Routing in Sensornetzen II

Philipp Tölke
Betreuer: Alexander Klein
Seminar Sensorknoten: Betrieb, Netze und Anwendungen Sommersemester 2010
Lehrstuhl Netzarchitekturen und Netzdienste, Lehrstuhl Betriebssysteme und Systemarchitektur
Fakultät für Informatik, Technische Universität München
Email: toelke@in.tum.de

KURZFASSUNG

Das Routing in drahtlosen Sensornetzen, die zumeist aus vielen kleinen Knoten bestehen, stellt große Anforderungen. Die Anwendung von klassischen Routingprotokollen ist auf Grund der Ressourcenknappheit nicht möglich. Diese Ausarbeitung untersucht einige Routingkonzepte anhand der Betrachtung von konkreten Routingprotokollen.

Schlüsselworte

Routing, Survey, drahtlose Sensornetze

1. EINLEITUNG

In Netzwerken aus kleinen Sensorknoten, die zunehmend in der Automatisierung, in Industrie, Haushalten und zur Umweltüberwachung eingesetzt werden, werden Daten von den Sensoren per Funk zur Auswertung transportiert.

Speicher, Rechenleistung und zur Verfügung stehende Energie ist in diesen Sensorknoten sehr beschränkt. Deshalb sind die aus dem Internet bekannten Routingprotokolle, wie zum Beispiel das Border Gateway Protocol (Bgp), dass im Internet das Inter-Autonomous-System-Routing verwaltet, nicht einsetzbar. Auch eine Eintragung der Routen von Hand verbietet sich bei Sensornetzen von 100 oder mehr Knoten. In manchen Anwendungen wird zusätzlich gefordert, dass sich die Knoten frei bewegen können, Routinginformationen müssen daher ständig erneuert werden.

Diese Seminararbeit soll einen Überblick darüber geben, wie die Anforderungen des Routings erfüllt werden können.

Zunächst wird ein Überblick über verschiedene Konzepte des Routings in Sensornetzen gegeben. Zu diesen Konzepten werden dann exemplarisch Protokolle vorgestellt, die zum Abschluss gegenübergestellt werden um eine Auswahl für einen gegebenen Anwendungsfall treffen zu können.

2. PROAKTIVE UND REAKTIVE ROUTING-PROTOKOLLE

Ein Routingprotokoll, das Routen bestimmt bevor sie wirklich gebraucht werden, nennt man „proaktiv“[6]. Ein Beispiel für ein proaktives Protokoll ist das Collection Tree Protocol (Ctp).

Erstellt ein Protokoll erst dann Routen, wenn ein Paket gesendet werden soll, nennt man es „reaktiv“. Ad-Hoc On-Demand Distance Vector (Aodv) ist ein reaktives Protokoll.

Reaktive Protokolle haben eine größere Latenz für das erste Paket einer „Verbindung“, verbrauchen aber dafür in der Regel weniger Energie Routinginformationen aufzubauen.

3. METRIKEN

Um Routingentscheidungen treffen zu können, müssen Knoten in einem Sensornetz „wissen“, wie weit sie vom Empfänger einer Nachricht entfernt sind, und welcher direkt erreichbare Knoten näher am Empfänger ist.

Die einfachste solcher Abstandsmetriken ist der Hop-Count. Zwei Knoten, die direkt kommunizieren können, haben den Abstand 1. Können zwei Knoten über genau einen Mittelsmann kommunizieren, haben sie den Abstand 2.

In die Routingmetrik können auch Umwelteinflüsse, wie beispielsweise die zur Verfügung stehende Energie einfließen: Knoten, die nur noch über wenig Energie verfügen, geben eine schlechtere Verbindung zu allen Empfängern an, als sie eigentlich haben. Das macht es unwahrscheinlicher, dass sie Pakete weiterleiten müssen und noch mehr Energie verbrauchen.

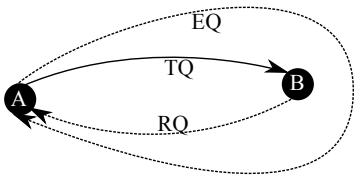

Abbildung 1: Batman: Die drei Werte TQ, RQ und EQ. EQ und RQ (gestrichelt) können direkt gemessen werden.

Im Protokoll Batman („Better Aproach To Mobile Ad-Hoch Networking“) Version IV wird eine Metrik vorgeschlagen, bei dem jeder Knoten ausrechnet, wie viele seiner Pakete auf dem Weg zu direkten Nachbarn verloren gehen. Dazu sendet jeder Knoten regelmäßig einen Broadcast um seine Position bekannt zu geben. Diese Broadcasts werden weitergeleitet und so im Netz verteilt. Weiterhin sind sie durchnummeriert, so dass ein Knoten erkennen kann, wenn er Broadcasts seiner Nachbarn nicht bekommen hat. Er verfügt damit über zwei Messwerte: die Qualität der Verbindung von seinen Nachbarn RQ („Receiving Quality“) und er kann messen wie viele seiner eigenen Broadcasts der Nachbar seinerseits weitergeleitet hat, EQ („Echo Quality“). Die gesuchte Metrik ist die TQ („Transmit Quality“). Sie beschreibt wie viele Pakete beim Nachbarn ankommen. Es gilt offensichtlich:

$$EQ = TQ \cdot RQ, \tag{1}$$

der Anteil der empfangenen wiederholten Broadcasts ist gleich der Anzahl der vom Nachbarn empfangenen Broadcasts multipliziert mit der Wahrscheinlichkeit überhaupt ein vom Nachbarn gesendetes Paket zu empfangen.

Durch Umformung lässt sich die gesuchte Sendequalität berechnen:

$$TQ = \frac{EQ}{RQ} \tag{2}$$

TQ kann als Metrik dafür eingesetzt werden, wie zuverlässig die Verbindung zum Nachbarn ist.[5]

4. PROTOKOLLE MIT WENIGEN EMPFÄNGERN

In vielen Anwendungen ist es ausreichend, wenn nicht jeder Knoten mit jedem anderen Knoten beliebig kommunizieren kann, sondern wenn nur bestimmte Knoten Daten empfangen können. So ist es beispielsweise zur Temperaturüberwachung eines Raumes nicht nötig, dass die Knoten untereinander Daten austauschen, sondern nur, dass die Temperaturdaten an bestimmte Steuerknoten gesendet werden. Diese Knoten nennt man „Senken“.

4.1 Collection Tree Protocol

Das „Collection Tree Protocoll“ (CTP) ermöglicht es einen logischen Routing-Baum zu erstellen, so dass jeder Knoten über einen möglichst kurzen Pfad Daten zu einer Senke schicken kann[2]. Es wurde als TinyOs Enhancement Proposal (TEP) entwickelt.

Mit welcher Metrik die Pfadlänge gemessen wird, hängt vom Anwendungsfall ab. Im ursprünglichen Vorschlag (TEP-123[1]) wird die erwartete Anzahl der Sendewiederholungen auf Schicht 2 herangezogen.

Das Protokoll stellt an die darunter liegende Schicht 2 die Anforderung, dass sich effizient Broadcasts an alle Knoten in der Nähe schicken lassen, also an alle Knoten, die eine Nachricht direkt empfangen können. Weiterhin fordert das Protokoll, dass Unicast-Nachrichten auf Schicht 2 bestätigt oder nötigenfalls wiederholt werden.

Jeder Knoten im Netzwerk sendet regelmäßig eine Nachricht („Beacon“) aus, in der er seinen Abstand zur nächsten Senke bekannt gibt. Erhält ein Knoten einen Beacon mit dem ein geringerer Abstand zur Senke bekannt gegeben wird, als dem Knoten bisher bekannt ist, wird er von jetzt an Nachrichten an die Senke über den Absender des Beacons senden.

Diese Beacons werden nach dem Trickle-Algorithmus[4] versandt. Dieser sorgt dafür, dass im stabilen Betrieb nur wenig Beacons versandt werden müssen, indem das Sendeintervall schrittweise vergrößert wird. Es wird in folgenden Fällen auf den kleinsten Wert zurückgesetzt:

Der Knoten bekommt ein Paket von einem Knoten, dessen Pfadlänge zur Senke nicht größer ist als die eigene zur Weiterleitung.

Dies kann passieren, wenn Nachbarn falsche oder veraltete Routinginformationen haben.

Die Pfadlänge verringert sich durch einen empfangenen Beacon erheblich.

Diese Änderung wird sofort allen Nachbarn weitergegeben, da es eine Verbesserung der Routen sein kann.

Ein Paket mit gesetztem P-Flag wird empfangen.

Ein Knoten, der aus irgendeinem Grund keine Routinginformationen besitzt wird zunächst Pakete mit gesetztem „Pull“-Flag senden, damit Nachbarn ihm ihre Pfadkosten mitteilen können.

4.2 Minimal Cost Forwarding Algorithm

Der MCFA ermöglicht es, wie das CTP Daten an einige Senken zu schicken. Anders als das CTP werden Datenpakete allerdings nicht direkt zum nächsten Hop gesendet, sondern per Broadcast an alle Knoten in der Nähe weitergegeben. Jeder Hop trifft dann lokal die Entscheidung, ob dieses Paket weitergeleitet werden muss.

Jeder Knoten kennt seine Entfernung zur nächsten Senke. Der Absender eines Paketes trägt seine Entfernung als „Kontigent“ in das Paket ein und sendet es per Broadcast ab.

Empfängt ein Knoten ein Datenpaket, überprüft er, ob das verbleibende Kontingent des Pakets größer oder gleich seiner Entfernung zur Senke ist. Ist es kleiner, verwirft er das Paket, er liegt nicht auf dem kürzesten Pfad vom Absender zur Senke. Andernfalls sendet er das Paket mit um die Weiterleitungskosten verringertem Kontigent weiter. Wie hoch die Weiterleitungskosten sind, hängt von der Metrik ab. Wird die Hopzahl als Metrik verwendet, verringert jeder Knoten das Kontingent eines Paketes jeweils um eins.

Anschaulich hat jeder Knoten eine Höhe, die den Abstand zur Senke angibt. Pakete laufen immer von hohen Knoten bergab in Richtung der Senke[7].

4.3 Kostenfindung im MCFA

Um die Höhe der einzelnen Knoten zu ermitteln, werden zu Beginn Broadcasts versendet. Die Senken schicken ein Broadcast-Paket in dem sie die Höhe 0 angeben. Knoten, die diese Pakete empfangen setzen ihre eigenen Kosten auf einen, je nach Metrik, höheren Wert und schicken ihrerseits einen Broadcast ab. Damit nicht alle Knoten kurz hintereinander einen Broadcast schicken müssen, der unter Umständen noch nicht die minimalen Kosten des Knotens enthält (siehe Abbildung 2), wartet der Algorithmus vor Versenden des Broadcasts für eine Zeit, die proportional zur Höhe des Knotens ist. Dadurch ist gewährleistet, dass Knoten, über die eine kürzere Verbindung zur Senke möglich ist, früher ihre Broadcasts abschicken. Es schickt also jeder Knoten nur sehr wenige (im Idealfall einen) Broadcasts ab, bis alle Knoten ihre Höhe kennen. [7]

Da nach einmal erfolgtem Broadcast ein Knoten kein weiteres Mal seine Position bekannt gibt, kann der MCFA nicht

auf Veränderungen des Netzes zu reagieren. Für Anwendungen bei denen Mobilität der Knoten wichtig sind, muss also ein modifizierter Algorithmus verwendet werden, und beispielsweise regelmäßig ein Broadcast geschickt werden.

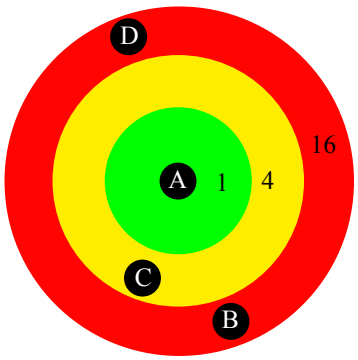

Abbildung 2: MCFA: der Knoten A sendet einen Broadcast aus. Als Metrik wird die Empfangsstärke angenommen, die quadratisch mit der Entfernung abnimmt. Knoten C nimmt also Entfernung 4 an, Knoten B und D die Entfernung 16. Würde Knoten B jetzt ohne zu warten einen Broadcast senden, müsste er nachdem Knoten C seinen Broadcast sendet erneut einen senden: Er hätte dann die Kosten 8.

Unverändert ist dieses Protokoll nur für die Konfiguration von statischen Netzen, zum Beispiel für die Umweltüberwachung, nutzbar.

5. PROTOKOLLE MIT BELIEBIGEN EMPFÄNGERN

Soll es den Knoten möglich sein, beliebig untereinander zu kommunizieren, muss jeder Knoten einen Eintrag in der Routingtabelle jedes andern Knoten haben.

5.1 AODV

AODV ist ein reaktives Protokoll. Die Routinginformationen werden erst dann erstellt, wenn ein Datenpaket gesendet werden soll.

Ein Knoten, der ein Paket zu einem andern Knoten schicken möchte, sendet zunächste eine RREQ-Nachricht („Route Request“) per Broadcast ab. Andere Knoten geben diese Nachricht weiter, bis sie einen Knoten erreicht, der einen gültigen Eintrag für das Ziel hat. Dieser Knoten kann natürlich auch das Ziel selbst sein. Er sendet jetzt eine RREP-Nachricht („Route Reply“), die in der Regel dengleichen Weg durch das Netzwerk zurück nimmt, den die RREQ-Nachricht genommen hat. Dazu schickt jeder Knoten die RREQ-Nachricht mit seiner Adresse als Absender ab und speichert für einige Zeit, dass er diese Nachricht gesehen hat. Bekommt er eine RREP, der zu einem RREQ passt, den er weitergeleitet hat, speichert er den Absender des RREP als nächsten Hop zum angefragten Knoten und sendet seinerseits einen RREP ab.

Damit ein RREQ nicht im kompletten Netz verteilt wird, setzt der anfragende Knoten im erstem Broadcast die TTL (Time To Live, wird bei jedem Hop dekrementiert. Ist sie 0 wird das Paket nicht mehr weitergeleitet) sehr niedrig, so dass das Paket früh verworfen wird. Trifft nach einiger Zeit kein RREP ein, sendet der Knoten erneut einen Broadcast mit erhöhter TTL. Die TTL wird so lange erhöht, bis innerhalb eines Timeouts eine RREP eintrifft. In Abbildung 3 ist beispielhaft der Weg von RREQ-Nachrichten von Knoten A zu Knoten B mit TTL 3 gezeigt.

Der ursprüngliche Absender des RREQ erhält irgendwann einen RREP von dem Knoten, der auf dem Pfad zum Zielknoten am nächsten liegt. Das Datenpaket kann jetzt über diesen Pfad losgeschickt werden.

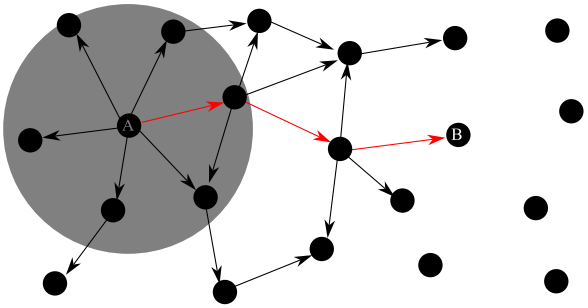

Abbildung 3: RREQ-Nachrichten von A zu B mit TTL 3. Der spätere Kommunikationskanal zwischen A und B ist rot gezeichnet. In grau ist der Broadcastradius von A angezeigt, der der anderen Knoten ist genauso groß.

Erhält ein Knoten ein Datenpaket zur Weiterleitung, kennt aber keinen nächsten Hop für das Ziel des Pakets, sendet er ein RERR-Paket („Route Error“) ab. Dieses läuft den Weg des Datenpakets zurück und sorgt dafür, dass alle Knoten auf dem Weg ihren Routeneintrag für das ursprüngliche Ziel löschen.

5.2 SBR – Statistics Based Routing

SBR ist ein Protokoll, das beliebigen Knoten ermöglicht untereinander zuverlässig zu kommunizieren. Es lässt sich einerseits als proaktives Protokoll einsetzen, jeder Knoten schickt dann regelmäßig so genannte „hello“-Nachrichten um seine Position bekannt zu geben. Diese Pakete werden weitergeleitet und die Routinginformationen darüber aufgebaut.

Andererseits kann es auch in einem hybriden Modus benutzt werden, bei dem, ähnlich wie im AODV eine Routenanfrage gestellt wird, die dann beantwortet wird. Im Unterschied zum AODV sendet der designierte Empfänger aber seine „hello“-Nachricht nicht nur einmal, sondern wiederholt sie regelmäßig, so dass eine Routenänderung sofort erkannt wird.[3]

Der Protokollablauf ist sehr ähnlich zum CTP bzw. zum AODV und wird deswegen hier nicht weiter dargestellt.

6. GEGENÜBERSTELLUNG DER PROTOKOLLE

Protokoll	re-/proaktiv	Struktur	Empfänger
CTP	proaktiv	Baumstruktur	wenige
MCFA	proaktiv	Unstrukturiert	wenige
SBR	einstellbar	kürzeste Wege	beliebig
AODV	reaktiv	kürzeste Wege	beliebig

Bis auf den MCFA erlauben alle Protokolle, dass sich die Knoten bewegen. SBR kann durch das periodische Absenden von „hello“-Nachrichten bei bestehender „Verbindung“

am schnellsten auf Änderungen reagieren, da ohne Fehlerbehandlung (RERR bei AODV bzw. erneutes Absenden eines Beacons bei CTP) die neue Route ermittelt wird.

7. ZUSAMMENFASSUNG

Es wurden viele Routingverfahren für drahtlose Sensornetze entwickelt, die unterschiedliche Anforderungen erfüllen. Bei der Planung eines Sensornetzes muss entschieden werden, welches dieser Verfahren für den Einsatzzweck optimal ist.

Die wohl wichtigste Fragestellung ist, ob die Knoten untereinander kommunizieren sollen oder ob es ausreicht, wenn nur einige spezielle Knoten Daten von allen anderen erhalten können. Sollen die Knoten untereinander kommunizieren, ist es wichtig abzuwägen, ob die Routinginformationen zu Beginn aufgebaut werden sollen, oder ob es reicht damit zu warten, bis sie benötigt wird.

8. LITERATUR

[1] R. Fonseca, O. Gnawali, K. Jamieson, S. Kim, P. Levis, and A. Woo. Tep123: The collection tree protocol. 2006.

[2] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*, pages 1–14, November 2009.

[3] A. Klein. Statistics based routing (sbr), technical report no. 453. `http://www.routingprotokolle.de/Routing/Publications/ResearchReport/SBR_TR453_paged.pdf`, January 2009.

[4] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *In Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI)*, pages 15–28, 2004.

[5] A. Neumann, C. Aichele, M. Lindner, and S. Wunderlich. Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.). Internet-Draft, pp. 1-24, April 2008. Network Working Group.

[6] G. Pei, M. Gerla, and T.-W. Chen. Fisheye state routing: a routing scheme for ad hoc wireless networks. In *Proc. IEEE International Conference on Communications ICC 2000*, volume 1, pages 70–74, June 2000.

[7] F. Ye, A. Chen, S. Lu, and L. Zhang. A scalable solution to minimum cost forwarding in large sensor networks. In *Proc. Tenth International Conference on Computer Communications and Networks*, pages 304–309, October 2001.