C9] Attila Takács, András Császár, Róbert Szabó, and Tibor Cinkler, “Thrifty traffic engineering through CSLLS,” in *Proceedings of the 18th International Teletraffic Congress ITC18*, Berlin, Germany, Sept. 2003, vol. 5 of *Teletraffic Science and Engineering*, pp. 61–70, Elsevier.

[C10] András Császár, Attila Takács, Róbert Szabó, Vlora Rexhepi, and Georgios Karagiannis, “Severe congestion handling with resource management in diffserv on demand,” in *Proceedings of Networking 2002 – The Second Intl. IFIP-TC6 Networking Conference*, Pisa, Italy, May 2002, vol. 2345 of *Lecture Notes in Computer Science*, pp. 443–454, Springer Verlag.

[C11] Lars Westberg, András Császár, Georgios Karagiannis, Ádám Marquetant, David Partain, Octavian Pop, Vlora Rexhepi, Róbert Szabó, and Attila Takács, “Resource management in diffserv (RMD): A functionality and performance behavior overview,” in *Proceedings of PfHSN’2002 – Seventh International Workshop on Protocols For High-Speed Networks*, Berlin, Germany, Apr. 2002, vol. 2334 of *Lecture Notes in Computer Science*, pp. 17–34, Springer Verlag.

[C12] András Császár, Csaba Lukovszki, and Róbert Szabó, “A differentiated services approach using CBQ for 3G communication,” in *Proceedings of the Polish-Czech-Hungarian Workshop on Circuit Theory, Signal Processing and Telecommunication Networks*, Budapest, Hungary, Sept. 2001, pp. 274–286.

[C13] András Császár, Attila Takács, Csaba Lukovszki, and Róbert Szabó, “Simulation study over IP based GSM backbone,” in *Proceedings of the IEEE International Conference on Telecommunications - IEEE ICT2001*, Bucharest, Romania, June 2001.

Patent Applications

[P1] András Császár, Attila Mihály, and Oktávián Papp, “Enhanced fast re-route for bandwidth-efficient link protection,” International Patent Application PCT/EP2007/057515, July 2007.

[P2] Gábor Enyedi and András Császár, “Loop-free fast interface based rerouting,” International Patent Application PCT/EP2007/057322, July 2007.

[P3] Attila Báder and András Császár, “Priority flow handling in stateless IP network domains,” International Patent Application PCT/EP2007/055178, May 2007.

[P4] Alpár Jüttner, András Császár, and Attila Mihály, “Concept and method for reducing the forwarding tables in network routers,” International Patent Application PCT/EP2007/051220, Feb. 2007.

[P5] Ferenc Pintér, Attila Báder, András Császár, and Attila Takács, “Explicit congestion control method for stateless domains,” International Patent Application PCT/EP2006/068317, Nov. 2006.

[P6] András Császár and Attila Báder, “A method for fast traffic measurement and monitoring,” International Patent Application PCT/IB2006/003405, Nov. 2006.

[P7] Attila Báder, András Császár, and Attila Takács, “Stateless congestion control,” International Patent Application PCT/EP2006/010820, nov 2006.

[P8] András Császár, Attila Báder, and Attila Takács, “A method for handling inter-domain re-routing between stateless quality-of-service domains,” International Patent Application, PCT/IB2006/050990, Mar. 2006.

[P9] András Császár, Attila Takács, and Attila Báder, “A method and system for aggregated resource reservation in internet protocol networks using generalised sliding window algorithm,” International Patent Application, Aug. 2005, PCT/EP2005/009094, WO-2007022789 A1.

[P10] András Császár, Attila Takács, Róbert Szabó, Attila Báder, and Lars Westberg, “Algorithms for fast handling servere congestion in an IP network using differentiated services,” International Patent Application, Nov. 2004, PCT/SE2004/001657, WO 2006/052174 A1.

Other Publications

[O1] Attila Báder, Lars Westberg, Georgios Karagiannis, Cornelia Kappler, Tom Phelan, Attila Takács, and András Császár, “RMD-QOSM - the Resource Management in Diffserv QoS model,” Internet Draft draft-ietf-nsis-rmd-12, IETF, Nov. 2007, Work in progress!

[O2] András Császár, Csaba Antal, Attila Mihály, and Árpád Szlávik, “Carrier-grade resilience in multi-service IP networks,” in *Proceedings of the High Speed Networking 2005 Spring Workshop*, may 2005, pp. 26–28.

[O3] András Császár, “Az analitikus hierarchikus eljárás alkalmazása befektetési döntéstámogató rendszerben,” M.S. thesis, Budapest University of Technology and Economics, Dept. of Information and Knowledge Management, Dec. 2004, in Hungarian.

[O4] Attila Takács, András Császár, Róbert Szabó, and Tamás Henk, “Re-routing in IP networks from the aspect of reduced state resource reservation,” in *Proceedings of the High Speed Networking 2004 Spring Workshop*, May 2004, pp. 102–105.

[O5] Attila Takács, András Császár, József Bíró, Róbert Szabó, and Tamás Henk, “Path integrity aware traffic engineering,” in *Proceedings of the High Speed Networking 2004 Spring Workshop*, May 2004, pp. 30–33.

[O6] András Császár, Attila Takács, Attila Báder, Róbert Szabó, and Csaba Antal, “RMD: Proof of concept,” Internal Technical Report ETH/RT-2004:0011, Ericsson, Jan. 2004.

[O7] András Császár, Attila Takács, and Róbert Szabó, “Severe congestion handling with resource management in diffserv on demand,” in *Proceedings of the High Speed Networking 2002 Spring Workshop*, May 2002, pp. 122–129.

[O8] András Császár, “Differentiated services for voice communication,” M.S. thesis, Budapest University of Technology and Economics, May 2001.

[O9] András Császár and Attila Takács, “Voice traffic control study over IP based GSM backbone,” Tech. Rep., Budapest University of Technology and Economics, Nov. 2000, Essay for Scientific Student Conference.

[O10] András Császár and Attila Takács, “VoIP szimuláció network simulator-ral,” Tech. Rep., Budapest University of Technology and Economics, Nov. 1999, Essay for Scientific Student Conference, in Hungarian.

Dissertation

[D] András Császár, *Reduced-State Resource Reservation*, Ph.D. thesis, Budapest University of Technology and Economics, Department of Telecommunication and Media Informatics, 2007.