Applying TCP-Friendly Congestion Control to Concurrent Multipath Transfer

Thomas Dreibholz, Martin Becke, Jobin Pulinthanath, Erwin P. Rathgeb
University of Duisburg-Essen
Institute for Experimental Mathematics
Ellernstrasse 29, 45326 Essen, Germany
{dreibh,becke,jp,rathgeb}@iem.uni-due.de

***Abstract*—The steadily growing importance of Internet-based applications and their resilience requirements lead to a rising number of multi-homed sites. The idea of Concurrent Multipath Transfer (CMT) is to exploit the existence of multiple paths among endpoints to increase application data throughput. However, handling the congestion control of each path independently lacks of fairness against non-CMT flows.**

In this paper, we describe our approach of combining CMT with the idea of Resource Pooling (RP) in order to achieve a performance improvement over non-CMT transfer while still remaining fair to concurrent flows on congested links. Unlike existing approaches which adapt classic TCP to a multi-homed CMT protocol, our approach does not depend on specific characteristics of TCP. Instead, we base on already entrenched functional blocks of CMT transfer, on the example of the CMT-enabled SCTP (Stream Control Transmission Protocol). In a simulative proof-of-concept analysis, we show that our approach – while relatively simple – is already quite effective.

Keywords: Multi-Homing, Congestion Control, Resource Pooling, Fairness, Proof of Concept

I. Introduction

Since the early days of the Internet, a client only needs one single destination address to request a service. The existence of multiple network interfaces – and with this more than one path to a destination service – has been out of scope for the development of the Internet Protocol (IP) as well as of Transport Layer protocols like TCP. During the last years there has been a silent revolution: more and more devices are offered with several network interfaces for improved resilience. But furthermore, the used protocols support only a single path access. However, new protocols like SCTP [1] comply with the requirements of multi-homing.

But within new possibilities also new issues for the distribution of load among paths are to investigate, especially if simultaneous utilization of multiple paths – denoted as Concurrent Multipath Transfer (CMT) [2] – is intended. Handling paths independently causes unfairness against concurrent non-CMT flows. The idea of Resource Pooling [3] is to bear the interaction among paths in mind in order to achieve a fair bandwidth share. Some research on combining CMT and RP has been made by [4], [5] for multi-homed TCP [6]. However, the focus on TCP – which has originally been designed as a single-homed protocol – requires a large set of complex additional features to support multi-homing and CMT and leads to – not yet fully researched – assumptions.

The goal of our approach presented in this paper is to propose a more clean slate approach to combine CMT and RP to improve the application data throughput while still remaining fair to concurrent TCP-like non-CMT flows on bottleneck links. The TCP-friendliness of our new congestion control scheme allows for the deployment of CMT in Internet setups – without discriminating other traffic. We demonstrate our approach by a simulative proof-of-concept analysis on the example of the multi-homed Stream Control Transmission Protocol (SCTP) [1] with CMT option [2]. Since SCTP already provides the multi-homing feature – which is also well-researched [7]–[9] and already deployed in real world setups – our new congestion control scheme remains very simple while already achieving a significant performance improvement over non-CMT transfer. We will demonstrate this by a simulative proof-of-concept analysis.

II. The SCTP Protocol

SCTP is a general-purpose, connection-oriented, unicast transport protocol which provides the reliable transport of user messages and a multi-homing concept out of the box.

An SCTP connection is denoted as *association*. Unlike TCP, each SCTP endpoint can use multiple IPv4 and/or IPv6 addresses to transmit to its peer. This feature is denoted as *multi-homing* and illustrated in figure 1. Each peer address defines a unidirectional *path*. The user data itself is segmented into units of so called DATA chunks, which are identified by unique Transmission Sequence Numbers (TSN). A Selective Acknowledgement (SACK) chunk is transmitted by the receiver to acknowledge received DATA chunks and report gaps (i.e. missing DATA chunks given by their TSNs) to the sender. The sender uses two different mechanisms of retransmission to fill gaps:

- Once a DATA chunk is gap-reported as missing for 3 times, it is retransmitted immediately on the same path (Fast Retransmission [1, subsection 7.2.4]).
- Further retransmissions (possibly on alternative paths) are triggered by a timer (Timer-Based Retransmission).

Currently, SCTP uses only one path in each direction to transmit DATA chunks on one time. This selected path is denoted as *primary path*. Alternative paths are only used

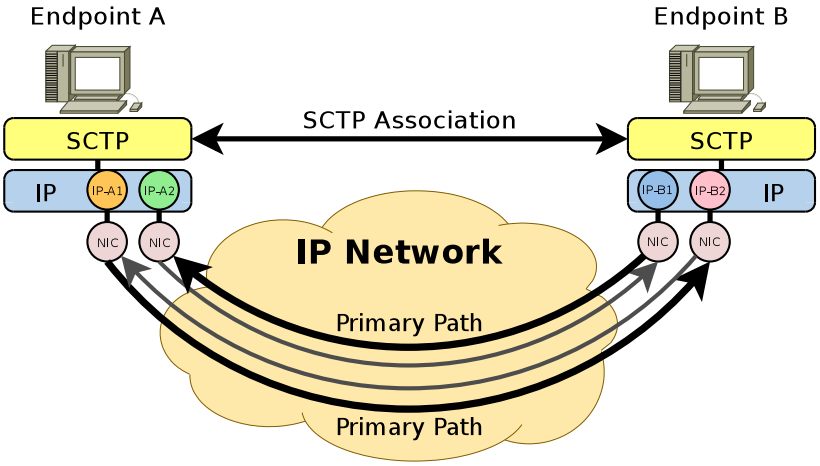


Figure 1. Multi-Homing

for retransmissions; the primary path may be changed e.g. in case of path errors. The reason for not sharing load among paths will be explained in section III.

SCTP shares the flow and congestion control mechanisms with TCP. That is, both protocols use the same AIMD (Additive Increase, Multiplicative Decrease) behaviour to adapt the congestion window to changing network conditions. Therefore, they fairly share the bandwidth on congested links. [10], [11] discuss SCTP congestion control in detail.

From the perspective of the Application Layer, SCTP provides a transport service covering all features provided by TCP plus a large set of additional features – particularly multi-homing, mobility [12], [13] and partial reliability [14]. In the long-term future, SCTP may completely replace TCP [15]. The advanced features of SCTP are also beneficial for applications like the data transport in grid scenarios [7] and the Reliable Server Pooling (RSerPool) framework [16]–[18].

III. Concurrent Multipath Transfer

Standard SCTP – as defined by its RFC [1] – transmits user data via a selected primary path. The idea of CMT for SCTP is to utilize *all* available paths. We denote this approach as CMT-SCTP. A particular application case for CMT-SCTP is distributed computing over MPI (Message Passing Interface) in large-scale setups [7].

[2] describes CMT-SCTP: the paths of an association are handled independently. As long as there is sufficient room in a path's congestion window, data may be transmitted over the corresponding path. Each path P has its own congestion window c_P and slow start threshold s_P. Just like for TCP, also SCTP adjusts c_P and s_P by using AIMD behaviour, upon changing network conditions on a path P with a Maximum Transmission Unit (MTU) of MTU_P (see also [1], [10] for a more detailed description):

- On Fast Retransmission on path P (i.e. a loss detected by gap reports):

$$s_P = \max(c_P - \frac{c_P}{2}, 4 * \text{MTU}_P) \quad (1)$$
$$c_P = s_P \quad (2)$$

- On Timer-Based Retransmission on path P (i.e. a loss detected by retransmission timeout):

$$s_P = \max(c_P - \frac{c_P}{2}, 4 * \text{MTU}_P) \quad (3)$$
$$c_P = \text{MTU}_P \quad (4)$$

  That is, in contrast to the Fast Retransmission the congestion window starts growing again from a single MTU_P (see also equation 2).
- On α acknowledged bytes in a fully-utilized congestion window during the slow start phase (i.e. for $c_P \leq s_P$):

$$c_P = c_P + \min(\text{MTU}_P, \alpha)$$

- On each fully acknowledged window in congestion avoidance phase (i.e. $c_P > s_P$):

$$c_P = c_P + \text{MTU}_P$$

The *per-path* behaviour of the SCTP congestion control is TCP-friendly, i.e. saturated TCP and SCTP flows will fairly share the bandwidth of the path [11]. However, in case of multipath transmission sharing a single bottleneck link, the *overall* behaviour of a CMT-SCTP association will be unfair: in case of n SCTP paths, the SCTP association will get n times the bandwidth share of a non-CMT SCTP or TCP connection over the same bottleneck. Therefore, [2] requires all CMT-SCTP paths to be disjoint. However, this condition cannot be assured for communications over the Internet. Therefore, the open topic has been whether it is possible to utilize multiple paths while remaining fair to other flows.

IV. CMT/RP-SCTP – Our Resource Pooling Approach for CMT-SCTP

A mathematical analysis on fairness in multipath congestion control is provided by [6]; an informal description of the general idea is given in [3]: the Resource

Pooling (RP) principle. Instead of handling paths independently, their interaction has to be taken into account. In particular: a loss on one path P_1 may have occurred on a bottleneck shared also by path P_2 of the multi-homed association. It is therefore necessary to adjust the transmission rate accordingly.

Based on the Resource Pooling ideas from [3], [5], we propose a modified Resource Pooling congestion control for CMT-SCTP, which we denote as CMT/RP-SCTP. However, our approach is generic and may be adapted to other protocols quite easily as well. Let $C = \sum_i c_i$ the overall congestion window of the association and $S = \sum_i s_i$ its overall slow start threshold. Then, our new congestion control behaves as follows:

- On Fast Retransmission on path P:

$$s_P = \max(c_P - \frac{C}{2}, 4 * \text{MTU}_P * \frac{s_P}{S}, \text{MTU}_P) \quad (5)$$
$$c_P = s_P \quad (6)$$

  That is, we incorporate the possibility of shared bottlenecks by trying to halve the overall congestion window on the lossy path. Instead of decreasing s_P to at least $4 * \text{MTU}_P$ (as for standard SCTP, see equation 1), we scale this lower limit by the slow start threshold fraction of the path P. We assume the slow start threshold to be a useful metric for the capacity of a path, and fairness is ensured by not exceeding $4 * \text{MTU}_P$ on all paths in a shared bottleneck case. A single MTU_P is the lower limit, since forcing segmentation to less than a full MTU_P makes no sense. Unless sending an overly large amount of paths over a really small-bandwidth bottleneck, this should not cause any fairness issue.
- On Timer-Based Retransmission on path P:

$$s_P = \max(c_P - \frac{C}{2}, 4 * \text{MTU}_P * \frac{s_P}{S}, \text{MTU}_P) \quad (7)$$
$$c_P = \text{MTU}_P \quad (8)$$

  Similar to standard SCTP (see equation 4), the congestion window c_P is also reduced to a single MTU_P.
- On α acknowledged bytes in a fully-utilized congestion window during the slow start phase:

$$c_P = c_P + \lceil \min(\text{MTU}_P, \alpha) * \frac{s_P}{S} \rceil \quad (9)$$

  Again, we assume the slow start threshold to be a useful metric for the capacity of a path. Therefore, we only increase the congestion window by the fraction of s_P and S, i.e. the capacity share of path P.
- On each fully acknowledged window in congestion avoidance phase:

$$c_P = c_P + \lceil \text{MTU}_P * \frac{s_P}{S} \rceil \quad (10)$$

  That is, we scale the increment value by $\frac{s_P}{S}$, analogously to our slow start procedure in equation 9.

The goals of our new approach – which are used as performance metrics for our proof-of-concept analysis in section VI – are as follows:

1) The application data throughput should not be smaller than for a non-CMT association.
2) Bandwidth fairness: on a shared bottleneck link, the bandwidth share of a multi-homed CMT/RP-SCTP association should be similar to a non-CMT association.

V. Our SCTP Simulation Model

In order to evaluate our Resource Pooling approach for CMT-SCTP, we have used the OMNeT++-based INET framework [19]. The SCTP simulation model in INET – described in [20] – has been extended by CMT-SCTP support according to [2] (see [21] for details) and our Resource Pooling approach as described in section IV. We have validated the CMT-SCTP part against the CMT implementation of FreeBSD 8.0. The SimProcTC [22], [23] tool-chain has been used for parameterization and results processing. The results plots in this paper show the average values of at least 24 runs and their 95% confidence intervals.

Figure 2 illustrates our simulation setup: the client and the server are connected via two paths. Path #2 can be configured to either use the link between the two upper routers (i.e. "disjoint paths" scenario) or the bottleneck link shared with path #1 between the two lower routers (i.e. "shared bottleneck" scenario). Two data flows are running from the client to the server:

- **Background Flow:** This flow is used to cause congestion. By option, it can use standard SCTP, CMT-SCTP or CMT/RP-SCTP.
- **Reference Flow:** Used to evaluate the fairness of the transport. It uses non-CMT SCTP, i.e. it behaves similar to a single TCP flow. Path #1 is used as primary path.

The SCTP data transfer application NetPerfMeter [21, subsection 4.4] measures the payload data rate.

Unless otherwise specified, the basic simulation setup, as illustrated in figure 2, uses the following configuration parameters:

- The senders are saturated (i.e. they try to transmit as much data as possible); the message size is 1,452 bytes at an MTU of 1,500 bytes (i.e. the DATA chunk packets fully utilize the MTU [1]). The advertised receiver window is large enough to accept all data generated by the sender.
- After association establishment and transmission start, the actual throughput measurement is started after 19s. The duration of the throughput measurement is 30s.
- The bandwidths of the bottleneck and disjoint links between the routers of each path are configurable; their delay is 1ms (realistic for an Ethernet MPI setup [7]). All other links and the routing are delay-free. The bottleneck network interfaces use RED queues [24] ($w_q = 0.002$, $\text{min}_{\text{th}} = 20$, $\text{max}_{\text{th}} = 80$, $\text{max}_p = 0.02$).

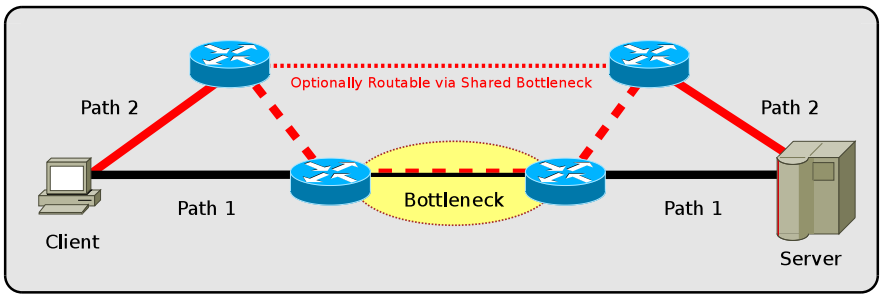


Figure 2. The Simulation Setup

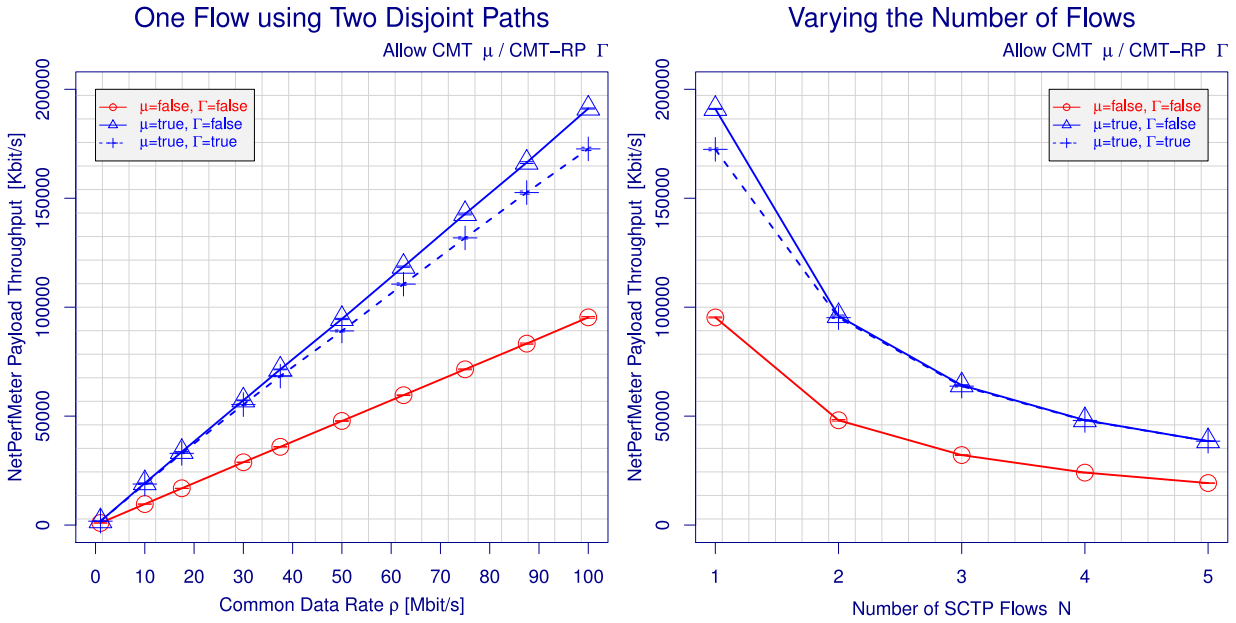


Figure 3. Payload Throughput of Background Flow(s) for Exclusive Usage of Two Disjoint Paths

VI. A Proof-of-Concept Analysis

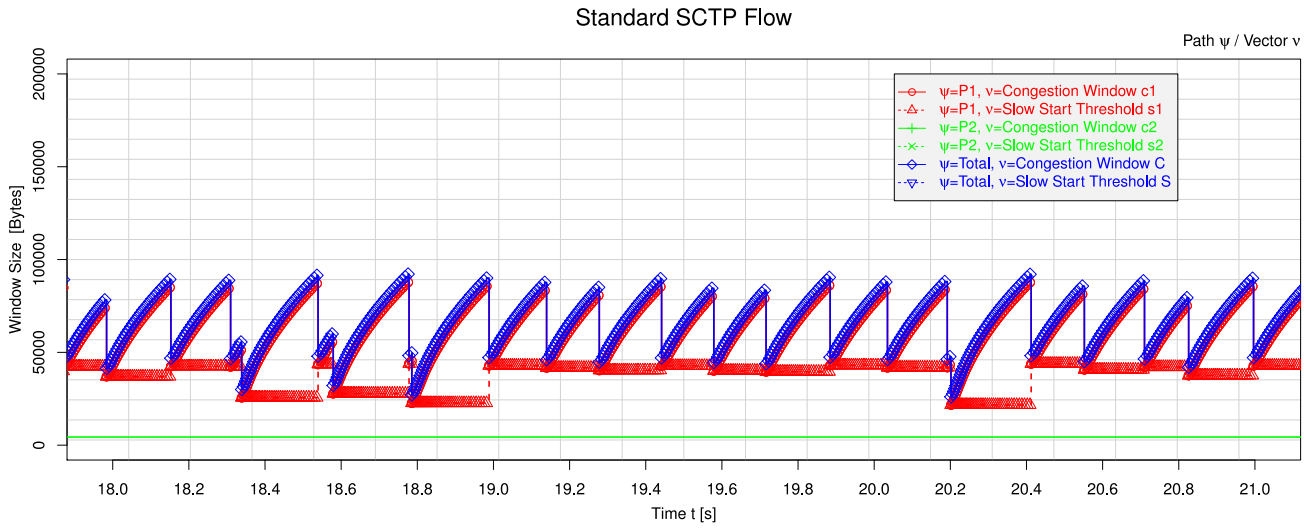
A. Exclusive Usage of Two Disjoint Paths

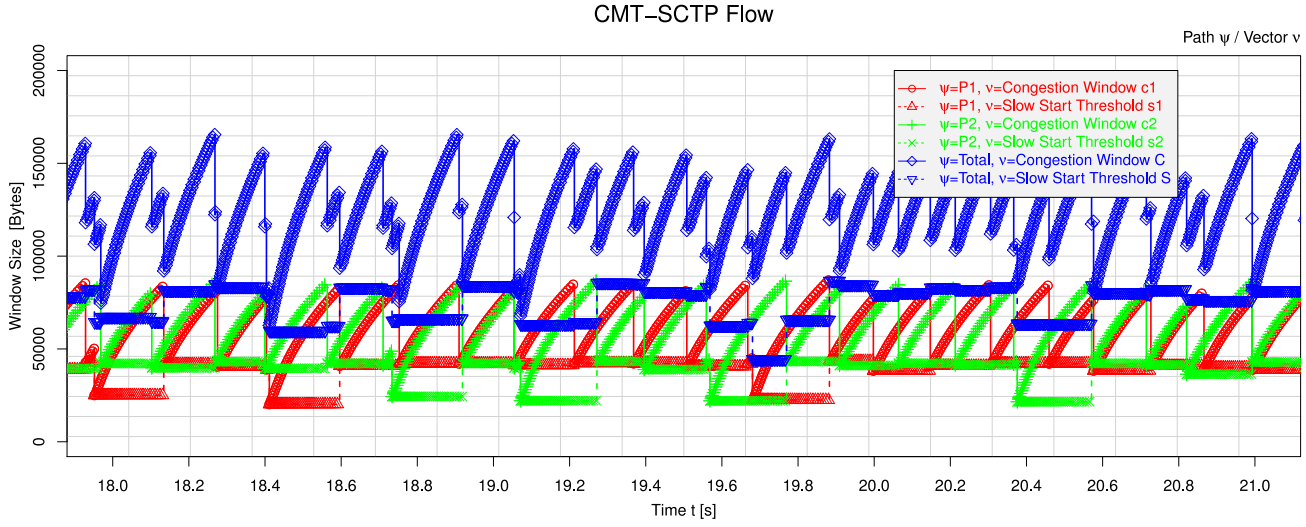
At first, we show that CMT and CMT/RP are actually working when a flow can use two disjoint paths exclusively. That is, we only use the "background flow" in our setup (see section V) and vary the data rate ρ of both router-to-router links. That is, the total usable data rate between the client and the server is $2 * \rho$ Mbit/s. The left-hand plot of figure 3 shows the resulting application payload throughput.

For CMT turned off (i.e. μ=false – denoted by red lines on a colour plot), the bandwidth linearly scales with the link data rate ρ – which is the expected behaviour. The payload throughput almost doubles when turning on CMT (i.e. μ=true – denoted by blue lines on a colour plot), e.g. from 95,000 Kbit/s (red line) to about 190,000 Kbit/s (solid blue line) at a router-to-router link bandwidth of ρ=100 Mbit/s on both paths. Turning on CMT/RP (denoted by dotted blue line), the achieved payload throughput at ρ=100 Mbit/s is about 171,000 Kbit/s. This is significantly better than for non-CMT SCTP (i.e. our first performance goal from section V is achieved). Of course, due to the less-aggressive congestion control, the throughput is smaller than for pure CMT at high bandwidths (here: $\rho \geq$50 Mbit/s). The quick congestion window reduction of CMT/RP on a packet loss (see equation 5 and equation 7) and its slower congestion window increase (see equation 9 and equation 10) lead to short periods of under-utilized paths.

In order to further investigate the implications of CMT and CMT/RP usage, figure 4 shows three seconds example plots of the congestion control states for non-CMT (upper plot), CMT (middle plot) and CMT/RP (lower plot) SCTP at ρ=100 Mbit/s. Solid lines represent the congestion windows c_1 (red colour), c_2 (green colour) and their total sum C (blue colour) for the paths P_1 and P_2; dotted lines show the corresponding slow start thresholds s_1 and s_2 as well as their total sum S.

Clearly, the results for standard SCTP (upper plot in figure 4) are not surprising: path P_1 is the primary path and utilized for the data transmission. Therefore, its congestion window and slow start threshold show the typical AIMD behaviour – analogously to TCP. Since path P_2





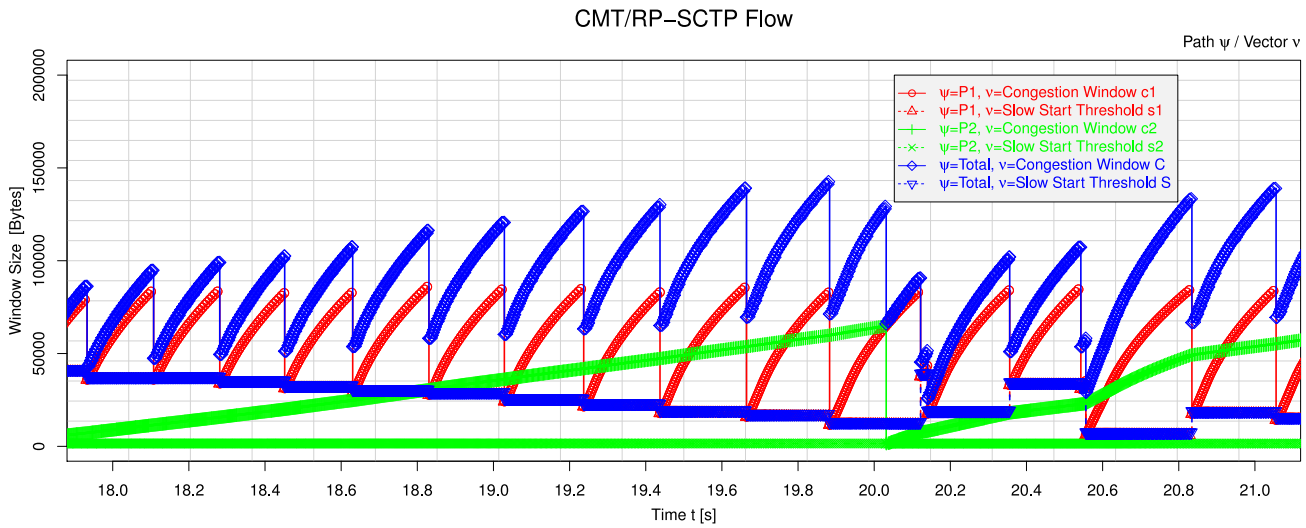


Figure 4. Congestion Window and Slow Start Threshold Behaviour Examples

is only used for some rare Timer-Based Retransmissions, its congestion window is still at the minimum; the slow start threshold has not even been set yet (no Timer-Based Retransmission of a Timer-Based Retransmission has occurred yet).

For CMT-SCTP (middle plot in figure 4), the congestion control behaviour on the two paths is like for two independent flows. Therefore, CMT-SCTP is able to fully utilize the two exclusively used links (see left-hand plot of figure 3). Note, that the behaviour of the total congestion window C and slow start threshold S is somewhat different from the AIMD behaviour of a single path: e.g. there may be a loss on path P_1 leading to a congestion window reduction, while the window for path P_2 still remains growing. Therefore, the total congestion window C may grow again from a level above S (e.g. around t=19.3s).

Comparing the plot for CMT/RP-SCTP (lower plot in figure 4) to the CMT-SCTP results (middle plot in figure 4), the curves of the two congestion windows c_1 and c_2 are eye-catching: in the example plot, c_2 grows very slowly in comparison to c_1. This effect is caused due to a Timer-Based Retransmission on path P_2 at a small setting of c_2 (in comparison to C): due to a large value of $\frac{C}{2}$, equation 7 sets c_2 to the minimum value. This value is also used in equation 8 to reset the slow start value s_2. Since this leads to a small fraction of s_2 in the total slow start threshold S, the following congestion window increases during congestion avoidance phase are small – due to the small scale factor $\frac{s_2}{S}$ (see equation 10). Therefore, the data rate on path P_2 is – for a short time – lower than it is for CMT-SCTP. Note, that the effect on path P_2 is only temporary. At some time later (not shown in the example plot), the slow start ratio changes (due to a Timer-Based Retransmission on path P_1) and leads to a quickly growing window c_2 while c_1 increments slowly now.

While the described effect of under-utilized paths reduces the throughput when there is a single flow only, it becomes negligible when there are multiple flows. The right-hand plot of figure 3 presents the throughput results for increasing the number of the "background flows" from $N = 1$ to $N = 5$. Already at $N = 2$, no significant difference is visible between the throughputs of CMT-SCTP (solid blue line on a colour plot) and CMT/RP-SCTP (dotted blue line). That is, in realistic scenarios using multiple flows, the temporary under-utilization caused by the congestion window update of CMT/RP-SCTP does not cause problems.

In order to examine our second goal from section V – bandwidth fairness to non-CMT flows – we have to examine concurrency situations.

B. Concurrency Scenarios

We now use the non-SCTP "reference flow" (see section V) and vary the CMT parameters of the "background flow". For CMT/RP, the reference flow is expected to get a fair bandwidth share – despite of the multi-homed background traffic.

1) Two Disjoint Paths: In the first scenario, the two paths are disjoint (see figure 2). The left-hand side of figure 5 presents the application payload throughput of the reference flow for varying the router-to-router link bandwidth ρ. In case of pure CMT background traffic (i.e. Γ=false), the non-CMT reference flow achieves a throughput of about $\frac{\rho}{2}$. That is, for ρ=100 Mbit/s, the background flow may utilize path #2 (nearly) exclusively, while the non-CMT reference flow has to share the bandwidth on path #1. That is, only half of path #1's capacity remains for the reference flow.

Activating CMT/RP (i.e. Γ=true), the bandwidth is shared more fairly: congestion occurs on path #1, since this path is shared with the non-CMT reference flow. The background flow B experiences no congestion on path #2, which leads to larger slow start threshold s_2^B and congestion window c_2^B on this path. When a loss occurs on the congested path #1, it quickly reduces its already-smaller slow start threshold s_1^B by $\frac{c_1^B+c_2^B}{2}$ (see equation 6 and equation 8) and after that only slowly increases the congestion window c_1^B using the scale factor $\frac{s_1^B}{S^B}$ (see equation 9 and equation 10). Therefore, the background flow "concentrates" its load on the exclusively used path #2, leaving more capacity on the shared path #1 for the non-CMT reference flow (which cannot utilize path #2).

2) Two Paths over Shared Bottleneck: For the second scenario, the two paths share a single bottleneck link (see figure 2). The right-hand side of figure 5 depicts the resulting application payload throughput of the reference flow. As expected, pure CMT background traffic (i.e. Γ=false) reduces the throughput to about $\frac{1}{3}$ of the bottleneck link bandwidth ρ. The CMT flow just behaves like two separate TCP flows and utilizes $\frac{2}{3}$ of the capacity. Turning CMT/RP on (i.e. Γ=true) resolves this unfairness. The reference flow reaches a throughput of about $\frac{\rho}{2}$, leading to a fair 50%:50% share of the link capacity between the two concurrent flows.

C. Summary

As shown in our proof-of-concept evaluation, CMT/RP achieves the two goals from section V:

1) The application data throughput is better than for standard SCTP.
2) The bandwidth is fairly shared among the flows in concurrency situations.

VII. Conclusions

The idea of CMT is to utilize all paths of multi-homed endpoints in order to increase application data throughput. But handling the congestion control of each path independently lacks of fairness against non-CMT flows. In this paper, we have introduced our approach of combining CMT with the idea of Resource Pooling into an improved congestion control scheme which is aware of the path interaction. CMT/RP-SCTP is the realization of our approach for the SCTP transport protocol. In a simulative proof-of-concept analysis, we have shown that

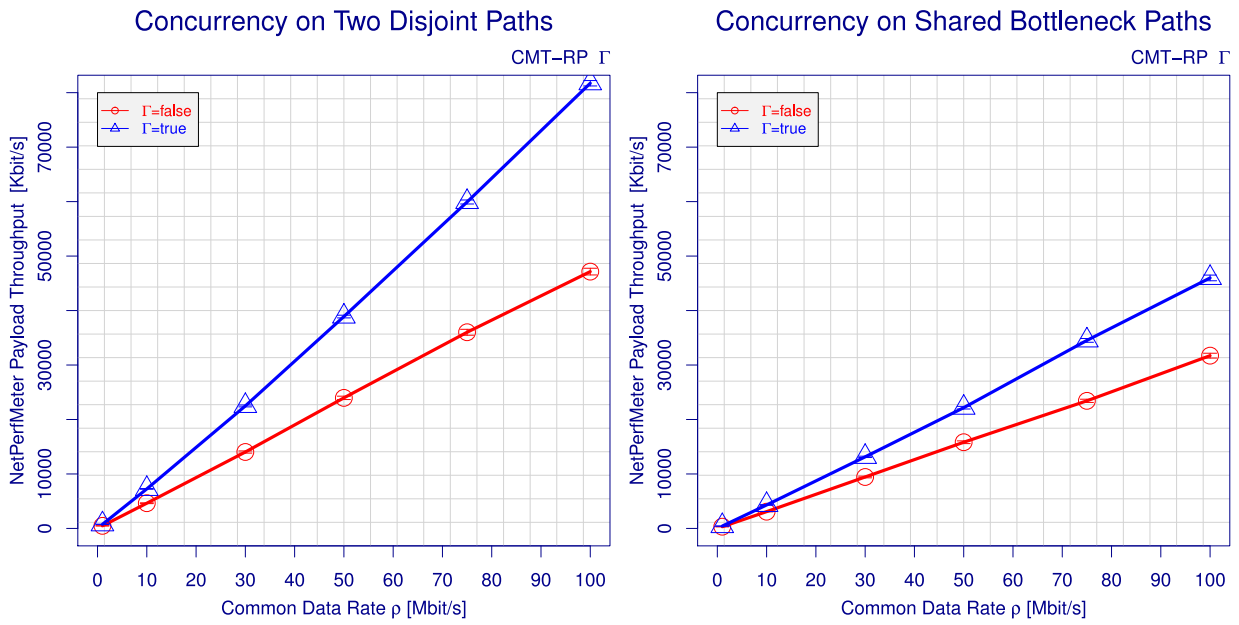


Figure 5. Payload Throughput of Reference Flow for Concurrency Scenario

CMT/RP achieves an improvement over standard SCTP while remaining fair to concurrent flows. Nevertheless, our approach is generic and may be transferred to other transport protocols as well.

As part of our ongoing work on CMT, we are going to perform detailed parameter studies in different scenarios in order to further analyse its behaviour and improve its performance. Also, we are going to realize our approach in the FreeBSD network stack, in order to test and analyse it in real life. Finally, we also intend to bring our ideas and improvements from research to application by contributing results into the IETF standardization process.

References

[1] R. Stewart, "Stream Control Transmission Protocol," IETF, Standards Track RFC 4960, Sept. 2007.

[2] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 951–964, 2006.

[3] D. Wischik, M. Handley, and M. B. Braun, "The Resource Pooling Principle," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 5, pp. 47–52, 2008.

[4] M. Honda, Y. Nishida, L. Eggert, P. Sarolahti, and H. Tokuda, "Multipath Congestion Control for Shared Bottleneck," in *Proceedings of the 7th International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT)*, Tokyo/Japan, May 2009.

[5] C. Raiciu, M. Handley, and D. Wischik, "Practical Congestion Control for Multipath Transport Protocols."

[6] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Overlay tcp for multi-path routing and congestion control," in *Proceedings of the ENS-INRIA ARC-TCP Workshop*, Paris/France, Nov. 2003.

[7] B. Penoff, M. Tsai, J. Iyengar, and A. Wagner, "Using CMT in SCTP-based MPI to Exploit Multiple Interfaces in Cluster Nodes," in *Proceedings of the EuroPVM/MPI*, Paris/France, Sept. 2007.

[8] A. Jungmaier, E. P. Rathgeb, and M. Tüxen, "On the Use of SCTP in Failover-Scenarios," in *Proceedings of the State Coverage Initiatives, Mobile/Wireless Computing and Communication Systems II*, vol. X, Orlando, Florida/U.S.A., July 2002, ISBN 980-07-8150-1.

[9] A. Jungmaier, M. Schopp, and M. Tüxen, "Performance Evaluation of the Stream Control Transmission Protocol," in *Proceedings of the IEEE Conference on High Performance Switching and Routing*, Heidelberg/Germany, June 2000, pp. 141–148.

[10] A. L. Caro, K. Shah, J. R. Iyengar, P. D. Amer, and R. R. Stewart, "SCTP and TCP Variants: Congestion Control Under Multiple Losses," Computer and Information Sciences Department, University of Delaware, Newark, Delaware/U.S.A., Tech. Rep. TR2003-04, Feb. 2003.

[11] I. Rüngeler, M. Tüxen, and E. P. Rathgeb, "Congestion and Flow Control in the Context of the Message-Oriented Protocol SCTP," in *Proceedings of the 8th International IFIP-TC 6 Networking Conference*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 468–481.

[12] R. Stewart, Q. Xie, M. Tüxen, S. Maruyama, and M. Kozuka, "Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration," IETF, Standards Track RFC 5061, Sept. 2007.

[13] T. Dreibholz, A. Jungmaier, and M. Tüxen, "A new Scheme for IP-based Internet Mobility," in *Proceedings of the 28th IEEE Local Computer Networks Conference (LCN)*, Königswinter/Germany, Nov. 2003, pp. 99–108, ISBN 0-7695-2037-5.

[14] R. Stewart, M. Ramalho, Q. Xie, M. Tüxen, and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial

Reliability Extension," IETF, Standards Track RFC 3758, May 2004.

[15] T. Dreibholz and E. P. Rathgeb, "Towards the Future Internet – An Overview of Challenges and Solutions in Research and Standardization," in *Proceedings of the 2nd GI/ITG KuVS Workshop on the Future Internet*, Karlsruhe/Germany, Nov. 2008.

[16] P. Lei, L. Ong, M. Tüxen, and T. Dreibholz, "An Overview of Reliable Server Pooling Protocols," IETF, Informational RFC 5351, Sept. 2008.

[17] T. Dreibholz and E. P. Rathgeb, "Overview and Evaluation of the Server Redundancy and Session Failover Mechanisms in the Reliable Server Pooling Framework," *International Journal on Advances in Internet Technology (IJAIT)*, vol. 2, no. 1, pp. 1–14, June 2009.

[18] T. Dreibholz, "Reliable Server Pooling – Evaluation, Optimization and Extension of a Novel IETF Architecture," Ph.D. dissertation, University of Duisburg-Essen, Faculty of Economics, Institute for Computer Science and Business Information Systems, Mar. 2007.

[19] A. Varga, *OMNeT++ Discrete Event Simulation System User Manual – Version 4.0*, Oct. 2009.

[20] I. Rüngeler, M. Tüxen, and E. P. Rathgeb, "Integration of SCTP in the OMNeT++ Simulation Environment," in *Proceedings of the 1st OMNeT++ Workshop*, Marseille/France, Mar. 2008, ISBN 978-963-9799-20-2.

[21] T. Dreibholz, M. Becke, J. Pulinthanath, and E. P. Rathgeb, "Implementation and Evaluation of Concurrent Multipath Transfer for SCTP in the INET Framework," in *Proceedings of the 3nd ACM/ICST OMNeT++ Workshop*, Málaga/Spain, Mar. 2010.

[22] T. Dreibholz, X. Zhou, and E. P. Rathgeb, "SimProcTC – The Design and Realization of a Powerful Tool-Chain for OMNeT++ Simulations," in *Proceedings of the 2nd ACM/ICST OMNeT++ Workshop*, Rome/Italy, Mar. 2009, ISBN 978-963-9799-45-5.

[23] T. Dreibholz and E. P. Rathgeb, "A Powerful Tool-Chain for Setup, Distributed Processing, Analysis and Debugging of OMNeT++ Simulations," in *Proceedings of the 1st ACM/ICST OMNeT++ Workshop*, Marseille/France, Mar. 2008, ISBN 978-963-9799-20-2.

[24] S. Floyd and V. Jacobsen, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.