Datenzentrisches Routing und Directed Diffusion in drahtlosen Sensornetzen

Sven Haidan

Seminar „Algorithmen für Sensornetze“ WS2004/05
Universität Karlsruhe (TH)

Zusammenfassung

Aufgrund technologischer Fortschritte in der Prozessortechnik, Speichertechnologie, sowie Funk- und Energietechnik ist es möglich geworden Netze aus kleinen und günstigen Sensorknoten herzustellen, welche dann zur verteilten Aufnahme von Phänomenen eingesetzt werden können. In dieser Seminararbeit wird das Kommunikationsparadigma „Directed Diffusion“ vorgestellt. Dazu wird zunächst ein Überblick über die Gegebenheiten in Sensornetzen gegeben und dann die Datenaggregation und das datenzentrisches Routing vorgestellt, welche dem „Directed Diffusion“ zugrunde liegen. Mit „Directed Diffusion“ sollen vor allem Energieeinsparungen errungen werden, da Energieknappheit in Sensornetzen ein großes Problem darstellt. Neben einer theoretischen Betrachtung werden auch noch Simulationsergebnisse zu Directed Diffusion vorgestellt.

Inhaltsverzeichnis

1 Einleitung

Sensornetze leiten einen Paradigmenwechseln im Einsatz von Computern ein. Statt wie bisher Daten manuell zu erheben, diese dann in ein Rechensystem einzugeben und zu verarbeiten, werden die Daten in Sensornetzen direkt durch das System erhoben. Dies geschieht an verschiedenen Orten gleichzeitig, so dass es sich bei einem Sensornetz um ein verteiltes System handelt. Durch diese Tatsache und die Möglichkeit, die Daten online, automatisch und in Realzeit zu erfassen ergibt sich eine ganz neue Qualität der erhobenen Daten. Um die durch die Sensoren gemessenen Werte verarbeiten zu können muss zuerst eine Signalvorverarbeitung erfolgen. Die Herausforderung an das Sensornetz besteht nun darin die Daten im Netz und somit dezentral und verteilt zu verarbeiten. Dies ist notwendig, da die Sensorknoten nur eine begrenzte Energie zur Verfügung haben, andererseits aber eine lange Lebensdauer des Sensornetzes erwünscht ist.

Sensornetze bestehen aus vielen Sensorknoten, welche nur lokal direkt miteinander kommunizieren können. Ein Sensorknoten besteht aus den vier Hauptkomponenten Verarbeitungseinheit (CPU), Speicher, Kommunikationseinrichtung und einem oder mehreren Sensoren. Alle diese Einheiten unterliegen stärkeren Ressourceneinschränkungen als dies bei herkömmlichen Netzknoten der Fall ist. Das Hauptproblem hierbei ist die stark beschränkte Energie, die zur Verfügung steht. Die Knoten werden meist zu Beginn des Netzbetriebes mit einer begrenzten Energie versorgt, z.B. durch eine Batterie, und haben wenn überhaupt nur geringe Möglichkeiten zur eigenen Energiegewinnung. Die Energiemenge wird desweiteren durch die gewünschte geringe Größe eines Sensorknotens wesentlich eingeschränkt. Ein Sensorknoten erfüllt in wesentlichen drei Aufgaben:

1. Auslesen der Sensordaten
2. Vorverarbeitung der Daten
3. Weiterleitung der Daten

In dieser Ausarbeitung gehe ich auf die ersten beiden Punkte nur insofern ein, als dass ich die gewünschte Darstellung der Daten nach dem Auslesen und der Vorverarbeitung angebe. Der Schwerpunkt liegt also auf dem dritten Schritt der Weiterleitung der Daten. Erst dieser Schritt und die Verarbeitung der Daten verschiedener Sensorknoten ermöglichen es Eigenschaften eines Umweltphänomens wie z.B. die räumliche Lage und Orientierung, Bewegungsrichtung- und Geschwindigkeit, sowie Form und Größe von Objekten zu bestimmen.
Sensornetze haben im allgemeinen eine dynamische Struktur und es kann keine Strukturannahme getroffen werden, da die Sensorknoten sich beliebig platzieren oder beliebig platziert werden können.

Die Anwendungsgebiete von Sensornetzen sind vielfältig. Ihr Einsatz ist vor allem dann sinnvoll, wenn ein Phänomen verteilt aufgenommen werden soll. Dies kann z.B. der Fall sein, wenn die Position des Phänomens nicht bekannt ist (Gebietsüberwachung), es die Position im Raum ändert (Objektverfolgung) oder aber die Auswirkungen des Phänomens auf unterschiedliche Teile untersucht werden sollen (z.B. Auswirkungen seismischer Aktivitäten auf Gebäude oder Brücken). An einer Gebietsüberwachung ist z.B. das Militär interessiert, das sicherheitskritische Anlagen oder unzugängliche Gebiete überwachen möchte. Und Biologen möchten das Verhalten von Tieren

untersuchen ohne in deren Lebensraum einzugreifen (unauffällige Präsenz). Die Anwendung bietet sich auch noch an bei entfernten unzugänglichen Regionen, giftigen, gesundheitsschädlichen oder schlecht zugänglichen Orten.

Sensornetze ähneln in gewisser Weise mobilen Ad-hoc-Netzen (MANETs). Wie bei MANETs so kommunizieren die Knoten auch in Sensornetzen meist über Funk und ihre Nachbarknoten sind nicht fest festgelegt und können sich über die Zeit ändern. Ebenso findet in beiden Netzen eine Multihopkommunikation statt.
Daneben gibt es aber auch einige Unterschiede: Während in MANETs voneinander unabhängige End-to-End-Kommunikation stattfinden, findet man in Sensornetzen dagegen Many-to-one-Datenflüsse von viele Quellen zu einer Senke (oder zumindest zu wesentlich weniger Senken). Da die Quellen in einem gewissen Gebiet dieselben Phänomene beobachten treten in diesen Daten Redundanzen auf. Obwohl sich die Struktur des Netzes auch bei den Sensornetzen über die Zeit ändern kann geht man davon aus, dass dies nur über äusserst lange Zeiträume geschieht, so dass man bei der Datenerfassung davon ausgehen kann, dass die Sensorknoten sich nicht bewegen. Stattdessen kann aber das beobachtete Phänomen über die Zeit seine Position im Raum ändern. Desweiteren sind die Energiebeschränkungen bei Sensornetzen wesentlich stärker als in MANETs, das in letzteren die Knoten in gewissen Abständen wieder aufgeladen werden können, wie z.B. im Falle eines Laptops. Bei Sensornetzen geht man jedoch davon aus, dass die Knoten mit der Energie, die sie zur Zeit des Netzaufbaus haben für die gesamte Lebensdauer des Netzes auskommen müssen.

Der Rest dieser Seminarsarbeit ist wie folgt aufgebaut:
Zuerst werde ich das datenzentrische Routing vorstellen und dabei auf die verwandten Themen der Datenaggregation und der benannten Daten eingehen. Das datenzentrische Routing wird dem adresszentrischem Routing gegenübergestellt und im Anschluss daran wird eine Simulation vorgestellt, welche die theoretisch gefundenen Ergebnisse überprüft.
Danach werde ich das Kommunikationsparadigma „Directed Diffusion“ vorstellen, welches stark auf dem datenzentrische Routing basiert. Hier wird kein fertiges implementiertes Verfahren vorgeführt, sondern es werden die dem „Directed Diffusion“ zugrundeliegende Struktur vorgestellt und mögliche Designentscheidungen aufgezeigt. Auch zum „Directed Diffusion“ werde ich Simulationsergebnisse vorstellen.

2 Datenzentrisches Routing und Datenaggregation

Da Sensornetze starken Energiebeschränkungen unterliegen muss ein Weg gefunden werden, wie man Energie einsparen kann. Im wesentlichen helfen hier die folgenden drei Beobachtungen um eine Lösung für das Problem zu finden:

1. In Sensornetzen gibt es viele Many-to-one-Datenflüsse (umgekehrtes Multicasting).

2. Aufgrund der Tatsache, dass mehrere Knoten dasselbe Phänomen beobachten gibt es viel Redundanz in den erfassten Daten, welche in benachbarten Knoten vorliegen.

3. Die Energiekosten sind wesentlich höher für die Übertragung von Daten von einem Knoten zu einem anderen, als für die Verarbeitung der Daten in einem Knoten.

Diese Beobachtungen legen es nahe die Daten schon in der Nähe ihres Beobachtungsortes zu verarbeiten. Diese Form der Datenverarbeitung im Netz bei der mehrere Daten von verschiedenen Eingangsknoten zu einem Datum zusammengefasst werden nennt man Datenaggregation. Hierdurch werden Übertragungen und somit Energie gespart. Es gibt verschiedene Möglichkeiten die Datenaggregation zu realisieren. Im einfachsten Fall werden einfach Duplikate unterdrückt und in komplexeren Aggregationsfunktionen werden aus verschiedenen Eingangsdaten Ausgangsdaten berechnet indem Redundanz eliminiert und Information aus den Eingangsdaten kombiniert wird.
Da bei den Anfragen, die an das Netz gestellt werden, nicht wichtig ist wer die Daten bereitstellt, geht man von einem adresszentrischen Routing mit Ende-zu-Ende-Verbindungen zu einem datenzentrischen Routing über. Dabei werden Routingentscheidungen aufgrund der vorliegenden Daten und ohne Angabe einer Netzadresse getroffen.

2.1 Adresszentrisches vs. datenzentrisches Routing

In diesem Abschnitt wird das adresszentrische Routing (address-centric routing ACR) mit dem datenzentrischen Routing (data-centric routing DCR) verglichen. Wir untersuchen folgendes Szenario: Eine Senke schickt eine Anfrage ins Netz und bekommt von mehreren Quellen passende Sensordaten zurückgeliefert. Dabei unterscheiden sich adresszentrisches und datenzentrisches Routing folgendermaßen:
Als **adresszentrisches Routing** bezeichnen wir das Routing, bei dem die Knoten unabhängig voneinander eine Ende-zu-Ende-Verbindung aufbauen, wobei sie die kürzeste Verbindung nehmen. Die Knoten haben dabei eine eindeutige Netzadresse. Beim **datenzentrischen Routing** hingegen senden die Quellknoten die Daten in Richtung Senke, wobei die Zwischenknoten jedoch die Daten auf dem Weg untersuchen und eine Datenaggregation durchführen. Die Knoten werden hier nur implizit angesprochen und der Datentransfer basiert auf einer Hop-zu-Hop-Basis. Dabei können, wie aus Abbildung 1 ersichtlich, beim datenzentrischen Routing Übertragungen eingespart werden. Interessant sind hierbei nur die Fälle, in denen die Quellen nichtdeterministisch Daten mit einer gewissen Redundanz senden. In dem Fall, dass die Quellen nur nichtredundante Daten senden kann keine Aggregation stattfinden und ACR und DCR sind gleich gut. Im Fall von totaler Redundanz kann die Senke alle bis auf eine Quelle auffordern nicht mehr zu senden und es ergeben sich weniger Übertragungen als im ACR. Ein Kernbestandteil des datenzentrischen Routings ist die Datenaggregation. Wesentliche Faktoren, die diese und damit auch das adresszentrische Routing beeinflussen, sind die Anzahl der Datenquellen, deren Platzierung und die Topologie des Kommunikationsnetzes. Deshalb werden zur Untersuchung adresszentrischen Routings zwei Quellenplatzierungsmodelle verwendet mit denen wir die Leistung des Routing beurteilen möchten.

Es werden das Event-Radius-Modell (ER) und das Radom-Sources-Modell (RS) verwendet. Dabei wird bei beiden Modellen von einem Einheitsquadrat ausgegangen, auf welches jeweils n Knoten zufällig verteilt werden. Jeder Knoten kann mit all jenen Knoten direkt kommunizieren, die sich in seinem Kommunikationsradius R befinden.

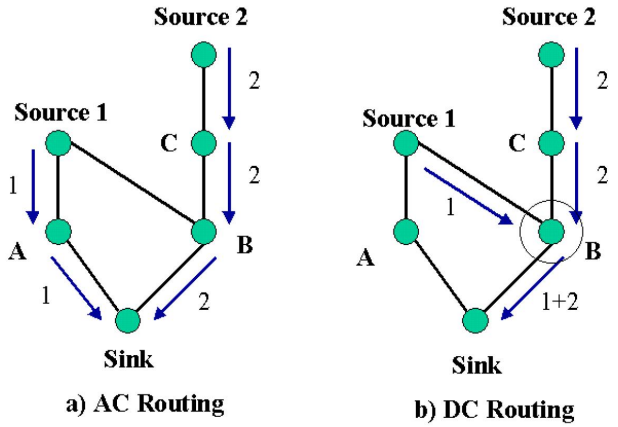

Abbildung 1: Vergleich ACR mit DCR - An Knoten B kann bei DCR eine Übertragung eingespart werden, da hier die Daten aggregiert und zusammen zu Knoten A übertragen werden. [Quelle: [3]]

Die Modelle unterscheiden sich darin, wie die Quellen auf das Einheitsquadrat verteilt werden.

Beim **Event-Radius-Modell** wird ein Punkt im Einheitsquadrat als Ereignisauslöser festgelegt und alle Knoten im Umkreis vom Radius S (sensing range) agieren als Quellen. Es gibt also ungefähr $\pi * S^2 * n$ Quellen.

Beim **Radom-Sources-Modell** hingegen sind k zufällig ausgewählte Knoten die Quellen und können also über das EInheitsquadrat verstreut sein.

Hier werden drei Maße zur Leistungsmessung der Datenaggreagtion vorgestellt.

- **Energieeinsparungen** entstehen durch die durch Datenaggregation hervorgerufene geringere Anzahl an Übertragungen zwischen den Knoten.
- **Verzögerungen** entstehen dadurch, dass Daten um aggregiert zu werden an den Knoten zurückgehalten werden müssen um mit anderen kombiniert werden zu können. Dadurch erreichen die Daten die Senke erst später.
- **Robustheit** gegenüber Dynamik im beobachteten Phänomen:
  Hierunter versteht man das Verhalten des Netzes falls z.B. neue Quellen beginnen Daten zu versenden oder das Phänomen sich räumlich verschiebt, sich also neue Quellen an- und alte abmelden.

Optimale und suboptimale Aggregation Unter Datenaggregation versteht man die Kombination von mehreren Eingangsdaten wobei eine Datenkompression erreicht werden soll. Die einfachste Form der Datenaggregation ist die Duplikatsunterdrückung, bei der gleiche mehrmals auftretende Eingangsdaten nur einmal weitergeleitet werden. Es können auch Funktion wie z.B. die Mittelwertbildung, die Maximum- oder Minimumsfunktion oder Funktionen, die komplexe Datenverarbeitungen durchführen, verwendet

werden.
Für die folgenden Betrachtungen nehmen wir an, dass wir einen Graphen $G = (V, E)$ vorliegen haben, der k Quellen $Q_1, ..., Q_k$ und eine Senke S hat. Die Kanten E sollen dabei alle direkten Kommunikationsverbindungen zwischen den Knoten V darstellen.

Optimale Aggregation Wenn wir davon ausgehen, dass jeder Knoten im Netz pro Übertragung eines Phänomens genau eine Übertragung durchführt, so kann der hierdurch entstehende Aggregationsbaum als ein inverser Multicastingbaum interpretiert werden. Wenn wir als Kostenfunktion einen konstanten Wert c für jede Übertragung annehmen, so ist der Multicastbaum mit einer Minimalanzahl an Verbindungen (Kanten) ein Steinerbaum (bzgl. der Quellen und der Senke). Ein Steinerbaum (bzgl. einer Menge $T \subseteq V$) ist ein Baum, der alle Knoten aus T verbindet und dabei die Kostenfunktion minimiert. Dabei kann der Baum auch aus weiteren Knoten des Graphens V bestehen. Die optimale Anzahl an Übertragungen mit DCR ist gleich der Anzahl Kanten in einem Steinerbaum bzgl. $\{Q_1, ..., Q_k, S\}$. Man kann zeigen, dass bei zufälliger Platzierung der Quellen das Problem den optimalen Steinerbaum zu finden NP-hart ist(vgl. [5]). Daraus folgt, dass auch DCR mit optimaler Datenaggregation NP-hart ist.

Suboptimale Aggregation Es folgen nun drei suboptimale Methoden um Datenaggregationsbäume aufzubauen. Ziel ist es damit zu zeigen, dass es durch geeignete Algorithmen Datenaggregationsbäume aufgestellt werden können, deren Kantenanzahl (entsprichte der Übertragungsanzahl im Netz) in der Nähe der Kantenanzahl im optimalen Aggregationsbaum liegt.

1. **Center at Nearest Source (CNS)**: Die Quelle, die am nächsten an der Senke liegt agiert als Aggregationspunkt für alle anderen Quellen, d.h. alle anderen Quelle senden ihre Daten auf dem kürzesten Weg zu dieser Quelle und diese Quelle schickt dann das Aggregat das aus diesen Daten erstellt wurde an die Senke.

2. **Shortest Path Tree (SPT)**: Jede Quelle sendet ihre Daten auf dem kürzesten Weg zur Senke. Falls sich diese Wege überlappen so wird an den entsprechenden Knoten aggregiert und die Knoten bilden eine Verzweigung im Aggregationsbaum.

3. **Greedy Incremental Tree (GIT)**: Dieser Baum wird sequentiell aufgebaut. Zu Beginn besteht er nur aus der Senke. In jedem Schritt wird dem Baum die ihm nächstgelegenste Quelle hinzugefügt.

2.2 Energieeinsparungen

In diesem Abschnitt untersuchen wir die Energieeinsparungen die durch Datenaggregation möglich sind. Wir werden sehen, dass man die größten Einsparungen erhält, wenn die Quellen nahe beisammen liegen und die Senke weit von ihnen entfernt ist.

Im weiteren betrachten wir das optimale ACR und das optimale DCR.
Definitionen:

- Wir definieren $\Sigma = \{Q_1, \ldots, Q_k\}$ als die Menge der Quellen und S sei die Senke.

- $SP(i, j)$ sei der kürzeste Weg von i nach j.
- Für jede Quelle Q_i definieren wir $d_i = SP(Q_i, S)$.
- Sei N_{AC} die Anzahl der Übertragungen für das optimale AC-Protokoll.
- Ebenso sei N_{DC} die Anzahl der Übertragungen für das optimales DC-Protokoll.
- Der Durchmesser $X(A)$ einer Menge A sei $X(A) = \max_{i,j \in A} SP(i, j)$.
- Weiter definieren wir für den Spezialfall $A = \Sigma$: $X := X(\Sigma)$
- Als Fractional Savings definieren wir die normierte Anzahl an Einsparungen mit DCR gegenüber ACR: $FS = (N_{AC} - N_{DC})/N_{AC}$. $FS = 1$ bedeutet also eine 100%-tige Einsparung und $FS = 0$ keine Einsparungen.

Für die Anzahl der Übertragungen im optimalen AC-Protokoll gilt $N_{AC} = \sum_{i=1}^{k} d_i$, da jede Quelle von den anderen unabhängig ihre Daten zur Senke übertragen muss. Für die Anzahl der Übertragungen im optimalen DC-Protokoll gilt $N_{DC} \leq N_{AC}$, da bei optimaler Aggregation nur Übertragungen eingespart werden können. Der schlechteste Fall ist also der, dass keine Aggregation stattfindet.
Gleichung 1 definiert eine untere Schranke für die Gesamtanzahl der Übertragungen des optimalen DC-Protokolls. Der kürzeste Pfad von der zur Senke nähesten Quelle muss im Steinerbaum vorkommen. Die Daten der anderen Quellen müssen mindestens einmal übertragen werden. Diese Anzahl der Übertragungen ist optimal.

$$\min(d_i) + (k - 1) \leq N_{DC} \tag{1}$$

Gleichung 2 stellt eine obere Schranke für die Gesamtanzahl der Übertragungen des optimalen DC-Protokolls dar. Der Beweis ist konstruktiv. $k - 1$ Quellen senden ihre Daten zu derjenigen Quelle, die der Senke am nächsten liegt. Also erhält man höchstens $(k - 1) * X$ Übertragungen zu dieser Quelle. Hinzu kommen dann noch die Anzahl der Übertragungen von der aggregierenden Quelle zur Senke. Der optimale Baum darf höchstens soviele Übertragungen haben.

$$N_{DC} \leq (k - 1) X + \min(d_i) \tag{2}$$

Für den Fall, dass für den Durchmesser X der Menge der Quellen gilt $X < \min(d_i)$, so ist das optimalen DC-Protokoll auf jeden Fall besser als das optimalen AC-Protokoll.
Beweis:

$$N_{DC} \leq (k - 1) X + \min(d_i) \leq (k) * \min(d_i) \leq \sum_{i=1}^{k} di = N_{AC} \tag{3}$$

Aus der Definition des Fractional Savings und den Gleichungen 1 und 2 folgen direkt durch einsetzen:

$$1 - ((k - 1) X + \min(d_i)) / \left(\sum_{i=1}^{k} d_i \right) \leq FS \tag{4}$$

$$FS \leq 1 - (\min(d_i) + k - 1) / \left(\sum_{i=1}^{k} d_i \right) \tag{5}$$

Betrachtet man nun den Fall , dass $\min(d_i) = \max(d_i) =: d$, so erhält man aus obigen Gleichungen 4 und 5 die Gleichung

$$1 - ((k-1)X + d) / (k * d) \leq FS \leq 1 - (d + k - 1) / (k * d) \tag{6}$$

Wenn nun der Durchmesser und die Anzahl der Quellen konstant bleiben, sowie die Senke immer weiter von den Quellen entfernt wird, so erhält man

$$\lim_{d \to \infty} FS = 1 - 1/k. \tag{7}$$

Dies bedeutet, je näher die Quellen zusammen liegen und je weiter entfernt sie von der Quelle liegen, desto mehr Einsparungen sind mit dem optimalen DC-Protokoll zu erreichen.
Beweis: Wir zeigen, dass obere und untere Schranke gegen den gleichen Grenzwert konvergieren.

$$\lim_{d \to \infty} \left(1 - \frac{(k-1)X + d}{(k * d)}\right) = \left(1 - \frac{(k-1)X}{k * d} - \frac{d}{k * d}\right) = 1 - 1/k \tag{8}$$

$$\lim_{d \to \infty} \left(1 - \frac{(d + k - 1)}{(k * d)}\right) = \left(1 - \frac{k-1}{k * d} - \frac{d}{k * d}\right) = 1 - 1/k \tag{9}$$

Satz 1 *Falls der Subgraph G des durch die Quellknoten $Q_1, ..., Q_k$ induzierten Kommunikationsgraphen verbunden ist, so kann der Datenaggregationsbaum in polynomialer Zeit berechnet werden.*

Der Beweis ist konstruktiv. Wir verwenden hierfür den GIT-Algorithmus. Über diesen ist bekannt, dass er in polynomialer Zeit läuft [6]. Da der Subgraph verbunden ist, ist sobald die erste Quelle dem Baum hinzugefügt wurde in jedem Schritt des Algorithmus' ein Quellknoten genau eine Kante entfernt. Somit entspricht die Anzahl der Kanten im Baum der unteren Schranke in 1 und ist somit optimal. In diesem speziellen Fall ist das finden eines optimalen Aggregationsbaumes also nicht mehr NP-hart.

Satz 2 *Falls im ER-Modell $R > 2 * S$ gilt, so kann der optimale Aggregationsbaum in polynomialer Zeit gebildet werden.*

Beweis: Wenn $R > 2 * S$ gilt, dann sind alle Quellen von jeder Quelle aus in einem Hop zu erreichen. Also liegt ein Spezialfall von Satz 1 vor. GIT und CNS erzeugen hier beide den optimalen Aggregationsbaum.

In [3] werden in einer Simulation die drei vorgestellten Methoden zum Aufbau eines Aggregationsbaumes mit der unteren Schranke für das optimale DCR (Gleichung 1) und dem ACR verglichen. In Abbildung 2 kann man erkennen, dass im ER-Modell das GITDC nahezu mit der unteren Schranke übereinstimmt. Dies ist der Fall, da bei moderatem Sensing-Range der durch die Quellen induzierte Kommunikationssubgraph verbunden ist. Dann ist der Aggregationsbaum nach Satz 1 optimal. Die CNSDC-Methode nähert sich bei wachsendem Kommunikationsradius ebenfalls dem Optimum. Dies entspricht der Aussage von Satz 2.
Beim ER- sowie dem RS-Modell ergeben sich die meisten Einsparungen je kleiner der Kommunikationsradius ist. Bei RS-Modell sind es weniger als beim ER-Modell, da die Quellen stärker über das Einheitsquadrat verteilt sind und sie so weniger Möglichkeiten zur Aggregation haben.

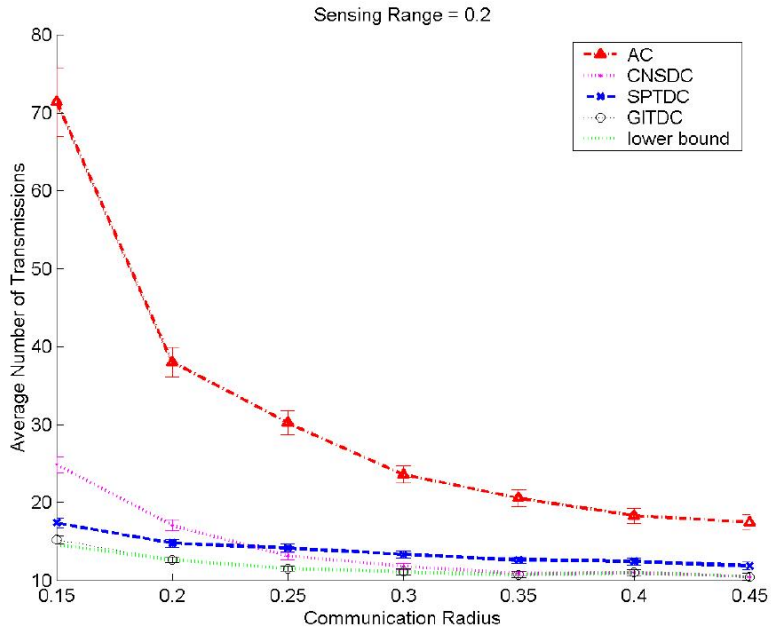

Abbildung 2: Vergleich von Energiekosten vs. Kommunikationsradius im ER-Modell [Quelle: [3]]

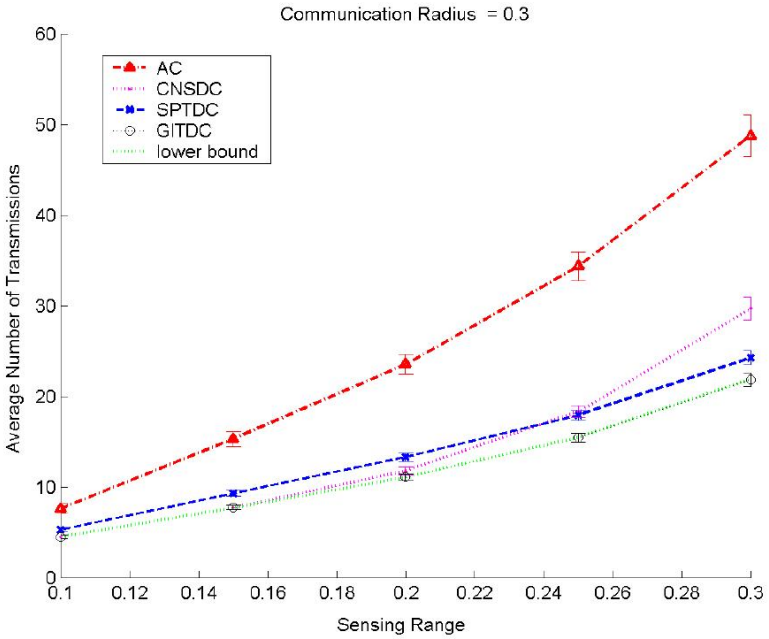

Abbildung 3: Vergleich von Energiekosten vs. Sensing-Range im ER-Modell [Quelle: [3]]

Aus Abbildung 3 kann man erkennen, das die Energieeinsparungen bei großem Sensing-Range am größten sind. Dasselbe gilt im RS-Modell, wenn die Anzahl der Quellen hoch ist.
Diese Beobachtungen stimmen mit den theoretischen Überlegungen überein. Man sieht, dass die Energieeinsparungen am höchsten sind, wenn die Quellen eng zusammen liegen und sie weit entfernt von der Senke sind. Das ist der Fall bei kleinem Kommunikationsradius.

2.3 Verzögerung

Mit den Energiegewinnen aufgrund der Datenaggregation handelt man sich aber möglicherweise auch Verzögerungszeiten bei den Ankunftsraten der Daten ein. Dies hängt allerdings von der Aggregationsfunktion ab. Bei der einfachen Aggregationsfunktion in Form der Duplikatsunterdrückung müssen keine Daten in den Knoten auf ihrem Weg zur Senke zurückgehalten werden. Sollen allerding Daten in den Knoten kombiniert werden, so müssen Daten zurückgehalten werden um sie mit später eintreffenden Daten aus entfernteren Knoten kombinieren zu können.
Im schlimmsten Fall muss auf das Eintreffen der Daten aus der am weitesten Entfernten Quelle gewartet werden. Im Falle des ACR muss nur auf die Daten der nähesten Quelle gewartet werden. Durch Differenzbildung dieser beiden Größen kann man den Effekt der Datenaggregation abschätzen. Die größte Differenz ergibt sich wenn der Kommunikationsradius groß ist und die Anzahl der Quellen groß.
Hier wird nur die Verzögerung durch Datenaggregation betrachtet, nicht jedoch die Verzögerung die Entstehen könnte durch die Verarbeitungszeit in den Knoten während des Aggregationsschrittes. Es ist aber unwahrscheinlich, dass diese gegenüber der anderen Verzögerung ist Gewicht fällt. Ebenfalls nicht beachtet wurde Verzögerungen die Auftreten könnten, wenn das Netz überlastet ist. Es ist aber nicht klar ob sich diese Verzögerungen in DC-Protokollen anders aüßern würden als in AC-Protokollen.

2.4 Robustheit

Durch Änderungen im beobachteten Phänomen, z.B. Ausbreitung oder Eintreten neuer Knoten in das Sensornetz kann sich die Anzahl der Quellknoten erhöhen. Gegen solche Effekte ist das DCR mit Datenaggregation sehr robust. Dies kann man sich besonders gut am GITDC-Modell klarmachen. Jede neu hinzugefügte Quelle sorgt nur für einen erhöhten Energieverbrauch die proportional zur minimalen Distanz der Quelle zum Aggregationsbaum ist, wohingegen sich beim ACR der Energieverbrauch proportional zur Distanz der neuen Quelle zur Senke erhöht. Dies lässt sich in Abbildung 4 am Beispiel des RS-Modells erkennen. Beim ER-Modell ergibt sich ein ähnliches Verhalten. Diese Energieeinsparungen können auch zu einer größeren Robustheit gegenüber Störungen führen die z.B. verhindern, dass ein Großteil der Quellen das Phänomen korrekt wahrnehmen kann. Da mit der gleichen Energie bei der Datenaggregation mehr Quellen zur Phänomenerfassung verwendet werden können senden mehr Quellen korrekte Daten an die Senke.

3 Directed Diffusion

Directed Diffusion ist ein Kommunikationsparadigma für Sensornetze. An Sensornetze sollen Anfragen gestellt werden können und diese sollen durch das Netz und die vor-

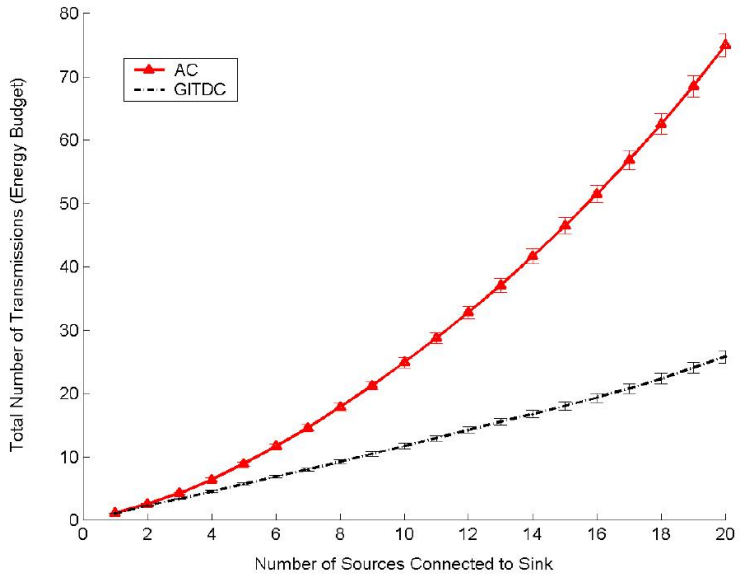

Abbildung 4: Vergleich der Übertragungskosten in Abhängigkeit von der Quellenanzahl beim RS-Modell [Quelle: [3]]

handene Sensorik beantwortet werden. Bei Directed Diffusion werden nicht konkrete Knoten angesprochen um die Aufgabe zu lösen, sondern die Sensorknoten müssen anhand der Anfrage des sogenannten „Interests“ erkennen, ob sie in der Lage sind die Anfrage zu beantworten. Directed Diffusion besteht aus drei Phasen. In der ersten Phase (Interestpropagation) wird die Anfrage (der Interest), die an einem Knoten des Netzes eingegeben wurde durch das Netz propagiert um die geeigneten Sensorknoten zur Bearbeitung der Anfrage zu finden. Den Knoten, an dem die Anfrage gestellt wurde, nennen wir Senke. Damit die Daten ihren Weg zurück zur Senke finden, werden während der Propagierung der Interests sogenannte Gradienten aufgesetzt, welche die Richtung des Datenflusses zurück zur Senke angeben. In der zweiten Phase speisen diejenigen Knoten, welche Daten zur Beantwortung der Anfrage gesammelt haben, diese Daten in das Netz ein, damit sie den Weg zurück zur Senke finden. Hierbei werden die Daten über verschiedene Pfade gesendet. In der dritten Phase (dem Reinforcement) wählt die Senke geeignete Datenleitungen aus, um die Wege, auf denen die Daten durch das Netz zur Senke finden sollen, einzuschränken. Wir werden sehen, dass Directed Diffusion skalierbar, robust und energieeffizient ist. Directed Diffusion ist ein datenzentrisches Kommunikationsparadigma und basiert auf benannten Daten. Im folgenden Abschnitt werden wir benannte Daten vorstellen und danach Designmöglichkeiten für die drei Phasen des Directed Diffusions vorstellen.

3.1 Benannte Daten

Directed Diffusion verwendet benannte Daten in Form von Attribut-Wert-Paaren. Dabei werden benannte Daten ebenso für die Aufgabenbeschreibung (Interest) wie auch für die Beschreibung der Daten, welche von den Sensorknoten in das Netz eingespeist

werden verwendet. Hier ist ein Beispiel für benannte Daten angegeben:

Typ = Vierbeiner
Instanz = Elefant
Intensität = 0.6
Vertrauen= 0.85
Ort = [-100,100,200,400]
Zeitstempel (ZS) = 01:20:40

Die einzelnen Attribute haben jeweils einen Wertebereich, aus dem die Attributwerte gewählt werden können. Es können sowohl einzelne Werte als auch Teilmengen des Wertebereichs benutzt werden. Für die Gestaltung der Datenbenennung gibt es verschiedene Möglichkieten. Hier soll allerdings nicht weiter darauf eingegangen werden, da hier die Idee, die hinter Directed Diffusion steckt, im Vordergrund stehen soll.

3.2 Interestpropagierung

Es wird nun anhand eines Beispiels die Funktionsweise der Interestpropagierung erklärt. Gegeben sei folgende Interessensbeschreibung, welche an jedem Knoten in das Netz eingegeben werden kann. Dieser Knote heisst Senke.

Typ = Vierbeiner
Intervall = 10ms
Rechteck = [-100,200,200,400]
Zeitstempel = 01:20:40
Ablaufsdatum = 01:30:40

Die Senke speichert diesen Interest und entfernt ihn wenn das Ablaufsdatum überschritten ist. Um nun den Interest im Netz zu verteilen sendet die Senke ihn an alle ihre Nachbarn. Dabei wird der initiale Interest mit einer größeren Intervallangabe als die originale Anforderung es vorgibt abgesendet. Dies wird gemacht um sicherzustellen, dass alle möglichen Quellen von der Anfrage erfahren und um gleichzeitig zu verhindern, dass das Netz überlastet wird. Da auch diese initialien Interests verloren gehen können sendet die Senke die niederfrequenten Interests periodisch solange weiter, bis das Ablaufdatum erreicht wird. Somit erfahren z.B. auch Quellen, die sich erst nach der ersten Interestpropagation anschalten bzw. in das Netz eintreten, von der Anfrage. Die gewünschte Frequenz (Datenrate) wird dann in der dritten Phase durch das Reinforcement erreicht, worauf wir später eingehen.
Jeder Sensorknoten hat einen Interestcache in dem die eingetroffenen Interests gespeichert werden. Für jeden Interest, der eine noch nicht vorhandene (Typ, Rechteck)-Kombination aufweist, wird ein neuer Cacheeintrag eingerichtet. Interests, die gleiche Typ- und Rechteckwerte haben, können aggregiert werden. Ein Cacheeintrag besteht aus den Typ-, Rechteck- und Zeitstempelfeldern und einer Menge von Gradienten, von denen jeder auf den Nachbarn zeigen, von dem ein Interest erhalten wurde. Die Gradienten besitzen ein Feld für die Datenrate (ermittelt aus der Intervallangabe des Interests) sowie eine Gültigkeitsdauer (= Ablaufsdatum - Zeitstempel). Kommt ein neuer Interest in den Knoten und es existiert noch kein passender Eintrag, so wird ein neuer Cacheeintrag eingerichtet und aus den Feldern des Interests initialisiert. Falls schon ein passender Eintrag existiert, so werden der Zeitstempel und alle Gültigkeitsdauern der Gradienten des Cacheeintrags aktualisiert. Falls noch kein Gradient in die Richtung aus

der der Interest kam existiert, so wird ein neuer eingerichtet, andernfalls wird der vorhandene aktualisiert. Sobald ein Gradient seine Gültigkeitsdauer überschreitet wird er entfernt. Wenn alle Gradienten abgelaufen sind, so wird auch der Cacheeintrag selbst entfernt.
Die einzelnen Knoten kennen nur ihre unmittelbaren Nachbarn mit denen sie kommunizieren können. Sie können nicht feststellen, ob der Urheber einer empfangenen Interests ihr Nachbar oder eine weiter entferntere Senke ist. Ebenso können sie nur ihre lokale Information nutzen um zu entscheiden, ob sie eintreffende Interests an ihre eigenen Nachbarn (mit Ausnahme dessen von dem sie den Interest erhalten haben) weiterschicken. Falls er keine weitere Information vorliegen hat, kann ein Knoten den Interest nur weitersenden, was einem Fluten des Interest durch das Netzwerk entspricht. Eine andere Möglichkeit wäre geographisches Routing zu verwenden um die Richtung der Interestverbreitung einzuschränken und somit Energie einzusparen. Ebenso kann ein Knoten einen Datencache nutzen in dem er von anderen Knoten empfangene Antworten speichert und den Interest dann in jene Richtungen weitersenden aus der schon zu der Anfrage passende Antworten gekommen sind. Falls die Interests durch das Netz geflutet werden, so werden Gradienten in alle Richtungen aufgebaut. Erst das Reinforcement kann dann Pfade auswählen über die die Daten zielgerichtet zur Senke gesendet werden. Die vielen Gradienten ermöglichen eine Robustheit gegenüber Änderungen in der Pfadqualität oder bei Wegfall oder Unterbrechung eines Pfades. Solch ein Pfad kann dann lokal repariert werden und sich damit selbst heilen. Weiterhin kann man die Gradienten auch dazu nutzen, die Daten probabilistisch auf verschiedenen Pfaden zu versenden und damit eine gleichmäßigere Auslastung des Netzes erreichen.

3.3 Datenpropagierung

Bekommt ein Sensor ein Interest in dessen Rechteckbereich er sich befindet und hat er die erforderlichen Sensoren, so kann er seine Sensoren abfragen. Um einen Kodebucheintrag zu erstellen vergleicht der Sensorknoten die Sensordaten mit vorhandenen Beispielen und ordnet der Klassifizierung dann eine Vertrauenswahrscheinlichkeit und speichert diese zusammen mit der Signalstärke und dem Kodebucheintrag für die Klassifizierung. Entdeckt der Sensorknoten ein Ziel, so durchsucht er seinen Interestcache nach Anfragen zu diesem Ziel. Falls diese vorhanden sind, so berechnet er die höchste Datenrate und schickt die Daten mit dieser Frequenz in die entsprechende Richtung. Es folgt ein Beispiel für einen solchen Dateneintrag:

Typ = Vierbeiner
Instanz = Elefant
Ort = [125,220]
Signalstärke = 0.6
Vertrauenswahrscheinlichkeit = 0.85
Zeitstempel = 01:20:40

Wenn ein Knoten nun ein Datum empfängt, so überprüft er seinen Interestcache um zu sehen, ob er an diesen Daten interessiert ist. Ist dies nicht der Fall, so wird das Datum verworfen. Wenn Interesse an den Daten besteht, so überprüft der Knoten seinen Datencache in dem er zuvor gesehene Daten speichert. Aufgrund des Datencaches kann nun entschieden werden, ob das Datum weitergesendet wird oder nicht. Falls das Datum schon einmal gesehen wurde, so liegt eine Schleife vor und es wird verworfen. Es kann auch temporär zurückgehalten werden um eine Datenaggregation durchzu-

führen. Bevor der Knoten die Daten aber weitersendet muss er noch die gewünschte Datenrate je Gradient in dem Interestcache betrachten. Falls diese größer oder gleich der Dateneingangsrate (bestimmt durch die Einträge im Datencache)ist, so sendet er die Daten mit der Eingangsrate weiter (möglicherweise zeitverzögert um Aggregation durchführen zu können). Falls ein Gradient eine geringere Datenrate wünscht (z.B. die Hälfte der Dateneingangsrate), so kann der Knoten in dessen Richtung nur jedes zweite Datum oder eine Interpolation aus zwei aufeinanderfolgenden Daten schicken.

3.4 Reinforcement

Hat eine Senke also ein Interest in Netz gesendet und haben einige Quellen passende Daten in Netz abgegeben so treffen diese Daten auf potentiell mehreren Wegen bei der Senke ein. Im Falle eines Interestflutens durch die Senke werden die Daten sogar zu jedem Knoten im Netz geschickt. Diese Datenflut muss nun eingeschränkt werden. Dies geschieht indem die Senke sich lokal passende Verbindungen aussucht über die sie die Daten gesendet bekommen möchte. Dies teilt sie den ausgesuchten Nachbarknoten mit (Reinforcement) und diese suchen sich ihrerseit geeignete Verbindungen aus über die sie die Daten geliefert bekommen möchten. Die Pfadauswahl wird also sukzessive lokal unter Zuhilfenahme der jeweiligen Datencaches getroffen. Diese Pfadauswahl kann z.B. realisiert werden indem das Interest mit der eigentlich gewünschten Datenrate an die ausgewählten Nachbarn gesendet wird und so ein oder mehrere Pfade mit hoher Datenrate ausgewählt werden. Mögliche lokale Regeln für die Pfadauswahl sind:

1. Wähle den Nachbarn, der zuerst das letzte neue Datum geliefert hat.
2. Wähle alle Nachbarn aus, von denen neue Daten (also zuvor noch nicht gesehene) geliefert wurden.
3. Wähle die Nachbarn, die die meisten neuen Elemente geliefert haben.
4. Wähle diejenigen Nachbarn, die konsistent zuerst neue Elemente liefern.

Durch diese Regeln wird ein Pfad ausgewählt, der eine kurze Verzögerung in der Datenlieferung aufweist. Die Regeln 1 und 2 reagieren sehr stark auf Änderungen in der Pfadqualität, was zu einer erhöhten Energiebelastung führen könnte. Diesem wird durch die Regeln 3 und 4 entgegengewirkt, indem nicht auf jede kleine Änderung sofort reagiert wird. Die genauen Auswirkungen dieser Regeln müssen erst noch untersucht werden.
Um nun Pfade, die zuvor verstärkt wurden, wieder abzuschalten, z.B. weil die Qualität der Datenlieferung abgenommen hat, braucht man einen Mechanismus mit dem man die Pfade „negativ verstärken“ kann. Eine Möglichkeit wäre Pfade mit hoher Gradienten mit hoher Datenrate mit einem Timeout zu versehen, so dass nach einer gewissen Frist in der keine Verstärkung erfolgte die hochfrequenten Gradienten entfernt werden. Die zweite Möglichkeit ist die, dass man explizit Interests mit niedriger Frequenz versendet und so die entsprechenden Gradienten zurückgesetzt werden. Wenn der Knoten seinen letzten Gradienten (bzgl. eines Interests) mit hoher Datenrate entfernt, so muss er seinen Nachbarn ebenfalls eine Aufforderung zur verringerung ihrer Datenrate senden. Hiermit kann ein Pfad schnell degradiert werden, dies erfordert aber einen erhöhten Ressourceneinsatz.
Für das negative Reinforcement gibt es ebenfalls wieder mehrere Möglichkeiten die negativ zu verstärkenden Knoten auszuwählen. Die Wahl der Knoten ist dabei unabhängig von der Art wie negativ verstärkt wird.

1. Wähle den Nachbarn, von dem kein neues Datum in einem Fenster von N Daten oder T Sekunden gesendet wurde.

2. Wähle diejenigen Nachbarn, die am wenigsten neue Daten liefern.

Wie beim Reinforcement ist auch hier eine weitere Untersuchung der Auswahlmöglichkeiten erforderlich um zu sehen, welche Regeln am energiesparendsten sind.
Beim Reinforcement stellen sich dabei grundsätzlich folgende Fragen:

- Wann soll (negatives) Reinforcement angewendet werden?
- Wieviele Knoten sollen ausgewählt werden?
- Welche Knoten sollten ausgewählt werden?

Es ist auch eine Strategie nötig mit der man vermeiden kann, dass eine Senke von zwei Quellen auf verschiedenen Pfaden Daten zieht, die über diese Pfade geringfügig früher eintreffen anstatt den Pfad zu favorisieren auf dem die Daten der beiden Quellen aggregiert werden könnten. Einfach hingegen ist es für eine zweite Senke, die das gleiche Interest hat wie eine andere. Sie kann die ausgerichteten Gradienten der ersten Senke sofort nutzen um mit hoher Datenrate Daten aus dem Netz zu ziehen.
Senken nutzen Reinforcement um Datenpfade mit hoher Datenrate zu erzeugen. Diese Pfade werden sukzessive durch die Zwischenknoten zwischen Senke und Quellen aufgebaut. Zwischenknoten können Reinforcementregeln aber auch dazu nutzen einen unterbrochenen Pfad (z.B. durch Energieerschöpfung oder Zerstörung eines Knotens) zu flicken, indem sie einen Umweg um das Hinternis suchen. Hierfür können sie Reinforcement verwenden, um aus ihren Nachbarn den geeignetesten Knoten auszuwählen. Neben der Verstärkung der gewünschten Daten als Antwort auf ein ausgesendetes Interest ist es auch denkbar mit Reinforcement spontan von Quelleknoten kundgetane Ereignisse zu verstärken.

3.5 Evaluierung

In [2] führen die Autoren eine Simulation eines Senornetzes mit Directed Diffusion durch, um die Effizienz des Paradigmas zu testen. Als Aggregationsstrategie wird die Duplikatsunterdrückung angewendet, wobei die Quellen nur gleiche Daten senden. Verstärkt (Reinforcement) werden die Knoten, die ein neues Ereignis senden, und negativ verstärkt werden nur die Pfade die konsistent keine neuen Ereignisse liefern. Weitere Einzelheiten zum Testaufbau können [2] (Kapitel 4) entnommen werden. Die Leistung wird mit Hilfe von zwei Metriken untersucht, die wir bereits schon beim datenzentrischen Routing kennengelernt haben.

1. Die **durchschnittlich verbrauchte Energie** misst das Verhältnis von totaler verbrauchter Energie pro Knoten zu der Anzahl von Ereignissen die die Senken sehen.

2. Die **durchschnittliche Verzögerungszeit** gibt die durchschnittliche Latenz an, die auftritt wenn ein Ereignis von einer Quelle zur Senke geschickt wird.

Im Falle von Dynamik im Netz wird auch noch das **Ereigniszustellungsverhältnis** untersucht, welches das Verhältnis zwischen zugestellten und von den Quellen abgegebenen Ereignissen angibt. Um das Testszenario zu vereinfachen wird vorausgesetzt, dass das Sensornetz nicht überlastet ist.

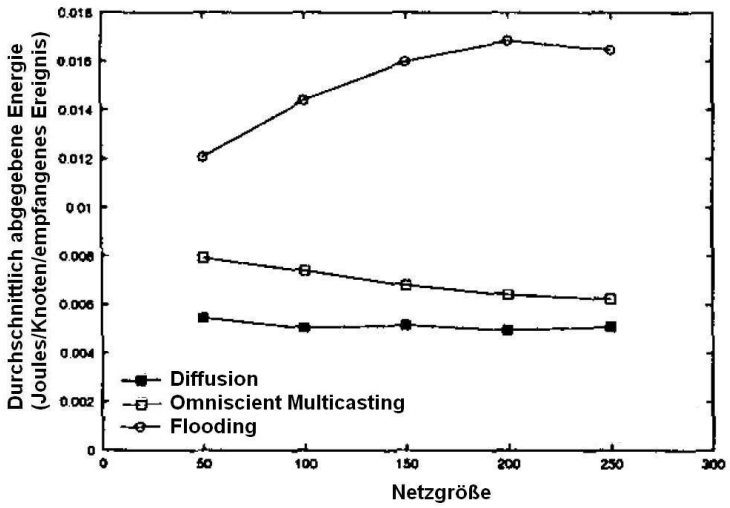

Abbildung 5: Durchschnittlich verbrauchte Energie von directed Diffusion im Vergleich zu Flooding und Omniscient Mulicasting [Quelle: [2]]

Zudem wurde Directed Diffusion mit Flooding und Omniscient Mulicasting verglichen um die Leistung von directed Diffusion mit Hilfe dieser oberen und unteren Schranken einzuschätzen. Da Flooding alle Knoten mit den gleichen Daten versorgt ist dies der denkbar schlechteste Fall. Beim Omniscient Mulicasting wird davon ausgegangen, dass die Daten ihren optimalen Zustellpfad im Netz kennen und es ist somit eine untere Schranke für Protokolle ohne Datenaggregation. Wie man aus Abbildung 5 erkennen kann ist Omniscient Mulicasting ungefähr doppelt so gut wie Flooding. Dies ist darauf zurückzuführen, dass hier die Ereignisse nur über einen Pfad ausgeliefert werden. Die zusätzlichen Gewinne von directed Diffusion treten auf, da es von Datenaggregation im Sensornetzwerk zusätzlich zur Auslieferung der Ereignisse über nur einen Pfad profitiert. Die Energieeinsparungen bei directed Diffusion sind nicht linear in der Anzahl der Quellen, da auch Energie für das Warten auf Daten verwendet wird und ausserdem werden über mehrere Datenpfade Daten zu der Senke gesendet, da die Reinforcementregel recht aggressiv ist und die Regel für negatives Reinforcement recht konservativ. Sobald nämlich ein Knoten ein neues Ereignis sendet wird er der zugehörige Datenpfad verstärkt und negativ verstärkt werden nur die Pfade, die konsistent keine neuen Ereignisse liefern.
Abbildung 6 zeigt, dass die Verzögerung bei directed Diffusion der von Omniscient Mulicasting entspricht. Das schlechte Abschneiden von Flooding ist auf die besonderen Versuchsbedingungen zurückzuführen (vgl. [2]).

Der Einfluss von Dynamik Um den Einfluss von Dynamik auf directed Diffusion zu testen wurden zufällig Knoten im Netz ausgeschaltet, sowie Knoten auf den Pfaden der Ereignisauslieferung an die Senke. Um den Einfluss von Dynamik besser zu erkennen, wurden in diesem Szenario unterschiedliche Daten an der Quellen eingespeist.
Die Autoren kommen zu den Ergebnissen, dass die Knotenausfälle keinen übermässigen Einfluss auf die zeitliche Auslieferung der Ereignissdaten haben. Die Energie-

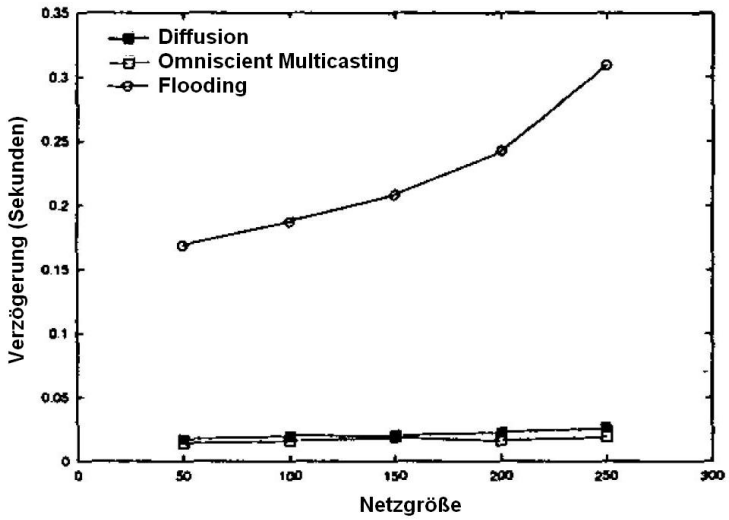

Abbildung 6: Durchschnittliche Verzögerungszeit bei directed Diffusion im Vergleich zu Flooding und Omniscient Mulicasting [Quelle: [2]]

kosten gehen sogar bei Knotenausfällen zurück. Dies ist so, da die verwendeten Reinforcementregeln mehrere Pfade zu den Datenquellen etablieren, so dass im Falle von Knotenausfällen keine neuen aufgebaut werden müssen und somit keine zusätzlichen Energiekosten entstehen. Einen Rückgang der Ereignisauslieferung von $\approx 85\%$ Auslieferung bei keinen Knotenausfällen auf $\approx 60\%$ bei 20% Knotenausfällen bewerten die Autoren als ausgezeichnet.

Energieeinsparungen Die Abbildungen 7 und 8 zeigen worauf die Energieeinsparungen bei directed Diffusion zurückzuführen sind. Abbildung 7 zeigt, dass directed Diffusion vom negativem Reinforcement profitiert. Selbst bei der sehr konservativen negativen Reinforcementregel kann der Energieverbrauch halbiert werden, indem Pfade mit hoher Verzögerung geprunt werden.

Abbildung 8 zeigt, dass in diesem Szenario directed Diffusion in kleinen Netzen 5-mal soviel Energie benötigt, wenn keine Duplikatsunterdrückung angewendet wird und in größeren Netzen 3-mal soviel. In großen Netzen gibt es nämlich längere Pfade, so dass über diese eine längere Verzögerungszeit bzgl. der Ereignisübertragung an die Senken entsteht. Diese werden durch die konservative Regel für negatives Reinforcement geprunt und verringern so den Energiebedarf in größeren Netzen.

4 Zusammenfassung und Ausblick

Wie gezeigt wurde, ist directed Diffusion ein robustes und skalierbares Kommunikationsparadigma für Sensornetzwerke. Es basiert auf benannten Daten und datenzentrischem Routing dessen Kern die Datenaggregation im Netz ist. Ein Sensornetzwerk mit directed Diffusion ist also ein verteiltes System.
Im Gegensatz zu herkömmlichen Netzen findet die Kommunikation auf Nachbar-zu-

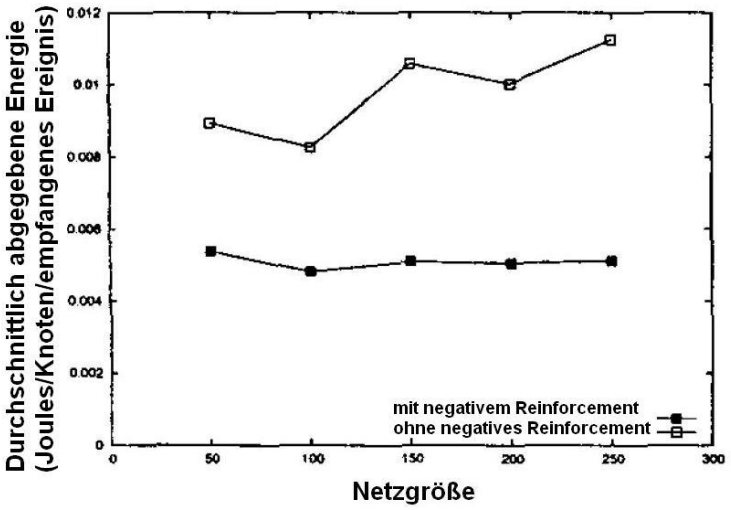

Abbildung 7: Negatives Reinforcement [Quelle: [2]]

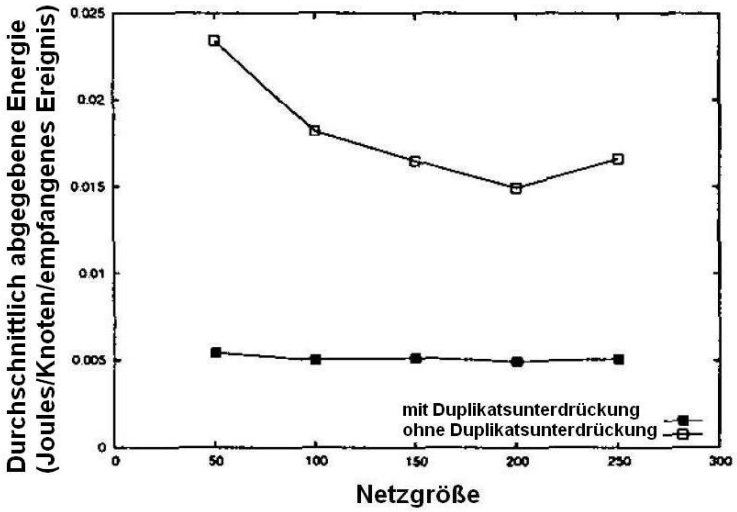

Abbildung 8: Duplikatsunterdrückung [Quelle: [2]]

Nachbar-Ebene (Hop-to-Hop-Ebene) statt. Es gibt also auch keine Router und ebensowenig werden global eindeutige Adressen benötigt. Es reicht aus, wenn jeder Knoten seine Nachbarn lokal unterscheiden kann. Sie sind die einzigen Netzteilnehmer, die er kennt. Aufgrund dieser Lokalität kann schnell und energieeffizient auf dynamische Änderungen reagiert werden und es muss keine Topologieänderung global verbreitet werden. Allerdings sind die resultierenden Pfade dadurch suboptimal, was allerdings durch effiziente Aggregationstechniken ausgeglichen werden kann.
Directed Diffusion ist robust gegenüber Änderungen in beobachteten Phänomenen und in der Netzstruktur. Ebenfalls ist es ein skalierbares Kommunikationsparadigma, da die Speicherung der Interessen nicht von der Anzahl der Knoten im Netz sondern nur von der Anzahl der zu dem jeweiligen Zeitpunkt gültigen Anfragen und der durch die Knoten geschleusten Daten abhängt.
Das Netz ist im Gegensatz zu traditionellen Netzen aufgabenspezifisch. Es kann Daten, die im Netz versandt werden, ändern und somit auch komprimieren und damit die Energiebelastung des Netzes reduzieren. Aufgrund dieser Tatsache ist es allerdings auch stärkeren Sicherheitsrisiken ausgesetzt. So kann neben in die Knoten des Netzes eingeschleustem feindlichen Kode auch die Einschleusung von feindlichen Netzknoten ein Sicherheitsrisiko darstellen.
Durch Directed Diffusion kann die Verarbeitung der Sensordaten bereits im Netz verfolgen. Dies ist neu gegenüber herkömmlichen Netzen in denen entweder ein Phänomen aus der Ferne beobachtet wird und dann Störgrößen eliminiert werden müssen oder ein Phänomen lokal durch mehrere Sensoren beobachtet wird, die Verarbeitung aber weit entfernt in einer leistungsfähigen Rechenanlage stattfindet.
Im Vergleich zu mobilen Ad-hoc-Netzen (MANETs) findet man in Sensornetzen Many-to-one-Datenflüsse von vielen Quellen zu einer Senke und in ihnen treten Redundanzen in den Daten auf. Ebenso bleibt die Struktur der Sensorknoten eher stabil, während sich in MANETs die Topologie durchaus öfter ändern kann. Stattdessen kann aber das beobachtete Phänomen über die Zeit seine Position im Raum ändern. Desweiteren sind die Energiebeschränkungen bei Sensornetzen wesentlich stärker als in MANETs, da in letzteren die Knoten in gewissen Abständen wieder aufgeladen werden können. Bei Sensornetzen müssen die Knoten jedoch mit der Energie, die sie zur Zeit des Netzaufbaus haben für die gesamte Lebensdauer des Netzes auskommen.

Diese Seminararbeit hat einen Überblick gegeben, wie ein System mit directed Diffusion konzipiert werden kann. Dabei müssen aber noch viele Einzelheiten untersucht werden. Anzuführen währen hier z.B. die Regeln für Reinforcement und negatives Reinforcement, welche auf Energieeffizienz untersucht werden müssen. Ebenso wäre es von Interesse ein Rahmenwerk auf directed Diffusion aufzubauen mit dem dann verschiedene Aufgaben gelöst werden können, so dass die immer wiederkehrenden nicht anwendungsspezifischen Aspekte von directed Diffusion nicht immer wieder neu implementiert werden müssen. Ein solches „Diffusionssubstrat“ ist von den Autoren der zugrundeliegenden Artikel (vgl. [2]) geplant.
Desweiteren sollten noch Techniken entwickelt werden, mit denen sichergestellt werden kann, dass die Energiebelastung des Netzes gleichmäßig auf die einzelnen Knoten verteilt wird. Dadurch würde verhindert werden, dass einige Knoten weitaus früher als andere aufgrund von Energieverlusten dem Netz nicht mehr zu Verfügung stehen.

Literatur

[1] Friedemann Mattern, Kay Römer: *Drahtlose Sensornetze*, GI Informatik-Lexikon, 2003. HTML

[2] Chalermek Intanagonwiwat, Ramesh Govindan und Deborah Estrin: *Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*, International Conference on Mobile Computing and Networking (MobiCom 2000), Seite 56-67, ACM, 2000.

[3] Bhaskar Krishnamachari, Deborah Estrin, Stephen Wicker: *Modelling Data-Centric Routing in Wireless Sensor Networks*, Proceedings of the Twenty First International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002), IEEE, 2002.

[4] John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin und Deepak Ganesan: *Building Efficient Wireless Sensor Networks with Low-Level Naming*, 18th ACM Symposium on Operating Systems Principles, ACM, 2001.

[5] M. R. Garey und D. S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, San Francisco, 1979. 3.5, 5, 3.5, 6, 7, 8, 4

[6] H. Takahasi und A. Matsuyama: *An approximate solution for the steiner problem in Graphs*, Math. Japonica, Band 24, Nr. 6, Seite 573-577, 1980. 1, 2.2, 2, 3, 4