Scheduling für Multipath-Transportprotokolle

Daniel Sarnow*

daniel.sarnow@haw-hamburg.de

31. August 2017

Abstract – Mit der drastischen Zunahme von mobilen Endgeräten – die über mehrere Netzwerkschnittstellen verfügen – und einer vermehrten Nutzung von internetbasierten Diensten, wurden *Concurrent Multipath Transfer* (CMT) Erweiterungen für Transportprotokolle geschaffen. Diese Erweiterungen erlauben es gleichzeitig über alle verfügbaren Netzwerkschnittstellen Applikationsdaten zu übertragen. Mit CMT kann nicht nur der Durchsatz erhöht, sondern auch ein besserer Schutz gegen Netzwerkausfälle gewährleistet werden.

Die Relevanz dieser Entwicklung wird besonders in den Forderungen der Europäischen Kommission deutlich. Die Europäische Kommission fordert im Wirtschaftsprogramm Europa 2020, dass *alle* Bürger bis zum Jahr 2020 die Möglichkeit bekommen sollen das Internet mit einer Downloadrate von *mindestens* 30 Mbit/s nutzen zu können (Europäische Kommission, 2010).

Um Daten gleichzeitig über mehrere Netzwerkschnittstellen versenden zu können benötigen CMT-Erweiterungen unter anderem einen *Scheduler*. Die Aufgabe des *Schedulers* ist es Datenpakete auf die Netzwerkschnittstellen zu verteilen.

Diese Ausarbeitung beschreibt die Grundlagen und Herausforderungen für die Datenübertragung mit CMT-Transportprotokollen, die Notwendigkeit von effizienten Schedulingalgorithmen und bereits existierende Scheludingansätze, um den aktuellen Forschungsstand aufzuzeigen.

* *Grundseminar Master, Department Informatik, HAW Hamburg*

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

1 EINLEITUNG

Am Anfang des Internetzeitalters wurde mit dem RFC 793 (Postel, 1981) das Transmission Control Protocol (TCP) bei der Internet Engineering Task Force (IETF) standardisiert. Bis heute ist es eins der wichtigsten und meistgenutzten Transportprotokolle im Internet. Doch die Anforderungen und das Nutzerverhalten im Internet verändern sich laufend, was auch auf Seiten der Transportprotokolle eine Weiterentwicklung notwendig macht.

Die Notwendigkeit für die Weiterentwicklung von Transportprotokollen lässt sich aus den steigenden Anforderungen der Nutzer ableiten. So fordert die Europäischen Kommission im Wirtschaftsprogramm Europa 2020, dass *alle* Bürger bis zum Jahr 2020 die Möglichkeit bekommen sollen das Internet mit einer Downloadrate von *mindestens* 30 Mbit/s nutzen zu können. Dies sei nach Auffassung der europäischen Kommission notwendig, um dem Nutzerverhalten gerecht zu werden und auf dem internationalen Markt konkurrenzfähig zu bleiben (Europäische Kommission, 2010).

Das Ziel bis zum Jahr 2020 europaweit einen flächendeckenden Zugang zum Internet zu schaffen und die Downloadrate entsprechend zu steigern kann auf zwei Wegen erfolgen: Eine Möglichkeit ist der Ausbau beziehungsweise die Erneuerung der bestehenden Infrastruktur. Die andere Möglichkeit ist eine effizientere Nutzung der vorhandenen Ressourcen.

Wie aus Eurostat, 2017 ersichtlich wird gibt es in Europa durchaus noch Haushalte, die sehr schlecht bis gar nicht an das Internet angeschlossen sind. Daher wird es notwendig sein die bestehende Infrastruktur zu erneuern beziehungsweise neue Regionen zu erschließen. Da diese Maßnahmen jedoch mit erheblichen Kosten verbunden sind ist es zudem sinnvoll die zur Verfügung stehenden Ressourcen optimal zu nutzen.

Wie gut die zur Verfügung stehenden Ressourcen genutzt werden können hängt zu einem großen Teil von der Effizienz der Transportprotokolle ab. Eine Verbesserung im Bereich der Transportprotokolle hat den großen Vorteil, dass diese Änderungen auf eine Vielzahl von Endgeräten angewendet werden und somit global wirken können. Im Gegensatz dazu können Infrastrukturänderungen immer nur regional wirken.

Somit könnte, wie auch in Detal, 2017 beschrieben, *Concurrent Multipath Transfer* ein möglicher Weg sein, um die vorhandenen Ressourcen besser zu nutzen und die Ziele der Europäischen Kommission zu erreichen. *Concurrent*

Multipath Transfer (CMT) erlaubt es gleichzeitig über mehrere Pfade Applikationsdaten zu versenden. Transportprotokolle, die CMT unterstützen, werden im Folgenden Multipath-Transportprotokolle genannt. Die gleichzeitige Nutzung mehrerer Pfade kann nicht nur den Durchsatz erhöhen, sondern auch einen besseren Schutz gegen Netzwerkausfälle gewährleisten. Da in den vergangenen Jahren der Anteil der mobilen multipathfähigen Endgeräte, wie zum Beispiel Smartphones, stark zugenommen hat, ist es möglich diese zusätzlichen Ressourcen durch CMT nutzbar zu machen.

Implementierungen für CMT sind bereits für die Transportprotokolle TCP und das Stream Control Transmission Protocol (SCTP), welches in RFC 4960 (Stewart, 2007) beschrieben ist, vorhanden. Die CMT-Erweiterung für TCP ist unter dem Namen Multipath TCP (MPTCP) bekannt und in RFC 6824 (Ford u. a., 2013) beschrieben. Die CMT-Erweiterung für SCTP heißt CMT-SCTP und ist in Iyengar, Amer und Stewart, 2006 beschrieben. Auch das QUIC Protokolls sieht eine CMT-Erweiterung vor, siehe hierzu Hamilton u. a., 2016. Auch wenn schon existierende Implementierungen für CMT-Erweiterungen vorhanden sind, gibt es besonders in heterogenen Umgebungen noch einige Herausforderungen, die es zu bewältigen gilt.

Diese Herausforderungen werden in Abschnitt 2 – Grundlagen & Herausforderungen genauer beschrieben. In Abschnitt 3 – Schedulingalgorithmen werden Ansätze für effizientere Schedulingalgorithmen vorgestellt, welche die beschriebenen Herausforderungen überkommen sollen. In Abschnitt 4 – Experimente & Experimentierplatformen werden die Beiträge bezüglicher der durchgeführten Experimente und ausgewählten Experimentierplatformen verglichen. Anschnitt 5 – Relevante Konferenzen gibt einen kurzen Überblick über relevante Konferenzen in diesem Forschungsgebiet. Abschnitt 6 – Ausblick fasst den aktuellen Forschungsstand zusammen und gibt einen Ausblick für die Projektarbeit im Master.

2 GRUNDLAGEN & HERAUSFORDERUNGEN

In diesem Abschnitt werden die Grundlagen und Herausforderungen bei der Datenübertragung mit *Concurrent Multipath Transfer* (CMT) erläutert.

Ein einfaches CMT Szenario ist in Abbildung 1 dargestellt. Das Internet wird in Abbildung 1 sehr stark vereinfacht durch ein Ansammlung von Routern dargestellt. Zwischen den beiden Kommunikationsteilnehmern A und B können Applikationsdaten über den rot oder blau markierten Pfad übertragen werden.

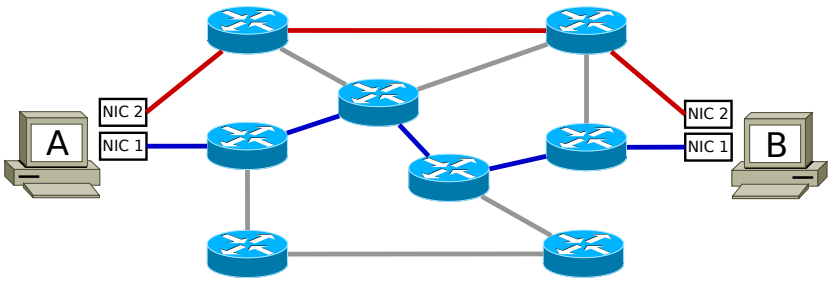

Abbildung 1: Einfaches CMT Szenario

Die einzelnen Pfade können sich in ihren Eigenschaften – *Quality-of-Service* – teilweise stark unterscheiden, was zum Beispiel bei mobilen Endgeräten wie Smartphones eher die Regel als die Ausnahme ist (Ferlin u. a., 2016).
Die Eigenschaften eines Pfades sind wie folgt definiert:

- Der **Durchsatz** beschreibt die pro Zeiteinheit übertragene Datenmenge.
- Die **Latenz** beschreibt die Verweildauer eines Paketes innerhalb des Netzwerkes.
- Die **Paketverlustrate** gibt Auskunft über den prozentualen Anteil verlorengegangener Pakete.
- **Jitter** beschreibt die Laufzeitschwankungen zwischen übertragenen Datenpaketen.

Die Leistung und Effizienz von Multipath-Transportprotokollen kann durch viele Faktoren beeinflusst werden, wie die Arbeit von Raiciu u. a., 2012 am Beispiel von Multipath TCP zeigt. Ein wesentlicher Einflussfaktor ist hierbei der gewählte Schedulingalgorithmus. Der Scheduler hat die Aufgabe Daten auf die einzelnen Pfade zu verteilen.

Die Abbildung 2 zeigt ein Schedulingbeispiel bei dem Round-Robin als Schedulingalgorithmus eingesetzt wird. Bei dem Round-Robin Schedulingalgorithmus werden die Datenpakete reihum auf die Pfade verteilt. Die Datenpakete werden also sehr gleichmäßig auf die Pfade verteilt. Durch gleichmä-

ßige Verteilung der Daten können die Ressourcen der einzelnen Pfade somit voll ausgeschöpft werden.

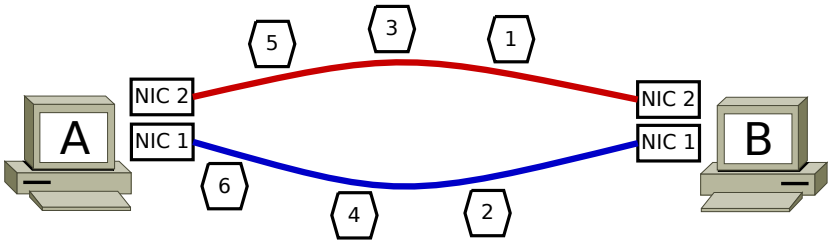

Abbildung 2: Round-Robin Schedulingbeispiel 1

In einem Szenario annähernd gleicher Pfade kann dieser Schedulingalgorithmus durchaus den Durchsatz erhöhen und die zur Verfügung stehenden Ressourcen optimal nutzen. Unterscheiden sich jedoch die Eigenschaften der Pfade, wie es in Abbildung 3 angedeutet ist, dann ist der Round-Robin Schedulingalgorithmus nicht unbedingt die beste Wahl. In Abbildung 3 ist ein Datenpaket auf dem roten Pfad wesentlich länger im Netzwerk unterwegs bevor es bei B eintrifft als auf dem blauen Pfad. Die Latenz des roten Pfades ist also größer als die des blauen Pfades.

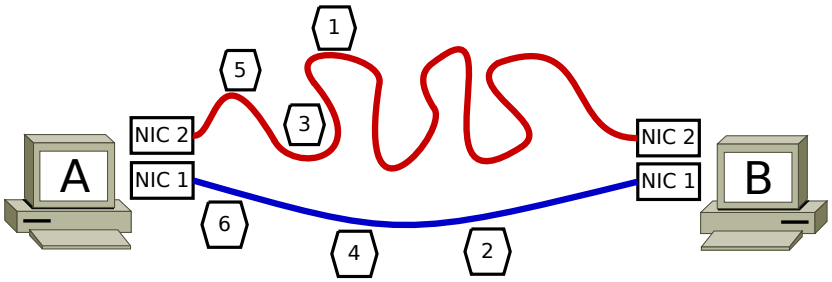

Abbildung 3: Round-Robin Schedulingbeispiel 2

Im Beispiel aus Abbildung 3 würde das Scheduling der Datenpakete dazu führen, dass die Datenpakete 2,4 und 6 vor dem Datenpaket 1 bei B eintreffen. Die Datenpakete 2,4 und 6 kommen also *out-of-order* bei B an und müssen im *Receive Buffer* zwischengespeichert werden, da Multipath TCP Daten vollständigen in Reihenfolge gesichert überträgt. Dieses Phänomen wird auch als Head-of-line (HOL) Blocking bezeichnet. Da die Daten nur *in-order* an die Applikation weitergereicht werden dürfen, kommt es bei HOL Blocking zu stoßartigen Auslieferungen von Daten an die Applikationsebene. HOL Blocking ist somit besonders für interaktive Applikationen problematisch.

Ein weiteres Problem, das in Paasch u. a., 2014a beschrieben wird, nennt sich Receive-Window Limitation. Sowohl im TCP als auch im SCTP Stack gibt es einen *Receive Buffer* für *out-of-order* Datenpakete, die durch Umsortierungen – wie in Abbildung 3 angedeutet – oder Paketverluste entstehen können. Um alle Pfade optimal nutzen zu können ist also die Größe des *Receive Buffers* ein entscheidender Faktor. Für Multipath TCP wurde eine Buffergröße von Barré, Paasch und Bonaventure, 2011, wie in Gleichung 1 dargestellt, definiert.

$$\text{Buffer} = \sum_{i}^{n} bw_i \cdot \text{RTT}_{max} \cdot 2 \tag{1}$$

Diese Buffergröße ist ausreichend um jeden Pfad im Zeitintervall der größten Round-Trip Time maximal auszulasten ($\sum_i^n bw_i \cdot \text{RTT}_{max}$) und Paketverluste in einem gewissen Umfang verkraften zu können ($\cdot 2$). Die Round-Trip Time (RTT) gibt die Zeit an, die ein Datenpaket benötigt um von A nach B zu gelangen und wieder zurück. Besonders in Szenarien ungleicher Pfade kann es durch Umsortierungen der Datenpakete dazu kommen, dass die Buffergröße nicht ausreichend ist, um alle *out-of-order* Datenpakete aufzunehmen. Zudem gibt es Endgeräte, die nicht in der Lage sind ausreichend Speicherplatz für den *Receive Buffer* zu allozieren (Paasch u. a., 2014a).

Schlechte Schedulingentscheidungen können somit zu Head-of-Line Blocking und Receive-Window Limitations führen, woraus folgt, dass Daten nur stoßartig ausliefert werden und der Gesamtdurchsatz gemindert werden kann. Besonders in Szenarien stark ungleicher Pfade kann der Gesamtdurchsatz so stark abfallen, dass dieser geringer ist als der Durchsatz der nur über den besten Pfad erreicht werden würde. Ein solches Verhalten verletzt zudem die erste von drei Design Regeln, die für die Entwicklung von Multipath-Transportprotokollen etabliert wurden.

Diese Design Regeln wurden in Wischik, Handley und Braun, 2008 und Raiciu, Wischik und Handley, 2009 definiert. Die für Schedulingalgorithmen relevanten Regeln, lauten wie folgt:

- **Improve Throughput** fordert, dass der Durchsatz in einem Multipath-Szenario mindestens so gut sein soll, wie der Durchsatz über den besten Pfad in einem Single-Path-Szenario.

- **Balance Congestion** fordert, ein stark genutzter Pfad – *congested path* – entlastet werden soll, indem die folgenden Datenpakete auf weniger stark genutzte Pfade verteilt werden.

Die bisherigen Ausführungen haben gezeigt, dass schlechte Schedulingentscheidungen negative Auswirkungen auf die Leistung und Effizienz von Multipath-Transportprotokollen haben können und es somit notwendig ist effiziente und robuste Schedulingalgorithmen zu entwickeln.

3 SCHEDULINGALGORITHMEN

Dieser Abschnitt stellt einige Schedulingalgorithmen für Multipath-Transportprotokolle vor. Aufgrund der großen Anzahl der Arbeiten in diesem Bereich können nicht alle Schedulingansätze in dieser Ausarbeitung vorgestellt werden. Die ausgewählten Ansätze beschränken sich auf Arbeiten, die entweder als sehr relevant erscheinen oder den aktuellen Forschungsstand besonders gut widerspiegeln. Für den interessierten Leser sind der Vollständigkeit halber noch einige der nicht ausgewählten Arbeiten aufgeführt: Chan, Tseng und Yen, 2016, Oh und Lee, 2015, Kuhn u. a., 2014, Ni u. a., 2014 und Sarwar u. a., 2013.

3.1 Multipath TCP Scheduler

Wie bereits in Abschnitt 2 dargestellt wurde ist der Round-Robin Schedulingalgorithmus nicht die beste Wahl, besonders nicht in heterogenen Umgebungen. Somit ist es in heterogenen Umgebungen sinnvoller Schedulingentscheidungen anhand der Latenz der Pfade – *delay-based* – zu treffen. In Multipath TCP wird daher der LowRTT Scheduler eingesetzt.

Der LowRTT Scheduler nutzt zuerst den Pfad mit der geringsten Round-Trip Time und sendet so lange Datenpakete über diesen Pfad bis das Congestion Window *cwnd* erschöpft ist. Danach wird der Pfad mit der nächst größeren Round-Trip Time ausgewählt (Raiciu u. a., 2012). Ein mögliches Schedulingbeispiel ist in Abbildung 4 dargestellt. Das Congestion Window ist eine Größe, die angibt wie viele Bytes auf einem Pfad unterwegs sein dürfen. Diese Größe

ist Teil des Congestion Control Mechanismuses, der das Ziel hat einen fairen Wettbewerb um Ressourcen zwischen den Teilnehmern zu gewährleisten.

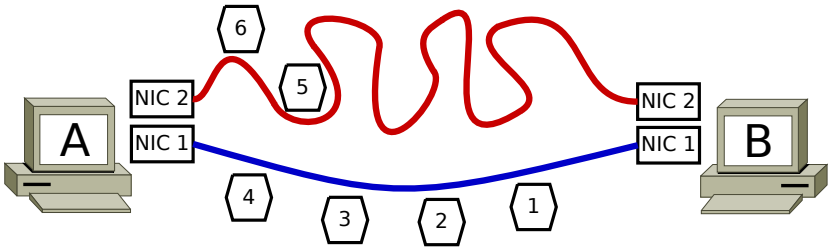

Abbildung 4: Low-RRT Schedulingbeispiel

Paasch u. a., 2014a hat sich mit der experimentellen Evaulation, basierend auf Paasch, Khalili und Bonaventure, 2013, des LowRTT Schdulers von Multipath TCP beschäftigt und folgendes festgestellt:

Der LowRTT Scheduler zeigt eine bessere Leistung als der Round-Robin Scheduler in heterogenen Umgebungen. Jedoch besteht – wie auch bei dem Round-Robin Scheduler – das Problem, dass das Scheduling *ack-clocked* wird sobald alle Congestion Windows gefüllt sind. *Ack-clocked* beschreibt einen Phänomen, bei dem neue Datenpakete über den Pfad versendet werden, dessen Congestion Window infolge eines Acknowledgments wieder frei geworden ist. Dies muss aber nicht immer die beste Schedulingentscheidung sein.

Außerdem gibt es noch zwei Erweiterungen für den LowRTT Scheduler, die auf HOL Blocking und Receive-Window Limitations reagieren sollen, die Experimente aus Paasch u. a., 2014a haben jedoch gezeigt, dass diese Maßnahmen nicht immer ausreichend sind und in einigen Szenarien sogar negative Auswirkungen haben können.

Paasch u. a., 2014a fordert, dass Schedulingentscheidungen so getroffen werden sollen, dass Datenpakete idealerweise *in-order* beim Empfänger eintreffen. Ein solches Vorgehen würde das Auftreten von HOL Blocking und Receive-Window Limitations stark minimieren.

3.2 OTIAS

Der von Yang, Wang und Amer, 2014 vorgeschlagene Schedulingansatz versucht Daten *out-of-order* zu verschicken, damit sie *in-order* beim Empfänger ankommen, daher steht die Abkürzung OTIAS auch für *Out-of-order Transmission for in-order Arrival Scheduler*.

Die grundlegende Idee dieses Ansatzes ist es mehr Datenpakete pro Pfad zur Versendung zu planen, als zu dem Zeitpunkt tatsächlich versendet werden können. Um mehr Pakete pro Pfad planen zu können werden Queues benötigt. Die sich in der Queue befinden Datenpakete können sobald versendet werden, wie wieder Platz im Congestion Window vorhanden ist.

Um ein Datenpaket einem Pfad zuordnen zu können wird die vermutliche Ankunftszeit des Datenpaketes pro Pfad berechnet und dann der Pfad mit der frühsten Ankunftszeit ausgewählt. Sofern es einen Pfad gibt, dessen Congestion Window noch nicht ausgeschöpft ist, wird das Datenpaket über diesen Pfad sofort versendet. Andernfalls wird anhand der Round-Trip Time, des Congestion Window und der Anzahl der Datenpakete, die sich in der Queue befinden, die Ankunftszeit errechnet und der entsprechende Pfad ausgewählt.

Im Vergleich zum LowRTT Scheduler nutzt der OTIAS Scheduler mehr Informationen der Pfade um Schedulingentscheidungen zu treffen. Allerdings kann der OTIAS Scheduler durch die verwendeten Queues pro Pfad nicht so dynamisch auf verspätete oder verlorengegangene Datenpakete reagieren. Die warteten Pakete in der Queue eines blockierten Pfades können erst später als erwartet ausgeliefert werden und können somit die Paketplanung durcheinander bringen. Des Weiteren ist die Wartezeit von Datenpakete die erneut gesendet werden müssen aufgrund der recht starren Planung unter Umständen größer als beim LowRTT Scheduler, da diese ebenfalls in eine Queue eingereiht werden müssen.

3.3 BLEST

Der *Blocking Estimation-Based MPTCP Scheduler* (BLEST) wurde von Ferlin u. a., 2016 vorgeschlagen und versucht HOL Blocking zu minimieren und somit besonders in heterogenen Umgebungen die Leistung von MPTCP zu steigern.

Dieser Ansatz versucht anhand der Wahrscheinlichkeit, dass HOL Blocking auftritt zu bestimmen über welchen Pfad ein Datenpaket versendet wird. Basierend auf der RTT der Pfade berechnet der BLEST Algorithmus, ob es zu HOL Blocking kommen könnte, wenn das Datenpaket über den langsameren Pfad versendet wird.

Somit folgt der BLEST Ansatz grundlegend dem LowRTT Scheduler, in dem Pfade mit kleinen RTT denen mit größeren RTTs vorgezogen werden. Al-

lerdings nutzt der LowRTT Scheduler jeden Pfad, sofern noch Platz im Congestion Window vorhanden ist. Der BLEST Scheduler kann auf der anderen Seite den langsamen Pfad überspringen und warten bis der schnellere Pfad wieder frei ist. Die Wahrscheinlichkeit des Auftretens von HOL Blocking kann durch Warten kleiner sein, als wenn das Datenpaket sofort über den langsameren Pfad versendet wird.

Experimente zur Evaluation dieses Ansatzes wurden sowohl in einer Simulationsumgebung als auch in einem Real-World Umgebung durchgeführt. In den gezeigten Experimenten ist der BLEST Ansatz den anderen Ansätzen stets überlegen. Allerdings scheint der Ansatz nur für Szenarien mit zwei Pfaden ausgelegt zu sein.

Ferlin u. a., 2016 sehen den BLEST und OTAIS Ansatz als eine gute Grundlage um in Zukunft robuste und effektive Schedulingalgorithmen für heterogene Umgebungen zu entwickeln. Des Weiteren wird angemerkt, dass die Evaluation dieser Ansätze noch weiter, basierend auf den Ansätzen von Paasch, 2014b, ausgebaut werden sollte.

3.4 Adaptive Data Chunk Scheduler

In der Arbeit von Verma und Kumar, 2017 wird ein Schedulingansatz vorgeschlagen, der die Latenz und den Durchsatz eines Pfades als Faktoren einbezieht um Schedulingentscheidungen zu treffen. Ziel soll es somit sein mehr Datenpakete über den Pfad mit der geringsten Latenz und höchsten Bandbreite zu übertragen.

Der vorgeschlagene Ansatz bestimmt aus der RRT, RTT_{min} und dem Congestion Window $cwnd$ pro Pfad i die tatsächliche Übertragungsrate, erwartete Übertragungsrate und deren Differenz, wie Gleichung 2 zeigt.

$$\begin{aligned} Actual_i &= \frac{cwnd_i}{RTT_i} \\ Expected_i &= \frac{cwnd_i}{RTT_{min_i}} \\ Difference_i &= (Expected_i - Actual_i) \cdot RTT_{min_i} \end{aligned} \tag{2}$$

Die erwartete Übertragungsrate ist die theoretische Übertragungsrate des Pfades i, wenn der Netzwerkpfad kaum genutzt wird. Die tatsächliche Übertragungsrate beschreibt die momentane Übertragungsrate des Pfades i. Wenn

sich nun die Auslastung des Pfades ändert, dann ändert sich auch die Differenz. Wenn die Differenz sehr groß ist, dann ist auch die Auslastung des Pfades sehr groß. Die Differenz repräsentiert also die Auslastung des Pfades und kann somit für das Scheduling als Faktor genutzt werden. Das Congestion Window $cwnd$ wird in diesem Ansatz entsprechend der Differenz manipuliert, wie in Gleichung 3 dargestellt, um Schedulingentscheidungen hervorzurufen.

$$cwnd_{i+1} = \begin{cases} cwnd_i & \text{if} \quad Difference_i > \delta \\ cwnd_i + \frac{MTU}{2} & \text{if} \quad \lambda \leqslant Difference_i < \delta \\ cwnd_i + MTU & \text{else} \end{cases} \tag{3}$$

Die Schwellenwerte $\lambda = 1$ und $\delta = 3$ wurden von Verma und Kumar, 2017 durch Experimente ermittelt und werden benötigt, um zu entscheiden wie stark der Pfad ausgelastet ist und wie dementsprechend das Congestion Window $cwnd$ verändert werden muss, siehe Gleichung 3. Ist die Differenz kleiner als λ dann ist der Pfad kaum ausgelastet und das $cwnd$ wird entsprechend um eine MTU – *Maximum Transmission Unit* – erhöht werden. Ist die Differenz größer als δ dann ist der Pfad schon stark ausgelastet und das $cwnd$ wird nicht mehr verändert.

Dieser Schedulingansatz wurde von Verma und Kumar, 2017 durch Experimente in der Simulationsumgebung Network Simulator evaluiert und hat eine bessere Leistung gezeigt als der Scheduler der von SCTP verwendet wird. Allerdings gibt es noch keine Kernelimplementierung dieses Ansatzes und somit auch keine Real-World Experimente.

4 EXPERIMENTE & EXPERIMENTIERPLATFORMEN

Die in den Arbeiten verwendeten Experimentierplattformen und durchgeführten Experimente unterscheiden sich in Auswahl und Umfang teilweise sehr stark.

Es gibt durchaus einige Ansätze (Verma und Kumar, 2017, Kuhn u. a., 2014 und Sarwar u. a., 2013), die für die Evaluation nur Experimente in einer Simulationsumgebung durchgeführt haben und nicht wie andere Arbeiten zu-

sätzlich noch in einer kontrollierten physikalischen Experimentierumgebung oder gar einer Real-World Umgebung wie NorNet.

Des Weiteren ist zu bemerken, dass sich die Anzahl und Auswahl der durchgeführten Experimente mitunter deutlich unterscheidet, was den Vergleich der einzelnen Ansätze deutlich erschwert. Es gibt lediglich einen Ansatz, der die Experimente in einer Simulationsumgebung, der Real-World Umgebung NorNet, nach dem *Experimental Design* Ansatz und dem daraufaufbauenden Empfehlungen von Paasch, Khalili und Bonaventure, 2013 durchführt.

Der *Experimental Design* Ansatz beschreit einen Prozess, in dem in einer kontrollierbaren Umgebung Experimente durchgeführt werden um Informationen über das System zu erlangen. Basierend auf dieses Informationen können Entscheidungen getroffen werden um den untersuchten Algorithmus zu verbessern und die angestrebten Ziele zu erreichen. Im Fall von CMT können die Ziele, die erreicht beziehungsweise eingehalten werden sollen, die Design Regeln sein. Die Design Regeln wurden im Abschnitt 2 vorgestellt.

5 RELEVANTE KONFERENZEN

Die folgende Auflistung gibt eine Übersicht einiger relevanter Konferenzen, in denen es regelmäßig Beiträge in den Bereichen *Concurrent Multipath Transfer* und Transportprotokolle gibt:

IFIP NETWORKING 2017 Die IFIP Networking Conference fand dieses Jahr vom 12.-15. Juni 2017 in Stockholm, Schweden statt.

IETF 99 Die diesjährige Konferenz der Internet Engineering Task Force (IETF) fand vom 16.-21. Juli in Prag, Tschechien statt. Die Konferenz ist von großer Bedeutung, da sich auf dieser Konferenz die Multipath TCP (MPTCP) Arbeitsgruppe getroffen hat. Bei diesem Treffen wurde unter anderem über einen neuen Internetdraft (Ford u. a., 2017) gesprochen, der den RFC6824 (Ford u. a., 2013) ablösen könnte. Der RFC6824 (Ford u. a., 2013) beschäftigt sich mit der Multipath-Erweiterung für TCP.

ACM SIGCOMM 2017 Die Special Interest Group on Data Communications der ACM fand dieses Jahr vom 21.-25. August 2017 in Los Angeles, USA

statt. Im Beitrag von Detal, 2017 würde gezeigt, wie Multipath TCP für Hybrid Access Networks genutzt werden kann.

IEEE GLOBECOM 2017 Die Global Communications Conference der IEEE findet dieses Jahr vom 4.-8. Dezember in Singapur statt. Das Programm der GLOBECOM 2017 ist zum jetzigen Zeitpunkt noch nicht im Detail einsehbar.

6 AUSBLICK

Die Entwicklung effizienter und robuster Schedulingalgorithmen für Multipath-Transportprotokolle ist ein wichtiger Bestandteil, um die bestehenden Ressourcen – besonders in heterogenen Umgebungen – optimal nutzen zu können.

In den vergangenen Jahren wurden einige vielversprechende Ansätze vorgestellt, die auf unterschiedlichen Metriken basieren um Schedulingentscheidungen zu treffen. Es ist deutlich geworden, dass es sich bei der Entwicklung von robusten Schedulingalgorithmen um eine nicht triviale Aufgabe handelt.

Die Experimente mit denen die Ansätze evaluiert wurden haben sich jedoch teilweise sehr unterschieden. Somit könnte im kommenden Grundprojekt eine Technologieanalyse durchgeführt werden, um eine Simulations- und eine Real-World-Experimentierplatform auszuwählen. Des Weiteren wäre es nützlich auf Basis des *Experimental Design* Ansatzes eine Sammlung von Experimenten zusammenzustellen, mit denen Schedulingalgorithmen umfassend evaluiert und verglichen werden können.

LITERATUR

Europäische Kommission (2010). *Europa 2020*. URL: https://ec.europa.eu/digital-single-market/en/broadband-strategy-policy (besucht am 20. 08. 2017).

Postel, Jon (1981). *Transmission Control Protocol*. RFC 793. IETF.

Eurostat (2017). *Internet Access and Use Statistics*. URL: http://ec.europa.eu/eurostat/statistics-explained/index.phpInternet_access_and_use_statistics_-_households_and_individuals (besucht am 20. 08. 2017).

Detal (2017). *Leveraging Multipath TCP to create Hybrid Access Networks*. URL: http://conferences.sigcomm.org/sigcomm/2017/files/program-industrial-demos/sigcomm17industrialdemos-paper4.pdf (besucht am 20. 08. 2017).

Stewart, R. (2007). *Stream Control Transmission Protocol*. RFC 4960. http://www.rfc-editor.org/rfc/rfc4960.txt. RFC Editor.

Ford, A. u. a. (2013). *TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6824. http://www.rfc-editor.org/rfc/rfc6824.txt. RFC Editor.

Iyengar, Janardhan R, Paul D Amer und Randall Stewart (2006). "Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths". In: *IEEE/ACM Transactions on networking* 14.5, S. 951–964.

Hamilton, Ryan u. a. (2016). *QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2*. Internet-Draft draft-tsvwg-quic-protocol-02. http://www.ietf.org/internet-drafts/draft-tsvwg-quic-protocol-02.txt. IETF Secretariat.

Ferlin, Simone u. a. (2016). "BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks". In: *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*. IEEE, S. 431–439.

Raiciu, Costin u. a. (2012). "How hard can it be? designing and implementing a deployable multipath TCP". In: *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, S. 29–29.

Paasch, Christoph u. a. (2014a). "Experimental evaluation of multipath TCP schedulers". In: *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop*. ACM, S. 27–32.

Barré, Sébastien, Christoph Paasch und Olivier Bonaventure (2011). "Multipath TCP: from theory to practice". In: *NETWORKING 2011*, S. 444–457.

Wischik, Damon, Mark Handley und Marcelo Bagnulo Braun (2008). "The Resource Pooling Principle". In: *ACM SIGCOMM Computer Communication Review* 38.5, S. 47–52.

Raiciu, Costin, Damon Wischik und Mark Handley (2009). "Practical Congestion Control for Multipath Transport Protocols". In: *University College London, London/United Kingdom, Tech. Rep*.

Chan, Min-Cheng, Chien-Chao Tseng und Li-Hsing Yen (2016). "Jitter-aware packet scheduler for concurrent multipath transmission in heterogeneous wireless networks". In: *Wireless Communications and Networking Conference (WCNC), 2016 IEEE*. IEEE, S. 1–7.

Oh, Bong-Hwan und Jaiyong Lee (2015). "Constraint-based proactive scheduling for MPTCP in wireless networks". In: *Computer Networks* 91, S. 548–563.

Kuhn, Nicolas u. a. (2014). "DAPS: Intelligent delay-aware packet scheduling for multipath transport". In: *Communications (ICC), 2014 IEEE International Conference on*. IEEE, S. 1222–1227.

Ni, Dan u. a. (2014). "Fine-grained forward prediction based dynamic packet scheduling mechanism for multipath TCP in lossy networks". In: *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*. IEEE, S. 1–7.

Sarwar, Golam u. a. (2013). "Mitigating receiver's buffer blocking by delay aware packet scheduling in multipath data transfer". In: *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, S. 1119–1124.

Paasch, Christoph, Ramin Khalili und Olivier Bonaventure (2013). "On the benefits of applying experimental design to improve multipath TCP". In: *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, S. 393–398.

Yang, Fan, Qi Wang und Paul D Amer (2014). "Out-of-order transmission for in-order arrival scheduling for multipath tcp". In: *Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on*. IEEE, S. 749–752.

Paasch, Christoph u. a. (2014b). "Improving multipath TCP". In: *UCLouvain/-ICTEAM/EPL*.

Verma, Lal Pratap und Mahesh Kumar (2017). "An adaptive data chunk scheduling for concurrent multipath transfer". In: *Computer Standards & Interfaces* 52, S. 97–104.

Ford, Alan u. a. (2017). *TCP Extensions for Multipath Operation with Multiple Addresses*. Internet-Draft draft-ietf-mptcp-rfc6824bis-09. http://www.ietf.org/internet-drafts/draft-ietf-mptcp-rfc6824bis-09.txt. IETF Secretariat.