Severe Congestion Handling with Resource Management in Diffserv on Demand

András Császár[1,2], Attila Takács[1,2], Róbert Szabó[1,2], Vlora Rexhepi[3], and Georgios Karagiannis[3]

[1] *N*et*L*ab, Ericsson Research HUNGARY
{robert.szabo, andras.csaszar, attila.takacs}@eth.ericsson.se
[2] High Speed Networks Laboratory,
Department of Telecommunications and Telematics,
Budapest University of Technology and Economics,
{robert.szabo, andras.csaszar, takacs}@ttt.bme.hu,
[3] ELN, Ericsson EuroLab Netherlands
{vlora.rexhepi, georgios.karagiannis}@eln.ericsson.se

Abstract. Quality of Service (QoS) for the Internet has been discussed for a long time without any major breakthrough. There are several reasons, the main one being the lack of a scalable, simple, fast and low cost QoS solution. A new QoS-framework, called resource management in differentiated services (RMD), aims to correct this situation. This framework has been published in recent papers and is extending the IETF differentiated services (diffserv) architecture with new admission control and resource reservation concepts in a scalable way. This paper focuses on proposing and investigating two resource reservation solutions on the problem of severe congestion situation within a diffserv-aware network utilizing an admission control scheme called Resource Mananagement in Diffserv (RMD). The different severe congestion solutions are compared using extensive simulation experiments.

1 Introduction

Internet QoS has been the most challenging topic of the networking research for several years now. The two existing Internet Protocol (IP) quality of service (QoS) architectures, Integrated Services (intserv) and Differentiated Services (diffserv) [1] are the results of the research work in this area.

Currently, the increasing popularity of the Internet as well as the growth of mobile communications have driven the development of IP-based solutions for wireless networking. The introduction of IP-based transport in radio access networks (RANs) is one of these networking solutions. When compared to traditional IP networks, an IP-based RAN has specific characteristics (see e.g. [2]) that impose stricter requirements on resource management schemes. Independently of the transport network, the cellular user expects to get the same service as in STM-based transport networks. In addition to this requirement, the situation is further complicated by the fact that the RAN is large in terms of its geographic size and the number of inter-connected nodes (hundreds or even thousands of nodes) with high cost of leased transmission lines, and the proportion of real-time traffic may get up to 100%. Resource management and CAC schemes working in IP-based RANs will have to enable dynamic admission control, fast resource reservation and at the same time they need to be simple, have low cost and easy to implement along with good scalability properties.

This paper focuses on proposing and investigating with simulation experiments two resource reservation solutions on the problem of severe congestion situation within a diffserv-aware network utilizing an admission control scheme called Resource Mananagement in Diffserv (RMD) [3] described in the following section. Severe congestion can be considered as an undesirable state, which may occur as a result of a route change or a link failure. Typically, routing algorithms are able to adapt to reflect changes in the topology and traffic volume. In such situations the re-routed traffic will traverse a new path. Nodes located on this new path may become overloaded, since they suddenly might need to support more traffic than their capacity. Moreover, when a link fails, traffic passing through it may be dropped, degrading its performance.

The rest of the paper is organized as follows: Section II lists related work in the resource management field. The severe congestion requirements and solutions are described in Section III. Section IV presents the simulation experiment results and their analysis. Finally, Section V concludes.

2 Related Works

Resource provisioning and traffic control algorithms use a signaling protocol to communicate the resource needs from end systems to routers, which either rely on information collected by measurements [4,5] or maintain some sort of reservation state. Generally, one group of approaches requires from every network entity to maintain per-flow state related information [6,7]. Another broad class of algorithms does not require per flow state related information, but is rather maintaining aggregated states in network core nodes, e.g., [8]. These mechanisms generally assume soft reservation status in the network, and either aim to periodically update it or try to harmonize the actions of routers along the path or take an economic approach to handle congestion [9].

2.1 Resource Management in DiffServ – RMD Framework

Currently, none of the available existing approaches satisfy the requirements for an appropriate resource management scheme within an IP-based RAN. In several recent papers [10,11] and IETF drafts [3,12] a new QoS framework, called Resource Management in DiffServ (RMD), is specified that aims to correct this situation. RMD extends the diffserv architecture with dynamic admission control and resource provisioning, and has good scaling properties and as such has low cost of implementation. Moreover, this framework has a wide scope of applicability in different types of diffserv networks.

In compliance with diffserv concepts, the RMD framework distinguishes between the problem of a complex reservation within a domain and handling a simple reservation within a node. Accordingly, there are two types of protocols defined within the RMD framework, the Per Domain Reservation (PDR) and Per Hop Reservation (PHR) protocol groups. *Per Domain Reservation* (PDR) is implemented only at the edges of the RMD domain and it handles the resource management in the entire diffserv domain. *Per Hop Reservation* (PHR) is used to perform resource reservation per diffserv class or Per Hop Behaviour (PHB) in each node of the diffserv domain. PHR aware nodes are not able to differentiate between individual traffic flows, as for e.g., RSVP, because no per-flow information is stored and the packet scheduling is done per aggregate. This way, PHR is optimized to reduce the functionality requirements of interior nodes.

In the following, we describe the simplified PHR operation. Before a new user data flow is admitted into the domain on one of the ingress edge nodes, it first has to signal its resource requirement (*QoS Request*). The ingress node classifies it into an appropriate PHB. These resource requests are transformed to discrete bandwidth values. Then the ingress edge node sends a *PHR Resource Request* packet to the egress edge, which is marked by any of the intermediate routers if they have not enough free resources. The egress edge node reports the reservation status back to the ingress, as a result of which the ingress can admit or reject the QoS request. If the flow is admitted, then periodic reservation refreshes are sent between the ingress and egress edge nodes.

The RMD framework [3] defines two different PHR groups: the reservation-based and the measurement based groups, which differ in the method a core node determines whether to mark a resource request packet, along with some signaling needs for this purpose. Here, we solely focus on the reservation-based PHR methods, where nodes maintain a per PHB reservation state. This is accomplished by using a combination of the reservation soft state and the explicit release principles. This means that the reserved resources can be released either when they are not refreshed regularly (1 refresh packet in every *PHR refresh period*), or when they are explicitly released by a predefined release message. In order to decrease the signaling traffic load on the network, the number of PHR refresh messages has to be minimized. Therefore, the PHR refresh period has to be chosen as large as possible, e.g., 30 seconds. The admittance decision is based on a threshold of maximum available resource units set for each PHB. Currently, there is one reservation-based PHR protocol defined, the Resource Management in Diffserv On DemAnd (RODA) protocol specified in [12].

3 Severe Congestion

3.1 Problem Definition and Requirements

Severe congestion can be considered as an undesirable state that may occur as a result of a route change or a link failure. The severe congestion situation will severely degrade the performance of the real time traffic and therefore, it has to be detected and solved very fast. Typically, in a RAN where majority of traffic is real-time traffic, the severe congestion situation has to be detected by the ingress edges within one second. Subsequently, the ingress edge has to undertake predefined policing actions to lower the incoming traffic volume in order to solve the severe congestion situation. The severe congestion solution can be decomposed in four subsequent phases:

- *Detection of severe congestion by interior nodes:* An RMD interior node has to detect the severe congestion situation using one of the following methods:
  - Volume measurements: by using measurements on the data traffic volume. If the volume of the data traffic increases suddenly, it is deduced that a possible route change and at the same time, a severe congestion situation occurred.
  - Counting: using a counter that counts the number of dropped data packets. The severe congestion state is activated when this number is higher than a pre-defined threshold. This method is similar to the previous one but is much simpler. However, it can only be applied when the traffic characteristics are known.
  - Increased number of refreshes: if the number of resource units per PHB, refreshed by PHR refresh messages is much higher than the number of resources refreshed previously, then the node deduces that a severe congestion occurred. This procedure is very efficient, but it can only be used when the PHR refresh period is small.

  The first three detection methods can be applied on both RMD schemes, i.e., reservation-based and measurement-based. The last method can only be applied on the reservation-based RMD scheme.

- *Propagation of severe congestion state to egresses*: In this phase, an interior node notifies the severe congestion situation to an egress node. Due to the fact that the interior node does not store and maintain any flow related information, it is not possible to identify the ID of the passing flow and the IP address of the ingress node. Therefore, the interior node is not able to directly notify the ingress node that a severe congestion situation occurred. One of the following methods is applied:
  - Greedy marking: all packets which are passing through a severe congested interior node and are associated to a certain PHB will be somehow remarked to indicate severe congestion;
  - Proportional marking: this method is similar to the previous method, with the difference that the number of the remarked packets is proportional to the detected overload;
  - PHR message marking: only PHR signaling messages that are passing through a severe congested interior node will be marked. The marking is done by setting a special flag in the protocol message, i.e., "S" (see [12]). This procedure is efficient, but it can only be used when the PHR refresh period is small.

  The last method can only be applied on the reservation-based scheme, while the other two can be applied on both RMD schemes.

- *Egress to ingress state propagation:* In this phase, the egress node has to process the severe congestion information received from the interior nodes. Moreover, it notifies the ingress node that a severe congestion occurred. The type of the received severe congestion information depends on the propagation method used by the interior nodes (see above). When either the "greedy marking" or the "PHR message marking" method is used, the severe congestion information simply notifies the egress node that a severe congestion situation occurred. When the "proportional marking" method is used, the egress node is informed that a percentage of the incoming traffic is overloading a certain communication path. The egress node forwards this information to the proper ingress node for all congestion marked flows.

- *Ingress actions on severe congestion:* the ingress node processes the severe congestion information received from the egress node and undertakes certain actions to solve the severe congestion situation. These actions depend on the method used by the interior nodes. One of the following set of actions can be undertaken:
  - Re-allocation: an ingress node is blocking new traffic flows and re-initiates all on-going flows that are affected by severe congestion. During the re-allocation procedure the ingress nodes will temporarily release and subsequently re-initiate all on-going flows that were affected by severe congestion.
  - Stochastic blocking: an ingress node is blocking new traffic flows and is terminating some of the on-going flows based on a probabilistic competition. The termination probability of a connection is proportional to its severe congestion marked traffic volume.

  The first method is applied when either the "greedy marking" or "PHR message marking" procedure is used. The later can only be applied when the "proportional marking" detection procedure is used.

3.2 Approaches to Handle Severe Congestion

In all of the consecutive approaches we commonly assume the followings: *i*) interior nodes use the *counting* detection method. In particular each interior node performs packet drop ratio measurements for every S interval per DSCP; *ii*) ingress nodes maintain per flow information that includes the flow ID, the requested amount of resources, i.e., bandwidth units; *iii*) egress nodes maintain per flow information that includes the flow ID and the IP address of the ingress node; *iv*) each node is capable of remarking each standardised DSCP, into locally defined DSCPs in order to signal severe congestion; *v*) egress nodes are checking the DSCP field of each passing data packet in order to identify its severe congestion status.

Solution A – Re-allocation of resources. The main characteristics (Fig. 1) of this scenario are that each interior node uses the *greedy marking* procedure and

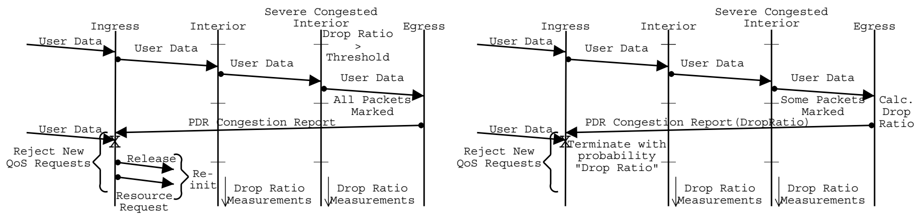


Fig. 1. Solution A and B Operation

each ingress node uses the *re-allocation* procedure to solve the severe congestion situation. Whenever the measured drop ratio in an S interval is higher than a pre-configured threshold value, the interior node deduces that a severe congestion occured. It remarks the DSCP field of all passing packets into a locally defined DSCP to indicate severe congestion. The egress nodes monitor the DSCP fields of each passing data packet. If re-marked DSCP fields are detected, then the egress node will deduce that a severe congestion occured. For each affected flow, the egress node will report the severe congestion situation to the corresponding ingress node by using a signaling PDR congestion report message.

When the ingress node receives a PDR congestion report message from the egress, it will block new incoming flows for a certain amount of time. Moreover, the on-going flows that are affected by severe congestion have to re-initiate their reservations: they must temporarily terminate their user data flow, deallocate their reservation with an explicit *release request* message, and try to reallocate their original resource usage along the path. Note that the *release request* and *reservation request* messages are not necessarily sent immediately after congestion notification as we will show later on, though user data must immediately be terminated.

Interior nodes receiving a *release request* message, decrease their aggregate reservation states with the number of units indicated in the corresponding message but not below zero. Unfortunately, in the case of a link failure and re-route event, flows previously accommodated on different paths will try to release resources on links where they have not yet allocated. In order to cope with this problem, interior nodes are disallowed to release more resources than previously allocated. Upon receiving a *reservation request* message, the interior node must see whether the sum of already existing reservations plus the new request is within a threshold. If so, the interior node updates its reservation state or else it marks the packet to indicate reservation failure.

The new reservation might be admitted by all interior nodes and signaled back by the egress to restart the user data. However, if the re-initiated allocation fails or time-outs (one may allow several retries before permanently terminating any connections) then the connection must be terminated by the end hosts.

Therefore, the *ingress node* must initiate the final termination of the flow at the end host.

Solution B – Stochastic (distributed) blocking. The main characteristics (Fig. 1) of this scenario are that each interior node uses the *proportional marking* procedure and each ingress node uses the *stochastic blocking* procedure to solve the severe congestion situation.

When the *ingress node* receives the PDR congestion report message, it will block new incoming flows for a certain amount of time. Moreover, it will also terminate some of the ongoing connections: it will realize a probabilistic drop on each individual flow that has received congestion report message according to the following formulas:

- Algorithm B1: $P_{\text{drop}}^{\text{B1}} = \frac{\text{\# of marked bytes}}{\text{\# of marked bytes+\# of unmarked bytes}}$. The underlying idea here is to purely base the blocking estimation on measured data. In this algorithm the blocking probability per connection (per flow) is calculated as the ratio between the dropped bytes and the maximum number of bytes that can be supported by the interior node (dropped / received).
- Algorithm B2: $P_{\text{drop}}^{\text{B2}} = \frac{\text{\# of marked bytes}}{rS_e8}$, where r [bps] is the allocated rate for the connection, S_e [sec] is the time base used at the egress edge and 8 is for bit/byte conversion. This version aims to eliminate the packet drops of connections by using the administrated reservations (dropped / sent-administrative).
- Algorithm B3: $P_{\text{drop}}^{\text{B3}} = \frac{\text{\# of marked bytes}}{2*\text{\# of marked bytes+\# of unmarked bytes}}$. Here, the blocking probability per connection (per flow) is calculated as the ratio between the dropped bytes and the total volume of user data (associated to the same connection), entering the RMD domain (dropped / sent-measured).

4 Numerical Results

In this paper we compare the severe congestion solutions described in Section 3 by using performance evaluation with the help of simulations. For these simulation experiments we used the network simulator (ns) [13] environment. This section describes the used traffic models, network topology, the performed experiments and their results.

4.1 Simulation Model

Traffic models: Based on the operational description of the RMD protocol, resource demands are handled in bandwidth units, which we will also use when describing our traffic models. First of all, one unit was set to represent 2000 bytes/second rate allocation. The reason behind was that one unit should represent the rate required by an encoded voice communication, e.g., GSM coding. Altogether we examined three different scenarios where i) calls requested only 1 units homogeneously, where ii) calls requested bandwidths of $\{1, 2, \ldots, 20\}$

units and iii) where call demands were selected from $\{1, 2, \ldots, 100\}$ units. Calls arrived according to a Poisson process with parameter λ_i for calls requesting i units of bandwidth. The average call holding time was set to $\frac{1}{\mu} = 90$ seconds. The call and bandwidth unit requests were generated in a way that the demanded load for each reservation unit class was eqalized on the average, or more formally $\frac{\lambda_i}{\mu} BW_i = \frac{\lambda_j}{\mu} BW_j$, where $\mathrm{BW}_i = i$[unit] is the bandwidth demand and $\frac{1}{\lambda_1} = 0.9$ sec as per default. Hence higher bandwidth demands arrived less frequently than smaller ones. This is not unrealistic since higher bandwidth requests are probably more expensive. Packet sizes (L) of the connections were determined according to their reservations in the following way: $L_i = 40$ [bytes]BW_i. For the sake of simplicity packet inter arrival times were kept constant (constant bit rate - CBR), hence every flow sent one single packet in each 20 msec interval.

Network topology: For the evaluation of the methods we used a simple delta topology with the $G(V, E)$ graph, where E denotes edges $\{e0, e1, e2\}$ and V denotes the vertices $\{(e0, e1), (e0, e2), (e1, e2)\}$. With the former re-routing and its effects are easily traceable. For the sake of simplicity only $e0$ generated traffic for the other two edges. In order to have effective multiplexing of flows, the capacity (C) of the links was set to be able to accommodate at least 100 flows of the highest bandwidth demands, i.e., to 100, 2‘000 and 100‘000 units.

As discussed earlier, the severe congestion detection is based on packet drop ratio measurements. Hence, it was important to find the proper dimensioning for the network buffers. As our traffic model was based on CBR traffic with 20 msec packet inter arrival times, we determined the queue lengths so that no packet loss can occur during normal operation. We dimensioned for a target load level of 80% link capacity, hence the buffer sizes (B) were determined using the following formula: $B = C * 0.02 * 0.8$ [bytes].

Network events: In our simple delta network after the system achieves stationarity, the link between nodes $e0$ and $e2$ goes down at 350 sec of simulation time. Afterwards, the dynamic routing protocol (OSPF) updates its routing table at 352.0 sec and all flows previously taking the $e0 - e2$ path will be re-routed to the $e0 - e1 - e2$ alternate path. Hence, node $e0$ will suffer a serious overload resulting from the re-route event.

4.2 Numerical Evaluation

Network utilization: It can be seen in Fig.2/a that with solution A, more reservation messages were accepted by the severe congested node during the re-initiation procedure than the target admission threshold (80%). This phenomenon is due to certain properties of the protocol related to the explicit release of soft state resources, whose basic idea is discussed in [11]. Nevertheless, the above problem can briefly be reasoned by the following operation:

First of all, in order to achieve better utilization the soft state refresh period (T), which is in all cases set to 30 sec, is sub-divided into cells (10), where a sliding (or time) window algorithm is used to smooth out the T long discrete time steps. If a re-route event happens in the system after the link failure, all of the traffic originally traversing along a different path will flow through the single operating link. This will evidently make severe congestion situation. Here however, not only data packets but protocol signaling packets (see [12]) are involved, which will affect the administrated reservations in the following way. With solution A, release messages of the re-initialization procedure (see section 3.2) will decrease the number of registered reservations, however not only the originally accommodated flows but also the re-routed ones will try to release their allocations. Hence the volume of release must be limited, which is done by permitting releases until the administered reservation state is above zero. Unfortunately, connections that have not yet been notified of severe congestion (longer round trip times (RTT)) will keep on sending their periodic reservation refreshes that increase the administered reservation. This affects the same reservation state that the release messages compete for, hence allowing more release and reallocation than desired. This overshoot will only leave the system after a refresh period (see it at around 382 sec in Fig. 2/a).

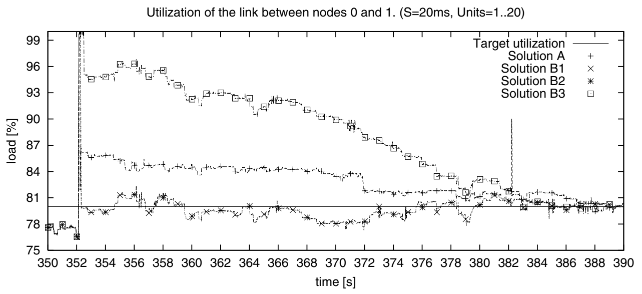
On the other hand, the descendant algorithms of solution B differ in the calculated call dropping probability (see section 3.2) and do not stop and re-allocate the connections but instead immediately drop some of them in proportion to the detected overload. This blocking probability is the smallest with algorithm $B3$, and highest with $B1$. It can be seen that -as expected- the lower the call blocking ratio is, the higher the maintained utilization is. Depending on the measurement time base (S), the retained utilization is above, upon or below the target level. It can be seen that with different measurement time bases, different drop ratio approximations perform best while solution A is almost indifferent to the measurement time base (see Fig. 2/a and b).

Fig. 3 shows short term transients of the algorithms. Solution A is shown in Fig. 3/a for three different measurement time bases. It is interesting to see that since user traffic had a well defined period of 20 msec, measurements with 50 msec time base introduced high level of oscillation. It can be also seen that it takes a couple of measurement periods to bring the load back around the desired level though the control time is still in the order of 100 msec.

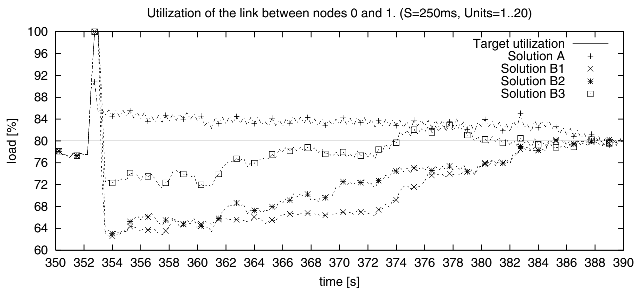
Solution B variants react very similarly (see Fig. 3/b) where $B1$ and $B2$ operations result in exactly the same transients.

Packet drop ratios: Here, only some representative results are presented due to the space limitations. As expected, the shorter the measurement time base is the shorter the congestion period will be (faster actions), hence packet drop periods decrease (see Fig. 4/a-b). The difference in operation between solution A and B can be seen when increasing the measurement windows. Since solution A stops user data flow its packet drop ratio more rapidly decreases with increasing measurement time base.

Signaling overheads: Fig. 4/c shows the protocol messages and the reservation status for solution A. It can be seen that after the detection of severe congestion all the 160% traffic load is released and tried to be reallocated (see the almost coinciding curves at 352 in Fig. 4/a). It can also be well seen that due to the synchronized re-initiation, refresh messages arrive in bursts. This

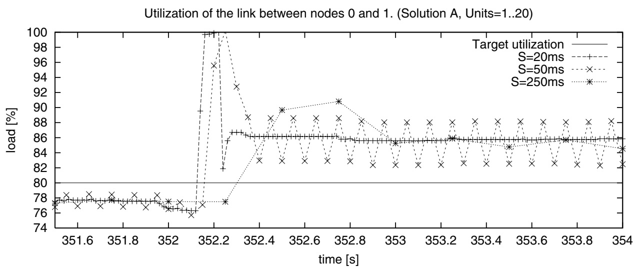


a) Utilization with $S = 20$ msec

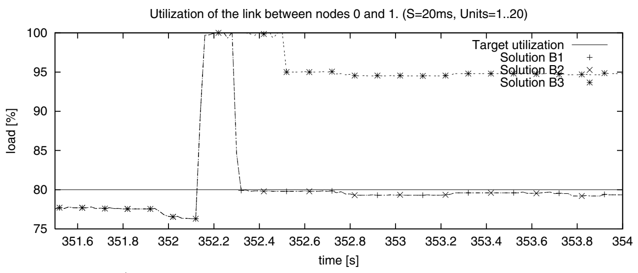


b) Utilization with $S = 250$ msec

Fig. 2. Utilization

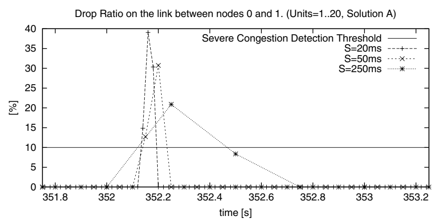


a) Short term transients for solution A

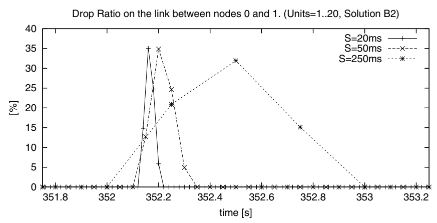


b) Short term transients for solution B descendants

Fig. 3. Short term transients

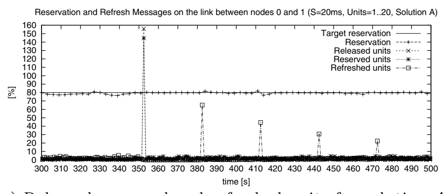


a) Solution *A*

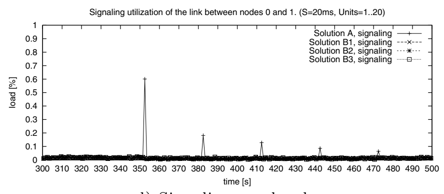


b) Solution *B*2

a-b) Drop ratios of the different algorithms



c) Released, reserved and refreshed units for solution *A*



d) Signaling overheads

c-d) Protocol messages

Fig. 4. Protocol performances

characteristic will only fade out with several refresh times, as connections are terminating by nature. Fig. 4/d shows the signaling overhead for the various algorithms. It is obvious that for solution *B* descendants there are no increase in signaling overhead as shown in Fig. 4/d. On the other hand, due to the excess signaling introduced by solution *A* the overhead bristles appear. This however, is still quite negligible compared to the link capacity (see Fig. 4/d).

5 Conclusions

In this work we have shown some aspects of severe congestion handling with the RODA protocol [12]. We have designed and presented two basic algorithms that could cope with severe congestion situations in the order of network round trip times. This reaction time can be considered as close to optimal due to the transmission constraints imposed by the system. The presented algorithms differ in their transients but we can conclude that two of our solution *B* derivatives performed best in all situations with measurement time base equal to the framing time of the data traffic. In this very special case the two algorithms resulted in the same operation due to the traffic characteristics and differences only appeared with higher measurement time bases. Overall, we are aware of the need for further analysis in this area with more general traffic models (e.g. VBR); with multiple traffic classes (e.g. voice, video and best-effort); with more complex network topology (e.g. a concrete RAN topology) and with a comparsion to other resource management protocols (e.g. RSVP). Nevertheless, we believe that our current results can already be applied to certain special networks like RANs. We suppose that these results will trigger new dialogs from the community.

Acknowledgments. The authors owe special thanks to the *RMD research team* located in Sweden, Netherlands and Hungary. This work has partially been done at the High Speed Networks Laboratory, at the DTT of BUTE.

References

1. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An architecture for differentiated service. Request for Comments 2475, Internet Engineering Task Force (1998)
2. Partain, D., Karagiannis, G., Wallentin, P., Westberg, L.: Resource reservation issues in cellular access networks. Internet Draft, Internet Engineering Task Force (2002) Work in progress.
3. Westberg, L., Jacobson, M., Karagiannis, G., Oosthoek, S., Partain, D., Rexhepi, V., Szabó, R., Wallentin, P.: Resource management in diffserv (RMD) framework. Internet-draft: draft-westberg-rmd-framework-xx.txt, Internet Engineering Task Force (2002) work in progress.
4. Elek, V., Karlsson, G., Ronngren, R.: Admission control based on end-to-end measurements. In: Proceedings of the Conference on Computer Communications (IEEE Infocom), Tel-Aviv, Israel (2000)
5. Breslau, L., Jamin, S., Shenker, S.: Comments on the performance of measurement-based admission control algorithms. In: Proceedings of the Conference on Computer Communications (IEEE Infocom), Tel Aviv, Israel (2000)
6. Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., Jamin, S.: Resource ReSerVation protocol (RSVP) – version 1 functional specification. Request for Comments 2205, Internet Engineering Task Force (1997)
7. Feher, G., Nemeth, K., Maliosz, M., Cselenyi, I., Bergkvist, J., Ahlard, D., Engborg, T.: Boomerang - a simple protocol for resource reservation in ip networks. In: IEEE Workshop on QoS Support for Real-Time Internet Applications, Vancouver, Canada (1999)
8. Baker, F., Iturralde, C., Faucheur, F.L., Davie, B.: RSVP reservations aggregation. Internet Draft, Internet Engineering Task Force (2001) Work in progress.
9. Gibbens, R.J., Kelly, F.P.: Resource pricing and the evolution of congestion control. Automatica **35** (1999) 1969–1985
10. Heijenk, G., Karagiannis, G., Rexhepi, V., Westberg, L.: Diffserv resource management in ip-based radio access networks. In: Wireless Personal Multimedia Communications (WPMC'01), Aalborg, Denmark (2001)
11. Ádám Marquetant, Pop, O., Szabó, R., Dinnyés, G., Turányi, Z.: Novel enhancements to load control - a soft-state, lightweight admission control protocol. In: to appear at QofIS2001 - 2nd International Workshop on Quality of future Internet Services, Coimbra, Portugal, COST263 (2001)
12. Westberg, L., Jacobsson, M., Karagiannis, G., Oosthoek, S., Partain, D., Rexhepi, V., Wallentin, P.: Resource management in diffserv on demand (RODA) PHR. Internet Draft, Internet Engineering Task Force (2001) Work in progress.
13. The network simulator - ns-2. (http://www.isi.edu/nsnam/ns/)