A TCP FRIENDLY, RATE-BASED MECHANISM FOR NACK-ORIENTED RELIABLE MULTICAST CONGESTION CONTROL

Joseph P. Macker[1] and R. Brian Adamson[2]
[1] Information Technology Division, Naval Research Laboratory
[2] Newlink Global Engineering Corporation

Abstract-In this paper, we describe ongoing work in adding congestion control extensions to an existing negative acknowledgement (NACK) oriented reliable multicast protocol. Our previous work adopted and used the concept of a dynamic worst path representative for equation-based rate adaptation at the multicast source and we have further refined this approach and present results here. We present an overview of these extensions implemented within a working reliable multicast protocol (mdp-cc) and we present simulation results. Our analysis of interflow fairness with TCP unicast sessions demonstrates friendly behavior across a set of scenarios and results with more dynamic flows show that the worst path representative approach adapts rapidly to changing congestion conditions.

I. INTRODUCTION

The successful day-to-day operation and proliferation of Internet Protocol (IP) technology worldwide has been in a large part due to the existence and wide scale use of a standardized, reliable unicast transport protocol (i.e., Transport Control Protocol (TCP)). In addition to reliable data transport mechanisms, TCP also provides effective, end-to-end congestion control mechanisms [1,2]. At present, reliable multicast transport mechanisms lack such "best practice" approaches to end-to-end congestion control. Effectively addressing congestion control issues remains a key requirement for widespread Internet deployment of reliable multicast (RM) solutions and applications. Another specific concern for the Internet community, at large, is the impact RM traffic has on other coexistent Internet traffic (particularly TCP flows) during times of congestion [3]. There are several important classes of RM protocols and applications and there is no “one size fits all” solution to the set of problems across all of these design spaces. Our particular work described in this document targets congestion control mechanisms for NACK-oriented reliable multicast (NORM) protocols, but some of the techniques can viewed as more general and independent of specific reliability mechanisms.

II. APPROACH AND PREVIOUS WORK

In recent years, research results relating to equation-based TCP throughput models [Floyd, Umass] have indicated that a reasonable low complexity, steady state model(s) existed for predicting TCP behavior. Also, previous work exploring TCP fairness definitions and methods for applying models to multicast situations [Whetten] outlined a discussion of fairness models for application to multicast transport. The TCP worst path fairness model is based on having equation-based TCP throughput estimates for all source-receiver paths in the multicast session. By adapting the rate of the source to the worst TCP predicted path rate amongst the receiver group, a fairness bound on other paths is guaranteed. The concept of using a subset of the receiver group to provide more rapid feedback for congestion control purposes provides merit by trading off the need for rapid feedback for congestion control purposes (e.g., representative receivers) against the need to continue to preserve protocol scalability to potentially large receiver groups within a multicast session. Although, we take a different approach in electing and applying representative feedback, earlier work regarding receiver representative concepts for multicast congestion control was outlined in [DeLucia]. Also, recent work on pgmcc adopts a dynamically elected single acker concept for window-based control.

In past years, work and discussions within the Internet Research Task Force (IRTF) Reliable Multicast Research Group (RMRG) group also significantly contributed to establishing research goals and ideas for applying fairness models and equation-based approaches to multicast flows. Other results demonstrating a fairness and equation-based congestion control model for unicast have been recently published in [Handley] and the single-rate pgmcc congestion control method outlined in [Rizzo]. Our preliminary work in applying rate-based TCP friendly congestion control and congestion control representatives to NORM protocols was previously presented at the June 1999 RMRG meeting in Pisa [1] and was also documented briefly in [2]. That previous work described a novel approach to use path loss and round trip time (RTT) estimates collected at the source to dynamically elect one worst path representative amongst the receiver set. This elected receiver provides a rapid feedback control loop for rate-based congestion control. To improve representative switching performance, we also maintain rapid control loop state on a small number of additional candidate receiver paths. In this paper, we expand on that previous work and describe more recent design refinements, experiments, and TCP friendliness results.

III. DESIGN CHALLENGES

An overarching design tradeoff in applying congestion control mechanisms to NORM style protocols is balancing the inherent reduced group feedback mechanisms against the increased need for timely and accurate receiver feedback for dynamic congestion control. Another challenge is that while an equation-based TCP model gives us a steady state target to achieve TCP fairness within a rate-based protocol, such a model does not tell us how to effectively collect metrics and/or how to integrate and achieve such a design within a dynamic multicast protocol framework. The typical NORM protocol makes this more challenging because the potentially infrequent NACKing is the fundamental receiver feedback mechanism to the source. In addition, for enhanced scalability, NORM protocols historically implement some form of NACK suppression among receivers to further reduce the feedback channel load and reduce the likelihood of NACK implosion within large group scenarios. NORM protocols also typically adapt some form of forward error correction (FEC) based packet repairing technique to replace or enhance explicit packet retransmission schemes. These inherent NORM design mechanisms compete against the need to provide timely and accurate receiver feedback for congestion control mechanisms. We review our approach to addressing this tradeoff below.

IV. Design approach

In targeting TCP friendliness behavior end-to-end, we adopted a worst path fairness model. As mentioned, the worst path TCP fairness model requires that only the minimum of the equation-based TCP throughputs across the set of receiver paths be used as a target rate goal. In order to responsively adapt to dynamics in congestion, timely feedback should be provided on the source-receiver path(s) of interest. Maintaining rapid feedback of congestion control metrics for all paths within a scaled multicast session can be prohibitive to efficient operation of the protocol. We hypothesized that maintaining more timely feedback state for a dynamically-elected subset of source-receiver paths and only reacting to the worst path receiver amongst this group could provide a reasonable compromise to the set of competing feedback requirements. In addition to the challenge of tracking and estimating worst path control loop metrics, we needed a robust and safe methodology for implementing multicast source rate adjustment. The following sections provide more details regarding our design strategies, solution, and simulation results.

To investigate and implement our protocol ideas, we developed extensions to the existing Multicast Dissemination Protocol (MDP) software [ref]. We refer to the extended congestion control version as *mdp-cc*. Many of the mdp-cc extensions can be generalized to other NORM protocols and the techniques can be adopted outside of the MDP protocol framework. However, starting with MDP as a software framework provided some advantages in development and experimentation. First, the framework is a well-tested open source implementation of an end-to-end, rate-controlled NACK-based protocol with all the typical esoteric features used for improved scalability (e.g., NACK suppression, FEC repairing, etc). Second, the existing codebase provided us with a detailed protocol simulation model already embedded within the ns2 framework to evaluate various protocol components and TCP fairness issues.

A. Core Design Components

The mdp-cc design extension can be broken down into four principal areas.

1) Receiver loss fraction measurement and collection

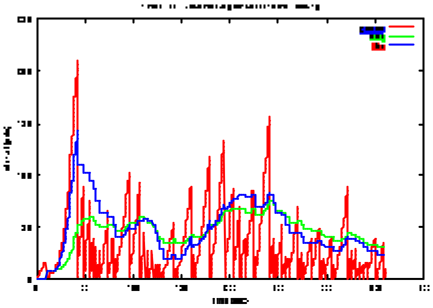
2) Source-receiver path RTT measurement and collection

3) Congestion control representative selection and timely feedback mechanisms

4) Source transmission rate adjustment algorithm

To predict and expected TCP source-receiver path throughput, we require a loss estimation input for the receiver path in question. Each MDP receiver maintains a running estimate of the current loss event fraction from each sender. The loss event fraction corresponds to the inverse of the average interval (in terms of a packet count) between loss events. A loss event is distinct from a raw packet loss in that multiple, individual packet losses occurring within in one RTT "window" of packets are counted as only a single loss event. This loss event definition is consistent with the definition used in equation-based TFRC work [REF TFRC paper]. Whenever a receiver provides any form of feedback to a given sender, the receiver provides its current estimate for that sender as part of the feedback message. The sender uses this estimate to update its list of congestion control representative candidates and to potentially feed into the rate adjustment algorithm. To facilitate maintenance of a loss fraction estimate, all source packets include a monotonically increasing sequence number that receivers use to detect missing packets. The mdp-cc implementation also keeps track of packets arriving out-of-order and delays counting losses until the possibility of an out-of-order arrival is eliminated. The delay depth for out-of-order packet tracking is dynamically updated when out-of-order packets arrive. The effectiveness of this technique in networks (e.g. mobile wireless) where out-of-order arrivals may be more common and its effect on congestion control operation (including impact on TCP-fairness) is a subject of future investigation.

The mdp-cc protocol implementation currently includes two options for loss event estimation. The first of these is a technique similar to the Average Loss Interval with discounted, weighted history similar to that described previously in [REF TFRC]. The other technique is an adaptively smoothed exponentially weighted moving average of the loss event interval. In preliminary evaluation, both techniques produce very similar estimates. This is illustrated in Figure 5. The performance and complexity trade-offs of these two techniques are being studied, but both are implemented within the present design allowing cross comparison and tradeoff analysis.



In MDP, the sender is responsible for collecting RTT measurements from receivers to determine both NACK suppression and repair cycle timing based upon the greatest observed RTT. In mdp-cc, RTT information is needed by the sender as part of the congestion control algorithm to calculate the TCP throughput estimate for different receivers. For general protocol operation, receivers provide the opportunity for the sender to collect individual RTT measurements when they transmit NACK messages. For reliability purposes, this technique seems sufficient, but rate-based congestion control requires consideration of additional issues. While it may be sufficient to receive feedback only from NACKing receivers during steady-state operation, an additional mechanism may be necessary at least for protocol startup purposes and when one considers the potential for operation among more heterogeneous (e.g., various *RTT,loss combinations*) source-receiver paths.

Within the present mdp-cc experimental implementation, the sender uses a longer term feedback mechanism to excite explicit feedback responses in addition to general NACK and the rapdi congestion control representative collection processes. The longer-term feedback excited from the group at large provides an opportunity for receivers not NACKing (or whose NACKs are suppressed) to provide feedback and be considered for inclusion in the representative set. Once such a receiver is treated as a representative and provides more timely congestion-oriented feedback, the sender will adjust its rate properly even when the receiver is not sending any NACK messages, because periodic loss event estimation and RTT timing is actively collected from even non-NACKing representatives.

From the loss event estimates and the RTT measurements gathered from received feedback, the sender calculates the estimated steady state throughput rate predicted by the analytical model of TCP for individual receivers. The sender keeps a list with state for a small set of receivers with the lowest predicted TCP throughput rates. This list is dynamically updated as feedback is progressively received from the group. These congestion control "representatives" are the expected candidates for worst path TCP fairness and are more rapidly probed by the sender for continued loss estimate and RTT measurement updates. The sender uses the feedback from the representative set and the previous methods described to find the receiver with the minimum transmission rate predicted by the TCP throughput model. This receiver is identified as the worst path receiver (WPR). The sender maintains smoothed RTT estimates for the representatives and tracks RTT variation for calculation of the retransmission timeout value (T0) used in the analytical TCP throughput model.

The sender uses an MDP_CMD message to excite the response from the current representative set. This probe message contains a list of the current representatives along with their respective RTT measurements. This allows the representatives to properly filter detected packet losses as loss events (i.e. counting multiple packet losses within one RTT as a single loss event). The sender also advertises its current transmission rate and the current representative set metrics (e.g., path RTT estimates) to assist receivers with more accurate loss event estimation. The probe message also contains a flag to mark when feedback is expected from the group at large. A field dictating the random backoff time window for receivers to respond to this *wildcard* probe is also included in the message. This time window is to be set to an appropriate value given the group size to keep the volume of this feedback at a low level. The non-representative receivers backoff their response with a uniform random distribution and the period of the *wildcard* probing also corresponds to this interval.

The current algorithm for selecting the representative set is very simple. The sender keeps state for a small number (currently 5) of receivers with the lowest transmission rate predicted by the TCP analytical model. Results collected in simulation and real networks with this technique have been promising. There is potential risk that receivers lying behind a common bottleneck link within the network may monopolize the current representative set which defeats the role of rapidly exciting congestion control feedback from multiple receivers simultaneously. However, NACK suppression helps mitigate this risk and this issue is currently under further investigation. Additionally, algorithms to dynamically populate the sender representative list with receivers with uncorrelated metric sets (e.g. RTT, loss event fraction, loss patterns) are being considered. If such approaches can be refined they would help the source select the most significant, heterogeneous paths to monitor with rapid feedback.

In addition to congestion control feedback collection, the mdp-cc source needs to have a method for adjusting the multicast transmission rate that provides a degree of TCP friendliness. The sender uses the rate predicted for the WPR to establish a goal rate for transmission. The sender begins adjusting its rate towards this goal rate. A rule-based approach is used to adjust the rate. This approach includes techniques for dealing with cases of missing expected feedback from the representative set and receivers leaving the group. As noted previously, the sender transmits congestion control probes to the group at a rate of once per RTT of the current WPR, denoted as WPR_RTT. At startup, before any receivers have responded, the sender transmits "wildcard" probes that are acknowledged by the group at large within a distributed random backoff time. Unlike unicast transport, multicast startup issues tend to involve a consideration for longer delay in order to capture heterogeneous group effects and to preserve efficient protocol overhead characteristics.

Some of the present rules used for rate adjustment in mdp-cc are as follows. When the sender receives a response from the WPR in a timely fashion (within 2*WPR_RTT of the time the corresponding probe was sent) and its current transmission rate is less than the predicted WPR rate, the sender increases its rate quickly (exponentially) towards the goal WPR rate. If a response is received later than 2*WPR_RTT after the corresponding probe was sent, no rate adjustment occurs. If the predicted WPR throughput rate is less than the current sender transmission rate, the sender rapidly (exponentially) reduces its rate towards the goal bottleneck rate. Rate decreases occur at the timeout interval at which the congestion control representative probing is done. The sender will automatically decrease its transmission rate if no response is received from the current WPR within 4*WPR_RTT of the probe transmission time. The rate is decreased once per WPR_RTT when the response is late. This measure serves as a congestion collapse avoidance measure. However, if a representative fails to respond at all after a large number of probes, the representative is bumped from the list and the next ranking candidate assumes the bottleneck role.

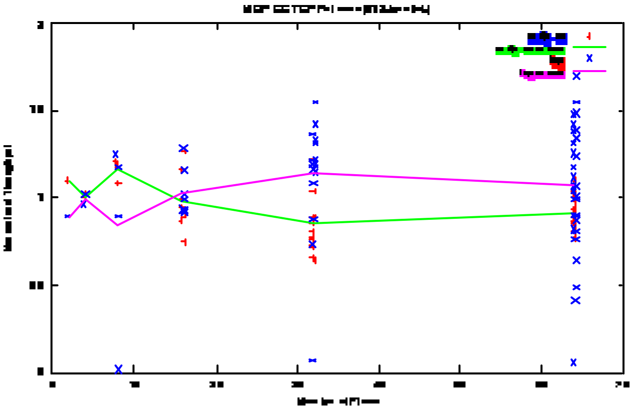
When the representative list is completely emptied due to lack of response, the protocol quickly reverts to a minimal transmission rate with long term, wildcard probing of the group in preparation for resuming steady-state operation. It is anticipated the rule-based algorithm to control rate adjustment will help maintain stability when network dynamics or measurement uncertainties cause the bottleneck to flip-flop among more than one representative candidate. Although the rate adjustment of this approach for multicast is slower in response than TCP, early simulation results show that this approach maintains good long-term steady state interflow fairness with co-existing TCP flows, even when new flows are dynamically initiated and terminated.

Another design factor that should be mentioned is the provision for dynamic packet sizes within the equation-based approach. In the present mdp-cc design, potential for varying source packet sizes is accommodated in the TCP throughput equation by keeping an weighted EWMA value for transmitted source packets and providing this number as a variable to the TCP throughput equation calculation.

V. SIMULATION PERFORMANCE RESULTS

A. Basic TCP Friendliness Trials and Results

To test the TCP friendliness of the approaches outlined here for mdp-cc, we began with simple simulation topologies and congestion scenarios to gain insight on steady state behavior of the protocol and to examine interflow fairness. In one set of tests, we adopted graphing methods used in [5] to help examine rate-based unicast congestion control (tfrc) and TCP friendliness. The results present long term steady state interflow fairness between tcp and mdp-cc. The value of 1 on the y axis represents the fair normalized average throughput given that there are *n* flows competing on the congested bottleneck. For example, if there are 64 intercompeting flows, as shown in the right part of the graph, the expected steady state throughput per flow is 512kbps/64 = 8kbps. The graph plots the ratio of the observed value for a flow and the expected value. The solid lines are averages for all sample points of a particular flow type. Figure x is an example output of a fairness trial of mdp-cc and tcp. Even under high degrees of statistical multiplexing the interflow fairness is quite acceptable in our opinion. In additional tests, we have seen that the results using mdp-cc are very comparable to those presented in [5]. Even though these initial tests are simple scenarios, it is important to note that this is a fully functioning reliable protocol with dynamic NACKing, feedback suppression, representative feedback, reelection,etc.



B. *WPR Switching Tests and Representative Plotting*

Simulations were constructed to evaluate the operation of mdp-cc in environments with dynamic changes in the worst path rate and location within a multi-bottleneck topology. Figure X illustrates an example topology generated by our simulation toolset. In this particular example, a source cluster (Nodes 0-7) populated with a mix of mdp-cc and tcp generator agents sends traffic to five other receiver clusters. Persistent steady-state tcp flows and a single mdp-cc flow are transmitted across the five links feeding the receiver cluster. These five links dynamically play bottleneck roles in the simulation through the start and stop of additional tcp flows across those links. The simulation toolset is capable of random and/or deterministic generation of these additional dynamic, congesting tcp flows as needed.

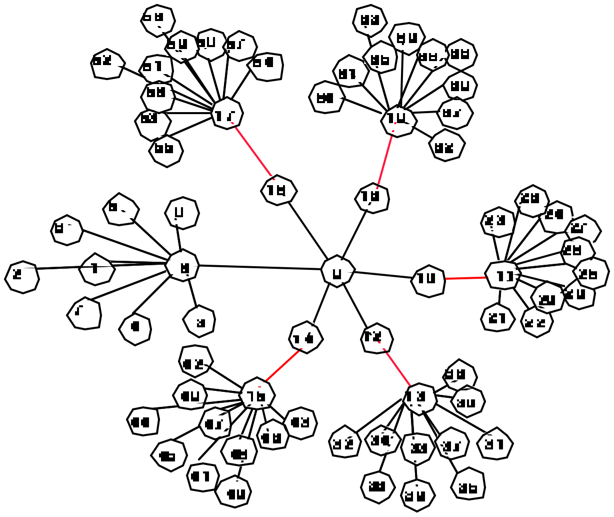
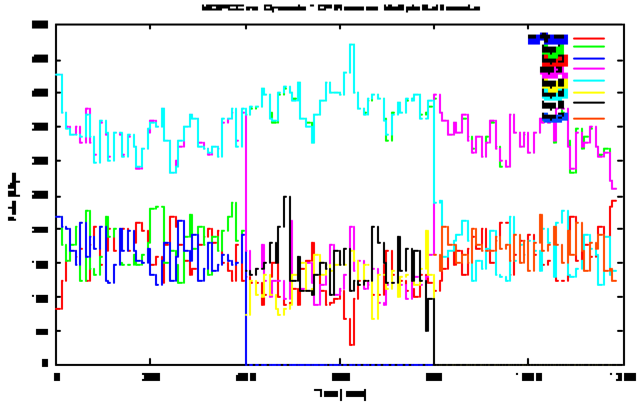


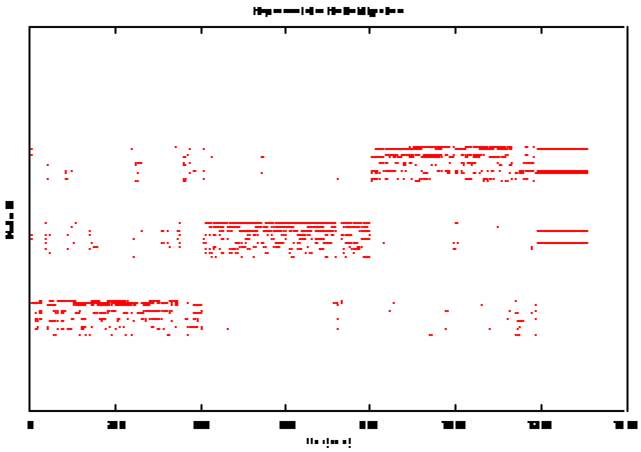
Fig. X - Dynamic Multi-bottleneck Simulation Topology

Figure XX is a plot of the observed transmission rates of mdp-cc and tcp flows during a simulation using the above topology with 500 kbps receiver cluster feed links. In this simulation, a single tcp flow (in addition to the persistent, background flow per bottlneck link) was added to one of the feed links from time 0 to 400 sec, creating a congestion bottlneck. Then, as that added flow terminated, two tcp flows were added to another feed link from time 400 to 800 sec, creating a different, slightly more severe bottleneck as the mdp-cc flow is forced to share the link with three other tcp flows (one steady-state and the two added flows). Finally, from time 800 to 1200 sec, a single additional tcp flow was placed on yet another feed link, once again changing the worst path within the topology and the congestion control rate.



As the bottom portion portion of the graph in Figure XX depicts, the transmission rate of the mdp-cc flow is fairly intermixed with the transmission rates of the tcp flows with which it was competing. The flow rate plots in the top portion of the graph correspond to tcp flows on the remaining uncongested feed links. Note that the transmission rate of mdp-cc appropriately adapts in response to changes in the bottleneck link location and congestion rate. These results are representative of what has been observed in mdp-cc simulations run to date.

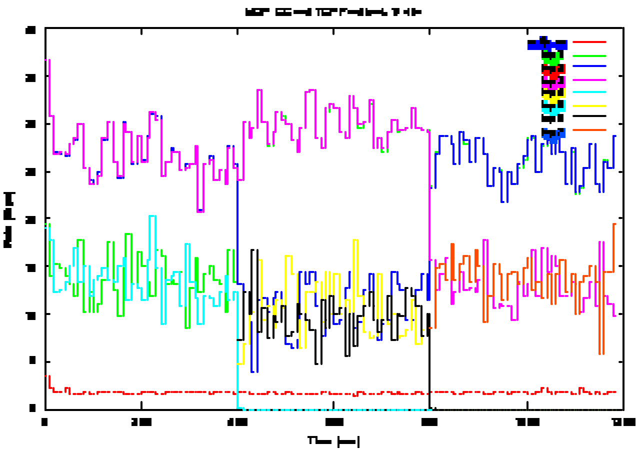
Figure XX illustrates which receiver nodes comprise the sender's representative receiver list at different points in time during the simulation. The mdp-cc software was configured for maximum of five representative receivers in its candidate list at one time. Thus, there is a maximum of five parallel points on the plot shown at any one point in time. The simulation topology assigns consecutive node identifiers to receivers within the same cluster which allows for easy interpretation of the graph below.



As this plot shows, the representative list membership generally includes the location of the congestion bottleneck well over the course of the simulation run. However, note that occasionally, the list includes receivers which are not behind the current bottleneck link. Furthermore, it was observed, that even the selected WPR, would sometimes be a receiver from a non-bottleneck cluster. This is likely due to dynamics of interacting with the steady-state tcp flow on the other corresponding feed link. It is exp ected that the rapid probing of multiple representative receivers in the mdp-cc approach helps maintain fairness and stability in light of such phenomena by allowing the sender to quickly correct its choice of WPR. Further simulations will be conducted and more data collected to specifically evaluate the value of the parallel, multiple representative approach. And, as the above graph suggests, further work to provide a technique to select appropriately decorrelated representative receivers to better span independent candidate worst paths should be performed. The potential issue of the representative list clustering to a single bottleneck is highlighted in the graph above, although there may be some value in the implicit hysteresis provided by having multiple representatives from the same congestion path. These tradeoffs will be examined in further work.

C. Feedback Loading results

As previously mentioned, maintaining the scalability of a NACK oriented protocol is an important design goal for an applicable congestion control scheme. The quantity of feedback traffic to the sender in a reliable multicast session is a principal factor in determining scalability. Figure XXX is graph of the volume of feedback traffic from the receivers to the senders of the mdp-cc and tcp flows from the simulations described above.



The flat line near the bottom of the graph plots the rate of all mdp-cc feedback traffic, including NACK messages for reliability as well as ACK messages from the representative receiver set. In this simulation, there were 50 receivers in the mdp-cc group. The other plots in the graph represent the feedback traffic of tcp flows during the simulation. It is interesting to note that the feedback traffic volume of mdp-cc to a group of 50 receivers is far less than the feedback generated by any of the competing tcp flows. Note that the quantity of tcp feedback traffic is relative to the transmission rate of the tcp flow. The principal source of mdp-cc feedback is the explicit response by representative receivers to sender congestion control probes. The volume of this traffic is a function of the topology RTT and the length of the sender's representative receiver list. This portion of the feedback will remain constant irrespective of group size.

VI. Further Work

(*rewrite this in short order or incorporate added work ideas throughout paper to save space*).

Timing issues, group size additions, enhancements to the RTT and loss event estimation approaches. More pathological analysis as in pgmcc.

VII. Conclusions

We have presented ongoing design work of mdp-cc, which provides rate-based TCP friendly congestion control within a NORM protocol framework. We have described a number of mechanisms that we believe effectively tradeoff protocol scalability and overhead, while providing improved rapid response for congestion control reaction.

We have applied this scheme to a working NACK-based protocol and have performed both operational and simulation tests demonstrating effective TCP fairness and inter-protocol fairness across a number of network scenarios. We feel that this protocol has demonstrated robustness and fairness issues to a degree that it would be safe to deploy for reasonably scaled usage. A public version of the protocol with the mdp-cc extensions is available at [ref] and runs on a variety of OS platforms (e.g., Win32, BSD, Linux, Solaris).

We have mentioned further work desired and ongoing investigations to refine a number of mechanisms, such as the inclusion of group size estimation to improve protocol efficiency. Also, additional work is planned to improve robustness issues under highly heterogeneous case scenarios of deployment. One area actively being worked on is an algorithm to improve distributed receiver loss event estimation using purely source-based RTT estimators.

VIII. References

[1] V. Jacobson, *"Congestion Control and Avoidance"*, Proc. of SIGCOMM 1988, pp. 314-328.

[2] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", Internet RFC 2001, January 1997.

[3] A. Mankin, A. Romanow, S. Bradner, V. Paxson, “IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols”, RFC 2357, June 1998.

[4] http://www.ietf.org/html.charters/rmt-charter.html

[5] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, “Equation-based Congestion Control for Unicast Applications”, ACM SIGCOMM 2000, Aug 2000.

[6] Luigi Rizzo, “pgmcc: a TCP-friendly single rate multicast congestion control scheme”, ACM SIGCOMM 2000, Aug 2000.

[7] J. Macker, R. B. Adamson, "Reliable Multicast Congestion Control", in IEEE Proc. Military Communications International Symposium (MILCOM) 2000, Los Angeles, USA, Oct 2000.

[8] J. Macker, R. B. Adamson, *"The Multicast Dissemination Protocol Toolkit"*, Proc. IEEE MILCOM 99, Oct 99.

[9] Dante DeLucia, Katia Obrascka, “Multicast Feedback Suppression using Representatives”, IEEE Infocom 97’.

[10] D. DeLucia, K. Obraczka, *"Congestion Control Performance of a Reliable Multicast Protocol.,"*. Proc. of IEEE ICNP'98, August 1998.

[11] A. Mankin, A. Romanow, S. Bradner, V. Paxson, “IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols”, RFC 2357, June 1998.

[12] Floyd, S., and Fall, K., "*Promoting the Use of End-to-End Congestion Control in the Internet"*, IEEE/ACM Transactions on Networking, August 1999. .

[13] J. Macker, "Reliable Multicast Transport and Integrated Erasure-based Forward Error Correction", Proc. IEEE MILCOM 97, Oct 1997.

[14] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, *“Modeling TCP Throughput: A Simple Model and its Empirical Validation”*, Univ of Mass, Technical Report CMPCSI TR 98-008.

[15] B. Whetton, J. Conlan. *“A Rate Based Congestion Control Scheme for Reliable Multicast”*, White Paper, Globalcast Communications, Oct 1998.

M. Handley, and S. Floyd, "*Strawman Specification for TCP Friendly* (Reliable) Multicast Congestion Control (TFMCC), Reliable Multicast Research Group, December 1998.