

Data Management and Routing in General Networks

Harald Racke*
Universitat Paderborn

1 Einleitung

Im Bereich des Hochleistungsrechnens ist in den vergangenen Jahren ein Trend weg von dem Einsatz dedizierter Parallelrechner, hin zur Verwendung moderner Clustersysteme zu verzeichnen. Die Popularitat dieser Systeme basiert auf ihrem im Vergleich zu speziellen Parallelrechnern extrem gunstigen Preis-Leistungsverhaltnis, welches durch die vorwiegende Verwendung von Standardkomponenten erreicht wird.

Wahrend dieser Ansatz von der Hardwareseite her, eine sehr hohe Rechenleistung zu einem vernunftigen Preis verspricht, stellt er auf der Softwareseite neue Anforderungen an die Entwicklung von Basisdiensten wie Latverteilung, Datenverwaltung, Monitoring, etc. Derartige Basisdienste sind fur die effiziente Nutzung von verteilten Systemen unentbehrlich, damit sich der Anwender, d.h. der Programmierer einer verteilten Anwendung, nicht um die Kleinigkeiten und Schwierigkeiten einzelner paralleler Systeme kummern muss.

Die gegenwartig sehr aktuellen Forschungsgebiete des Grid-Computing und der Peer-to-Peer Netzwerke beschaftigen sich mit der Entwicklung von Basisdiensten fur Clustersysteme, die entweder durch ein proprietares Netzwerk oder durch ein nicht-proprietares Netzwerk wie z.B. dem Internet verbunden sind. In beiden Fallen liegt eine Hauptschwierigkeit in der, im Vergleich zu dedizierten Parallelrechnern, sehr unregelmaigen Struktur des Verbindungsnetzwerks.

Im Rahmen der Dissertation wurde eine theoretische Analyse von Routing- und Datenverwaltungsdiensten in verteilten Systemen durchgefuhrt. Die vorgestellten Algorithmen sind selbst in sehr unregelmaigen Topologien nahezu optimal, weshalb sie insbesondere fur moderne Clustersysteme geeignet scheinen.

Ein *Routingdienst* bildet die Basis fur den Informationsaustausch zwischen den Knoten eines verteilten Systems. Es werden sowohl *Unicastalgorithmen* (wobei Informationen von einer Quelle zu genau einem Ziel geschickt werden), als auch *Multicastalgorithmen* betrachtet (bei denen Daten von einer Quelle zu mehreren Zielen geschickt werden).

Ein *Datenverwaltungsdienst* realisiert einen transparenten Zugriff auf globale Datenobjekte, auf die von den Knoten des Netzwerks lesend und schreibend zugegriffen werden kann. Beispiele fur derartige globale Datenobjekte sind z.B. globale Variablen in einem parallelen Programm, Dateien in einem verteilten Dateisystem oder Webseiten im Internet. Ein Datenverwaltungsdienst ist ublicherweise erheblich komplizierter als ein einfacher Routingdienst, da er z.B. Antworten auf die folgenden Fragen geben muss:

*Harald Racke, Carnegie Mellon University, Pittsburgh, PA 15213, Email: harry@cs.cmu.edu

- Auf welchen Knoten des Netzwerks soll eine Kopie eines globalen Objektes platziert werden?
- Wann sollen Kopien gelöscht oder verschoben werden?
- Wie können Kopien in einem dynamischen Szenario, in dem sich die Verteilung der Kopien zu Laufzeit ändert, effizient lokalisiert werden?
- Mit Hilfe welcher Kopie wird eine konkrete Leseanfrage bearbeitet, und entlang welchen Routingpfades innerhalb des Netzwerks werden die Daten zu dem anfragenden Knoten geschickt?

Insbesondere der letzte Aspekt zeigt, dass Datenverwaltung eine Verallgemeinerung des Routingproblems ist.

In der Arbeit werden Strategien für zwei unterschiedliche Szenarien vorgestellt. In dem ersten Szenario ist es das Ziel Basisdienste zu entwickeln, die die *Performance* einer verteilten Anwendung optimieren. Hierfür ist es erforderlich die Belastung des Kommunikationsnetzwerkes so gering wie möglich zu halten, da das Netzwerk in den meisten Parallelrechnern den Hauptengpass darstellt, der einen erheblichen Einfluss auf die Gesamtleistung haben kann. Letzteres gilt in besonderem Maße für Clustersysteme, die auf Standardhardwarekomponenten basieren.

Um in solchen bandbreitenlimitierten Netzwerken eine gute Performance zu erzielen, müssen die Basisdienste die Kommunikationslast minimieren. Allerdings ist es nicht ausreichend einfach die Gesamtzahl der Nachrichten innerhalb des Systems zu minimieren (die sog. Gesamtkommunikationslast), da dies nicht verhindert, dass Engpässe innerhalb des Netzwerks entstehen. Stattdessen ist es erforderlich die Kommunikationslast so gleichmäßig wie möglich auf alle Netzwerkressourcen zu verteilen. Deshalb wird in der Arbeit das Kostenmaß *Congestion* untersucht, das die maximale Belastung einer Netzwerkverbindung beschreibt.

In der Arbeit wird ein allgemeines Verfahren zur Congestionminimierung in beliebigen ungerichteten Netzwerken vorgestellt. Das Herzstück des Verfahrens ist eine Netzwerkzerlegung, die es gestattet ein beliebiges Netzwerk in ein *gleichwertiges* Baumnetzwerk (den sog. Zerlegungsbaum) zu transformieren, d.h., dass der Zerlegungsbaum nahezu die gleichen Kommunikationseigenschaften wie das ursprüngliche Netzwerk hat (d.h. der Baum hat die gleiche Kapazität zwischen Knotenpaaren, hat die gleichen Engpässe etc.). Dank der einfachen Struktur von Baumnetzwerken lassen sich viele Probleme, die darauf abzielen die Congestion zu minimieren, sehr einfach und elegant auf Bäumen lösen. Unser allgemeines Verfahren erlaubt es aufgrund der Ähnlichkeit von Baumnetzwerk und ursprünglichem Netzwerk, eine Baumlösung für ein spezielles Problem in eine Lösung für beliebige ungerichtete Graphen zu transformieren. Diese Technik, die in Abschnitt 3 näher erläutert wird, bildet das Hauptergebnis der Arbeit und hat schon zahlreiche Anwendungen für die theoretische Analyse von Kommunikationsproblemen gefunden [CKS04, AAA⁺04, MMP⁺05, AGMM04].

In einem zweiten Szenario werden Netzwerke betrachtet, in denen die Ressourcennutzung nicht umsonst ist, sondern eine Gebühr in Abhängigkeit der genutzten Bandbreite und Speicherkapazität erhoben wird. Dieses Modell ist hauptsächlich durch die Entwicklung

der Grid-Technologie motiviert, die darauf abzielt standardisierte, einfach zu nutzende Methoden für die gemeinsame Nutzung von Ressourcen durch Institutionen und Organisationen anzubieten. Gegenwärtig kommen die beteiligten Institutionen grösstenteils aus dem wissenschaftlichen Sektor, und der Anreiz sich an dem System zu beteiligen (und damit Ressourcen zu Verfügung zu stellen) resultiert aus dem wissenschaftlichen Interesse. Mit einem zunehmenden Interesse an dem Grid-Computing Ansatz werden Verfahren in denen das Angebot und die Nutzung von Ressourcen auf finanziellen Transaktionen basiert an Bedeutung gewinnen.

In solch einem Szenario misst sich die Qualität eines Basisdienstes nicht allein an der erzielten Performance sondern auch an den damit verbundenen Kosten. In der Dissertation werden statische Datenverwaltungsstrategien untersucht (d.h. Strategien, die für ein à priori bekanntes Zugriffsmuster eine feste Datenplatzierung berechnen), die versuchen die Gesamtkosten, die mit der gewählten Platzierung verbunden sind, zu minimieren. Es wird gezeigt, dass es für Bäume in Polynomialzeit möglich ist eine optimale Platzierung zu bestimmen. Für beliebige Netzwerke ist dieses Problem NP-hart. Deshalb werden für diesen Fall Approximationsalgorithmen untersucht. Es wird ein Algorithmus vorgestellt dessen Lösung nur um einen konstanten Faktor schlechter als die optimal mögliche Lösung ist. Aus Platzgründen wird im Rahmen dieser Zusammenfassung nicht näher auf diesen zweiten Teil der Arbeit eingegangen.

2 Problemdefinition

In diesem Abschnitt wird das allgemeine Verfahren für die Congestionminimierung in Netzwerken am Beispiel des Routingproblems detailliert dargelegt. Zunächst führen wir hierfür die folgenden Begriffe und Definitionen ein. Gegeben ist ein Graph $G = (V, E)$ mit Knotenmenge V und Kantenmenge E , der ein Computernetzwerk repräsentiert. Die *Kapazität* oder *Bandbreite* von Netzwerkverbindungen wird durch eine Gewichtsfunktion $b : E \rightarrow \mathbb{R}_0^+$ auf den Kanten modelliert. Das Ziel ist es in diesen Netzwerken die *Congestion* zu minimieren, die wie folgt definiert ist. Die *absolute Last* einer Netzwerkverbindung $e \in E$ ist das Datenvolumen, dass für ein gegebenes Routingverfahren über e verschickt wird. Die *relative Last* oder *Last* der Kante ist die absolute Last geteilt durch die Kantenkapazität $b(e)$. Die *Congestion* ist definiert als die maximale Last einer Kante innerhalb des Netzwerks.

Wir beschränken unsere Überlegungen in dieser Zusammenfassung auf das folgende Routingproblem. In dem *Virtual Circuit Routing* Problem ist eine Sequenz von Routingaufträgen $(s_1, t_1), (s_2, t_2), \dots$ gegeben wobei jeder Auftrag aus einem Quellknoten s_i und einem Zielknoten t_i besteht. Die Aufgabe besteht darin für jeden Routingauftrag einen Pfad in G festzulegen, der die Quelle mit dem Ziel verbindet. Jeder Routingauftrag induziert Last 1 auf jeder benutzten Kante. Das Ziel ist die Minimierung der Congestion.

Die Sequenz von Routingaufträgen muss hierbei *online* abgearbeitet werden, d.h. es wird angenommen, dass die Routingaufträge nacheinander gestellt werden, und ein Routingauftrag $\sigma_i = (s_i, t_i)$ bearbeitet werden muss bevor der nächste Auftrag σ_{i+1} vorliegt. Diese Einschränkung modelliert, dass ein realer Routingalgorithmus nicht alle Routingaufträge à priori kennt sondern kontinuierlich neue Routingaufträge bekommt und bearbeiten muss.

Im Rahmen der Competitive-Analyse wird die Qualität eines Online-Algorithmus durch den Vergleich mit einem optimalen Algorithmus gemessen. Ein Online-Algorithmus ist *c-competitive* oder hat *Competitive-Ratio* c , falls für jede Sequenz σ von Routingaufträgen

$$C_{\text{onl}}(\sigma) \leq c \cdot C_{\text{opt}}(\sigma)$$

gilt, wobei $C_{\text{onl}}(\sigma)$ die Congestion des Online-Algorithmus für Sequenz σ bezeichnet, und $C_{\text{opt}}(\sigma)$ die Congestion eines optimalen Algorithmus ist, der die gesamte Sequenz σ à priori kennt.

Dieses ist die Standarddefinition für Onlinealgorithmen wie sie von Sleator und Tarjan [ST85] eingeführt wurde. Diese allgemeine Definition ist für unsere Problemstellung allerdings aus den folgenden Gründen ungeeignet.

Ein Onlinealgorithmus ist im Vergleich zu einem optimalen Algorithmus eingeschränkt, da der Routingpfad, der z.B. für einen Auftrag σ_i gewählt wird, nicht von nachfolgenden Aufträgen σ_j , $j > i$, abhängen darf, da diese zu Bearbeitungszeit von σ_i noch nicht bekannt sind. Allerdings darf der Routingpfad für σ_i von gewählten Routingpfaden für Aufträge σ_j , $j < i$ abhängen. Diese Tatsache gibt einem Onlinealgorithmus immer noch eine Menge Freiheiten, so dass in beliebigen, gerichteten (!) Netzwerken Onlinealgorithmen mit einem Competitive-Ratio $O(\log n)$ existieren [AAF⁺97].

In einem verteilten System ist es allerdings sehr unrealistisch, dass ein Routingalgorithmus die Entscheidung über einen zu wählenden Routingpfad von allen bisher ausgeführten Routingaufträgen abhängig machen kann. Dies würde nämlich insbesondere voraussetzen, dass ein einzelner Knoten, der z.B. eine Nachricht verschicken will, Information über alle anderen im System vorhandenen Nachrichten besitzt. Dies ist nur mit einem unverhältnismäßig hohen Verwaltungsaufwand zu erreichen, der wiederum die Kommunikationslast des Netzwerks erhöhen würde.

In der Dissertation werden deshalb Routingverfahren betrachtet, die nicht nur online sondern *oblivious* arbeiten. Für einen Oblivious-Routingalgorithmus darf der für einen Routingauftrag gewählte Pfad von *keinem anderen* Auftrag abhängen. Dies bedeutet, dass der zu wählende Routingpfad de facto nur von dem Quell- und Zielknoten, und von Zufallsbits abhängen darf.

Oblivious-Routingalgorithmen haben eine extrem einfache Struktur und können deshalb sehr effizient implementiert werden. Ein Hauptergebnis der Dissertation besteht in dem Beweis, dass für beliebige ungerichtete Netzwerke ein Routingalgorithmus existiert, der oblivious arbeitet und ein Competitive ratio von $O(\log^3 n)$ erzielt.

3 Das Bisimulationsverfahren

Im Folgenden skizzieren wir die allgemeine Technik für die Congestionminimierung in beliebigen Netzwerken. Diese Technik wurde in [MMVW97] eingeführt, wo sie dafür benutzt wurde congestionbasierte Probleme auf Gitternetzwerken zu lösen.

Das Verfahren ermöglicht es ein Baumlösung für ein Problem in eine Lösung für beliebige ungerichtete Netzwerke zu transformieren. Es basiert auf einer Bisimulation zwischen dem physikalischen Netzwerk $G = (V, E)$ und einem virtuellen Baumnetzwerk $T = (V_t, E_t)$, das als Hilfsnetzwerk dient. Das Hilfsnetzwerk wird speziell so konstruiert,

dass es im wesentlichen die gleichen Kommunikationseigenschaften wie G hat.

Das Bisimulationsverfahren arbeitet wie folgt. In einem ersten Schritt wird gezeigt, dass das Baumnetzwerk das Netzwerk G auf die folgende Art simulieren kann. Angenommen wir haben eine Problem Instanz (z.B. eine Folge von Knotenpaaren für das Virtual Circuit Routing Problem) für das Netzwerk G gegeben. Diese Instanz wird mittels einer Abbildung $\pi_1 : V \rightarrow V_t$ in eine Problem Instanz I_t auf T überführt. Nun kann man eine Lösung für I benutzen um eine Lösung für I_t zu erhalten. Für jede Nachricht, die die Originallösung in G zwischen zwei Knoten u und v verschickt, wird eine Nachricht zwischen den zugehörigen Bildknoten $\pi_1(u)$ und $\pi_1(v)$ verschickt. Es wird gezeigt, dass die Congestion der so generierten Lösung für I_t auf T höchstens so groß wie die Congestion der Lösung für I auf G ist.

Dieser erste Schritt zeigt, dass das Baumnetzwerk das "bessere" Kommunikationsnetzwerk ist, da für alles was in G mit Congestion C geroutet werden kann auch in T eine Lösung mit höchstens dieser Congestion existiert.

In einem zweiten Schritt wird gezeigt, dass das Baumnetzwerk nicht viel "besser" ist. Für eine Baumlösung in T existiert eine probabilistische Simulation in G so dass für jede Kante die erwartete Last höchstens einen kleinen Faktor f größer als die Congestion innerhalb des Baumes ist. Formal wird gezeigt, dass eine randomisierte Abbildung $\pi_2 : V_t \rightarrow V$ existiert, die eine Problem Instanz auf T , die mit Congestion C_t gelöst werden kann, in eine Problem Instanz auf G überführt, so dass für deren Lösung $E[L(e)] \leq f \cdot C_t$ für jede Kante e gilt. Hierbei bezeichnet $E[L(e)]$ die erwartete Last der Kante e .

In diesem zweiten Simulationschritt muss ein Knoten in G eventuell mehrere Knoten aus T simulieren, da üblicherweise das Baumnetzwerk deutlich mehr Knoten enthält als das Originalnetzwerk G . Zusätzlich zu der Abbildung $\pi_2 : V_t \rightarrow V$ wird für diese Simulation auch beschrieben, wie die Routingpfade zwischen zwei Bildknoten in G gewählt werden. Dies ist für den ersten Simulationschritt nicht erforderlich, da dort das Routing in einem Baum erfolgt und somit die Pfade zwischen Knoten eindeutig sind.

Die Abbildungen π_1 und π_2 für die beiden Simulationschritte erfüllen $\pi_2(\pi_1(v)) = v$.

Mit Hilfe dieser Simulationen kann man nun wie folgt congestionbasierte Probleme in G lösen. Eine Vorbedingung für den Einsatz des Verfahrens ist es, dass das jeweilige Problem auf Bäumen gut lösbar ist. (Das Virtual Circuit Routing Problem ist auf Bäumen z.B. sehr einfach optimal lösbar, da die Routingpfade eindeutig sind).

Angenommen wir haben einen Baumalgorithmus der eine Problem Instanz I_t auf T mit Congestion höchstens $f_t \cdot C_{\text{opt}}(I_t)$ löst. Man erhält wie folgt eine Strategie für G . Zu Anfang haben wir eine Problem Instanz I auf G gegeben. Diese wird mit Hilfe der ersten Abbildung in eine Instanz I_t auf T übertragen. Wenn wir für diese Instanz den vorausgesetzten Baumalgorithmus verwenden gilt

$$C_{\text{opt}}(I) \geq C_{\text{opt}}(I_t) \geq \frac{1}{f_t} C_{\text{alg}}(I_t) ,$$

wobei $C_{\text{alg}}(I_t)$ die Congestion bezeichnet die der Baumalgorithmus in T generiert.

Nun kann man mit der Abbildung π_2 die Problem Instanz I_t zusammen mit der Lösung wieder zurück auf G übertragen. Die Eigenschaft $\pi_2(\pi_1(v)) = v$ stellt sicher, dass bei dieser Abbildung sich wieder die ursprüngliche Problem Instanz I ergibt. Für die Congestion gilt dann

$$C_{\text{alg}}(I) \leq f \cdot C_{\text{alg}}(I_t) \leq f \cdot f_t \cdot C_{\text{opt}}(I) .$$

Dies bedeutet, dass die Simulation der Baumstrategie auf G ein Competitive-Ratio von $f \cdot f_t$ besitzt.

Die große Herausforderung für die Anwendung dieses Simulationsprinzips ist die Konstruktion des Hilfsnetzwerks T . Glücklicherweise hängt dieser Teil nicht von dem jeweiligen Problem ab. Er muss nur einmal für ein Netzwerk G gelöst werden, und danach kann der Baum für jedes Problem verwendet werden, das darauf abzielt die Congestion zu minimieren.

4 Hierarchische Graphzerlegungen

Das Baumnetzwerk wird so gewählt, dass die Menge der Blattknoten der Knotenmenge V des physikalischen Netzwerks entspricht. Die inneren Knoten des Baumes entsprechen dann Teilmengen von V (die Menge der Blattknoten des entsprechenden Unterbaums), und die Wurzel korrespondiert zu V selbst. Diese Teilmengen werden im folgenden Cluster genannt. Wir bezeichnen die Menge, die zum Baumknoten v gehört mit H_v .

Man kann solch einen Baum als eine rekursive Zerlegung des Netzwerks betrachten. Auf dem ersten Level wird die Knotenmenge V in eine Menge von disjunkten Clustern zerlegt, welche auf den folgenden Leveln rekursiv weiter zerlegt werden. Auf dem untersten Level bestehen alle Cluster aus einzelnen Knoten von G .

Die Kantenkapazitäten in dem Baumnetzwerk T werden wie folgt festgelegt. Eine Kante von einem Knoten $u_t \in T$ zu einem Vaterknoten $v_t \in T$ erhält die Kapazität aller Kanten, die H_u in G verlassen. Aufgrund dieser Definition ergibt sich sofort der folgende Satz, der den ersten Simulationschritt enthält.

Satz 1 *Jede Lösung auf dem Originalgraphen G kann im Baumnetzwerk simuliert werden ohne die Congestion zu erhöhen.*

Beweis. Die Abbildung π_1 für die Simulation ist die Bijektion zwischen V und den Blattknoten von T . Wenn man nun Nachrichten in T zwischen zwei Blattknoten verschickt, gilt die folgende Aussage. Eine Nachricht benutzt eine Kante von u_t nach v_t (wobei v_t der Vater von u_t ist) nur, falls die korrespondierende Nachricht in G die Menge H_u verläßt. Aufgrund der gewählten Kantenkapazitäten für T folgt der Satz. ■

Die Hauptschwierigkeit besteht in der Realisierung der zweiten Simulation von T auf G . Das allgemeine Prinzip für diese Simulation ist das Folgende. Angenommen eine Nachricht wird im Baum zwischen zwei Blattknoten u_t und v_t verschickt. Seien $u_t = s_1, s_2, \dots, s_\ell = v_t$ die Knoten, die diese Nachricht im Baum passieren muss. In G wird für jeden Knoten s_i ein zufälliger Knoten r_i aus H_{s_i} gewählt und dann wird die Nachricht in G nacheinander von r_i nach r_{i+1} weitergeleitet. Für die Wegewahl zwischen r_i und r_{i+1} wird ein sorgfältig gewähltes Wegesystem benutzt. Die Qualität eines solchen Ansatzes hängt von den folgenden Faktoren ab.

- das Baumnetzwerk T , insbesondere dessen Höhe
- die Wahrscheinlichkeitsverteilung für die Wahl der zufälligen Knoten eines Clusters H_{s_i}

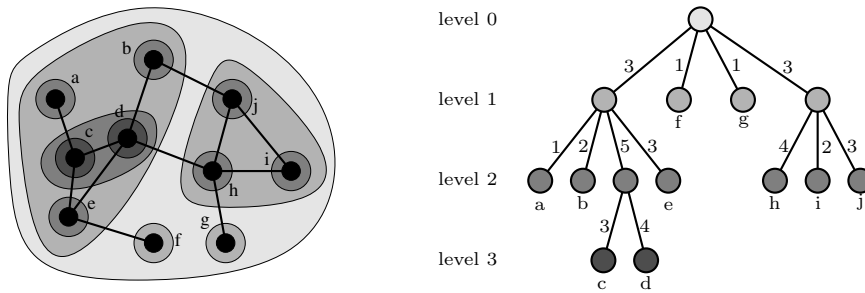


Abbildung 1: Ein Graph und ein möglicher Zerlegungsbaum.

- dem verwendeten Wegesystem zwischen zufällig gewählten Knoten aus H_{s_i} und $H_{s_{i+1}}$

Der große Vorteil des Ansatzes besteht darin, dass durch die Aufspaltung einer Nachricht in verschiedene Teilnachrichten, alle Nachrichten in G die gleiche Form haben. Sie gehen von einem zufällig gewählten Knoten eines Clusters H_{s_i} zu einem zufällig gewählten Knoten eines anderen Clusters $H_{s_{i+1}}$. Dies erleichtert die theoretische Analyse enorm.

In der Dissertation wird gezeigt, dass man alle oben genannten Faktoren aufeinander abstimmen kann, so dass man einen Faktor $f = O(\log^3 n)$ zwischen der Congestion in T und der erwarteten Last in G erreichen kann. Allerdings wird im wesentlichen nur die Existenz einer Zerlegung mit dieser Schranke bewiesen. Der Algorithmus für die Berechnung hat exponentielle Laufzeit. Polynomialzeit kann nur für eine Version gezeigt werden, die einen Faktor $f = O(\log^4 n)$ garantiert.

5 Optimale Routingalgorithmen

Für das Virtual Circuit Routing Problem ist die Frage ob die Zerlegung effizient berechenbar ist, nicht relevant. Für dieses Problem reicht der Nachweis der Existenz einer guten Zerlegung (und damit eines guten Routingalgorithmus) aus, da in einem weiteren Abschnitt der Dissertation gezeigt wurde, dass der optimale Oblivious-Routingalgorithmus, (d.h. der, der das beste Competitive-Ratio garantiert) in Polynomialzeit berechnet werden kann.

Für diesen Nachweis werden Techniken aus der linearen Optimierung verwendet. Es wird gezeigt, dass man das Problem des Findens eines optimalen Routingalgorithmus als ein lineares Programm mit (überabzählbar vielen) Bedingungen formulieren kann. Dann kann man zeigen, dass nur exponentiell viele dieser Bedingungen von Bedeutung sind, d.h. man kann alle anderen wegfällen lassen, ohne den Wert der LP-Lösung zu verändern. Das verbleibende LP kann man durch ein Separationsorakel mit Hilfe der Ellipsoidmethode in Polynomialzeit lösen.

Dieses Verfahren berechnet für ein gegebenes Netzwerk den optimalen Routingalgorithmus und das Competitive-Ratio, dass dieser Routingalgorithmus erzielt. Es kann allerdings nicht dazu verwendet werden allgemeine Schranken für das Competitive Ratio von

Oblivious-Algorithmen in bestimmten Netzwerkklassen zu bestimmen.

6 Weiterführende Arbeiten

Die in der Dissertation vorgestellten Ergebnisse sind von einer Vielzahl von Autoren weitergeführt bzw. angewendet worden.

Harrelson, Hildrum und Rao [HHR03] haben den Algorithmus für die hierarchische Graphzerlegung (d.h. für die Bestimmung des Baumnetzwerks) verbessert, so dass sich ein Competitive-Ratio von $O(\log^2 n \log \log n)$ für das Virtual Circuit Routing Problem und die anderen in der Dissertation betrachteten Probleme ergibt. Dieser Zerlegungsalgorithmus hat sogar polynomielle Laufzeit.

Applegate und Cohen [AC03] führen empirische Untersuchungen bzgl. des erreichbaren Competitive Ratios von Oblivious-Algorithmen auf internetähnlichen Netzwerken durch, in dem sie für diese Netzwerke den optimalen Routingalgorithmus bestimmen. Es wurde gezeigt, dass der Competitive-Ratio für die meisten der betrachteten Netzwerke zwischen 2 und 4 liegt, was einen überraschend niedrigen Wert darstellt.

Außerdem wurden die gezeigten Techniken für die theoretische Analyse verschiedenster Schnitt- und Kommunikationsprobleme in Graphen verwendet [CKS04, AAA⁺04, MMP⁺05, AGMM04].

Literatur

- [AAA⁺04] Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., und Naor, J. S.: A general approach to online network optimization problems. In: *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*. S. 577–586. 2004.
- [AAF⁺97] Aspnes, J., Azar, Y., Fiat, A., Plotkin, S. A., und Waarts, O.: On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*. 44(3):486–504. 1997. Also in *Proc. 25th STOC*, 1993, pp. 623–631.
- [AC03] Applegate, D. und Cohen, E.: Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In: *Proceedings of the ACM Symposium on Communications Architectures & Protocols (SIGCOMM)*. S. 313–324. 2003.
- [AGMM04] Andreev, K., Garrod, C., Maggs, B. M., und Meyerson, A.: Simultaneous source location. In: *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*. S. 13–26. 2004.
- [CKS04] Chekuri, C., Khanna, S., und Shepherd, B.: The all-or-nothing multicommodity flow problem. In: *Proceedings of the 36th ACM Symposium on Theory of Computing (STOC)*. S. 156–165. 2004.
- [HHR03] Harrelson, C., Hildrum, K., und Rao, S. B.: A polynomial-time tree decomposition to minimize congestion. In: *Proceedings of the 15th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. S. 34–43. 2003.
- [MMP⁺05] Maggs, B. M., Miller, G. L., Parekh, O., Ravi, R., und Woo, S. L. M.: Finding effective support-tree preconditioners. In: *Proceedings of the 17th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. S. 176–185. 2005.

- [MMVW97] Maggs, B. M., Meyer auf der Heide, F., Vöcking, B., und Westermann, M.: Exploiting locality for networks of limited bandwidth. In: *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS)*. S. 284–293. 1997.
- [ST85] Sleator, D. D. und Tarjan, R. E.: Amortized efficiency of list update and paging rules. *Communications of the ACM*. 28(2):202–208. 1985.

Wissenschaftlicher Werdegang

Harald Räcke ist geboren am 16.10.73 in Rheda-Wiedenbrück, wo er 1993 das Abitur erlangt hat. Nach seinem Zivildienst studierte er ab 1994 Informatik an der Universität Paderborn und schloss das Studium 1999 mit Diplom ab. Danach arbeitete er als Mitglied der Arbeitsgruppe “Algorithmen und Komplexität” unter Friedhelm Meyer auf der Heide an der Universität Paderborn. Im Rahmen dieser Forschungstätigkeit entstand die hier zusammengefasste Dissertation mit der er 2003 promovierte. Momentan arbeitet er als Postdoc an der Carnegie Mellon University in Pittsburgh.