

Anfragespezifische Routingmechanismen

Seminar „Informationsverwaltung in Sensornetzen“

Universität Karlsruhe (TH)
Fakultät für Informatik
IPD - Lehrstuhl Prof. Böhm

von Maria Kopaigorenko

Betreuer: Markus Bestehorn

Zusammenfassung Da die drahtlose Kommunikation den Energieverbrauch der Sensorknoten dominiert, ist die Wahl eines energieeffizienten Routingmechanismus für die Gewährleistung der möglichst langen Lebensdauer eines Sensornetzes sehr wichtig. Im Rahmen des Seminars „Informationsverwaltung in Sensornetzen“ wird in dieser Arbeit zuerst der übliche Aufbau eines Routingbaumes skizziert. Ausgehend von den Problemen, die von dem Ansatz ungelöst bleiben, werden zwei weitere Routingverfahren (Directed Diffusion und Broadcasting-Based query Schema) vorgestellt.

2

1 Einleitung.....	3
2 Routing in Sensornetzen	4
2.1 Anfrageverteilung.....	5
2.2 Anfrageverarbeitung.....	7
3 Directed Diffusion.....	9
3.1 Interestverbreitung.....	10
3.2 Datenweiterleitung.....	11
3.3 Verstärken/Abschwächen einzelnen Pfade.....	13
3.4 Bewertung.....	14
4 Broadcasting-Based query Scheme.....	14
4.1 Begriffseinführungen.....	15
4.2 Annahmen.....	16
4.3 Algorithmus.....	16
4.3.1 Konstruktion lokaler Routingbäume.....	17
4.3.2 Datenweiterleitung.....	19
4.3.3 Sonderfälle.....	19
4.4 Bewertung.....	21
5 Zusammenfassung und Ausblick.....	22

1 Einleitung

Unter einem Sensornetz versteht man ein Rechnernetz, welches aus einer großen Anzahl unabhängiger, meist über Funk kommunizierender, (potentiell) kostengünstiger Miniaturcomputern besteht. Die so genannten Sensorknoten sind mit Sensoren ausgestattet, die bestimmte physikalische Größen messen. Ein Beispiel für einen solchen Sensorknoten ist der Mica2 Mote [15] [14], welcher an der Universität Berkeley entwickelt wurde und von der Firma Crossbow Technology, Inc. hergestellt und vermarktet wird. Heutzutage oft mit ein paar AA-Batterien betrieben, sollen Sensorknoten über Monate oder Jahre hinweg ohne menschlichen Eingriff ihre Aufgaben erfüllen.

Basisstationen (oder Datensinken) sind spezielle Knoten, die durch z.B. Kabelgebundene- oder Satellitenkommunikationssysteme mit Komponenten außerhalb vom Sensornetzwerk verbunden sind. Die Kommunikation mit Benutzern vom Sensornetzwerk erfolgt meistens über diese „Gateway“-Knoten.

Seit Jahren werden Sensornetze intensiv erforscht, weil die Möglichkeit grosse Anzahl an billigen Sensorknoten fast überall zu verteilen und Sensordaten zur Auswertung einzusammeln neue Perspektiven für verschiedene Anwendungen schafft. Vor allem werden sie für die Überwachungs- und Kontrollaufgaben eingesetzt. So kann man zum Beispiel kleine Sensorknoten am Eingang der Bruthöhlen von Vögeln auslegen, um deren Brutverhalten ungestört zu beobachten, wie es auf einer kleinen Insel an der Küste von Maine gemacht wurde (Great Duck Island [16]). Einsatzmöglichkeiten im Bereich der Gebäudeautomatisierung sind auch denkbar. So können Sensorknoten zur Messung der von einzelnen elektrischen Geräten entnommenen Energie (anhand des Magnetfelds) und zur Bestimmung von Temperatur und Qualität (z.B. Kohlendioxid-Gehalt) der Luft in Räumen verwendet werden [17]. Es existieren viele andere Anwendungsgebiete (Medizin, Militär, Verkehrsüberwachung etc.) für die Sensornetze ein hochinteressantes Forschungsthema sind.

Um Überwachungsaufgaben zu erfüllen werden meist sehr viele Sensorknoten vor Ort in die Umgebung eingebettet. Jeder Sensor kann dabei als eine separate Datenquelle betrachtet werden, die einen kontinuierlichen Datenstrom erzeugt [13]. Daten von verschiedenen Knoten, die über gleiche Sensoren verfügen, können in einer verteilten relationalen Tabelle gespeichert werden. Somit liegt es nahe ein Sensornetz als ein verteiltes Datenbanksystem zu behandeln.

Die einfachste Art in diesem Fall mit dem Sensornetz zu kommunizieren ist, Anfragen an das Netzwerk zu stellen, wobei diese durch das Netz und die vorhandene Sensorik beantwortet werden sollen.

Das nächste Kapitel bietet einen Überblick über das Routing in Sensornetzen, dessen Herausforderungen und Probleme. Im Kapitel 3 wird ein Verfahren

namens Directed Diffusion [2] vorgestellt, das auf eine energieeffiziente Anfrageverbreitung abzielt. Das Kapitel 4 beschäftigt sich mit dem Broadcasting-Based query Schema [1], das das Kommunikationsaufkommen der Sensorknoten nahe der Datensenke zu reduzieren versucht, um deren frühzeitigen Ausfälle zu vermeiden und somit die Lebensdauer des Sensornetzes als Ganzes zu verlängern. Abgeschlossen wird diese Seminararbeit mit einer Zusammenfassung im Kapitel 5.

2 Routing in Sensornetzen

Den Routingmechanismen kommt eine sehr große Bedeutung zu, da ein Sensornetz erst mit ihrer Hilfe seine Aufgabe erfüllen kann. In einem Sensornetz müssen ständig Nachrichten verschickt werden, sei es um die einzelnen Sensorknoten untereinander zu koordinieren oder von Außen kommende Anfragen zu beantworten.

Herausforderungen Sensorknoten stehen nicht in direktem Funkkontakt mit allen anderen Knoten im Netzwerk, deswegen sollen Routingprotokolle die Kommunikation über mehrere Hops erlauben. Also hat ein Sensorknoten nicht nur Daten auszulesen, sondern muss er diese auch vorverarbeiten und weiterleiten können. Für andere Knoten fungiert er dann als Router.

Die Energieeffizienz von Routingprotokollen stellt die Hauptherausforderung bei deren Entwicklung dar, da Sensorknoten typischerweise nur so lange funktionsfähig bleiben, bis ihre Energievorräte ausgeschöpft sind. Weil die Wartung bzw. der Batteriewechsel oft nicht durchführbar ist, bedeutet der Ausfall mehrerer Knoten meistens den Zusammenbruch des ganzen Netzes. Um die Lebensdauer eines Sensornetzes als Ganzes zu maximieren und Ausfälle zu vermeiden, muss eine gleichmäßige Belastung aller Sensorknoten gewährleistet sein.

Es ist dabei zu beachten, dass die Kommunikation meistens energieaufwändiger als das Erfassen und Berechnen der Messwerte ist. Nach [4] ist das Senden eines Bits über 100m so energieaufwändig, wie das Ausführen von 3000 Prozessor Instruktionen.

Ein weiteres Problem, mit dem man zu kämpfen hat, ist die Dynamik eines Sensornetzes. Sensorknoten fallen infolge Defekte, schädlicher Umwelteinflüsse oder aufgebrauchter Energie häufig aus. Die beobachteten Phänomene können ihre Position ändern und somit werden neue Datenquellen hinzugefügt bzw. alte fallen weg. Also müssen die Protokolle robust gegenüber diesen Änderungen sowie Knotenausfällen sein .

Skalierbarkeit ist eine weitere wichtige Anforderung an Routingverfahren, weil Sensornetze aus Tausenden von Knoten bestehen können.

2.1 Anfrageverteilung

Eine Aufgabe von Routingverfahren ist die Auslieferung einer Anfrage an die Sensorknoten, die entsprechende Daten bereitstellen können. Die Schwierigkeit hier entsteht durch die Tatsache, dass es oft Daten von mehreren Quellen benötigt werden um eine Anfrage zu beantworten und es i.A. nicht bekannt ist, welche Knoten die Daten liefern können. Also ist es nicht immer möglich, den Empfänger einer Nachricht mit einer eindeutigen Adresse zu identifizieren. Außerdem ist die Identität von Knoten, welche die Daten bereitstellen, meist weniger von Interesse, als die erhobenen Daten selbst.

Beispiel 2.1: Als Beispiel betrachte man folgende Anfrage: „Wie hoch ist die Durchschnittstemperatur in der geographischen Region $X = (100, 100, 200, 200)$?“ Wobei es sich viele Sensorknoten in dem jeweiligen Gebiet befinden können.

```
select avg(temp)
from sensors
where location in (100,100,200,200) //  $x_1, y_1, x_2, y_2$ 
```

Oft wird dann Flooding [18] mangels anderer Lösungen für die Anfrageverteilung eingesetzt. Beim Flooding wird eine Nachricht via Broadcast versendet. Alle Sensorknoten, die diese Nachricht hören, senden sie wiederum als Broadcast an alle Nachbarn weiter. Nach einer bestimmten Anzahl von Hops oder beim Erreichen des Ziels erfolgt keine Wiederholung mehr. Dieser Mechanismus ist zwar einfach, erfordert keine Routingentscheidungen und gewährleistet, dass alle Knoten immer erreicht werden, verschwendet aber die Bandbreite und die Energie enorm. Die Ineffizienz zeigt sich darin, dass viele Knoten Daten empfangen, die gar nicht an sie adressiert sind, oder dass verschiedene Daten Schleifen durchlaufen und somit unnötig lange im Netz bleiben.

Ein weit verbreiteter Ansatz ist die Konstruktion von Routingbäumen, entlang welcher Anfragen verbreitet werden, bzw. Daten zur Basisstation (Datensenke) fließen.

Aufbau eines Routingbaumes

Typischerweise initiiert die Senke (Knoten S in Abbildung 2.1) den Aufbau eines Routingbaumes mit einer Broadcast-Nachricht (*routing request*), die unter Anderem ihre ID, ihren Level (die Distanz zum Wurzelknoten, also hier auf „0“ gesetzt) enthält. Die Datensenke kann mit ihrem Broadcast oft nur einen Teil von Sensorknoten erreichen. Diejenigen Knoten, die diese Nachricht empfangen haben (Knoten A und B in Abbildung 2.1), erhöhen ihren Level um eins, wählen

die Wurzel als Vaterknoten, erzeugen ihre eigenen ID's und versenden die adoptierte Aufforderung einen Routingbaum zu konstruieren via Broadcast weiter. Die Vorgehensweise wird fortgesetzt, bis das ganze Netzwerk die Nachricht gehört hat.

Zu beachten dabei ist, dass ein Sensorknoten nur einen Sender nach bestimmten Kriterien als seinen Vaterknoten auswählt, falls er Nachrichten von mehreren Knoten empfangen hat. Beispielsweise bekommt der Knoten D „routing request“- Nachrichten von den Knoten A und B, so entscheidet er sich für den Knoten A zum Beispiel aufgrund von geringeren Kosten oder besserer Link-Qualität. Somit kennt jeder Knoten seine Eltern- und Kinderknoten.

Der routing request wird periodisch von der Senke wiederholt um den Routingbaum an mögliche Topologieänderungen anzupassen.

Nachdem der Routingbaum aufgebaut wurde, kann die Senke Anfragen entlang des Baumes verbreiten, indem jeder Knoten sie an alle seine Kinder verschickt.

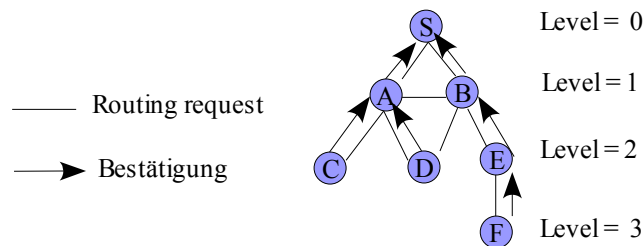


Abbildung 2.1: Aufbau eines Routingbaumes

Problem

Die Initialisierungsphase ist sehr kostenintensiv. Ausßerdem erhält jeder Knoten eine Anfrage unabhängig davon, ob er passende Daten bereitstellen kann oder nicht.

Andere Ansätze

Ein anderer Ansatz ist, Anfragen nicht anhand von globalen Adressinformationen zu verbreiten, sondern anhand von Eigenschaften der Daten, die angefragt sind.

Definition 2.1: Eine Adressierung heisst „Datenzentrisch“ [18], falls der Inhalt der zu übertragenen Daten in jedem Zwischenknoten ausgewertet wird um zu entscheiden, an welche Knoten (wenn überhaupt) eine Nachricht weitergeleitet werden soll. Die Empfänger einer Nachricht werden dabei in der Nachricht selbst mit Hilfe von Attributen spezifiziert.

Falls eine Anfrage geographische Ortsangaben enthält, kann sie bei Verwendung von entsprechenden Algorithmen sofort in die richtige Richtung weitergeleitet werden.

Eine andere Möglichkeit ist, die Datensinke mit einer „power-adjustable“-Funkeinrichtung auszustatten, so dass sie alle Sensorknoten im Netzwerk via Broadcast erreichen kann.

2.2 Anfrageverarbeitung

Nach Erhalt einer Anfrage beginnen die geeigneten Datenquellen ihre Messergebnisse an den Vaterknoten zu senden. Die dicht verteilten Sensorknoten erzeugen oft gleiche oder zumindest ähnliche Daten, so dass es zu hohem Datenaufkommen mit geringem Informationsgehalt kommen kann, wenn alle dieser Daten unkomprimiert an die Datensinke geschickt werden. Deswegen versucht man die Duplikate zu unterdrücken und ähnliche Daten durch Korrelation in den Knoten auf dem Weg zur Senke zu einem höherwertigen Ergebnis zu integrieren. Bei Aggregationsanfragen, wie im Beispiel oben, wird versucht die Aggregation bereits im Sensornetzwerk durchzuführen um die Nachrichtenanzahl und die versendete Datenmenge zu reduzieren. Damit spart man die limitierten Bandbreite und Energie.

Propagieren von Daten

In der Anfrage ist die Dauer enthalten, die angibt, wie lange ein Vaterknoten auf die Datensendung seiner Kinder warten wird. Wobei jeder Knoten diese Zeitspanne verkürzt, so dass alle seine Nachfolger Zeit hätten, die Anfrage zu bearbeiten, sie weiterzuleiten, gegebenenfalls Messungen zu tätigen, die von den Kindern erhaltenen Daten zu aggregieren und das Ergebnis an den Vaterknoten zu versenden.

Also wartet ein Knoten die in der Anfrage angegebene Zeitspanne auf die Antworten seiner Kinderknoten, kombiniert die erhaltenen Ergebnisse mit seinen eigenen Sensordaten und leitet das Resultat an seinen Vorgängerknoten weiter. Auf dieser Weise gelangen die Informationen zum Wurzelknoten.

Da Sensorknoten ausfallen, neue Knoten dazukommen oder Link-Qualitäten sich ändern können, muss der Routingbaum verwaltet werden.

Der Ausfall von einem Sensorknoten, nahe zu den Blattknoten in einem Routingbaum, kann toleriert werden, da die Nachbarknoten meistens ähnliche Daten liefern und diese Daten noch ein ziemlich niedriges Informationsniveau aufweisen. Der Ausfall von direkten Nachbarn der Datensinke kann dagegen zu größeren Datenverlusten führen und sogar zur Partitionierung des Sensornetzes.

Energieverbrauch von Sensorknoten nahe der Datensenke

In Sensornetzen mit zufällig verteilten homogenen Sensorknoten und einer stationären Datensenke verbrauchen Sensorknoten nahe der Senke ihre Energie viel schneller als der Rest des Netzwerkes, falls alle Sensorknoten ungefähr gleich viele Daten an die Datensenke liefern [1]. Die Sensorknoten, welche die Senke mit nur einem Hop erreichen können, haben laut [5] den höchsten Anteil an Weiterleitungsaufgaben. Die hohe Belastung führt dazu, dass die Lebenszeit solcher Knoten viel kürzer ist als die der übrigen Sensorknoten im Netz. Das Sensornetz wird somit partitioniert und die Lebenszeit vom Sensornetz wird verkürzt, obwohl der Rest der Knoten über einen ausreichenden Energievorrat verfügt. Dabei spielen nicht nur die Anzahl und die Größe von weitergesendeten Nachrichten eine Rolle, die Knoten verlieren ihre Energie beim Empfangen der weiterzuleitenden Nachrichten. Man kann sich diese Tatsache anhand der Abb.2.2 veranschaulichen.

Aus dem mathematischen Modell in [5] folgt, dass in großen Sensornetzen bis zu 90% der initialen Energie von Sensorknoten unverbraucht bleibt, nachdem die „ein-hop“-Nachbarn der Datensenke ausgefallen sind. Also ist es nicht ausreichend die Routingprotokolle nur auf die Reduktion der pro Sensorknoten verbrauchten Energie hin zu optimieren. Es müssen auch Lösungen gefunden werden, die außerdem die Lebensdauer des gesamten Sensornetzes maximieren.

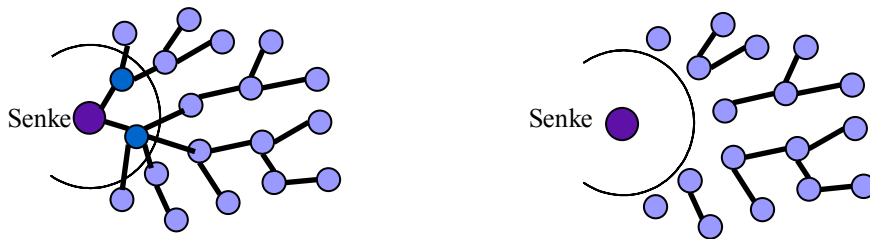


Abbildung 2.2: Routingbaum mit der Datensenke in der Wurzel

3 Directed Diffusion

Directed Diffusion versucht den Energieverbrauch bei der Anfrageverbreitung zu verringern, indem eine Anfrage gezielt in den für ihre Bearbeitung relevanten Teil des Sensornetzes verbreitet wird. Damit sparen die Sensorknoten, die keine passenden Informationen bereitstellen können, die Kosten, die durch den Empfang von nicht relevanten Anfragen entstehen würden. Außerdem können sie diese Zeit im Schlafmodus verbringen.

Directed Diffusion zeichnet sich durch die datenbasierte Adressierung aus. Dabei werden Routingentscheidungen in der Anwendungsschicht aufgrund der vorliegenden Daten und ohne Angabe einer netzweit eindeutigen Adresse getroffen.

Ablauf

Die Datensinke interessiert sich für Informationen eines bestimmten Types. Es ist am Anfang nicht bekannt, ob das Sensornetz diese Informationen liefern kann und falls ja, ob sie in einem oder mehreren Sensorknoten vorliegen. Directed Diffusion besteht aus drei Phasen:

1. **Interestverbreitung** - Eine Anfrage, welche die vom Sensornetz zu erfüllende Aufgabe beschreibt und von den Autoren von Directed Diffusion „*Interest*“ genannt wird, wird im einfachsten Fall im Netzwerk geflutet, um die geeigneten Sensorknoten zur Bearbeitung der Anfrage zu finden. Dabei lernen Zwischenknoten, in welcher Richtung sie die Daten dieses Types später weiterleiten müssen, indem sie sich merken, von welchem Nachbarknoten die Anfrage kam.
2. **Datenweiterleitung** - Die passenden Datenquellen speisen ihre Messergebnisse in das Sensornetz, welche dann auf eventuell mehreren Pfaden zur Senke fließen. Jeder Zwischenknoten muss anhand gespeicherter Informationen entscheiden, ob er ein Paket weiterleiten soll und an welchen Nachbarknoten die Daten gegebenenfalls gesendet werden müssen.
3. **Verstärken einzelner Pfade** - Um die Anzahl der Datenweiterleitungswege einzuschränken verstärkt die Senke einen/oder einige der Pfade, den/die sie für „gut“ hält.

3.1 Interestverbreitung

Die Empfänger einer Nachricht werden bei datenzentrischen Verfahren in der Nachricht selbst mit Hilfe von bestimmten Eigenschaften spezifiziert. Ein Interest wird bei Directed Diffusion im Wesentlichen als eine Liste von so genannten Attribut-Wert-Paaren dargestellt [2]. Das Verfahren wird im Folgenden anhand eines Sensornetzes erläutert, das die Aufgabe hat, Standorte von Tieren zu ermitteln.

Beispiel 3.1: „Detektiere ein vierbeiniges Tier in der Region (x_1, y_1, x_2, y_2) = (-100, 200, 200, 400) und schicke jede Sekunde ein Ergebnis in Richtung der Senke im Laufe der nächsten 10 Minuten“.

Der Interest sieht dann beschrieben in der in [3] vorgeschlagenen SQL-ähnlichen Form folgendermassen aus:

```

ON EVENT four-legged-animal-detect(location)
SELECT event.instance, event.location, time
FROM sensors
WHERE sensors.location in (-100, 200, 200, 400)
SAMPLE PERIOD 1s
DURATION 10 min //Gültigkeitsdauer der Anfrage

```

Jeder Sensorknoten kann als Datensenke auftreten. Zuerst wird ein initialer Interest mit einer relativ großen Intervallzeit (`sample period` – Zeitspanne zwischen zwei aufeinander folgenden Antwortnachrichten) im Netz verteilt. Dadurch will man herausstellen, welche Sensorknoten die Anfrage überhaupt positiv beantworten können.

Die Senke (Knoten A in der Abbildung 3.1) sendet den Interest an alle ihre Nachbarn (Abbildung 3.1a). Außerdem speichert sie diesen Interest und behält ihn im Speicher innerhalb von der im Duration-Attribut angegebenen Zeitspanne.

Jeder Knoten verwaltet einen Interestcache für die aktuell bestehenden Anfragen. Erhält ein Sensorknoten einen Interest, so richtet er für ihn einen Eintrag im Interestcache ein, falls er davor keinen Interest mit den identischen ON EVENT- und WHERE-Klauseln empfangen hat. Ein Eintrag besteht aus den im Interest selbst vorhandenen Attributen sowie einer Menge von Referenzen auf die Nachbarknoten, die schon gleiche Informationen angefragt haben. Diese Verweise bezeichnet man als *Gradienten*. Jeder Gradient beschreibt einen Nachbarknoten durch seine lokale ID und einen anwendungsspezifischen Wert. In diesem Beispiel ist es die Datenrate, mit welcher der Knoten Ergebnisse an den jeweiligen Nachbarknoten liefern soll. Sie lässt sich aus dem Wert des `sample period` herleiten. Ferner wird das Duration-Attribut festgehalten.

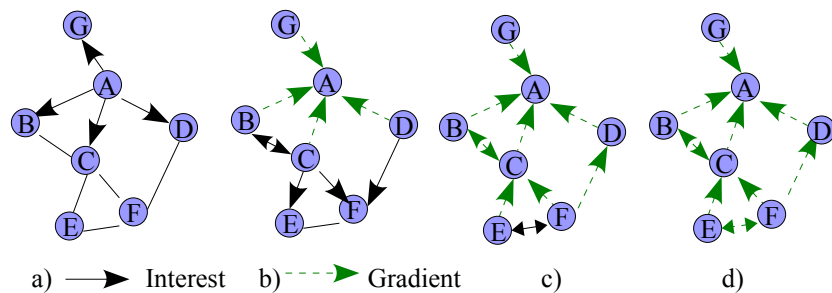


Abbildung 3.1: Interestverbreitung und Gradienten-Setup

Also erzeugen die Knoten B, C, D und G je einen Eintrag in seinen Interestcaches für den Interest aus dem Beispiel 3.1 und setzen jeweils einen Gradienten auf den Knoten A auf (siehe Abbildung 3.1b)).

Dann wird der Interest mit dem aktualisierten Duration-Feld an Nachbarknoten (evtl. abgesehen von dem Knoten A, von dem der Interest geliefert wurde) verschickt. Wenn eine geographische Region innerhalb eines Interests angegeben ist, wiederholen den Broadcast nur die Knoten, die sich näher an der Zielregion befinden als der Knoten, von dem sie die Nachricht erhalten haben. Dadurch wird das Fluten begrenzt.

Jetzt erfahren auch die Knoten E und F von der Anfrage (Abbildung 3.1c)). Der Knoten E richtet einen Gradienten für seinen Nachbar C ein. Der Knoten F muss Verweise auf zwei Knoten (C und D) speichern. Die Knoten B und C erhalten noch einmal den gleichen Interest. In jedem dieser Knoten existiert bereits ein Eintrag mit identischen type- und location-Attributen. In diesem Fall werden die sample period- und Duration-Felder der Gradienten aktualisiert. Zu bemerken ist, dass die Knoten B und C gegenseitig Gradienten aufbauen müssen (Abbildung 3.1d)), da sie aufgrund nicht vorhandenen globalen ID's nicht feststellen können, ob der Interest von der gleichen Senke kam und früher schon weitergeleitet wurde, oder ob es ein identischer Interest von einer anderen Datensenke ist. Genauso wie die Knoten E und F, die im nächsten Schritt auch aufeinander zeigende Gradienten aufsetzen. Eine Abhilfe dafür schafft der Datencache erst bei der Datenweiterleitung.

Da eine Nachricht im Sensornetz zum Beispiel aufgrund von Kollisionen verloren gehen kann, wiederholt die Senke den gleichen Interest periodisch während seiner Gültigkeitsdauer (*duration*). Somit erfahren auch Knoten, die davor unerreichbar waren oder neue Quellen, von der Anfrage.

3.2 Datenweiterleitung

Erfasst ein Sensorknoten Ereignisse (Knoten F in Abbildung 3.2), so durchsucht er seinen Interestcache und prüft, ob ein dazu passender Interest bereits gespeichert wurde. Falls ein solcher vorhanden ist, bestimmt der Knoten den Gradienten mit der höchsten Datenrate (d.h. mit der kleinsten Intervallzeit). Wenn Broadcastkommunikation benutzt wird, versendet der Sensorknoten seine Daten mit dieser Rate (Senderate). Werden die Daten direkt an einen Nachbarknoten geschickt, dann mit der Datenrate aus dem jeweiligen Gradienten.

Beispiel 3.2: Der Knoten F registriert ein Zebra an der Position (100, 350) und generiert eine Antwortnachricht:

```

type = four-legged animal
instance = zebra // welches Tier
location = (110, 350) // wo
intensity = 0.7 // Signalstärke
confidence=0.8 // Wahrscheinlichkeit, dass die Angaben stimmen
timestamp = 01:21:40

```

Also versendet der Knoten F jede Sekunde ein Datenpaket an die Knoten C, D und E (Abbildung 3.2a)). Jeder Zwischenknoten, bei dem ein Datenpaket ankommt, überprüft seinerseits, ob er Interesse an diesen Daten hat. Existiert kein passender Interest, werden die Daten einfach gelöscht. Bei den Knoten C, D und E ist ein passender Eintrag im Interestcache vorhanden, so speichern sie die empfangenen Daten im Datencache.

Nun müssen sie die Datenrate berechnen, mit der sie die Daten weiterleiten sollen. Dafür wird der stärkste Gradienten bestimmt sowie die Datenrate, mit welcher der jeweilige Knoten das Paket bekommen hat (Empfangsrate). Ist die Empfangsrate niedriger als die (bzw. gleich der) Senderate, so schickt der Knoten das Datenpaket mit der Empfangsrate weiter. Im Fall einer höheren Empfangsrate muss er diese auf die Senderate „runterrechnen“. Möchte z.B. ein Gradient Daten nur mit der halben Eingangsdatenrate seine Pakete erhalten, kann der Knoten lediglich jedes zweite Ereignis oder eine Interpolation aus zwei aufeinanderfolgenden Ereignissen an den Nachbar senden. Im Beispiel 3.1 sind alle Gradienten gleich stark. Deswegen senden die Knoten C, D und E je eine Antwortnachricht pro Sekunde (Abbildung 3.2b)).

Der Knoten C erhält die gleichen Daten, die von ihm bereits gesehen wurden und somit im Datencache gespeichert sind, noch einmal vom Knoten E, verwirft das Datenpaket aber, um Schleifen zu vermeiden.

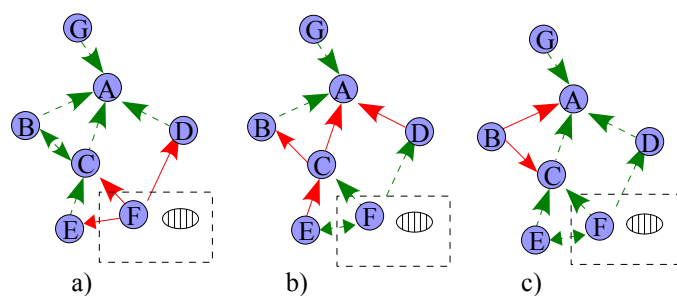


Abbildung 3.2: Datenweiterleitung

3.3 Verstärken/Abschwächen einzelnen Pfade

Nachdem die Senke ein Interest mit großem Intervall im Sensornetz verbreitet hat und Gradienten für redundante Pfade mit gleicher Datenrate aufgesetzt wurden, fließen die ersten Daten auf potentiell mehreren Pfaden in Richtung der Senke. Die Senke ist nun daran interessiert eine höhere Datenrate von dem Knoten zu erhalten, der das Ereignis am besten detektieren kann und zwar auf dem optimalen Weg. Deswegen richtet sie eine explizite Anfrage (*Reinforcement-Nachricht*) an den Nachbarknoten, den sie für fähig hält, ihren Anforderungen entsprechende Daten zu liefern. Eine Reinforcement-Nachricht ist ein identischer Interest für das Ereignis abgesehen von dem sample period, das jetzt einen kleineren Wert aufweist. Der Nachbarknoten muss dann den dazugehörigen Gradienten anpassen und mindestens einen Knoten auswählen, den er seinerseits verstärken will, es sei denn, er empfängt schon Daten mit geforderter Datenrate.

Dabei gibt es verschiedene Möglichkeiten, wie über die „guten“ Knoten lokal entschieden werden kann: beispielsweise kann man den Knoten C (Abbildung 3.3) bevorzugen, von dem man die neuen Daten zuerst erhält. Ferner kann der Knoten ausgewählt werden, der die meisten Ereignisse verschickt hat. Oder Reinforcement-Nachrichten können an mehrere Knoten verschickt werden, von denen man kürzlich neue Daten bekommen hat.

Indem jeder Zwischenknoten, der eine Reinforcement-Nachricht erhält, rekursiv gleiche Regeln anwendet, wird ein Pfad von der Quelle zur Senke verstärkt.

Der anfängliche Interest mit einer großen Intervallangabe verschwindet nach dem Ablauf der Duration-Zeitspanne. Ab dem Zeitpunkt fließen die Daten auf verstärkten Pfaden. Es kann allerdings sein, dass die Qualität eines Pfades abnimmt. In diesem Fall besteht die Möglichkeit, diesen Pfad abzuschwächen bzw. ganz abzuschalten. Dazu verschickt die Senke entweder einen neuen Interest mit der Aufforderung Daten mit niedrigerer Frequenz an sie zu leiten, oder die Reinforcement-Nachrichten werden mit einem time-out versehen, nach dessen Ablauf, falls keine neue Verstärkung angefordert wurde, die stärksten Gradienten entfernt werden.

Jeder Sensorknoten kann wiederum selbst lokal entscheiden, einen Nachbarn zu verstärken oder abzuschwächen, was die lokale Pfadreparatur erlaubt.

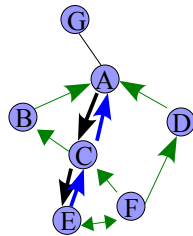


Abbildung 3.3: Reinforcement

3.4 Bewertung

Ein Vorteil dieses Verfahrens ist, dass es unter schlechten Bedingungen stabil bleibt. Auch wenn nur ein einziger Pfad zwischen Quelle und Senke weiterleitungsfähig bleibt, wird dieser gefunden und zwar recht schnell. Diese Eigenschaft verdankt Directed Diffusion vor allem der Lokalität des Algorithmus.

Falls kontinuierlich Daten von einem Typ benötigt werden, werden sie meistens auf einem Pfad geroutet. Trotzdem kann man mehrere alternative Routen aufrecht erhalten, um die Hindernisse auf dem primären Pfad umgehen zu können.

Ein weiterer Vorteil ist die gute Skalierbarkeit, da die Menge an Daten, die in Zwischenknoten gespeichert werden, nicht von der Anzahl der Knoten im Netz, sondern von der Anzahl der zum bestimmten Zeitpunkt gültigen Interests, abhängt.

Das Verfahren funktioniert auch bei mehreren Senken, vor allem, wenn Weiterleitungspfade (oder Teile davon) von mehreren Senken gemeinsam genutzt werden können, d.h. wenn sie Interesse an gleichen Daten haben.

Für einmalige Anfragen ist das Verfahren nicht geeignet, weil dann die fortlaufende Initialisierung des Netzes erforderlich ist, die Gradienten-Setup-Phase aber sehr energieintensiv ist. Die Kosten der Initialisierung sind der Hauptnachteil des Verfahrens.

Für Sensornetze mit einer hohen Ereignisfrequenz und einer vergleichsweise geringen Anzahl von gleichzeitig existierenden Anfragen im Netz erweist sich das Verfahren als besonders vorteilhaft, weil die Initialisierungskosten dann nicht mehr so stark ins Gewicht fallen.

Außerdem ist zu beachten, dass die Schleifenfreiheit der Pfade nur durch Speicherung von Daten garantiert wird. Speicherressourcen eines Sensorknotens sind aber sehr begrenzt (z.B. besitzt der Mica2 Mote 4 Kbyte RAM).

Ein weiterer Nachteil von Directed Diffusion ist, dass nach der Verstärkung eines Pfades die meisten Datenpakete durch die Knoten auf diesem Pfad zur Senke weitergeleitet werden. Dadurch verbrauchen diese Knoten ihre Energieressourcen eventuell sehr schnell und fallen dann aus.

Die Aufrechterhaltung mehrerer Pfade trägt einerseits der Robustheit bei, andererseits führt es zur Redundanz und wiederum zur schnelleren Energieerschöpfung in erster Linie der Sensorknoten nahe der Datensenke.

4 Broadcasting-Based query Scheme

Die Kommunikation ist der größte Kostenfaktor, der zur schnellen Energieerschöpfung der Sensorknoten um die Senke herum führt (siehe Kapitel 2.2). Das Hauptziel des Broadcasting-Based query Schema (BBS) ist es, diese Knoten zu entlasten, indem es versucht wird, die Kommunikation auf diesen

Knoten so gering wie möglich zu halten. Besonders gut gelingt es bei so genannten „Zonen-basierten“ Anfragen (siehe Definition 4.1) durch den Aufbau von lokalen Routingbäumen, die eine Anfrage so weit wie möglich verarbeiten und dann das Ergebnis in Richtung der Senke weiterleiten. Die Vorverarbeitung zielt darauf ab, die Menge an versendeten Daten in Richtung der Knoten um die Senke zu verringern. Eine solche Vorverarbeitung bietet sich vor allem bei Aggregationsanfragen an. Dabei hängt die Vorgehensweise davon ab, ob das Aggregat *holistisch* oder *nicht-holistisch* ist (siehe Definition 4.2).

4.1 Begriffseinführungen

Definition 4.1: Falls Antworten auf eine Anfrage aus Sensordaten von einer bestimmten Sub-Region (Anfragegebiet) eines Sensornetzes stammen, nennt man solche Anfragen „Zonen-basiert“ (*zone-based query*).

Definition 4.2: Eine Aggregatfunktion F ist *holistisch*, falls es keine konstante Grenze für den notwendigen Platz zur Beschreibung eines Sub-Aggregates gibt. Das heißt, dass es keine Konstante M existiert, so dass ein M -Tupel die Berechnung von F charakterisiert.

Also darf auf dem Weg von der Datenquelle bis zur Datensenke keine Aggregation in den Zwischenknoten erfolgen. Rohdaten müssen bei der Senke ankommen, um ein richtiges Ergebnis zu bekommen. *Median* und *häufigsterWert* sind Beispiele für holistische Anfragen.

Beispiel 4.1: Man betrachte die Funktion *häufigsterWert*(Temperatur) und die dazugehörigen gesammelten Werte: (10, 11, 9, 10, 8) kommen bei einem Knoten an; (9, 11, 12, 11, 7, 11) bei einem zweiten und (9, 10, 11, 10) beim dritten Knoten. Falls jeder Knoten die bei ihm eingegangenen Daten aggregieren würde, hätte man beim ersten Knoten „10“, beim anderen - „11“, beim letzten – auch „10“ als Ergebnis. Man nehme an, diese drei Knoten haben einen gemeinsamen Vaterknoten und leiten ihre Ergebnisse an ihn weiter. Dieser hätte dann „10“ als den häufigsten Wert berechnet. Die korrekte Antwort ist aber „11“. Das Problem nennt man auch das „*Sub-Aggregations Problem*“.

Alle Anfragen, bei denen die In-Netzwerk-Aggregation möglich ist, werden als *nicht-holistisch* bezeichnet. Die am meisten verbreiteten nicht-holistischen Funktionen sind *Max*, *Min*, *Avg*, *Sum*.

4.2 Annahmen

Es wird angenommen, dass alle Sensorknoten eindeutig identifiziert sind. Da BBS geographisches Routing verwendet, muss jeder Knoten seine eigene Position, die der Datensenke, sowie die ID's und die Position seiner Nachbarn, d.h. der Sensorknoten mit denen er direkt kommunizieren kann, kennen. Diese Positionsinformationen können durch lokale Berechnungen oder Nachrichtenaustausch mit stationären Geräten gewonnen werden (Triangulation). Die Ausstattung einiger oder mehreren Knoten mit Positionssystemen ist auch denkbar.

In dem einfachen Netzwerkmodell, das zur Beschreibung von BBS benutzt wird, existiert eine einzige stationäre Datensenke mit unbegrenztem Energievorrat. Die Sensorknoten sind in einem festen, rechteckigen Gebiet zufällig verteilt. Jedes Anfragegebiet ist durch ein achsenparalleles Rechteck eingegrenzt. Bidirektionale Kommunikation wird außerdem vorausgesetzt.

4.3 Algorithmus

BBS besteht aus fünf Basisschritten:

1. **Anfrageverbreitung:** Dies geschieht entweder mit Hilfe existierender Flooding-Algorithmen oder die Datensenke sendet eine Broadcast-Nachricht an alle Sensorknoten im Netzwerk, falls sie über eine entsprechende Ausstattung verfügt.
2. **Konstruktion von lokalen Routingbäumen:** In einem Anfragegebiet werden unter Umständen mehrere lokale Routingbäume konstruiert.
3. **Propagieren von Daten innerhalb lokaler Routingbäume:** Nach Aufbau von Routingbäumen können Daten innerhalb dieser Bäume propagiert werden. Bei nicht-holistischen Anfragen aggregiert man die Daten in den Zwischenknoten. Bei holistischen Anfragen werden lediglich Rohdaten der Kinderknoten in ein Paket zusammengefasst.
4. **Datenweiterleitung vom lokalen Wurzelknoten zur Datensenke:** Nachdem die Daten bei der lokalen Wurzel angekommen sind, leitet sie sie an die Datensenke weiter.
5. **Analyse von aktuellen lokalen Routingbäumen:** Als letztes muss die Datensenke nach Erhalt der Daten prüfen, ob es mehrere lokale Wurzelknoten innerhalb von einem Anfragegebiet gibt. Bei holistischen Anfragen führt das Vorhandensein mehrerer Wurzelknoten zu einem gefälschten Endergebnis, weil Datentupeln dann mehrfach in die Berechnung eingehen.

Optional können lokale Routingbäume zu einem Baum kombiniert werden. Unter dem Namen „Route Redirection“, bietet BBS die Möglichkeit den Energieverbrauch von Sensorknoten nahe der Senke in bestimmten Sonderfällen

noch mehr zu reduzieren. Die Abbildung 4.1 bietet einen Überblick über den Algorithmusablauf.

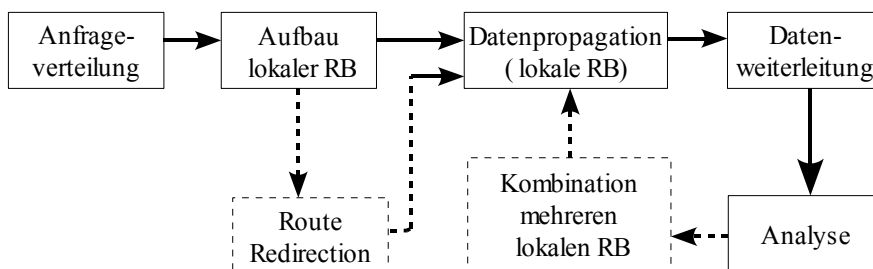


Abbildung 4.1: Ablauf von BBS (- - - optionale Ausführungsschritte) [1]

4.3.1 Konstruktion lokaler Routingbäume

Die Haupteigenschaft von BBS ist die Konstruktion lokaler Routingbäume innerhalb des jeweiligen Anfragegebiets. Dies hängt nicht vom Anfrageverbreitungsverfahren ab. Es werden keine Pfadinformationen von den Sensorknoten zur Datensenke benötigt. Nur Sensorknoten, die sich im Anfragegebiet befinden und selbst geeignete physikalische Größen messen, schließen sich dem Routingbaum an.

Berechnung des „Root Reference Point“

Als Erstes berechnen alle Sensorknoten, die an der Konstruktion eines Routingbaumes beteiligt sind, einen eindeutigen geographischen Punkt innerhalb des Anfragegebiets, den so genannten „*root reference point*“. Falls das Anfragegebiet die Datensenke enthält, wird die Position der Senke als „*root reference point*“ gewählt. Somit wird die Senke zur „lokalen“ Wurzel. Sonst wird abhängig vom Anfragetyp vorgegangen.

holistische Anfragen:

1. Fall: Das Anfragegebiet liegt komplett innerhalb des Sensornetzes. Dann wird der Mittelpunkt des Anfragegebiets zum „*root reference point*“.
2. Fall: Das Anfragegebiet umschließt einen Bereich, der sich über die Grenzen des Netzwerkes erstreckt. In diesem Fall wird der Mittelpunkt vom Teil des Anfragegebiets, das noch Sensorknoten enthält, als dieser eindeutige geographische Punkt identifiziert. (R_1 in Abb. 4.2)

nicht-holistische Anfragen:

„Root reference point“ ist der Schnittpunkt der Geraden, die die Datensenke mit dem Mittelpunkt des Anfragegebiets verbindet, mit einer der Kanten des Anfragegebiets (R_2 in Abb. 4.2). Dies impliziert, wie wir weiter unten sehen werden, dass die am nächsten zur Datensenke liegenden Sensorknoten mit höherer Wahrscheinlichkeit lokale Wurzeln werden. Somit reduziert man sowohl Weiterleitungskosten zur Senke als auch die Verzögerung.

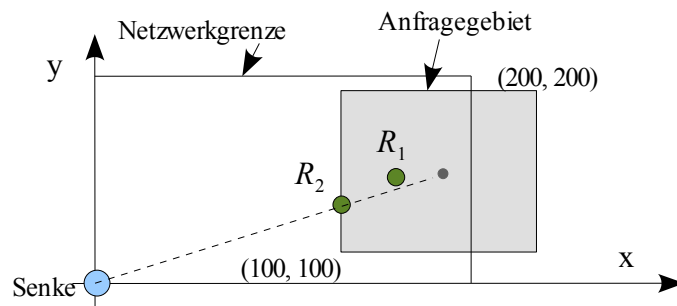


Abbildung 4.2: Berechnung des "root reference point"

Baumaufbau

Jeder Sensorknoten weist sich eine eindeutige Prioritätsstufe zu. Sie ist desto höher je kleiner der euklidische Abstand vom Knoten zum „root reference point“ ist. Im Fall von zwei gleich weit vom „root reference point“ entfernten Nachbarknoten, hat der Knoten mit der größeren ID eine höhere Priorität. Hat ein Sensorknoten die höchste Priorität unter allen seinen Nachbarn, so wird er zum lokalen Wurzelknoten.

Die eben beschriebenen Berechnungen können lokal in jedem Knoten durchgeführt werden, da laut Voraussetzung alle Sensorknoten über ihre eigenen Positionsinformationen sowie die der Senke und die der Nachbarknoten als auch über die ID's der Nachbarn verfügen. Also entscheidet ein Sensorknoten selbst,

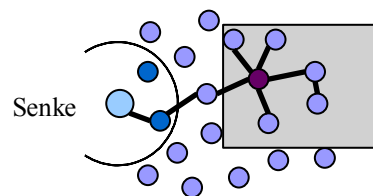


Abbildung 4.3: Aufbau eines lokalen Routingbaumes

ob er die lokale Wurzel ist. Trifft es nicht zu, wählt er den Nachbarknoten mit der höchsten Priorität als seinen Vaterknoten aus.

4.3.2 Datenweiterleitung

Nachdem ein lokaler Routingbaum konstruiert ist, fließen Daten entlang dieses Baumes zur lokalen Wurzel.

Wurde eine Anfrage durch Fluten verbreitet, so ist der Pfad von der lokalen Wurzel zur Datensenke bereits beim Fluten etabliert worden. Ist dagegen die Datensenke in der Lage mit ihrem Broadcast alle Sensorknoten im Netz zu erreichen, so muss jeder lokale Wurzelknoten einen Pfad für Datenweiterleitung finden. BBS schlägt vor, geographisches Routing hierfür anzuwenden. In Sensornetzen mit dicht verteilten stationären Sensorknoten benutzt man den Greedy-Ansatz [6]. D.h. eine lokale Wurzel verschickt die Daten an denjenigen Nachbarknoten, der sich geographisch am nächsten zur Datensenke befindet. Dieser Knoten leitet die Daten dementsprechend weiter. Der Greedy-Ansatz kann allerdings in Sackgassen führen. Deswegen werden für Netzwerke mit einer geringen Knotendichte die Verfahren wie GFG [7] oder GPSR [8] empfohlen.

4.3.3 Sonderfälle

Unter Umständen kann es zur Konstruktion mehrerer lokalen Routingbäume in einem Anfragegebiet kommen, weil

1. ein Anfragegebiet nicht direkt benachbarte Sensorknoten umfassen kann, die nichts von einander wissen. Deswegen meinen mehrere Sensorknoten, sie seien die Wurzel.
2. oder das Anfragegebiet einer nicht-holistischen Anfrage ein-hop-Nachbarn der Senke enthält, die Senke selbst aber außerhalb vom Anfragegebiet liegt. Dann werden alle direkten Nachbarknoten der Senke ausnahmsweise zu lokalen Wurzeln, um unnötigen Datentransfer zwischen diesen Knoten zu vermeiden.

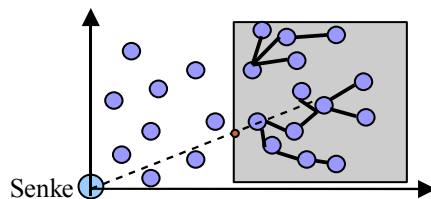


Abbildung 4.4: Aufbau mehrerer lokalen Routingbäume in einem Anfragegebiet

Das Vorhandensein mehrerer Wurzelknoten heißt im ersten Fall für nicht-holistische Anfragen, dass es mehrere Pfade von einem Anfragegebiet zur Senke existieren, welche durch mehrere one-hop-Nachbar der Senke führen und somit der schnelleren Energieerschöpfung von denen beitragen. Um dies zu erkennen und die zur gleichen Anfrage gehörenden Daten, die in einem Zwischenknoten zusammenlaufen, gegebenenfalls zu aggregieren, enthalten die Antwort-Nachrichten die Anfrage-ID und eine Sequenznummer sowie ein Sub-Feld REP#. Beim Generieren einer Antwortnachricht setzt die lokale Wurzel REP# auf „1“. Nach der Ankunft von Daten mit der identischen Anfrage-ID und Sequenznummer in einem Zwischenknoten werden die REP#-Werte aufsummiert und mit den aggregierten Daten weitergeschickt. Es repräsentiert somit die Anzahl von lokalen Wurzelknoten, die ihre Daten durch diesen Knoten weiterleiten.

Es ist natürlich möglich, dass die Nachrichten verschiedener Wurzelknoten auf disjunkten Pfaden zur Senke geroutet werden. Aber spätestens die Senke stellt die Existenz mehrerer lokalen Routingbäume fest. Bei holistische Anfragen, bei denen die Existenz mehrerer lokalen Wurzeln zum Sub-Aggregations-Problem führen kann, kann die Datensenke die Wurzelknoten auffordern Rohdaten an sie zu schicken, um ein richtiges Ergebnis zu berechnen. Eine andere Möglichkeit, die von den Autoren des BBS vorgeschlagen wird, ist es, lokale Routingbäume zu einem zu kombinieren. Außerdem wird auch versucht mit der Wahl vom „root reference point“ in der Mitte des Anfragegebiets die Wahrscheinlichkeit, dass mehrere lokale Bäume erzeugt werden, zu minimieren.

In Bezug auf den zweiten Sonderfall wird im nächsten Abschnitt unter der Bezeichnung „Route Redirection“ eine Möglichkeit vorgestellt die Empfangskosten der direkten Nachbarknoten der Datensenke weiter zu senken.

Route Redirection

Enthält das Anfragegebiet einer nicht-holistischen Anfrage one-hop-Nachbarn der Senke, die Senke selbst aber nicht, so werden sie zu lokalen Wurzeln gewählt. Route Redirection (RR) hat das Ziel die Anzahl der von den direkten Nachbarn der Senke empfangenen Nachrichten zu reduzieren. Vorausgesetzt, Empfangskosten sind vergleichbar hoch.

Man betrachte das Sensornetz in Abb. 4.3a), wo im Anfragegebiet drei one-hop-Nachbarn der Senke liegen (S_1 , S_2 , S_3). Die Anzahl der Datenpakete, die diese Knoten zu versenden haben, kann nicht reduziert werden, da sie auf jeden Fall ihre eigenen Messergebnisse übertragen müssen. Die Empfangskosten können dagegen durch eine gezielte „Umleitung“ von ankommenden Nachrichten an die zwei-hop-Nachbarn der Senke verringert werden. In der Abb. 4.3b) leiten die zwei-hop-Nachbarn der Senke ihre Daten an den Knoten S_4 weiter, der sie dann aggregiert und an S_2 schickt. Also empfängt S_2 nur eine Nachricht anstatt von vier in Abb. 4.3a). Ähnlich wie der root reference point beim Aufbau von

Routingbäumen benutzt wird, berechnet man hier den so genannten „redirection reference point“ um neue Elternknoten zu bestimmen. Dieser Punkt soll so gewählt werden, dass die Zyklen in den konstruierten Routingbäumen vermieden werden und die Größe der Bäume reduziert wird.

Die Umleitung der Daten an drei (oder noch höhere)-hop-Nachbarn der Senke würde zu größeren Routingbäumen führen und somit zu längeren Verzögerungen.

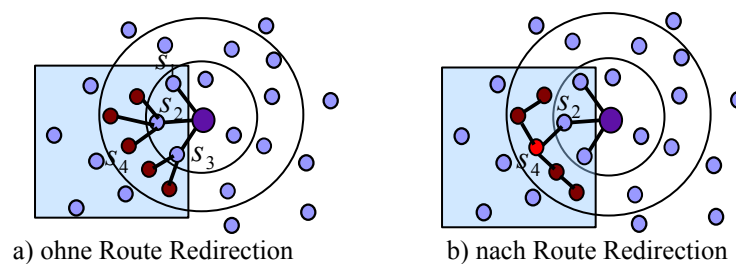


Abbildung 4.3: Routingbaum

4.4 Bewertung

Ein Vorteil von BBS ist seine Skalierbarkeit, da ein Sensorknoten nur seine Nachbarn und die Senke kennen muss.

Bei diesem Verfahren werden am meisten lokale Wurzelknoten belastet. Dadurch erschöpfen sie ihre Energie schnell und fallen eventuell aus. Bei der Wahl von einer lokalen Wurzel spielt nur der Abstand zum root reference point eine Rolle. Der aktuell vorhandene Energievorrat wird in die Entscheidung nicht miteinbezogen.

BBS ist besser für große und dichte Sensornetzwerke geeignet. Es reduziert den Energieverbrauch von Sensorknoten nahe der Senke im Vergleich zu Verfahren, die einen Routingbaum mit der Senke in der Wurzel konstruieren (z.B. TAG [9]), vor allem wenn das Anfragegebiet relativ klein ist. Dann ist die Wahrscheinlichkeit, dass Daten in einem einzigen Paket (pro Intervall) auf einem Pfad zur Senke geroutet werden, ziemlich hoch. Wenn ein Anfragegebiet groß ist, so bietet BBS weniger Vorteile gegenüber anderen Verfahren. Die Simulationen in [1] belegen das auch. Man betrachte die Abbildungen 4.5 und 4.6 mit Simulationsergebnissen für nicht-holistische Anfragen, wobei g die durchschnittliche Anzahl der Nachbarn eines Sensorknotens ist, tx - Kosten für das Senden einer Nachricht, rs - Empfangskosten, rs - der Senderradius. Man erkennt, dass bei BBS gegenüber TAG bis zu 50% mehr Anfragen während der Lebensdauer des Netzwerkes abgearbeitet werden. Allerdings hängt diese Zahl von der Dichte und der Größe des Netzwerkes.

In Szenarien mit nicht vernachlässigbaren Empfangskosten bringt der Route Redirection-Ansatz (BBS-RR) weitere Steigerung der Anzahl von Anfragen, die während der Lebensdauer vom Netzwerk abgearbeitet wurden.

Für holistische Anfragen lohnt es sich laut [1] für Netzwerke mit geringer Knotendichte, trotz der zusätzlichen Kosten, die Kombination der lokalen Routingbäume durchzuführen.

Die Simulationsergebnisse sind aber nur bedingt aussagekräftig, weil unter Anderem ein einfaches Netzwerkmodell mit quadratischen Anfragegebieten vorausgesetzt wurde und außerdem, Simulationen nur für zwei Anfragetypen, nämlich Max und Median, durchgeführt wurden.

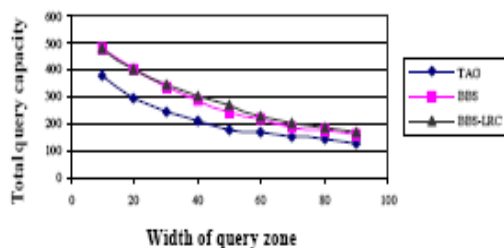


Abbildung 4.5: Query capacity

($g=10$, $tx=10$, $rx=0$, $rs=10$)

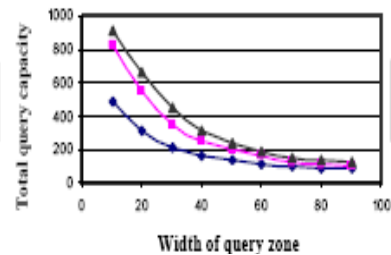


Abbildung 4.6: Query capacity

($g=20$, $tx=5$, $rx=5$, $rs=5$)

5 Zusammenfassung und Ausblick

Über das Routing in Sensornetzen wird viel geforscht, weil das Senden und das Empfangen von Nachrichten den Energieverbrauch von Sensorknoten meist dominieren. Es existieren inzwischen sehr viele Verfahren, die auf Energieeffizienz, Robustheit und Skalierbarkeit ausgelegt sind. Die meisten Protokolle verfolgen einen dezentralen Ansatz und sind für bestimmte Anwendungsszenarien geeignet.

In dieser Arbeit wurden zwei Routingverfahren vorgestellt. Directed Diffusion ist ein gradientenbasiertes Verfahren, das datenzentrische Adressierung verwendet. Das Ziel von Directed Diffusion ist, effiziente Kommunikationspfade zwischen einer oder mehreren Quellen und Senken zu etablieren. Dabei soll anwendungsspezifische In-Netzwerkbearbeitung von Anfragen ermöglicht werden. Es ist robust und skalierbar. Weiterhin gibt es Multipath-Routingalgorithmen, die auf Directed Diffusion basieren [10]. Allerdings wird die

pro Sensorknoten aufgebrauchte Energie als Leistungsmetrik benutzt anstatt auf die gleichmäßige Energiebelastung der Sensorknoten hin zu optimieren .

Broadcasting-Based query Schema zeichnet sich durch die Konstruktion von Routingbäumen mit der Wurzel in dem jeweiligen Anfragegebiet aus. Dadurch wird versucht den Energieverbrauch von one-hop-Nachbarn der Senke zu reduzieren und somit die Anzahl der abgearbeiteten Anfragen zu erhöhen. Wobei es ausreichende Simulationen, die die Effizienz von dem Verfahren belegen, noch fehlen.

Die beiden Verfahren sind vor allem für langlaufende Aggregationsanfragen geeignet. In diesem Fall können Flooding-Kosten für initiale Anfragenverteilung über die Zeit ausgeglichen werden. Zur Bearbeitung von Schappschussanfragen sei man auf andere Verfahren wie zum Beispiel [12] verwiesen.

Zusammenfassend lässt sich sagen, dass es noch kein Verfahren für alle Anwendungsszenarien gibt. Es wird versucht durch die Einbindung der Anwendungsschicht in die Routingentscheidungen den Energieverbrauch zu minimieren. Neben der eigentlichen Anwendung muss auch die Charakteristik des Netzwerkverkehrs analysiert werden und auf dessen Grundlage müssen Designentscheidungen für das Routing Protokoll getroffen werden.

Literatur

1. Jie Lian, Lei Chen, Kshirasagar Naik, M. Tamer Özsu, and Gordon B. Agnew. BBS: An Energie Efficient Localized Routing Scheme for Query Processing in Wireless Sensor Networks. Technical Report CS2005-15, University of Waterloo, 2005
2. Intanagonwivat, Govindan, Estrin, „Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks“, Proc. ACM MobiCom '00, Boston, August 2000
3. S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The Design of an Acquisitional Query Processor for Sensor Networks. In *Proc. of ACM SIGMOD*, 2003.
4. G. Pottie and W. Kaiser. Wireless Sensor Networks. *Communication of the ACM*, 43(5): 51-58, May 2000.
5. J. Lian, K. Naik, and G. Agnew. Data Capacity Improvement of Wireless Sensor Networks Using Non-Uniform Sensor Distribution, to appear in *International Journal of Distributed Sensor Networks*, 2005.
6. G.G. Finn. Routing and Addressing Problems in Large Metropolitan-scale Internetworks, *ISI Research Report ISU/RR-87-180*, 1987
7. P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with

- Guaranteed Delivery in Ad Hoc Wireless Networks. Proc. of ACM Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, 1999, pp. 48-55.
8. B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. *Proc. of ACM/IEEE MOBICOM 2000*, August 2000. pp. 243-254.
 9. S. Madden, M.J. Franklin, J. M. Hellerstein, and w. Hong. TAG: A Tiny Agregstion Service for Ad-Hoc Sensor Networks. IN *OSDI (to appear)*, 2002
 10. D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, „Highly-resilient, energy-efficient multipath routing in wireless sensor networks“, ACM SIGMOBILE Mobile Computing and Communications Review, vol. 5, no. 4, pp. 1125, 2001.
 11. J. N. Al-Karaki and A. E. Kamal. Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications*, 2004.
 12. N. Sadagopan et al., The ACQUIRE mechanism for e±cient querying in sensor networks, in the Proceedings of the First International Workshop on Sensor Network Protocol and Applications, Anchorage, Alaska, May 2003.
 13. Y. Yao and J. Gehrke, Query Processing for Sensor Net. In *Proc. of CIDR*, 2003.
 14. Crossbow, *Mica2 Wireless Measurement System*, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf
 15. Jason L. Hill, David E. Culler, ”Mica: a wireless platform for deeply embedded networks“, *IEEE Micro*, Novermer-December 2002
 16. Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, John Anderson, „*Wireless Sensor Networks for Habitat Monitoring*“, WSNA '02, 2002
 17. <http://people.inf.ethz.ch/roemer/papers/visionen0206.pdf>
 18. I. F. Akyildiz *et al.*, “Wireless Sensor Networks: A Survey,” *Elsevier Sci. B. V. Comp. Networks*, vol. 38, no. 4, Mar. 2002, pp. 393–422.