REDUCING LOAD DISTRIBUTION OVERHEAD WITH MESSAGE AGGREGATION

Vasil Hnatyshin and Adarshpal S. Sethi

Department of Computer and Information Sciences,
University of Delaware, Newark, DE 19716
{vasil, sethi}@cis.udel.edu

ABSTRACT

The current best effort approach to Quality of Service in the Internet can no longer satisfy a diverse variety of customer service requirements, because of which there is a need for alternative strategies. A promising approach for dealing with this problem is a method called Load Distribution Scheme (LDS) which dynamically adjusts traffic load at the network boundary based on feedback from the network. In order to fairly share available resources among individual flows, the load distribution scheme relies on a message exchange protocol which in certain cases may cause significant overhead in the system. In this paper, we examine the issues related to the problem of the message overhead in the LDS, propose solutions to the problem, and evaluate these solutions through simulation in OPNET.

Keywords: Quality of service, load distribution, network feedback, load overhead, message aggregation

1. INTRODUCTION

The current approach to providing Quality of Service in the Internet is no longer adequate because of the increasing emergence of applications with diverse customer service requirements. As people become willing to pay more for services that satisfy their application needs, the one-service-for-all approach of today's Internet will become obsolete, creating a need for alternative strategies. In order to tackle this problem, a number of service differentiation models have been

proposed. Integrated [3] and Differentiated [2] Service architectures introduced by IETF's IntServ and DiffServ working groups, core-stateless fair queuing [24], and proportional service differentiation framework [6-8] are currently among the most popular approaches. Unfortunately, these schemes often fail to provide proper service differentiation or may not be applicable to current networks. For example, Differentiated Services model may fail to provide fair resource allocation and fair service degradation during periods of congestion [9, 19, 21, and 23] because of static resource allocation. Integrated Service approach, on the other hand, guarantees per-flow QoS but does not scale well to large networks due to per-flow information stored in the network core. The proportional service differentiation model does not violate relative guarantees under any network condition. However in this model, the lack of mechanisms for limiting the amount of data injected into the network can reduce the absolute level of QoS below user expectations [6-8].

We believe that one way to deal with this problem is to introduce *a load distribution scheme* (LDS) [11 – 13] at the network boundary. The main objectives of the load distribution scheme are to satisfy minimum per-flow QoS guarantees and to fairly distribute excess resources among the flows. In particular, the LDS guarantees that each active flow in the network will receive at least its minimum requested amount of bandwidth.

In this paper, we extend work published in [11 – 13] and further study the performance of the LDS under more realistic conditions. In particular, we examine the behavior of the LDS in a network with a large number of small flows. Since the LDS relies on a message exchange protocol to distribute QoS requirements among individual nodes, it is expected that in a network with a large number of small flows the control message overhead will be proportional to the number of active flows. Subsequently, in such cases the message exchange protocol of the LDS might cause too much overhead in the system and would raise scalability concerns.

In this paper, we introduce a technique for reducing the total number of control message generated upon activation or termination of a flow. This approach, which we call *message aggregation*, merges multiple control messages into a single packet. We examine how effectively message aggregation reduces the overhead of the control message exchange and study its influence on the resource allocation by the LDS.

The rest of the paper is organized as follows. Section 2 provides a definition of fairness and an overview of the message exchange protocol of the LDS. In Section 3 we introduce the idea of message aggregation and we present its evaluation through simulations with OPNET in Section 4. Section 5 provides a related work overview and finally we conclude in Section 5.

2. THE LDS OVERVIEW AND DEFINITION OF FAIRNESS

In order to satisfy minimum per-flow guarantees and to provide fair resource allocation the load distribution scheme relies on the network feedback and admission control. When a new flow activates, the boundary node probes the network to determine if the new flow could be admitted into the network without violation of the minimum QoS requirements of the currently active flows. In this paper we will not discuss admission control. However, we imply that a new flow can enter the network only if there are enough resources to satisfy minimal QoS requirements of all the flows in the domain. Figure 1 illustrates the idea of the LDS. As the figure shows, traffic enters the network domain at the boundary router[1] B1, traverses the network in some fashion, and then exits this network domain at the boundary router B2. When a new flow activates or terminates, the boundary node advertises the change in the QoS requirements on the path. If congestion arises, the core routers distribute the aggregated QoS requirements through the congestion notifications sent to the network boundaries. Based on this feedback, the boundary routers fairly adjust the amount of traffic admitted into the domain.

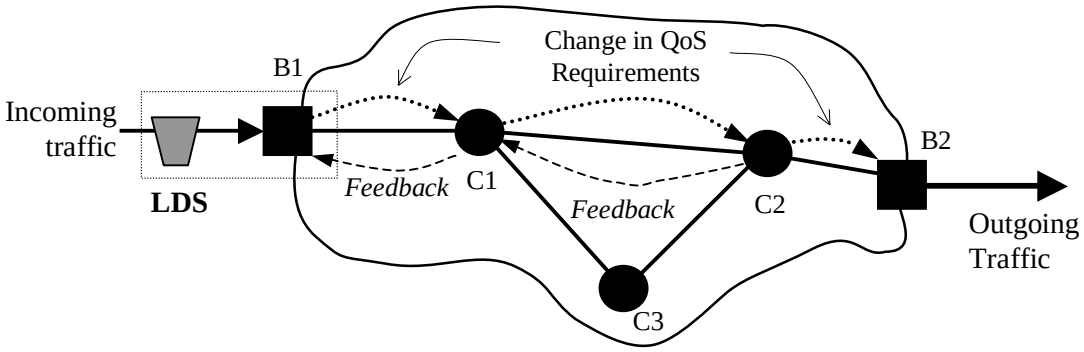

Figure 1. Scenario for Load Distribution Scheme

In order to determine a permissible sending rate of a flow, each boundary node maintains a ***R**equested **L**oad **R**ange*, $RLR = [b_f, B_f]$, for the flows that enter the network domain through it. A *flow's RLR* consists of two values: a minimum rate, b_f, below which the flow cannot operate normally and the maximum rate, B_f, that the flow can utilize. The flow's sending rate, R_f, is limited by its RLR and lies within this requested range. Throughout the paper we will often refer to numerous definitions of the RLR aggregates, which we define as follows.

In addition to the flow RLRs, each ingress node keeps track of the path RLRs. The *path RLR*, $[b_i^P, B_i^P]$, or the RLR of the ingress node i on the path P, is a load range where b_i^P corresponds

[1] Throughout this paper we will also refer to the boundary nodes at which traffic enters a domain as ingress routers and we will call the boundary nodes where traffic leaves a domain as egress routers.

to the sum of the minimum requested rates of the flows that originate from the ingress node i and traverse the path P, while B_i^P is the sum of the corresponding requested maximum rates. Using $f \to P$ to denote that flow f traverses path P, we define the path RLR as follows:

$$b_i^P = \sum_{\forall f | f \to P} b_f \qquad B_i^P = \sum_{\forall f | f \to P} B_f \qquad (1)$$

where only flows originating at boundary node i are considered.

Similarly, we define *interface* and *aggregated interface RLRs* for the core router interfaces. The interface RLR of interface k for the ingress node i, $\left[b_i^k, B_i^k\right]$, is the sum of the path RLRs of the ingress node i, subject to the condition that the paths include interface k.

$$b_i^k = \sum_{k \in P} b_i^P \qquad B_i^k = \sum_{k \in P} B_i^P \qquad (2)$$

To avoid confusion, we will use an upper-case letter (e.g. P) for a path and a lower-case letter (e.g. k) for an interface. Finally, the aggregated interface RLR of interface k, $\left[b^k, B^k\right]$, is the sum of its interface RLRs.

$$b^k = \sum_i b_i^k \qquad B^k = \sum_i B_i^k \qquad (3)$$

Ingress nodes obtain the flow RLRs from the service level agreements established with the user, and they compute the path RLRs based on these values. Each core interface obtains interface RLRs from the ingress node's advertisements and computes an aggregated interface RLR. Ingress nodes maintain information about individual flows (e.g. flow's RLR) and their corresponding paths (e.g. path RLRs), while the core routers maintain only per-ingress node information (e.g. interface RLRs). A more detailed overview of the data structures maintained in the ingress and core nodes is provided in [12, 13].

Congestion notification messages, which are sent by a congested core interface to ingress nodes, carry interface and aggregated interface RLRs. These values allow ingress nodes to fairly distribute available resources among individual flows. The fair shares on congested interface k of ingress node i and of flow f are computed as follows:

$$FS_i^k = \min\left(b_i^k + \left(C^k - b^k\right)\frac{B_i^k - b_i^k}{B^k - b^k}, B_i^k \right) \qquad (4)$$

$$FS^f = \min\left(b^f + \left(FS_i^k - b^f\right)\frac{B^f - b^f}{B_i^k - b_i^k}, B^f \right) \qquad (5)$$

where C^k is the capacity of the outgoing link on interface k, while FS_i^k and FS^f denote the fair shares of ingress node i and of flow f on congested interface k, respectively. Other alternative definitions of fairness within the framework of LDS were examined in [12, 13].

The message exchange protocol consists of three distinct phases. During the first phase, called *path probing*, the ingress node attempts to learn about the current state of a path or to learn the path itself if the route to the flow's destination is unknown. The probe messages collect the current arrival rate of the traffic and the aggregated interface RLR for each traversed link. The probe messages are generated either periodically or when a new flow is activated. Periodic probing is used to determine if the ingress node can increase its sending rate on the path. In the presence of excess bandwidth, the ingress node increases transmission rates of the flows that travel on the corresponding path proportionally to the individual flow's RLR. The path probing initiated due to the flow activation determines if the new flow can be admitted into the network.

Admission of a newly activated flow into the network or a flow termination initiates the second phase called the *RLR change* phase. The purpose of this phase is to update the interface RLRs along the flow's path. If the admission of the new flow causes congestion anywhere along the path, then the ingress node initiates the third phase, called *the Rate Reduction Phase*. During the third phase congested interfaces notify ingress nodes to slow down. Upon arrival of the congestion notification message the ingress nodes compute their corresponding fair shares and adjust transmission rates of individual flows accordingly. A special case occurs when the ingress node that transmits data at the rate higher than its fair share due to the presence of the excess bandwidth on the path, receives a congestion notification. In this case, the ingress node might not need to reduce the transmission rate to its fair share. Instead, the ingress node reduces its rate proportionally to the RLR change on the path.

The message exchange protocol uses the following message types. In the first phase, ingress nodes generate PROBE packets and receive results of the path probing via PROBE_REPLY messages. In the second phase, ingress nodes advertise changes using RLR_CNG packets. CN and CN_CORE messages are used during the rate reduction phase to convey information about congested interfaces to the ingress and core nodes respectively.

3. MESSAGE AGGREGATION

The load distribution scheme relies heavily on the message exchange protocol to distribute the interface and the aggregated interface RLR among the boundary nodes. In a network with a small number of flows, the overhead due to the message exchange protocol of the LDS is negligible as reported in [12 and 13]. In such networks, the major cause of the overhead is the

periodic path probing. Since the probe messages are infrequent and their sizes are significantly smaller than the average size of the data packet, the total overhead due to the control messages is very small. However, in a network with a large number of small flows that activate and terminate very frequently, the LDS scheme will constantly remain in the RLR change phase of the message exchange protocol. As a result, the ingress nodes would generate the RLR_CNG packets for each flow activation or termination, which may cause a significant overhead. Furthermore, numerous RLR_CNG messages may cause frequent changes of the congestion status in the network, which would subsequently result in additional overhead due to congestion notification messages.

The key to reduction of the message exchange overhead is to limit the number of the RLR_CNG messages. To do that, the boundary nodes should combine frequent updates of the RLR information by carrying the information about the multiple requests for flow activation or termination in a single RLR_CNG message. We will call this technique *message aggregation.*

It should be noted that if the boundary nodes do not generate an RLR_CNG message upon each flow activation or termination, then the interface and aggregated interface information stored in the network core would not be accurate, which in turn may influence the fairness of the LDS. In fact, the message aggregation technique reduces the overall overhead due to the control messages at the cost of violating strict guarantees of the fair resource distribution of the LDS. Because of that, we conclude that the ingress node's fair share on that path is one of the parameters that determine if the flow's request for activation or termination could be aggregated in subsequent RLR_CNG messages.

Let us consider the situation when the boundary node i receives a request from the flow f to be activated on the path P, with the bottleneck link k. In this case, the boundary node i should compute and compare its fair shares on the path P, for the case when the RLR_CNG message was generated and when the flow's request was aggregated (e.g. RLR_CNG message was not generated):

$$ {}^{noAGGR}FS_i^k = \min\left(b_i^k + b^f + \left(C^k - b^k - b^f\right)\frac{\left(B_i^k - b_i^k\right) + \left(B^f - b^f\right)}{\left(B^k - b^k\right) + \left(B^f - b^f\right)}, \left(B_i^k + B^f\right) \right) \quad (6) $$

$$ {}^{AGGR}FS_i^k = \min\left(b_i^k + \left(C^k - b^k\right)\frac{B_i^k - b_i^k}{B^k - b^k}, B_i^k \right) \quad (7) $$

where, ${}^{noAGGR}FS_i^k$ is the fair share of the ingress node i on the path P in the case when the ingress node advertises RLR change on the path and ${}^{AGGR}FS_i^k$ is the fair share of the ingress node i on the path P when the flow's request was aggregated. It should be noted that if the

request for the flow activation was aggregated then the flows that follow the path P receive the amount of resources slightly below their corresponding fair shares. On the contrary, if the request for the flow termination was aggregated, then the flows that follow the path P receive the amount of resources slightly above their corresponding fair shares. If the deviation from the flow's fair share is within an acceptable range, then the node i should aggregate the flow's request in the subsequent RLR_CNG messages. Otherwise the ingress node i should initiate the RLR change phase.

This is an example of the simplest message aggregation policy that relies only on the ingress node's fair shares to determine if the flow's request could be aggregated. However, the boundary nodes are allowed to implement more complex message aggregation policies that would include the network status or other parameters in the decision making process.

4. EVALUTION OF THE MESSAGE AGGREGATION FOR THE LDS

4.1 SIMULATION SETUP

To study and evaluate the performance of message aggregation, we performed a simulation study using the OPNET network simulator [17]. The goal of the simulation study was to examine how the message aggregation technique influences the overhead in the system as well as to investigate its effects on the fairness of the LDS. In order to study how the message aggregation influences the fairness of the resource distribution in the network, we introduce a new term called *degree of fairness* defined as follows.

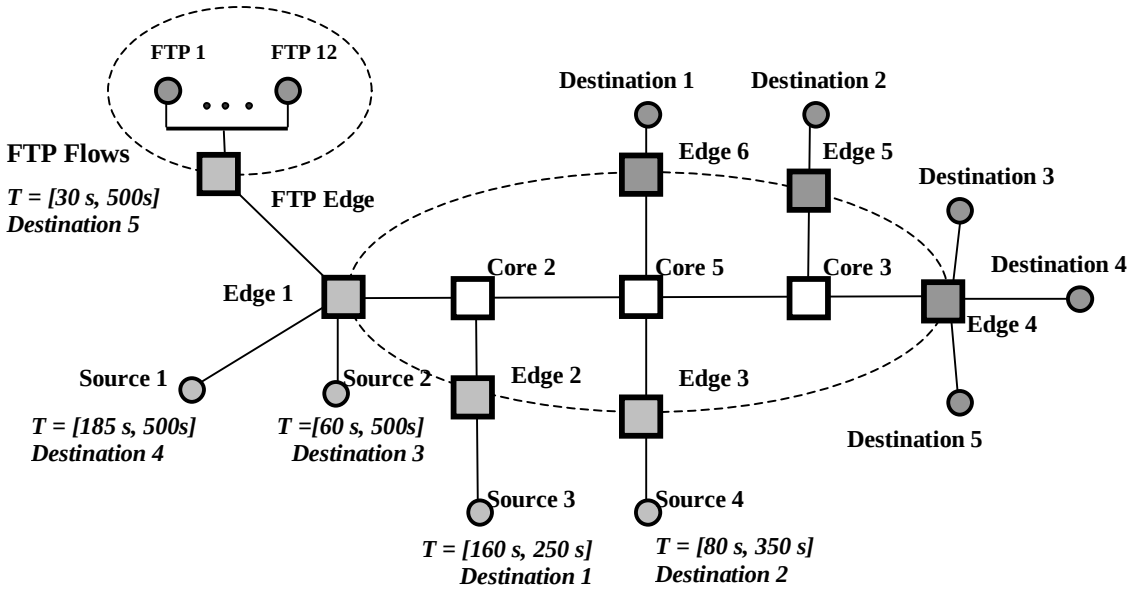

Figure 2. Simulation Topology

We will refer to the fair shares of the boundary node i on the path P at the time τ with and without the message aggregation as ${}^{ma}FS_i^P(\tau)$ and $FS_i^P(\tau)$, respectively. Then, the degree of fairness of the boundary node i on the path P at the time τ is defined as:

$$DF_i^P(\tau) = \left| 1 - \frac{FS_i^P(\tau)}{{}^{ma}FS_i^P(\tau)} \right| \qquad (7)$$

It should be noted that with this definition, a low degree of fairness (e.g. less than 5%) indicates that the resources are shared fairly among the flows, while a high degree of fairness indicates unfair sharing.

To study the performance of the message aggregation we used the topology of Figure 2 that shows the flow's activation/termination schedule and the point of destination for each flow. For example, the flow of Source 1 activates at time 185 seconds and travels to node Destination 4, while the flow of source 3 activates at time 160 seconds, terminates at time 250 seconds, and travels to Destination 1. The duration of the simulation was 500 seconds.

Flow Numbers	***Flow's Activation/ Termination Schedule***	***Flow's RLR***	***Ingress Node***
FTP 1-3	[30 s, 500 s]	[20 Kbps, 50 Kbps]	Edge 1
FTP 4-6	[30 s, 500 s]	[40 Kbps, 50 Kbps]	Edge 1
FTP 7-10	[30 s, 500 s]	[10 Kbps, 30 Kbps]	Edge 1
Video 1	[185 s, 500 s]	[400 Kbps, 1200 Kbps]	Edge 1
Video 2	[60 s, 500 s]	[200 Kbps, 1000 Kbps]	Edge 1
Video 3	[160 s, 250 s]	[800 Kbps, 2000 Kbps]	Edge 2
Video 4	[80 s, 350 s]	[500 Kbps, 1300 Kbps]	Edge 3

Table 1. Flow specification

In the simulation we used two types of applications: FTP and video traffic. FTP flows are small, short lived flows that activate and terminate very frequently. Each FTP flow randomly activates multiple times during the simulation and remains active for a random duration not longer than 30 seconds. FTP flows use TCP as their transport protocol. Video traffic consists of the large, long-lived flows that use UDP as the transport protocol. Each video flow activates only once during the simulation and remains active according to the schedule shown in Table 1. All the video traffic in the simulation is bi-directional; however, in order to avoid unnecessary confusion we will not discuss video traffic that travels from the boundary nodes Edge 4, Edge 5, and Edge 6 to their corresponding destinations.

4.2 LOAD DISTRIBUTION USING LDS

Let us examine the bandwidth allocation by the LDS in greater detail. The load distribution among ingress nodes for the scenario defined by Figure 2 and Table 1 is shown in Figures 3 – 5.

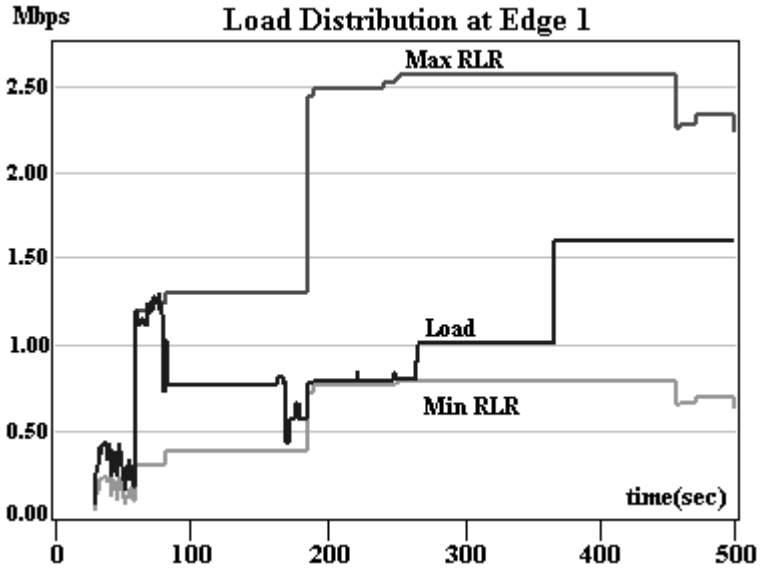

Figure 3. Load Distribution at boundary node Edge 1

At time 30 seconds the FTP flows begin to activate in a random fashion. At time 60 seconds flow Video 2 activates followed by activation of flow Video 4 at time 80 seconds. At this point the link between the nodes Core 5 and Core 3 becomes congested which forces the edge nodes 1 and 3 to throttle transmission rates of their corresponding flows. At time 160 seconds flow Video 3 activates which shifts the bottleneck for ingress node Edge 1 to link Core 2 – Core 5. As a result, ingress nodes Edge 1 and Edge 2 adjust transmission rates of their flows according to aggregated interface RLR on link Core 2 – Core 5, while ingress node Edge 3 benefits from the excess bandwidth created by throttling the flows of ingress node Edge 1.

All the active flows in the network adjust their transmission rates according to their corresponding bottleneck links upon activation of the flow Video 1 at time 185 seconds. However, after all the flows have adjusted their transmission rates, link Core 5 – Core 3 becomes underutilized which enables flow Video 4 to benefit from the excess bandwidth.

In should be noted that frequent activations of the FTP flows often cause congestion on both bottleneck links Core 2 – Core 5 and Core 5 – Core 3. As a result, all the active flows in the network adjust their transmission rates upon activation of the FTP flows. However, since link Core 5 – Core 3 does not limit transmission rate of the FTP flows, flow Video 4 need not reduce its transmission rate to its fair share on the link Core 5 – Core 3. Instead, flow Video 4 reduces its transmission rate proportionally to the RLR of the newly activate FTP flow.

As long as link Core 2 – Core 5 remains the bottleneck for the traffic from ingress node Edge 1, link Core 5 – Core 3 will contain excess bandwidth and flow Video 4 would utilize it. If upon reception of every CN message, flow Video 4 adjusts its transmission rate to the corresponding fair share then later it would increase its sending rate because of the excess bandwidth available on the link Core 5 – Core 3, resulting in unnecessary load fluctuations. Thus, upon CN message arrival, the boundary node may reduce its transmission rate proportionally to the RLR change on the congested interface, instead of sending traffic at its fair share.

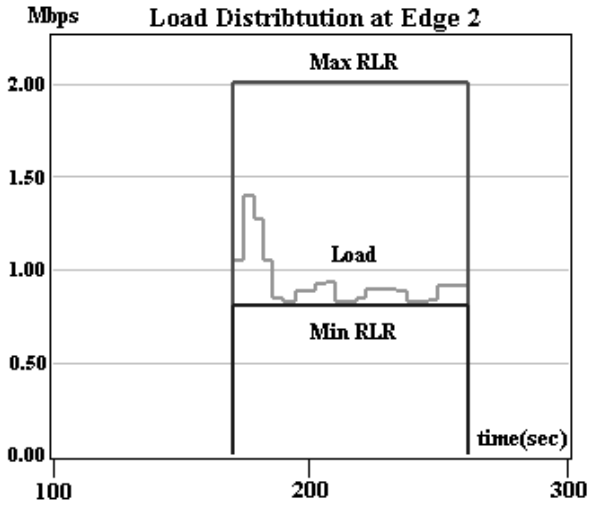

Figure 4. Load Distribution at the boundary node Edge 2

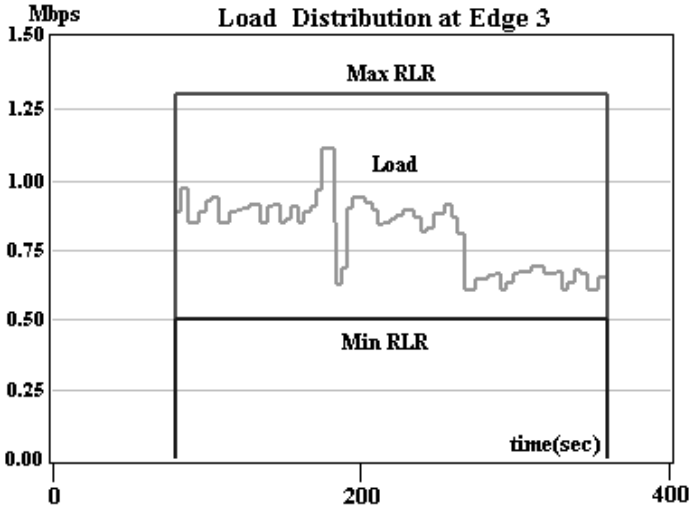

Figure 5. Load Distribution at the boundary node Edge 3

At time 250 seconds flow Video 3 terminates which causes the bottleneck for the traffic from Edge 1 to shift back to link Core 5 – Core3. Consequently, all the active flows adjust their transmission rates accordingly. In particular, flow Video 4 adjusts its sending rate to its fair share in order to accommodate traffic from Edge 1. Finally, at time 350 seconds, flow Video 4 terminates and traffic from Edge 1 utilizes all the available bandwidth on the path.

It should be noted that throughout the simulation each active flow receives an amount of bandwidth that is within its requested load range. Furthermore, the per-flow bandwidth allocation in the network satisfies (within small error limits) the fairness criteria defined by equation 5. In the next section we examine the influence of the message aggregation on the fairness of the load distribution by the LDS.

4.3 EVALUATION OF THE MESSAGE AGGREGATION

To evaluate the message aggregation technique and its influence on the fairness of the load distribution, we implemented the following aggregation policies for the scenario of Figure 2. As

mentioned before, the goal of the message aggregation is to reduce the total number of control messages (RLR_CNG) generated upon the flow activation or termination. The following message aggregation rules specify under what conditions the RLR_CNG message should be generated and when it should be aggregated.

Rule 1. Always generate RLR_CNG message upon activation or termination of the video flow.

Rule 2. Always generate RLR_CNG message if the flow's activation does not cause congestion.

Rule 3. Generate RLR_CNG message if upon the activation/termination of the FTP flow, the ratio between ${}^{noAGGR}FS$ and ${}^{AGGR}FS$ is larger than the aggregation threshold, where, ${}^{noAGGR}FS$ is the fair share of the FTP flows on the path when the RLR_CNG message was sent and ${}^{AGGR}FS$ is the fair share of the FTP flows on the path when the flow's request was aggregated.

Rule 4. Otherwise do not generate control message.

We examined the reduction in the total number of RLR_CNG messages and the variation of the degree of fairness at the ingress node Edge 1 by changing the value of the aggregation threshold from 30% to 80%. Each scenario was executed 10 times and the averaged results presented in Figures 6 – 9. As expected, the reduction in the total of the RLR_CNG messages and the degree of fairness both increase as the aggregation threshold becomes larger. When the aggregation threshold increases, a larger number of flow requests are being aggregated, as a result of which fewer RLR_CNG messages are generated. Subsequently, the core nodes contain and advertise a less accurate value of the aggregated interface RLR, which influences the accuracy of the load distribution and causes the degree of fairness to increase (implying less fairness).

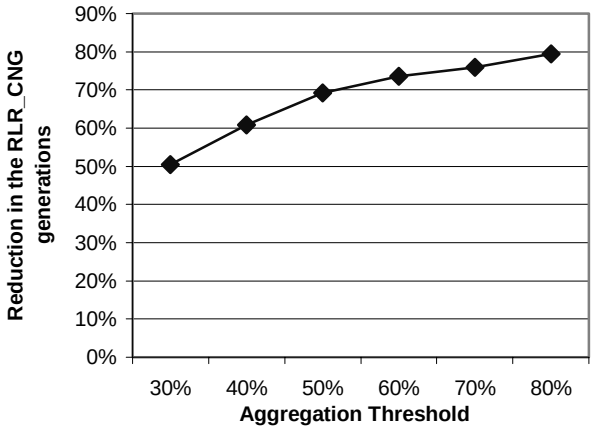

Figure 6. Reduction in RLR_CNG messages

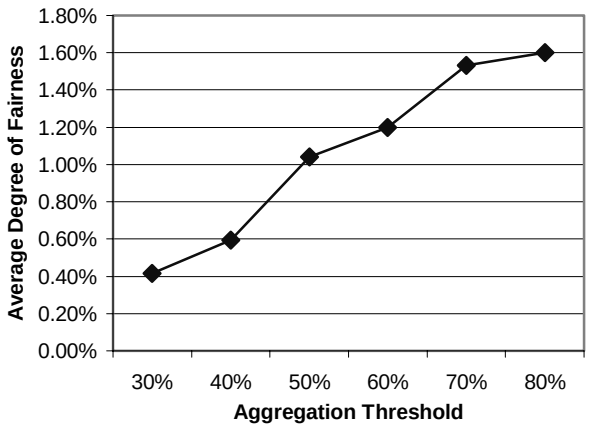

Figure 7. Average Degree of Fairness

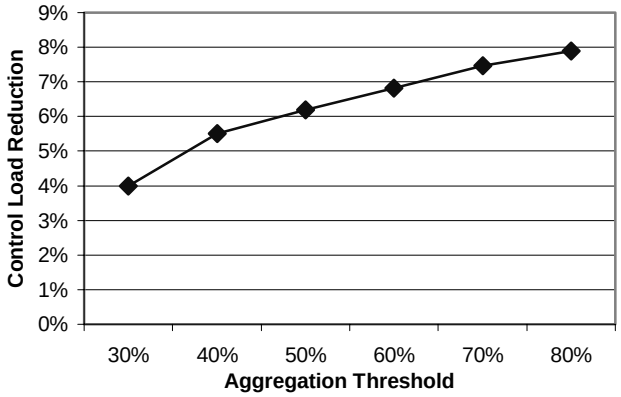

Figure 8. Control Load Reduction

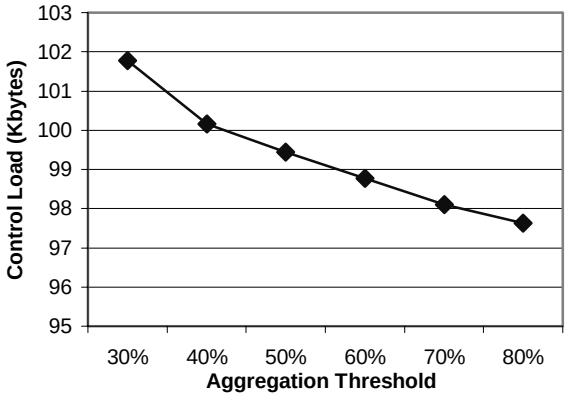

Figure 9. Total Control Load

Simulation results collected for the simulation topology of Figure 2 showed that the message aggregation significantly reduces the total number of the RLR_CNG messages generated. In the best case, when the aggregation threshold was set to 80%, the message aggregation technique reduced the total number of RLR_CNG messages by almost 80%. At the same time, the average degree of fairness during simulation was only 1.6%, which means the allocation bandwidth values did not deviate much from the optimally fair load distribution.

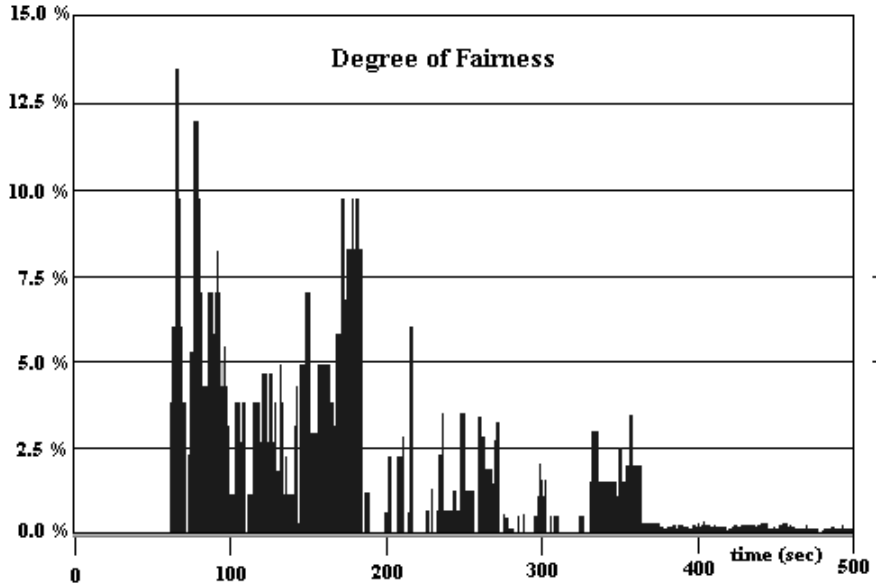

Figure 10. Variation of the Degree of Fairness

However, the reduction of the total control load due to message aggregation was significantly lower as shown in Figure 8. In the best case, when the aggregation threshold was set to 80%, the total load reduction was only 8%. Nevertheless, as Figure 9 shows, the total load of the control messages consistently decreases as the aggregation threshold is increased.

We observed such a small reduction in the amount of the control traffic because the simulation of Figure 2 was configured with a relatively small number of FTP flows. On average, during the simulation there were only 300 flow activation requests by the FTP sources. As a result, the PROBE messages were the dominant contributor to the total control load in the network. Since the goal of the message aggregation is to reduce the total number of RLR_CNG messages, its effects on the reduction of the total control load are very small , only 4% -- 8%. In the scenario where the total number of the flow activation requests is significantly larger, the RLR_CNG messages would dominate the control load and thus the message aggregation would significantly reduce the control load overhead. In addition, it should be noted that the overall overhead due to the control message exchange was less then 0.1% of the total load in the system. Thus, in networks with a small number of flows where the PROBE messages are the dominant contributor to the control load overhead, the message aggregation may not be needed.

It should also be noted that although the average degree of fairness varied between 0.4% and 1.6%, in certain instances during the simulation its value reached as high as 13%. Figure 10 shows the variation of the degree of fairness for the scenario of Figure 2, where the aggregation threshold was set to 80%. However, as Figures 10 and 11 show, such fluctuations do not happen very frequently and usually they last for a very short period of time. For example, value of the degree of fairness is larger than 10% only 0.66% of the time, while it varies from 5% to 10% only 8.47% of the time. As a result, we believe that such behavior of the LDS due to message aggregation is acceptable.

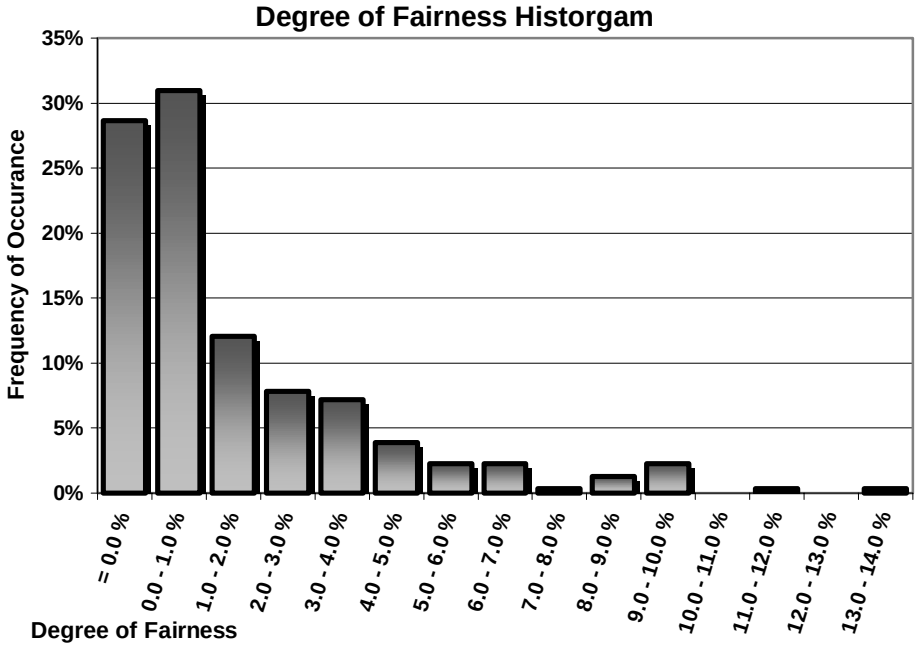

Figure 11. Distribution of the Degree of Fairness values

5. RELATED WORK OVERVIEW

This paper introduces a new idea for reducing the total overhead due to control message exchange within the framework of the LDS introduced in [11-13]. This work is a direct extension and improvement of [13] where a more detailed description of the LDS may be found. In [11] Hnatyshin et al provide an alternative approach to load distribution in the Internet. The LDS proposed in [11] does not require the core nodes to maintain aggregated interface RLR and instead relies on an approximation mechanism for computing the fair share of the ingress nodes. However, this approach cannot accurately compute the fair shares, takes a long time to converge, and does not guarantee fair load distribution among the edge nodes under all network conditions.

In [14], Kar et al provided an excellent definition of the dynamic rate control problem and introduced an iterative algorithm that solves it. In [14], individual sources adjust their sending rates based on the utility function and the network feedback which consists of information about the number of congested links on the path. However, the algorithm proposed in [14] converges to the optimal values slowly, operates on a per-flow basis, requires sources to communicate their sending rates to the core routers, and relies on the ACK packets to carry the feedback. In certain situations, the solution proposed in [14] becomes unacceptable because of these features.

Mirhakkak et al introduced a somewhat related idea in [18]. Their goal was to modify the resource reservation protocol RSVP for supporting dynamically changing QoS requirements in mobile ad hoc networks. The proposed dRSVP mechanism also assumes that each flow requests resources in a range. When a new flow enters the network and there are not enough resources to accommodate it, the congested link will adjust the reservations of other flows in order to accept the new flow's reservation. Unfortunately dRSVP also works on a per-flow basis and thus does not scale well. Furthermore, it does not guarantee that the links in the network will be fully utilized and it allows periods of QoS degradation.

The Explicit Congestion Notification (ECN) model [22] requires that the sources will reduce their rates upon reception of the CE marked packets. Both Explicit Congestion Notification approach and simple rate control algorithm [14] assume that the sources are well behaved and would reduce their sending rate upon congestion notification arrival. Unfortunately in the diverse Internet environment, we cannot be sure that all the sources will behave as requested. Thus neither of these approaches provides protection against denial-of-service attacks. On the contrary, the load distribution scheme that we have introduced deals with trustworthy boundary nodes that would adjust sending rates regardless of the user behavior and thus mitigates the possibility of misbehaving sources launching a denial-of-service attack.

The problem of admission control [10, 15-16] and controlled-load services [24] is somewhat related to the load distribution issues discussed in this paper. However they address a slightly different problem of determining when a new flow could be accepted into the network, while the LDS examines the problem of how to fairly distribute resources among the sources in order to accommodate the new flow's request.

6. CONCLUSIONS

In this paper we examined performance of the LDS in a network that contains a large number of small flows and introduced a message aggregation technique for reducing the control load overhead. The message aggregation reduces the total number of RLR_CNG control messages at the cost of violating the fairness requirements of the resource distribution. Simulation results reported that the LDS provides a fair and efficient resource allocation in the network with a large number of small flows. Furthermore the message aggregation is capable of significantly reducing the total number of RLR_CNG messages at the small cost of less than 2% of average deviation from the optimally fair load distribution.

Although the message aggregation showed very promising results, it should not be used in networks where the dominant part of the message exchange overhead comes from the periodic path probing. In this case, the message aggregation technique would be ineffective. Furthermore, simulation results showed that the overhead due to the periodic path probing is very insignificant, less than 0.1% of the total load. As a result, the message aggregation should be used only in networks where the RLR_CNG messages due to flow activation/ termination dominate control load overhead.

7. REFERENCES

1. M. Allman, "A Web Server's View of the Transport Layer," *Computer Communications Review*, vol. 30, no. 5, October 2000, pp. 10-20.
2. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. "An Architecture for Differentiated Services", December 1998. IETF RFC 2475.
3. R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", June 1994. IETF RFC 1633.
4. H. Chow and A. Leon-Garcia, "A Feedback Control Extension to Differentiated Services", March 1999. Internet Draft: draft-chow-diffserv-fbctrl.txt.
5. D. Clark and W. Fang, "Explicit Allocation of Best Effort Packet Delivery Service," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, August 1998, pp. 362-373.

6. C. Dovrolis and P. Ramanathan, "A Case for Relative Differentiated Services and Proportional Differentiation Model," *IEEE Network*, vol. 13, vo. 5, Sep./Oct. 1999, pp. 26-34.
7. C. Dovrolis and D. Stilliadis, "Relative Differentiated Services in the Internet: Issues and Mechanisms," Proceedings of ACM SIGMETRICS, May 1999.
8. C. Dovrolis, D. Stilliadis, and P. Ramanathan, "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling," Proceedings of ACM SIGCOMM '99 Conference, Cambridge, MA, Sep. 1999, pp.109-120.
9. W. Feng, D. D. Kandlur, D. Saha, and K. G. Shin "Understanding and Improving TCP Performance over Networks with Minimum Rate Guarantees," *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, April 1999, pp. 173-187.
10. R. Gibbens and F.P. Kelly, "Distributed connection acceptance control for a connectionless network," Proceedings of ITC'99, Edinburgh, UK, June 1999.
11. V. Hnatyshin and A.S. Sethi, "Avoiding Congestion Through Dynamic Load Control," SPIE's Int'l Symp. on The Convergence of Information Technologies and Communications, Aug. 2001, pp. 309-323.
12. V. Hnatyshin and A.S. Sethi, "Providing per-flow QoS using load distribution scheme," TR 2002-06, Department of Computer and Information Science, University of Delaware, February 2002.
13. V. Hnatyshin and A.S. Sethi, "Fair and Scalable Load Distribution in the Internet," Proceedings of the International Conference on Internet Computing, Las Vegas, NV, June 24-27, 2002, pp. 201-209.
14. K. Kar, S. Sarkar, and L. Tassiulas, "A Simple Rate Control Algorithm for Maximizing Total User Utility," Proceedings of INFOCOM 2001, Anchorage, USA, April 2001.
15. F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate Control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, no 3, pp. 237 - 252, March 1998.
16. F. Kelly, P.B. Key, and S. Zachary, "Distributed Admission Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 12, December 2000, pp. 2617-2628.
17. A.F. Lobo and A.S. Sethi, "A cooperative congestion management scheme for switched high-speed networks," Proceedings of ICNP-96, International Conference on Network Protocols, Columbus, Ohio (Oct.-Nov. 1996), pp. 190-198.
18. M. Mirhakkak, N. Schult, and D. Thomson, "Dynamic Quality-of-Service for Mobile Ad Hoc Networks," First Annual Workshop on Mobile and Ad Hoc Networking and Computing, 2000, pp. 137 –138.

19. B. Nandy, N. Seddigh, P. Pieda, and J. Ethridge, "Intelligent Traffic Conditioners for Assured Forwarding Based Differentiated Services Networks,“ Proceedings of IFIP High Performance Networking (HPN 2000), June 2000.
20. OPNET Modeler. OPNET Technologies Inc. http://www.mil3.com.
21. P. Pieda, N. Seddigh, and B. Nandy, "The Dynamics of TCP and UDP Interaction in IP-QOS Differentiated Services Networks," Proceedings of the 3rd Canadian Conference on Broadband Research, November 1999.
22. K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", March 2001. Internet Draft: draft-ietf-tsvwg-ecn-03.txt.
23. J. Rezende, "Assured Service Evaluation", Proceedings of Globecom’99, March 1999.
24. J. Wroclawski, “ Specification of the Controlled-Load Network Element Service,” September 1997. IETF RFC 2211.