A Practical Method for the Efficient Resolution of Congestion in an On-Path Reduced-State Signalling Environment

András Császár, Attila Takács, and Attila Báder

TrafficLab, Ericsson Telecommunication Hungary
Laborc utca 1., Budapest, Hungary, H-1037
{Andras.Csaszar,Attila.Takacs,Attila.Bader}@ericsson.com

Abstract. Currently, the standardisation of on-path signalling protocols is going on within the Next Steps in Signalling (NSIS) Working Group of the IETF. NSIS is responsible for the definition of a general IP signalling protocol. The first use case of the proposed protocol is flow-level resource management. One of the considered reservation methods, reduced-state mode, is based on the Resource Management in DiffServ (RMD) framework. Since it relies only on per-class state information in interior routers, it has a number of benefits including scalability, low complexity, and low memory consumption. However, the price of simplicity is decreased efficiency in case of exceptional situations. The most demanding task for RMD is the handling of congestion that may occur after a failure resulting in re-routing of flows onto a new path. Resolving a suddenly evolved overload without per-flow states is a highly non-trivial task. We present a low complexity mechanism which easily handles the undesirable situation, and we give guidelines to set the parameters of our scheme based on worst-case calculations.

1 Introduction

In the IETF the NSIS working group [1] is responsible for standardising a general IP signalling protocol for flow-level resource management as the first use case. A *flow* is a series of packets transmitted during a session of a specific application by a specific host. For example, the flow could be the packet series of a streaming audio or video session but an FTP file transfer will also generate a packet flow. The task of a flow-level resource management protocol is to implement admission control. That is, it has to decide whether the network has enough free resources to accommodate both the new and the previously admitted flows.

The intention of the NSIS QoS application (QoS-NSLP [2]) is to re-use, where appropriate, the protocol mechanisms of RSVP [3], while at the same time applying a more general signalling model suiting other, possibly simpler, resource management schemes as well. Currently, the working group considers implementing two models, a *stateful* and a *reduced-state* realisation. These terms reflect the complexity of interior nodes in a domain. The stateful solution store per-flow

state information in interior nodes suiting the IntServ/RSVP QoS model. On the other hand, the reduced-state solution relies on aggregated, per-class states in interior nodes, and only edge nodes are permitted to dispose of per-flow states. This mode implements the DiffServ/RMD model, where RMD stays for Resource Management in Diffserv [4–7].

RSVP, RMD, and the new QoS-NSLP are on-path resource management protocols because they all reserve resources, typically bandwidth, in a hop-by-hop manner. Before admitting the flow, a signalling message is sent between the two end-points. On every hop, this signalling message requests admission and reserves the resources if admitted. The admission decision for a link is generally very simple. If the sum of the newly arriving bandwidth request and the sum of admitted reservations is smaller than or equal to a pre-defined *admission threshold*, the new request is admitted and its reservation is added to the sum. The difference between a stateful and a reduced-state solution is coming from what kind of states are available and established by the signalling message in interior nodes.

In previous papers [6, 7] the authors argued 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 poses less processing and storage capacity requirements to core routers and it has a smaller protocol overhead, while the achievable performance is similar.

However, there is an exceptional situation where per-flow states give an advantage to stateful protocols. This situation occurs when routing protocols redirect flows from their original path to alternative paths. This might happen as a reaction to node or link failure. In general, on-path resource reservation protocols face the problem that the admission of re-routed flows to the new path is uncertain for example because there is not enough bandwidth to accommodate all re-routed flows. If this is the case *severe congestion* occurs [7–9].

We devote this paper to the severe congestion handling procedure of RMD. We detail the problem of severe congestion in Section 2. Then we focus on congestion handling mechanisms in Section 3, and introduce our improved solution. In Section 4 we present a worst-case model for the configuration of the proposed method. Finally, we conclude the paper.

2 Severe congestion: problem description

Severe congestion is considered as an undesirable state, which may occur as a result of a route change. Typically, routing algorithms are able to adapt and change their routing decisions to reflect changes in the topology (e.g., link or node failures) and traffic volume. In such situations the re-routed traffic will follow a new route. Nodes located on this new path may become overloaded, severe congestion may occur, since they suddenly might need to support more traffic than their capacity. The resource management protocol in reaction to

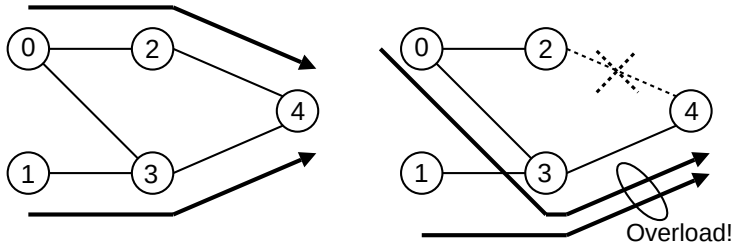

Fig. 1. Severe congestion example

severe congestion has to terminate[1] some flows on the congested path in order to ensure proper QoS for the remaining flows.

Figure 1 shows an example scenario. First, the traffic distribution was such that the upper and lower paths were both utilised under normal conditions. After a link failure, the traffic of the upper path was suddenly re-directed to the $0-3-4$ path without immediately initiating reservation requests to decide on the admission of the traffic to this new path. The joint load on link $3-4$ is higher than the admission threshold, hence a mechanism is needed to recover to an acceptable load rather than to accept QoS degradation. The incoming traffic volume has to be decreased by means of terminating flows so that if a flow remains admitted after congestion handling, it has to be given the full previously reserved bandwidth.

Generally, stateful resource reservation protocols like RSVP or YESSIR [10] store the destination address for all flows in their database. If the routing protocol announces which destination address have been assigned a new path, the reservation protocol can easily identify the re-routed flows. Then, it can try to re-reserve the needed resources for all these flows on the new path. For example, in Fig. 1 router 0 can initiate a new reservation for all flows headed towards destination address 4. If this new reservation fails, the refused flows will be terminated within a round-trip time (RTT) when the source node or the domain's ingress edge node is informed. In RSVP this feature is called *Local Repair*. This approach is advantageous since the whole procedure is as quick as it can get, the reaction starts as soon as the routing protocol re-directs the flows and finishes within a RTT. Additionally, Local Repair also re-reserves the resources for as many of the re-routed flows as possible according to the admission threshold (i.e., only the excess part of the traffic is terminated).

The above procedure is based on per-flow states. However, due to the absence of per-flow states in interior nodes, a reduced-state protocol is not able to distinguish the re-routed flows from those ones that were originally traversing that path. Similarly to RSVP, RMD or the RMD-based QoS-NSLP of NSIS use the

[1] Flow termination may mean to stop the flow, or to re-map it to a lower-quality (e.g., best-effort) class. In any case, the originally agreed resources are not guaranteed any longer.

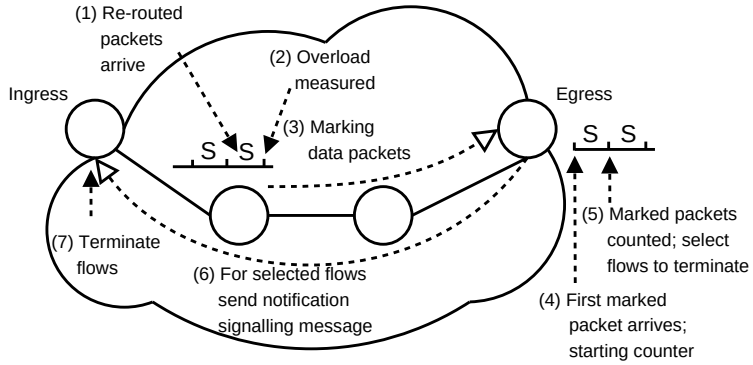

Fig. 2. Basic severe congestion handling mechanisms of RMD

soft-state principle to keep reservations up-to-date: if flows do not refresh their reservation within periodical intervals (30 seconds by default), their reservations time out. Considering re-routing, this means that the re-directed flows are going to try to refresh their reservation maximum within a refresh interval, when the excess flows that are above the *admission threshold* can be terminated.

However, during this time the quality of flows may be violated seriously, e.g., packet drops may occur. Hence, reduced-state protocols need an alternative mechanism, which is able to handle severe congestion quickly. Since resource reservation is foreseen to be applied to streaming traffic, primarily to VoIP applications, the expectation is to be able to handle severe congestion within a few hundred milliseconds so that the users of not terminated calls do not realise quality degradation.

3 Severe congestion handling

In [8] the authors gave a solution for severe congestion handling for the original, stand-alone RMD protocol. In this paper we are describing an enhanced, more precise solution to be used for the reduced-state NSIS QoS-NSLP protocol.

With reduced-state protocols, severe congestion in the communication path has to be notified to the edge nodes, since core nodes do not have per-flow identification and so cannot initiate flow termination. Edge notification is done by the following mechanism (see Fig. 2).

After re-routed packets arrive (1) core routers estimate the overload of the link by measuring arriving traffic (2) with a fixed measurement interval (S). If the estimated bandwidth is higher than the predefined admission threshold, denoted by T, the router concludes severe congestion. After that, the core router informs the edge nodes about the amount of the estimated overload by marking[2]

[2] Setting a bit in the header, or changing the DiffServ code-point to a domain specific administrative code-point, etc.

(3) regularly dequeued user data packets. The number of marked packets encodes the amount of overload. As an example, let us suppose that after a measurement interval the router estimated an arrival rate R, and $R > T$. The estimated overload is $R - T$ (e.g., in bytes per seconds). The core node transforms this metric to a number of bytes to mark as $B = (R - T) \cdot S$. The core router will mark that many leaving packets the total size of which corresponds to B, while a new bandwidth measurement cycle begins. With this procedure the core node can manage to encode and signal the amount of overload to the egress nodes without relying on per-flow information.

After receiving the first marked packet (4), every egress node counts the number of received marked bytes with counting interval S. After this (5), egress nodes transform the counted marked bytes back to an overload bandwidth value as follows. Let $\mathcal{E}$ denote the set of egress routers receiving marked packets. B_e is the amount of marked bytes counted by egress routers ($e \in \mathcal{E}$). Then $\frac{B_e}{S}$ gives the amount of overload seen by egress router e. Note that if marked packets are not dropped than the summarised overload bandwidth seen by all affected egress routers is the same as the overload estimated on the core node:

$$\sum_{\forall e \in \mathcal{E}} B_e = M \;\Rightarrow\; \sum_{\forall e \in \mathcal{E}} \frac{B_e}{S} = R - T.$$

After calculating $\frac{B_e}{S}$, any egress router e chooses a set of flows to terminate which will resolve the calculated overload. Preferably, the flows to be terminated are randomly selected among the flows that passed the congested link.

For every chosen flow, the egress sends back a congestion report (6) signalling message to the ingress node of the flow, requesting it to terminate the connection.

A shortcoming in the original RMD solution was how the egress identifies the set of flows that were passing through the congested node in order to select some of them to terminate. Trivially, ending an arbitrary flow that has a different route (i.e., did not contribute to the overload) makes no sense. The previous solution can only choose from those flows that received marked packets. However, the network operator may wish to choose flows for termination not only from those that received marked packets but from all flows involved in the congestion. E.g., within a voice traffic class the operator may prefer emergency calls over regular phone calls even though accidentally only the packets of the emergency calls got marked.

We propose to rely on an enhanced stamping procedure of the interior node. The previously explained procedure to encode and signal overload value is only slightly modified. The key idea is, to stamp all other packets in the congested node that are not carrying encoded overload information with a new *affected* stamp. That is, a packet can have three stamping states: *none*, *encoded*, and *affected*, respectively. The stamping can be done with the help of two bits in the packet header, e.g., the Explicit Congestion Notification (ECN) bits (described in [11]), or by separating two bits from the DiffServ Code Point (DSCP). Another solution is if the network operator locally assigns for every traffic class two other DSCPs, which are interpreted at its egress nodes as encoded or affected stamps.

3.1 Solving the over-reaction

The severe congestion handling algorithm defined in the original RMD concept (described in the previous section) has a major shortcoming which needs to be addressed to get an efficient solution.

The problem is over-reaction. With over-reaction we refer to the case when more flows are terminated than necessary by the congestion handling algorithm. This effect can be seen as an "undershoot" in the link utilisation graph of the affected links. This is highly undesirable as the result is lot of terminated flows (e.g., VoIP calls) and poor network utilisation after the congestion.

The reason of the over-reaction is the delayed feedback included in the congestion handling control loop. Since marking is done in core nodes, the decisions is made at egress nodes, and termination of flows are done in ingress nodes, a significant delay may be introduced until the overload information is learned by ingress nodes. 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 explicit signalling messages from egress to ingress (see Fig. 2). Moreover, until the overload decreases at the congested core node an additional trip-time from ingress to the core must expire. This is because immediately before the congestion notification and flow termination the ingress may have sent out packets in the flows that were selected for termination. That is, a terminated flow may contribute to the congestion for a trip-time from the ingress to the core node. Considering all the delays involved we get that signalling the congestion has no influence on the overload for as long as $S + \mathrm{RTT}$, where RTT is the round-trip time. However, with the original congestion handling method core nodes continue marking the packets until the measured utilisation falls below the congestion threshold. Thus, it can happen that the necessary number of flows are already being terminated but the core node still signals overload to the egress as shown in Fig. 3. This way, at the end more flows will be terminated than necessary.

To solve the problem of over-reaction a memory needs to be introduced in the core nodes keeping track of the signalled overload in a couple of previous measurement intervals. At the end of a measurement period before signalling the overload, the actual measured overload should be decreased with the sum of already signalled overloads stored in the memory since that overload is already being handled in the severe congestion handling control loop.

We propose to use a sliding window to store previously marked overloads. The memory consists of an integer number of cells. At the end of every measurement interval, the newest calculated overload is pushed into the memory, and the oldest cell is dropped. If R denotes the measured data rate at the end of a measurement period, T is the admission control threshold, and M_i is the ith memory cell ($i \in [1..N]$), then at the end of every measurement interval the considered overload, as shown in Fig. 4, is the following:

$$\hat{R} = R - T - \sum_{i=1}^{N} M_i \,.$$

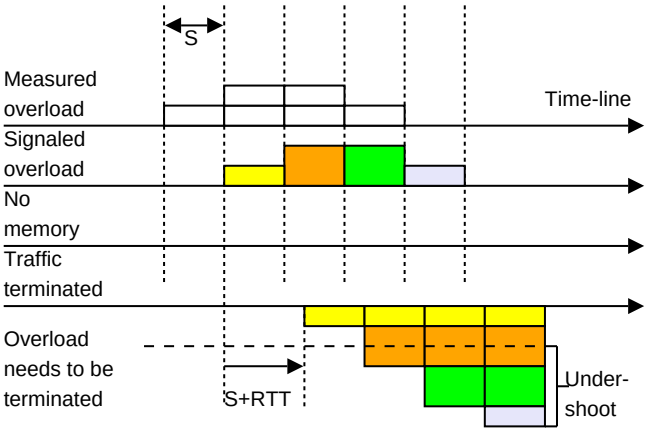

Fig. 3. Severe congestion handling without memory

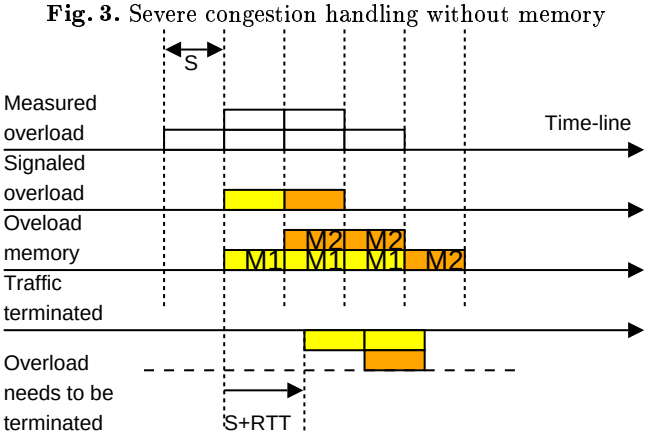

Fig. 4. Severe congestion handling with the proposed method

Next, the memory is updated as:

$$M_i := M_{i+1}\ \forall i = 1..(N-1), \text{ and } M_N := \hat{R}.$$

As discussed earlier, the time until a congestion signal (marked packets) affects the overload is $S + \mathrm{RTT}$. To make sure that already handled congestion signals do not cause over-reaction, the number of memory cells must be set according to:

$$N = \lceil \frac{S + \mathrm{RTT}}{S} \rceil = 1 + \lceil \frac{\mathrm{RTT}}{S} \rceil.$$

While S is the measurement period which is unique in the domain, flows are likely to have different RTT values. To guarantee the elimination of over-reaction considering all flow, the operator should overestimate the required memory cells using the maximal RTT should in the calculation.

4 Worst-case model for configuring the parameters of the congestion handling method

This section addresses the joint configuration of the number of memory cells (N) and the length of the measurement interval S, the two parameters of our severe congestion handling method. Our dimensioning tool is based on a worst-case severe congestion handling model. The model has been implemented as a simplified simulator that in a step-by-step manner calculates the load of the congested link at the end of every measurement interval. The model has three parameters; i) the measurement interval (S), ii) the maximal round-trip time ($\mathrm{RTT}_{\max}$), and iii) the number of memory cells (N).

For the dimensioning two aspects of severe congestion handling are considered. The volume of over-reaction and the time needed to resolve the congestion (handling time). First, eliminating over-reaction is necessary to keep network utilisation high even after a congestion occurred. Second, having a fast solution that resolves congestion within a few hundred milliseconds is essential for a resource management protocol being considered for deployment in a high-resilience carrier-grade network.

Undershoot is the volume of the unnecessarily terminated flows in percent of the threshold. If after congestion the load sinks back right to the admission threshold, undershoot is zero; however, if the load decreases to zero, i.e., all flows are terminated, undershoot is 100%. Handling time is defined as the time between the arrival of the first re-routed packet on the given link up to the arrival of the last packet of the last terminated flow at the link.

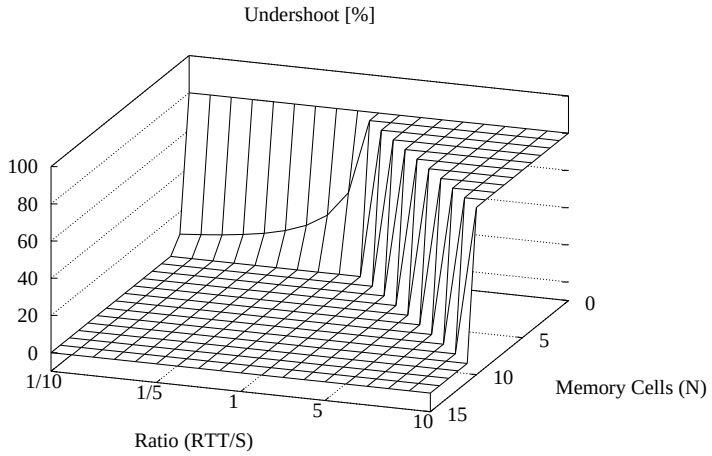

Fig. 5. Undershoot occurrance with different parameters

The simulator evaluates the worst-case where all flows are supposed to have the maximal RTT, i.e., the flow termination process will need in all cases the

maximal $S + \mathrm{RTT}_{\max}$ amount of time. The result is that the highest possible undershoot and the longest handling time is calculated.

First of all, we point out that regarding the worst-case calculation of the volume of over-reaction the particular values of S and RTT are irrelevant, only the ratio of these ($\frac{RTT}{S}$) affects the results. Intuitively, by having $S < \mathrm{RTT}$, multiple measurements are carried out during the feedback of congestion. Hence, congestion is detected fast but over-reaction is more likely to occur. On the other hand, if $S > \mathrm{RTT}$, the feedback delay is low compared to the slow measurements, hence for the price of higher handling time over-reaction will not be a likely phenomenon. Further consideration is left to the reader.

Figure 5 shows the dependence of the worst-case over-reaction from the number of memory cells (N) and the ratio of RTT and S. The figure shows that if $N \geq 1 + \lceil \frac{\mathrm{RTT}}{S} \rceil$, over-reaction is eliminated in all cases. Without memory, the algorithm will always over-react (N=0). An important message of the figure is that one needs the less memory cells the higher value the ratios takes. That is, the higher S is compared to $\mathrm{RTT}_{\max}$. To reduce processing and storage requirements in core routers, this would speak for configuring S to at least the maximal round-trip time, where only two memory slots are needed.

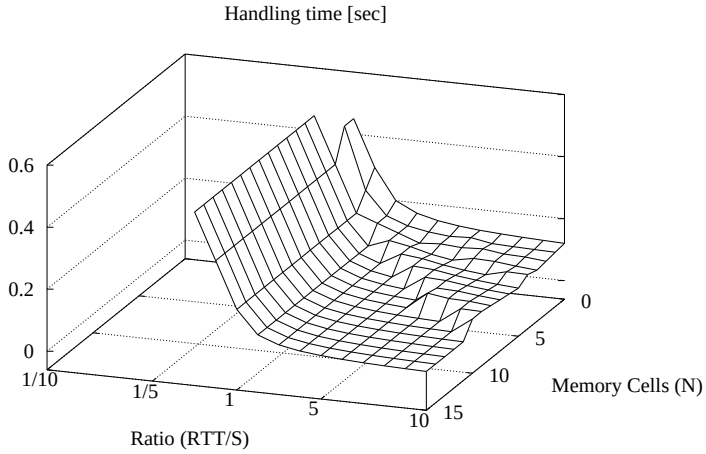

Fig. 6. Congestion handling time with different parameters

On the other hand, increasing the measurement period will result in increased handling time. Figure 6 shows the results for a maximal RTT of 50 milliseconds. It shows that adding memory cells has little influence on handling time. Opposed to over-reaction, Fig. 6 also shows that choosing higher S has bad influence onto the handling time. However, by considering both figures one can see that there are parameter setups where both handling time and undershoot are low. To quantify this observation we defined a wellness function ($W(.)$) which formalises our precedence relation between low over-reaction and low handling times. The

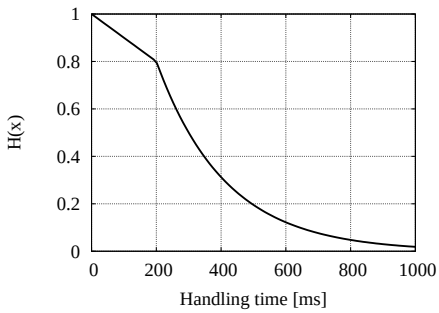

Fig. 7. $H(x)$

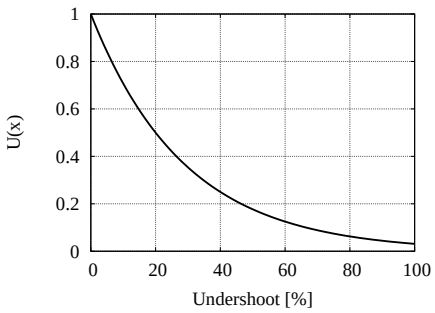

Fig. 8. $U(x)$

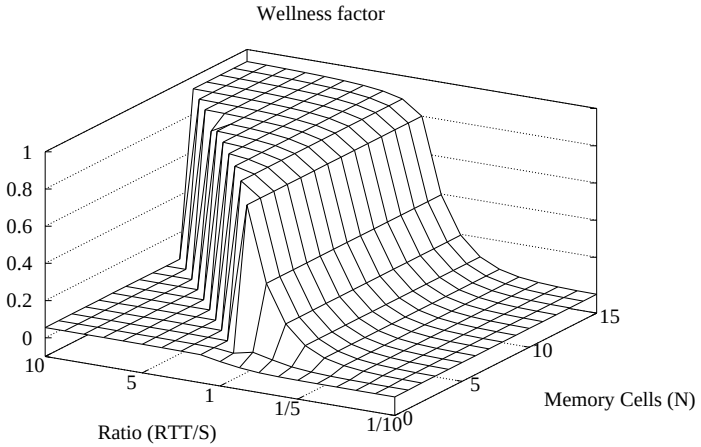

Fig. 9. Wellness factor with different parameters

measure is $W(h,u) = H(h) \cdot U(u)$, where h is the handling time, u is the undershoot. Function $H(.)$ is shown in Fig. 7 and $U(.)$ in Fig. 8 respectively. By formalising the functions $U(.)$ and $H(.)$ we had the following assumptions. First, naturally we need a method which do not over-reacts severely. Hence we selected an exponential measure which punishes high over-reactions radically. Second, we would like fast congestion handling but basically any value, say, below 200 ms is fairly acceptable. Hence in this region we have a linear relation between handling time and $H(.)$. However, if resolving the congestion takes more time the utility of such configurations is exponentially degrading.

The resulting wellness function is shown in Fig. 9[3]. It shows that the preferred setups should be lower S values with more memory cells. However, we should focus on the lower left edge of the highest plane. At this edge we have those configurations that need the least amount of memory cells and at the same time

[3] Note that this figure has a different angle from previous ones.

can have the highest possible measurement period. This way, we get for example, that if the maximal RTT is 50 ms, then, e.g., the S=50 ms and N=2 or S=25 ms and N=3 configuration reflect two good setups where handling time is below 200 ms and no over-reaction evolves. This way, using the worst-case calculations along with the wellness function we can select an appropriate S and N value based on the estimated $\mathrm{RTT}_{\max}$which will yield a proper configuration for the proposed congestion handling method.

5 Conclusion

As the last performance gap between stateful and reduced-state resource management protocols we addressed the issues of failure recovery with reduced-state resource management protocols, specifically with the RMD-based operation mode of the QoS-NSLP protocol, which is currently being specified in the NSIS working group of the IETF. We highlighted the problem of severe congestion as the result of re-routing, and we introduced a low complexity solution. We also investigated the efficiency of the proposed method with worst case calculations.

The given calculations can be used to configure the parameters of the severe congestion handling algorithm. As a guideline we derived that the ratio of the measurement period (S) and the RTT of the flows has a great influence on congestion handling precision. We found that fast handling time can be achieved when $\frac{S}{RTT} \leq 1$. However, in this case over-reaction (more flows are terminated than necessary) occurs. Introducing a small local memory in the handling algorithm reduces and even eliminates the undershoot without increasing handling time. Using our method and dimensioning equations a low severe congestion handling time without any undershoot can be achieved.

References

1. Hancock, R., Freytsis, I., Karagiannis, G., Loughney, J., den Bosch, S.V.: Next steps in signaling: Framework. Internet Draft draft-ietf-nsis-fw, IETF (2004) Work in progress.
2. den Bosch, S.V., Karagiannis, G., McDonald, A.: NSLP for quality-of-service signaling. Internet Draft draft-ietf-nsis-qos-nslp, IETF (2004) Work in progress.
3. Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.: Resource reservation protocol (RSVP) – version 1 functional specification. RFC 2205, IETF (1997)
4. Báder, A., Westberg, L., Karagiannis, G., Kappler, C., Phelan, T.: RMD-QOSM - the Resource Management in Diffserv QoS model. Internet Draft draft-ietf-nsis-rmd, IETF (2005) Work in progress!
5. Karagiannis, G., Báder, A., Pongrácz, G., Császár, A., Takács, A., Szabó, R., Westberg, L.: RMD – a lightweight application of NSIS. In: Proc. of the Networks 2004, Vienna, Austria (2004)
6. Westberg, L., Császár, A., Karagiannis, G., Marquetant, A., Partain, D., Pop, O., Rexhepi, V., Szabó, R., Takács, A.: Resource management in diffserv (RMD): A functionality and performance behavior overview. In: Proc. of PfHSN 2002, Berlin, Germany, (2002)

7. Császár, A., Takács, A.: Comparative performance analysis of RSVP and RMD. In: Proc. of the QoFIS 2003, Stockholm, Sweden, (2003)
8. Császár, A., Takács, A., Szabó, R., Rexhepi, V., Karagiannis, G.: Severe congestion handling with resource management in diffserv on demand. In: Proc. of the Networking 2002, Pisa, Italy, (2002)
9. Császár, A., Takács, A., Szabó, R., Henk, T.: State correction after re-routing with reduced state resource reservation protocols. In: Proc. of the Globecom 2004, Dallas, TX, USA, (2004)
10. Pan, P., Schulzrinne, H.: YESSIR: a simple reservation mechanism for the internet. ACM SIGCOMM Computer Communication Review **29** (1999)
11. Ramakrishnan, K., Floyd, S., Black, D.: The addition of explicit congestion notification (ECN) to IP. RFC 3168, IETF, Network WG (2001)