

Routing in verzögerungstoleranten Netzwerken

Benjamin Gufler
(benjamin.gufler@in.tum.de)

Seminar „Internetrouting“ ,
Technische Universität München

WS 2004/2005 (Version vom 9. Februar 2005)

Zusammenfassung

Diese Arbeit behandelt Routing in verzögerungstoleranten Netzwerken. Häufige Verbindungsausfälle mit daraus resultierender Netzwerkpartitionierung sowie begrenzte Puffer in den einzelnen Knoten machen den Einsatz von gängigen Algorithmen unmöglich, so dass neue Ansätze gefunden werden müssen. Einige solche werden hier angeführt, jeweils zusammen mit einigen Effizienzüberlegungen und Testergebnissen.

1 Einleitung

Mit der zunehmenden Verbreitung des Internets in den vergangenen Jahren ist auch die Wichtigkeit der Zugangsmöglichkeit stetig angestiegen. Unabhängig vom aktuellen Aufenthaltsort können geschäftliche und private Kontakte gepflegt, Recherchen getätigt, Daten gesammelt und analysiert, Angebote des eigenen Unternehmens publiziert und von anderen Unternehmen gesucht werden. Den Möglichkeiten sind nahezu keine Grenzen gesetzt. Doch funktioniert dies alles wirklich absolut standortunabhängig?

Gerade abgelegenen und/oder finanziell schwachen Gebieten, für die das Internet oft als *die* Möglichkeit gepriesen wird, die lokale Wirtschaft zu stärken, die Abwanderung zu verhindern oder stoppen, bleibt der Zugang häufig verwehrt: Die Telefonkosten sind zu hoch und deren Bandbreite zu niedrig, das Legen von schnelleren kabelbasierten Anbindungen zu aufwändig und teuer, geostationäre Satelliten nicht im Einzugsgebiet.

Diese Arbeit soll eine Einführung in die gegenwärtigen Bemühungen sein, solchen Gebieten einen kapazitätsstarken und kostengünstigen Zugang zum Internet zu bieten, wobei höhere Latenzzeiten in Kauf genommen werden. Das Thema soll dabei hinreichend allgemein betrachtet werden, um eine Übertragung der Ergebnisse auf ähnliche Situationen wie Netze von verteilten Sensoren oder — in fernerer Zukunft — interplanetare Verbindungen zu erlauben.

Die dabei entstehende Infrastruktur wird als VERZÖGERUNGSTOLERANTES NETZWERK (DELAY TOLERANT NETWORK, DTN, siehe [F 03]) bezeichnet. In diesen Netzen wird von häufigen und langen Verbindungsunterbrechungen ausgegangen, die auch zur Partitionierung führen können. Direkte Verbindungen zwischen (logisch) weit entfernten Knoten können durchaus auch nie zustande kommen.

2 Verbindungsmöglichkeiten

Welche Verbindungsmöglichkeiten existieren in derartigen Netzwerken? Für Sensornetzwerke wird in [SRJB 03] eine dreischichtige Architektur eingeführt, die größtenteils auf allgemeine DTNs übertragen werden kann. Zwischen die Schicht der Datenquellen und die der Datensinken wird die Schicht der sogenannten MULEs (mobile ubiquitous LAN extensions) geschoben. Bei der Wahl der MULEs ist dabei eine Vielzahl von Möglichkeiten gegeben:

- Satelliten mit niedrigem Orbit, die das mit dem Internet zu verbindende Gebiet gelegentlich überfliegen
- mit mobilen Speichergeräten (USB-Sticks, Festplatten, Notebooks) ausgestattete Kurier (siehe etwa [WIZZY])
- Busse, ausgestattet mit Speicher und kabellosen Übertragungsmöglichkeiten (siehe [HFP 03])
- frei umherziehende Tierherden, die mit Geräten zur automatisierten Datensammlung und -weitergabe versehen werden
- und viele weitere

Eine letzte, nicht ganz in das Konzept der MULEs passende Verbindungsmöglichkeit ist zudem wochentag- oder uhrzeitabhängige Einwahl über das Telefon (die resultierenden Kosten könnten etwa in Nachtstunden erheblich niedriger sein als tagsüber).

Bedingt durch die vielen, möglicherweise parallel vorhandenen MULEs, ist es angebracht, Algorithmen zur Wahl der „besten“ Route für Pakete in DTNs zu entwickeln (was dabei auch immer unter der „besten“ Route zu verstehen ist: geringste Verzögerung, wahrscheinlichste Ankunft am Ziel, kostengünstigste Variante, ...).

Herkömmliche Routing-Algorithmen kommen dabei — zumindest ohne Anpassung — nicht in Frage: Diese versuchen in der Regel, den besten Start- Ziel-Pfad aus den aktuell verfügbaren auszuwählen. In einem DTN ist es aber häufig der Fall, dass ein derartiger Pfad nie existiert. Diesem Umstand müssen DTN-Routing-Algorithmen Sorge tragen.

Im Folgenden werden einige solche Algorithmen vorgestellt und kurz analysiert. Sie entstammen (bis auf eine Ausnahme) [JFP 04]. Dabei wird größtenteils davon ausgegangen, dass KONTAKTE, das heißt Verbindungsmöglichkeiten zwischen zwei oder mehreren Knoten, periodisch und damit zu vorhersehbaren Zeitpunkten zustande kommen. Auf einige der oben aufgelisteten MULEs trifft diese Annahme offensichtlich zu. Das Bewegungsmuster der übrigen kann mit Hilfe der Wahrscheinlichkeitstheorie angenähert werden. Abweichungen von diesem berechneten Muster werden sicherlich zu einer „Störung“ der Algorithmen führen. Eine genauere Betrachtung dieses Themas würde aber den Rahmen dieser Arbeit sprengen.

Die Algorithmen aus [JFP 04] wurden in zwei Szenarien getestet. Die einfachere Testumgebung liegt in Südafrika: Es galt, das Routing zwischen dem Dorf Kwazulu-Natal und Kapstadt abzuwickeln. Dabei standen drei verschiedene direkte Routen zur Verfügung: Eine Telefonverbindung, die aus Kostengründen nur nachts genutzt werden kann (genauer: von 23:00 bis 06:00 Uhr mit einer Geschwindigkeit von 4 kbps und einer Verzögerung von 20 ms), PACSAT-Satelliten (die während eines Kontakts sowohl mit Kapstadt als auch mit Kwazulu-Natal Verbindung haben, bei etwa 10 bis 12 Kontakten zu je rund 10 Minuten am Tag, Übertragungsgeschwindigkeiten von 10 kbps und 25 ms Verzögerung) und drei Motorradfahrer, die mit USB-Sticks zu 128 MB ausgestattet sind und für eine Fahrt etwa zwei Stunden benötigen (Übertragungsgeschwindigkeit 1 Mbps, Kontaktdauer 5 min). Das Szenario wird in Abbildung 1 auf der nächsten Seite schematisch dargestellt.

Als komplexere Testumgebung wurde ein Ausschnitt aus dem Linienbusnetz von San

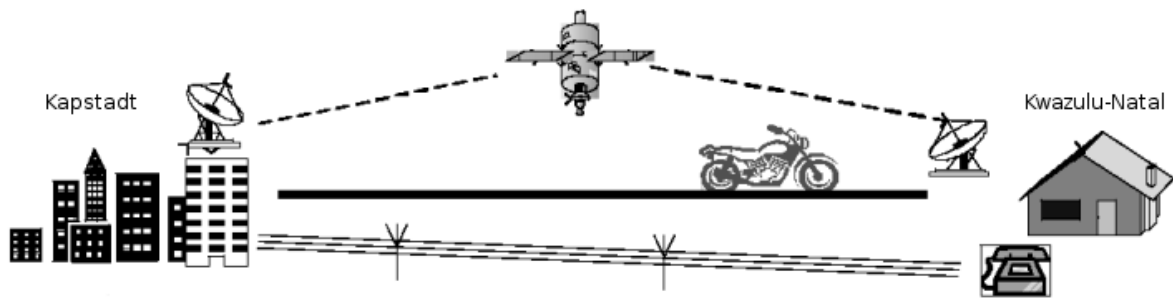


Abbildung 1: Routen zwischen Kapstadt und Kwazulu-Natal

Francisco (mit 20 Bussen) gewählt. Dieses Netz wurde aber nur am Simulator umgesetzt.

Es sei noch ein weiteres Netzwerk, „DakNet“ erwähnt, das in [HFP 03] vorgestellt wird und in dem ebenfalls Busse zum Transport von Datenpaketen eingesetzt wurden. Dak-Net wurde in Indien realisiert; es sind aber leider nicht genügend Messwerte vorhanden, um dieses Netz mit den beiden obigen Beispielszenarien zu vergleichen.

3 Triviale Ansätze

In diesem Abschnitt werden zwei Algorithmen betrachtet, die versuchen, ohne jegliche Kenntnis des zugrundeliegenden Netzes Pakete zu ihrem Ziel zu befördern.

3.1 First Contact

Dieser Algorithmus befördert wartende Pakete immer an den ersten sich ergebenden Kontakt weiter. Bei mehreren gleichzeitig entstehenden Kontakten wird einer davon gleichverteilt zufällig gewählt.

Dass dieser Algorithmus äußerst einfach umsetzbar ist, sollte unmittelbar klar sein, ebenso aber auch, dass die erzielte Performance im Mittel nicht akzeptabel sein wird. Übermäßig hohe Verzögerungen sind zu erwarten; zudem können in nichttrivialen Netzwerken Schleifen entstehen.

Dies wird auch durch die Ergebnisse aus dem Beispiel mit dem Busnetz untermauert. Die Resultate für das Südafrika-Beispiel sind zwar erheblich besser (zumindest in Bezug auf die mittlere, nicht auf die maximale Latenzzeit), dies erklärt sich aber umgehend aus der einfachen Topologie, die eine „falsche“ Routenwahl unmöglich macht.

Möglichkeiten zur Verbesserung dieses Algorithmus' sind recht schnell erkennbar, etwa die Einführung eines Pfadvektors zur Vermeidung von Schleifen. Der nächste hier vorgestellte Algorithmus verfolgt das Ziel, die maximale Latenzzeit von FIRST CONTACT zu reduzieren.

3.2 Epidemic Routing

Die hohe maximale Latenzzeit von FIRST CONTACT beruht im Wesentlichen darauf, dass aufgrund der trivialen Wegwahl Umwege und Schleifen entstehen. Ein (zugegeben naiver) Ansatz zur Abschwächung dieses Problems ist, ein Paket nicht nur an den ersten, sondern an mehrere Kontakte weiterzugeben. Diese Vorgehensweise wird in [VB 00] analysiert.

In nicht allzu komplexen Netzstrukturen und bei hinreichend großen Puffern in den Knoten erzielt dieser Algorithmus durchaus akzeptable Ergebnisse. So konnte in der

beschriebenen Testumgebung etwa eine Auslieferungsrage von 100% erreicht werden, während traditionelle Ansätze aufgrund fehlender End-zu-End-Verbindungen gänzlich versagten.

Durch das Vervielfachen der Pakete entsteht mit diesem Ansatz aber schon bei einer geringen Paketzahl viel Datenverkehr. Übersteigt die Anzahl der Pakete (ohne Berücksichtigung der Duplikate) die Puffergrößen der einzelnen Knoten deutlich, so bricht die Performance ein — was die Tests im genannten Dokument ebenso belegen wie das Gnutella-Netzwerk, in da auch dort Daten repliziert werden.

4 Routing bei Kenntnis zukünftiger Verbindungen

In diesem und dem folgenden Abschnitt werden jeweils zwei Algorithmen erläutert, die partielle Kenntnis des zugrundeliegenden Netzes haben und dieses Wissen in die Bestimmung der Routen mit einfließen lassen. In der Reihenfolge ihrer Präsentation steigt dabei der Bedarf der Algorithmen an Kenntnis über das Netzwerk. Die zwei Algorithmen aus diesem Abschnitt beschränken sich auf Kenntnis der Verbindungen zwischen einzelnen Knoten, die sich zukünftig ergeben werden.

Alle Algorithmen basieren auf einer Modifikation des bekannten DIJKSTRA-ALGORITHMUS zur Berechnung des kürzesten Wegs in einem Graphen. Die in diesem Algorithmus vorkommende Kantengewichtungsfunktion, die üblicherweise von der Menge der Kanten in eine Menge von Gewichtungen abbildet, wird durch eine Funktion ersetzt, die zusätzlich zu einer Kante auch die Versendezeit, die Paketgröße und den Knoten, der die Berechnung durchführt, mit berücksichtigt.

Grundalgorithmus: Dijkstra mit angepasster Kostenfunktion

Eingaben:

$G = (V, E)$ Graph (bestehend aus Knoten und Kanten), in dem der Weg gefunden werden soll

S Knoten, von dem aus das Paket versandt wird

T Zeitpunkt des Absendens

M Größe des Pakets

$w(e, t, m, s)$ Gewichtsfunktion

Ausgabe:

L Kostenvektor; für jeden Knoten k im Netzwerk enthält $L[k]$ die geringstmöglichen Kosten, die das Versenden des angegebenen Pakets zu diesem Knoten verursachen würde.

```
1:  $R = V$ 
2: for  $v \in R$  do
3:    $L[v] = \infty$ 
4: done
5:  $L[S] = 0$ 
6: while  $R \neq \emptyset$  do
7:   sei  $u \in \{x \in R : L[x] == \min_{y \in R} L[y]\}$  beliebig
8:   for  $e \in \{o \in E : o == (u, v)\}$  do
9:     if  $L[v] > L[u] + w(e, T + L[u], M, S)$  then
10:       $L[v] = L[u] + w(e, T + L[u], M, S)$ 
11:     fi
12:   done
13: done
```

Als Gewicht der Kanten wird hierbei die nötige Zeit verwendet, die zum Versenden des Pakets über die Kante nötig ist. Diese Zeit setzt sich aus zwei Komponenten zusammen:

- der nötigen Zeit, um das gesamte Paket vom Startknoten der Kante in die Kante zu „laden“
- der Verzögerung, die die Kante verursacht.

Für die Gewichtsfunktion ergibt sich somit

$$w(e, t, m, s) = \text{send}(e, t, m, s) + \text{delay}(e, t) - t$$

Die von der Kante verursachte Verzögerung $\text{delay}(e, t)$ ist als Eigenschaft dieser Kante bekannt. Die „Ladezeit“ $\text{send}(e, t, m, s)$ kann berechnet werden. Sie hängt neben der verfügbaren Kapazität der Kante natürlich auch von der Länge der Warteschlange ab, die vor dem Versand „unseres“ Pakets noch abgearbeitet werden muss:

$$\text{send}(e, t, m, s) = \min \left\{ t'' : \int_{x=t}^{t''} c(e, x) dx \geq (m + Q(e, t, s)) \right\}$$

Dabei sind

- $c(e, t)$ die verfügbare Kapazität der Kante e zum Zeitpunkt t und
- $Q(e, t, s)$ die vom Knoten s geschätzte Länge der Warteschlange am Anfangsknoten der Kante e zum Zeitpunkt t .

Die vorgestellten Algorithmen mit zeitabhängigen Kosten unterscheiden sich voneinander nur durch unterschiedliche Abschätzungen der Warteschlangenlängen (also durch die Funktion Q).

4.1 Minimum Expected Delay

Dieser Algorithmus arbeitet noch mit zeitunabhängigen Kosten. Seine benötigte Kenntnis des Netzes beschränkt sich auf die durchschnittlichen Wartezeiten bis zum Kontakt eines gegebenen Knotens mit einer bestimmten Kante. Der Weg für ein Paket wird vom Knoten, an dem das Paket ins Netz kommt, berechnet, und kann im weiteren Verlauf nicht mehr verändert werden (SOURCE-ROUTING). Dadurch kann weder auf Netzwerküberlastungen noch auf sich „plötzlich“ ergebende Abkürzungen reagiert werden.

Entsprechend schlecht schneidet MINIMUM EXPECTED DELAY in den zwei Testszenarien ab. Aufgrund der nachts durchgehend verfügbaren Telefonverbindung von Kwa-zulu-Natal nach Kapstadt ist die mittlere Wartezeit dieser Kante geringer als die der anderen Kanten. Folglich wird der gesamte Datenverkehr über die Telefonlinie abgewickelt. Die sich ergebenden Wartezeiten sind dadurch um ein Vielfaches höher als bei FIRST CONTACT.

Im Busnetzwerk kann MINIMUM EXPECTED DELAY die von FIRST CONTACT erzielten Ergebnisse zwar geringfügig verbessern, da er wenigstens eine zielgerichtete Routenwahl durchführt, aber annehmbar für eine breitere Verwendung sind auch diese Ergebnisse nicht.

4.2 Earliest Delivery

EARLIEST DELIVERY berücksichtigt die Zeit, zu der ein Paket versendet wird: Der Algorithmus ist über die genauen Zeitpunkte und Dauern aller Kontakte in Kenntnis und kann so die sich am schnellsten ergebende Route zum Ziel bestimmen. Warteschlangenlängen werden aber (noch) nicht berücksichtigt:

$$Q(e, t, s) \equiv 0$$

Dass EARLIEST DELIVERY damit bessere Ergebnisse erzielen sollte als MINIMUM EXPECTED DELAY, scheint offensichtlich. Es gilt sogar, dass dieser Algorithmus unter günstigen Bedingungen optimale Routen wählt, nämlich genau dann, wenn

- alle Warteschlangen entlang der gewählten Route leer sind (dann stimmt die Abschätzung Q), oder wenn
- Warteschlangen entlang der Route zwar nicht leer sind, aber die Kapazitäten der Kontakte hinreichend groß sind, um das betrachtete Paket noch im berechneten Kontakt zu übertragen (dann kann die Warteschlangenlänge vernachlässigt werden).

Wird ein eingeplanter Kontakt hingegen wegen Netzwerküberlastung verpasst, so hat dies wegen der fix vorberechneten Route für das Paket unter Umständen enorme Auswirkungen auf die Übertragungsdauer.

Diese Überlegungen werden durch die beiden Testsituationen untermauert. Unter geringer Netzlast kann EARLIEST DELIVERY durchaus mit den beiden im Weiteren vorgestellten Algorithmen mithalten. Bei steigender Last sinkt die Performance aber auf und sogar unter die von MINIMUM EXPECTED DELAY — im Beispiel zu Kwazulu-Natal nur die maximale Verzögerung, beim Busnetzwerk sogar die durchschnittliche. Die starr vorberechneten und nicht änderbaren Routen werden dem Algorithmus dann zum Verhängnis. Diese Erkenntnis soll auch in die nächste vorgestellte Variante von EARLIEST DELIVERY einfließen.

5 Routing unter Kenntnis der Netzauslastung

Um die Performance der Algorithmen auch unter hoher Last zu verbessern, ist eine Beobachtung der Auslastung notwendig. Dies geschieht nun — wie bereits angekündigt — durch Betrachtung der Längen der Warteschlangen für die einzelnen Kanten im Netzwerk, also durch geeignete Abschätzungsfunktionen Q .

5.1 Earliest Delivery with Local Queuing

Jeder Knoten in einem DTN kennt die Längen der Warteschlangen, die sich an seinen ausgehenden Kanten gebildet haben. EARLIEST DELIVERY WITH LOCAL QUEUING erweitert EARLIEST DELIVERY so, dass diese Information mit in die Routenwahl einfließt. Im Unterschied zu den bisher betrachteten Algorithmen ist die gewählte Route damit natürlich abhängig vom Knoten, der sie berechnet.

$$Q(e, t, s) = \begin{cases} \text{Länge der Warteschlange für } e \text{ zum Zeitpunkt } t & \text{falls } s \text{ Startknoten von } e \\ 0 & \text{sonst} \end{cases}$$

Angewandt nach dem bisherigen Schema des Source-Routing vermeidet dieser Algorithmus die Wahl von stark belasteten lokalen Kanten — natürlich nur, sofern bei Routing über alternative Kanten ein besseres Ergebnis erzielt werden kann.

Nach den bei EARLIEST DELIVERY beobachteten negativen Eigenschaften von fix vorberechneten Routen schlägt [JFP 04] für diese Variante vor, in jedem Knoten nur die nächste zu verfolgende Kante zu bestimmen. Mit diesem als PER-HOP-ROUTING bekannten Verfahren scheint sich zudem die lokal optimale Kantenwahl auf das gesamte Netzwerk auszudehnen. Durch den Einsatz von Per-Hop-Routing ergeben sich aber auch einige bekannte Probleme, etwa das mögliche Entstehen von Schleifen. Um die Schleifenbildung zu vermeiden, kann natürlich ein Pfadvektor in die Pakete aufgenommen werden — was größere Pakete und demzufolge höhere Netzlast ohne effektiven Nutzen für den Anwender zur Folge hat.

Die Ergebnisse aus den zwei Testszenarien sprechen — im Vergleich mit den bisher vorgestellten Alternativen — für EARLIEST DELIVERY WITH LOCAL QUEUEING. Die mittlere wie auch die maximale Verzögerung bleiben bei erhöhter Last weit unter den entsprechenden Werten der anderen Algorithmen.

Ein in [JFP 04] nicht betrachteter Aspekt, der sich aus dem Einsatz von Per-Hop-Routing ergibt, ist das sogenannte ROUTE-FLAPPING. Dieses bei Verwendung von Per-Hop-Routing in herkömmlichen Netzwerken beobachtete Phänomen bezeichnet den häufigen Wechsel zwischen zwei oder mehreren Routen. Zurückzuführen ist dies auf die Abhängigkeit der Routenwahl von den Warteschlangenfüllständen: Ist die Warteschlange an der optimalen Route gefüllt, so wird zur nächstschlechteren Route gewechselt. Sobald nun einige Pakete über die erstere Route versendet wurden und die Warteschlange deshalb wieder Platz frei hat, wechselt der Routing-Algorithmus wieder zurück, verursacht dadurch schnell wieder eine volle Warteschlange und muss erneut zur schlechteren Route wechseln.

Es ist offensichtlich, dass dieses Phänomen auch bei Per-Hop-Routing in verzögerungstoleranten Netzwerken auftreten kann. Fraglich ist jedoch, ob — oder wann — die Auswirkungen des Route-Flapping hier Nachteile bringen oder ob sie — wenigstens in bestimmten Situationen — nicht vielleicht akzeptabel oder sogar wünschenswert sind. Bei hoher Kontaktdichte entlang der besseren Route kann es durchaus besser sein, Pakete, die nicht in die entsprechende Warteschlange passen, zu verwerfen anstatt sie über alternative Routen zu senden. Allerdings müsste der Sender des Pakets darüber in Kenntnis gesetzt werden, um bereits nach kurzer Wartezeit das Paket erneut zu versenden. Im Allgemeinen ist eine schnelle Wiederholung von Sendevorgängen in verzögerungstoleranten Netzwerken natürlich nicht ratsam. Demzufolge sollte der sendende Knoten über verworfene Pakete explizit informiert werden — was zusätzlichen Verkehr und Aufwand (auch derartige Informationspakete könnten verworfen werden) verursacht. Um eine Klassifizierung der auftretenden Situationen durchführen und effektive Probleme durch Route-Flapping in verzögerungstoleranten Netzwerken klar erkennen zu können sowie Vermeidungsstrategien oder alternative Routenfindungsansätze zu finden, ist in diesem Bereich wohl noch einige Forschungsarbeit notwendig.

5.2 Earliest Delivery with All Queues

Die Schwierigkeiten mit EARLIEST DELIVERY WITH LOCAL QUEUEING treten nicht auf, wenn jeder Knoten die Warteschlangenfüllstände an allen Knoten im Netz kennt. Dann muss, um auf lokale Überlastungen reagieren zu können, nicht zu Per-Hop-Routing übergegangen werden. Der Startknoten weiß im Voraus, wann an welchen Knoten die Warteschlangen voll sein werden, und wählt, falls nötig, gleich eine andere Route. Als Abschätzungsfunktion der Warteschlangenlängen dient somit deren effektive Länge:

$$Q(e, t, s) = \text{Länge der Warteschlange für } e \text{ zum Zeitpunkt } t$$

Um auch zukünftige Warteschlangenfüllstände bestimmen zu können, wird in [JFP 04] ein Reservierungssystem vorgeschlagen: Nach der Bestimmung einer Route, die genügend Kapazität für ein Paket bereitstellt, werden die benötigten Kapazitäten an allen Knoten entlang dieser Route reserviert. Ein derartiges System setzt natürlich voraus, dass alle Knoten — zumindest über eine Verbindung mit niedriger Bandbreite — ständig verbunden sind. In allgemeinen verzögerungstoleranten Netzwerken, die derartige Verbindungen nicht bieten, ist EARLIEST DELIVERY WITH ALL QUEUES folglich nicht oder, aufgrund der Verzögerungen bei der Reservierung, nicht vollständig gemäß der obigen Beschreibung umsetzbar.

In den zwei Testszenarien konnte dieser Algorithmus trotzdem mit den übrigen vorgestellten verglichen werden: Im Beispiel zu Kwazulu-Natal treten keine Zwischenknoten auf, deren Warteschlangen berücksichtigt werden müssen; entsprechend sind die

Ergebnisse von EARLIEST DELIVERY WITH ALL QUEUES identisch zu denen von EARLIEST DELIVERY WITH LOCAL QUEUING. Das Busnetzwerk wurde „nur“ am Simulator erprobt, weshalb die Bestimmung der Längen der Warteschlangen auch hier kein Problem darstellte.

Erstaunlicherweise bestand zwischen den Ergebnissen dieser beiden letztgenannten Algorithmen nur ein geringer Unterschied — die Kenntnis der lokalen Warteschlangen zusammen mit dem Einsatz von Per-Hop-Routing scheint also (zumindest unter den gegebenen Testbedingungen) nahezu äquivalent zur Kenntnis aller Warteschlangen zu sein.

6 Zum Vergleich: Routing „mit Glaskugel“

Um die mit den entwickelten Routing-Algorithmen erzielten Ergebnisse besser bewerten zu können, wird in [JFP 04] eine lineare Optimierungsaufgabe formuliert, die unter Kenntnis aller Eigenschaften des Netzwerks (Verbindungszeitpunkte und deren Dauer, Warteschlangen in allen Knoten, gegenwärtigen und zukünftigen Bandbreitenbedarf) die Summe der Latenzzeiten aller Pakete minimiert. Als effektiver Routing-Algorithmus ist diese Formulierung also nicht einsetzbar, da eine genaue Kenntnis über alle zukünftig zu versendenden Pakete wohl nur in den seltensten Fällen gegeben sein dürfte.

Im Beispiel der Routenwahl zwischen Kapstadt und Kwazulu-Natal benötigt die Lösung der Optimierungsaufgabe mit Hilfe von [CPLEX] auf einem Rechner mit 8 Prozessoren zu je 700 MHz und 3 GB Arbeitsspeicher etwa 8 Minuten. Dabei treten rund 500000 Nebenbedingungen mit 550000 Variablen auf. Unter mäßiger Last entsprechen die berechneten Zeiten sowohl für die mittlere als auch für die maximale Latenz denen von EARLIEST DELIVERY WITH LOCAL QUEUING und EARLIEST DELIVERY WITH ALL QUEUES. Unter hoher Last trifft dies weiterhin auf die mittlere Latenzzeit zu. Die maximale Latenz hingegen ist in der berechneten Lösung zum Teil deutlich höher — verständlich, nachdem die Berechnung auf eine Minimierung der mittleren, nicht der maximalen Latenz abzielt.

7 Zusammenfassung und abschließende Bemerkungen

Nach einer Einführung in die Ideen und den Aufbau von verzögerungstoleranten Netzwerken und Motivation ihres Studiums durch Aufzeigen einiger Einsatzmöglichkeiten wurden verschiedene Ansätze für Routing-Algorithmen vorgestellt und kurz ihre Vor- und Nachteile aufgezeigt. Sofern in den zugrundeliegenden Dokumenten vorhanden, wurde die Präsentation jedes Verfahrens durch Nennung einiger Testergebnisse und Vergleich mit den Ergebnissen vorangegangener Algorithmen verglichen. Abbildung 2 auf der nächsten Seite soll diese Testergebnisse für das Beispielszenario zu Kwazulu-Natal noch einmal graphisch veranschaulichen (verwendete Abkürzungen in der Abbildung: MED: MINIMUM EXPECTED DELAY, ED: EARLIEST DELIVERY, EDLQ: EARLIEST DELIVERY WITH LOCAL QUEUING, EDAQ: EARLIEST DELIVERY WITH ALL QUEUES, LP: Vergleichswerte der linearen Optimierungsaufgabe (engl. linear programming problem)).

An dieser Stelle soll noch einmal darauf hingewiesen werden, dass den präsentierten Testergebnissen Einschränkungen zugrunde liegen, die im realen Einsatz nicht vorausgesetzt werden können. So wurden etwa Ausfälle von Knoten oder Kanten nicht berücksichtigt. Des Weiteren wurde implizit davon ausgegangen, dass während der gesamten Dauer von Kontakten Daten übertragen werden können. Aufgrund des endlichen Speichers von MULEs ist dies natürlich nicht immer gegeben; bei vollem Speicher ist mit nicht vorhersehbaren Verzögerungen (bei Kontrolle des freien Speichers vor der Übertragung) oder Datenverlust (keine Kontrolle des freien Speichers) zu rechnen. Der

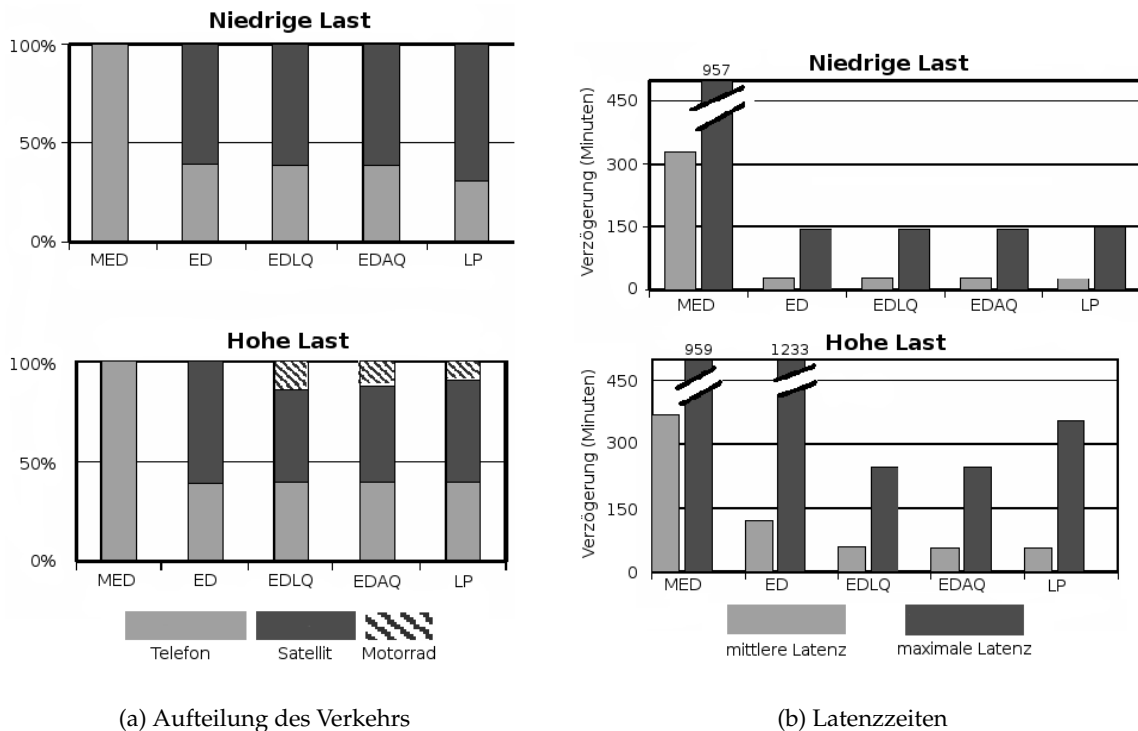


Abbildung 2: Ergebnisse des Tests zwischen Kwazulu-Natal und Kapstadt

Umgang mit derartigen Situationen ist in einigen der zugrundeliegenden Arbeiten als zukünftiges Forschungsgebiet erwähnt, zur Zeit scheinen aber noch keine fertigen Arbeiten zum Thema zu existieren.

Die Bilder in dieser Arbeit sind [J 04] entnommen, die Beschriftungen angepasst.

Literatur

- [JFP 04] Sushant Jain, Kevin Fall, Rabin Patra: *Routing in a Delay Tolerant Network*, SIGCOMM, September 2004
- [F 03] Kevin Fall: *A Delay-Tolerant Network Architecture for Challenged Internets*, Februar 2003
- [SRJB 03] Rahul C. Shah, Sumit Roy, Sushant Jain, Waylon Brunette: *Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks*, IEEE SNPA, Mai 2003
- [WIZZY] The wizzy digital courier, <http://wizzy.org.za>
- [VB 00] Armin Vahdat, David Becker: *Epidemic Routing for Partially-Connected Ad Hoc Networks*, Technical report, Duke university, 2000
- [CPLEX] CPLEX Linear Programming Solver, <http://www.ilog.com>
- [J 04] Sushant Jain: *Routing in a Delay Tolerant Network*, Präsentation, DTN Meeting, August 2004
- [HFP 03] Amir Alexander Hasson, Dr. Richard Fletcher, Dr. Alex (Sandy) Pentland: *Dak-Net: A Road To Universal Broadband Connectivity*, 2003, <http://courses.media.mit.edu/2003fall/de/DakNet-Case.pdf>