Routing in Ad-Hoc Netzen

Bär Urs

14. Mai 2001

Zusammenfassung

In Ad-Hoc Netzen geraten die Routingprotokolle für klassische Netzwerke an ihre Grenzen. Daher wurden eine Vielzahl neuer Routingalgorithmen für diese Art von Netzen entwickelt. In dieser Ausarbeitung werden einige ausgesuchte Routingverfahren erklährt.

1 Einleitung

In einer zunehmend mobilen und ubiquitären Umgebung wird drahtlose Kommunikation ohne grundlegende Infrastruktur immer wichtiger. Jedoch die Dynamik solcher Ad-Hoc Netze stellt den Routingprotokollen neue Anforderungen, denen die im Internet verwendeten Protokolle, wie Distance Vector Routing oder Link State Routing, nicht mehr genügen. Aus diesem Grunde wurden auf dem Gebiet der Ad-Hoc Netze eine Vielzahl neuer Routingverfahren entwickelt.
Das Hauptproblem beim Routing in Ad-Hoc Netzen ist die Änderung der Netztopologie. Diese kann durch die Mobilität der Knoten zu dynamisch für das Protokoll werden, insbesondere, wenn bei jeder Änderung alle Knoten benachrichtigt werden müssen.
Ein weiteres Problem bei Ad-Hoc Netzrouting liegt im Wesen deser Netze. Ad-Hoc Netze bestehen aus mobilen Knoten, die drahtlos miteinander kommunizieren. Diese Kommunikation geschieht ohne grundlegende Infrastruktur, wie zum Beispiel Basisstationen. Da diese Knoten mobil sind und wie gesagt drahtlos kommunizieren, werden sie im allgemeinen durch Batterien mit Strom versorgt. Ein Routingprotokoll sollte also möglichst wenige Pakete verschicken, um den Stromverbrauch niedrig zu halten.

2 Routingverfahren

Die vorhandenen Routingverfahren für Ad-Hoc Netze lassen sich in zwei Gruppen unterteilen, nämlich in die tabellenorientierten und in die on-Demand Routingverfahren. Während bei den tabellenorientierten Verfahren versucht wird ein konsistentes Bild der Netztopologie in Tabellen zu speichern und nachzuführen, wird bei den on-Demand Verfahren erst dann versucht einen Pfad zum Zielknoten zu finden, wenn einer gebraucht wird.

2.1 Tabellenorientiertes Routing

Die tabellenorientierten Routingverfahren basieren auf den heute im Internet verwendeten Verfahren Distance Vector und Linke State Routing. Allerding wurden sie an die veränderte

Umgebung angepasst und optimiert. Da die vorhin genannten Protokolle bekannt sein dürften, will ich in diesem Abschnitt nur kurz auf das Grundprinzip des Distance Vector Routing eingehen. Tiefergehende Betrachtungen würden den Rahmen dieser Ausarbeitung sprengen. Eine genaue Beschreibung des Distance Vector Algorithmus findet sich in [4], diejenige der Ad-Hoc Variante ist in [3] zu finden.
Das Prinzip des Verfahrens ist es, eine möglichst konsistente Sicht der Netztopologie in Tabellen zu speichern, und diese Tabellen zu verwenden, um die Pakete weiter zu leiten. Um die Konsistenz der Tabellen zu wahren, verschickt ein Knoten periodisch Routing-Packete an seine Nachbarn. Ein solches Routing-Paket enthält jeweils die momentane Routingtabelle des sendenden Knotens.
Der Empfänger eines solchen Routing-Paketes bringt mit den aus dem Paket gewonnenen Informationen seine Routingtabelle auf den neuesten Stand, und schickt weitere Routing-Pakete los, falls sich in seiner Tabelle etwas geändert hat. Nach einer gewissen Zeit wird das System wieder in einen stationären Zustand zurück finden. Es ist offensichtlich, dass die hohe Dynamik der Ad-Hoc Netze in diesem Protokoll das Hauptproblem ist.

2.2 On-Demand Routing

Bei den on-Demand Routingverfahren wird das Routingproblem in zwei Phasen unterteil. In der ersten Phase (*Route Discovery*) wird versucht, einen Pfad zum Empfängerknoten zu finden. Ist der Pfad gefunden, wird in der zweiten Phase (*Route Maintenance*) versucht, diesen Pfad während des Gebrauchs aufrecht zu erhalten. Es ist leicht einzusehen, dass, im Gegensatz zu den tabellenorientierten Routingverfahren, eine Änderung in der Topologie nur diejenigen Knoten beeinflusst, die direkt davon betroffen sind. Das heisst diejenigen Knoten, welche sich gerade bewegen, oder mit solchen kommunizieren.
Wie funktioniert nun das Finden und Unterhalten eines Pfades? Diese Frage wird anhand zweier ausgesuchter Routingverfahren, dem Dynamic Source Routing und dem Terminode Routing, erläutert.

2.2.1 Dynamic Source Routing

Wie bei allen on-Demand Verfahren, ist auch das Dynamic Source Routing in Route Discovery und Route Maintenance unterteilt. Während der Discovery-Phase versucht das Verfahren, einen Pfad zum Empfänger zu finden. Dies geschieht mittels eines Request/Reply-Verfahrens. Der Absender eines Paketes schickt einen **Route Request** an alle seine Nachbarn. Dieser Request hat einen eindeutigen Bezeichner. Dieser Bezeichner ermöglicht, dass im Folgenden Duplikate ein und desselben Requests erkannt werden.
Was geschieht nun in einem Knoten, wenn ein solcher Route Request ankommt? Als erstes wird der Knoten die Zieladresse des Requests mit seiner eigenen Adresse vergleichen. Ist es dieselbe, sendet er den Pfad im Request mittels eines **Route Reply** Packetes an den Initiator des Route Requests zurück.
Falls die Nachricht nicht für den Knoten war, gibt es zwei Möglichkeiten fort zu fahren. Im Falle, dass der Knoten den Request mit diesem Bezeichner erst kürzlich schon einmal gesehen hat, so weiss er, dass es sich beim eben erhaltenen Packet um ein Duplikat handelt, und wird es nicht mehr weiterschicken. Falls es ein neuer Request ist, fügt er seine Adresse an den Pfad im Route Request, und schickt diesen an alle seine Nachbarn. Dies führt allerdings dazu, das bei jedem Suchen eines Pfades der Route Request im ganzen Netzwerk verteilt wird (flooding).

Sobald ein Pfad vom Sender zum Empänger besteht, muss dieser während er gebraucht wird unterhalten werden (Route Maintenance). Dabei ist vor allem wichtig, heraus zu finden, ob die Gegenstation noch sendet oder nicht. Dies kann man auf verschiedene Arten tun. Die einfachste ist, man lässt sich von der Station, an die man soeben ein Paket sandte eine Bestätigung schicken, dass sie die Nachricht erhalten hat. Eine andere Möglichkeit ist es, in den Äther zu hören, ob die Gegenstation das Packet weiterleitet. In beiden Fällen kann der Knoten sicher sein, dass die Nachricht weitergeleitet ist, und die Gegenstation noch funktioniert.
Was ist aber, wenn eine Verbindung zwischen zwei Knoten nicht mehr funktioniert? In diesem Fall sendet der Knoten, der das Problem festgestellt hat, ein **Route Error** Packet an den ursprünglichen Sender. Dieser entfernt dann den unterbrochenen Pfad aus seinem Cache. Falls der Sender noch andere Pfade zum Empfänger in seinem Cache hat, kann er für die weitere Übertragung einen von diesen verwenden, oder aber einen neuen Route Request durchführen.

2.2.2 Terminode Routing

Wie DSR ist auch das Terminode Routing ein on-Demand Verfahren. Der Unterschied zum Dynamic Source Routing liegt in der Art, wie ein Pfad gefunden wird. Jeder Knoten, auch Terminode genannt, besitzt einen permanenten, eindeutigen Bezeichner (EUI = End System Unique Identifier) sowie eine temporäre positionsabhängige Adresse (LDA = Location Dependent Address). Diese LDA kann zum Beispiel aus einem Tripel der geographischen Länge, Breite und der Höhe bestehen, wie es von jedem GPS-Gerät geliefert werden kann.
Das Routingverfahren besteht aus zwei Protokollen, die zusammenarbeiten. Das so genannte *Terminode Local Routing* (TLR), welches für Paketzustellungen innerhalb eines lokalen Radius (momentan sind dies zwei Hops) zuständig ist, so wie dem *Terminode Remote Routing* (TRR), welches das Zustellen über lange Distanzen übernimmt.
Wie schon gesagt, wird das TLR verwendet, um Knoten innerhalb eines lokalen Radius, der sogenannten Nachbarschaft, zu erreichen. Dazu wird lediglich die EUI gebraucht. Denn jeder Knoten führt eine Tabelle mit all seinen Nachbarn, die innerhalb des lokalen Radius sind. Um die Nachbarn zu erkennen, sendet jeder Knoten HELLO-Nachrichten an all seine unmittelbaren Nachbaren. Diese HELLO-Nachrichten enthalten Paare von EUI und LDA der unmittelbaren Nachbarn des Knotens, sowie die des Knotens selbst. Man kann recht einfach einsehen, dass dadurch jeder Knoten seine Nachbarn im Umkreis von zwei Hops erkennt. Das Weiterleiten einer Nachricht geschieht nun aufgrund der Nachbarschaftstabelle.
Ist der Empfänger nicht in der Nachbarschaft des Senders zu finden, wird das Terminode Remote Routing verwendet, bis man zu einem Knoten in der Nachbarschaft des Empfängers kommt. Dazu verwendet man als Grundverfahren das *Geodesic Packet Forwarding*. Dieses Verfahren braucht die LDA des Zielknotens. Es wird in der Beschreibung des Algorithmus [5] angenommen, dass man irgendwie an diese Information kommt. Sei es durch Tracking des Knotens oder durch verteilte Datenbanken (weitere Informationen zu diesem Thema finden sich in [9]). Hat man also die LDA des Zielknotens gefunden, sendet der Knoten seine Nachricht an denjenigen Knoten, der am nächsten am Empfänger ist. Dieser Zwischenknoten schaut, ob er den Empfänger via TLR erreichen kann, oder nicht. Wenn dies möglich ist, sendet er die Nachricht direkt an den Empfänger, ansonsten sendet er die Nachricht wiederum an jenen Knoten weiter, der von ihm aus gesehen am nächsten beim Ziel ist. Dieses Verfahren hat allerdings ein Problem, wenn knotenfreie Zonen im Netz existieren. Somit kann es vorkommen, dass es zu einem Knoten keinen anderen Knoten mehr gibt, der näher

am Ziel ist als er selbst, der Knoten jedoch nicht mit dem Empfänger kommunizieren kann. Dies kann allerdings durch sogenannte Anker behoben werden. Die Anker sind geographische Punkte, die nicht unbedingt mit Positionen von Terminodes zusammenfallen müssen. Die zu verschickenden Packete werden nun von einem Anker zum anderen geschickt, bis sie am Ende beim Empfänger ankommen.
Eine Frage stellt sich nun unwillkürlich. Wie finde ich solche Anker? Zu diesem Zweck wird das so genannte *Friend Assisted Path Discovery* Verfahren verwendet. Dieses Verfahren beruht auf dem Konzept der "small world graphs" wie in [10] erläutert. Dies soll aber nicht Gegenstand dieser Ausarbeitung sein. Beim Friend Assisted Path Discovery Verfahren hat jeder Knoten eine gewisse Anzahl von Freunden. Ein Freund F ist dadurch definiert, dass der Knoten A einen guten Pfad zu F hat und F in der Liste der Freunde von A ist. Das Grundprinzip dieses Verfahrens besteht darin, das die Nachricht an den Freund geschickt wird, der am nächsten am Ziel ist. Allerdings gibt es auch hier Methoden, um die knotenfreien Zonen zu umgehen. Diese Methoden hier zu beschreiben, würde den Rahmen dieser Ausarbeitung sprengen. Ein genauer Beschrieb befindet sich in [5].

3 Routing in nichtkooperativer Umgebung

Den bis jetzt beschriebenen Routingverfahren lag die Annahme zugrunde, dass die Knoten kooperieren. Doch dies ist im alltäglichen Einsatz nicht immer erwünscht. Denn ein Weiterleiten der Nachricht verbraucht zwar Batterien, bringt aber keinen direkten Nutzen für den weiterleitenden Knoten. Daher wurden verschiedene Methoden entwichelt, um die Last möglichst gleichmässig auf die vorhandenen Knoten zu verteilen (siehe auch [6], [7] und [8]). Allerdings werde ich nur auf eines dieser Verfahren eingehen.

3.1 Routing mit Nuglets

Die Idee hinter dieses Verfahrens liegt darin, dass entweder der Sender oder der Empfänger für das Routing seiner Pakete zahlt. Zu diesem Zweck wurde eine virtuelle Währung eingeführt: *die Nuglets*.
Wie schon gesagt gibt es zwei Möglichkeiten, wer für das Routing bezahlt, der Sender oder der Empfänger. Ich will aus Platzgründen hier nur das Modell des bezahlenden Senders, *Packet Purse Model* genannt, behandeln. Das Modell des bezahlenden Empfängers, *Packet Trade Model*, wird in [6] beschrieben.
Wie schon gesagt, zahlt beim Packet Purse Model der Sender für die Weiterleitung der Pakete. Für dies läd er das zu sendende Paket mit genügend Nuglets, und schickt es an einen benachbarten Knoten. Jeder Knoten, der das Paket weiterleitet, entnimmt seinen Kosten entsprechend Nuglets aus dem Paket. Die Knoten auf dem Weg können im Verlauf der Zeit ihre Kosten dem inneren Zustand wie Restenergie in der Batterie oder ähnlichem anpassen. Natürlich gibt es in diesem Modell noch einige Probleme, die gelöst werden müssen. So ist zum Beispiel der Raub oder die Fälschung von Nuglets ein offensichtliches Problem.

Literatur

[1] Eliyabeth M. Royer, C-K Toh: *A Review of Current Routing Protocels for Ad-Hoc Mobile Wireless Networks*, April 1999

[2] David B. Johnson: *Routing in Ad Hoc Networks of Mobile Hosts*, Dezember 1994

[3] Charles E. Perkins, Pravin Bhagwat: *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers*, Oktober 1994

[4] Larry L. Petterson, Bruce S. Davie: *Computer networks, a systems approach*, Morgan Kaufmann Publishers, 2000

[5] Ljubica Blazević, Silvia Giordano, Jean-Yves Le Boudec: *Self Organized Terminode Routing*

[6] Levente Buttyán, Jean-Pierre Hubaux: *Nuglets: a Virtual Currency to Stimulate Cooperation in Self-Organized Mobile Ad Hoc Networks*, 18.01.2001

[7] Piyush Gupta, P. R. Kumar: *A System and Traffic Dependent Adaptive Routing Algorithm for Ad Hoc Networks*, 1997

[8] Jae-Hwan Chang, Leandros Tassiulas: *Energy Conserving Routing in Wireless Ad-hoc Networks*, März 2000

[9] J. Li, J. Jannotti, K. De Couto, D. Karter, R. Morris: *A Scalable Location Service for Geographic Ad Hoc Routing*, Mobicom'00, Boston, 2000

[10] D. J. Watts: *Small Worlds, The dynamics of networks between order and randomness*, Princeton University Press, 1999