

Algorithms for Topology-Aware Sensor Networks

Alexander Kröller

TU Braunschweig

Institut für Betriebssysteme und Rechnerverbund, Abteilung Algorithmenik
a.kroeller@tu-bs.de

Abstract: Ein wichtiges Problem bei Sensornetzwerken besteht darin, den Knoten ein Lokationsbewusstsein zu ermöglichen. Im klassischen Ansatz sind dafür NP-schwere Probleme zu lösen. Bei der Entwicklung von Alternativen entstanden drei Forschungsbereiche, die hier dargelegt werden. Zuerst stellen wir ein neuartiges Lokationsbewusstsein vor, das allen Knoten des Netzes eine globale Sicht auf die Netzwerktopologie ermöglicht. Danach untersuchen wir ein Netzwerkflußproblem, das auch im zentralisierten Modell noch nicht analysiert wurde. Schließlich stellen wir den Netzwerksimulator Shawn vor.

1 Einführung

Die vorliegende Arbeit beschäftigt sich mit algorithmischen und geometrischen Fragestellungen in Sensornetzwerken. In den 1990er Jahren entwickelte sich die Idee von unsichtbar kleinen, intelligenten und vernetzten Partikeln von futuristischer Träumerei hin zu greifbar nahen technischen Möglichkeiten. Das SmartDust-Projekt [KKP99] der UC Berkeley beschrieb detailliert, wie sogenannte „Motes“, winzige Computer in der Größe eines Kubikmillimeters, zu konstruieren seien. Bereits heute existieren Geräte mit einer Vielzahl von Anwendungen, und die fortschreitende Miniaturisierung und der mit steigender Produktion einhergehende Preisverfall werden in den nächsten Jahren große Sensornetzwerke ermöglichen.

Klassische Algorithmen verwenden Berechnungsmodelle, in denen ein einzelner Prozessor sequenziell Anweisungen abarbeitet, und dabei Zugriff auf die vollständigen Daten eines Problems hat. In Sensornetzwerken sind derartige Modelle bedeutungslos. Es werden verteilte Algorithmen benötigt, bei denen die einzelnen Prozessoren miteinander Daten austauschen, Berechnungen durchführen und gemeinsam eine Aufgabe bewältigen, zu der einzelne Geräte niemals in der Lage wären. Weitere Herausforderungen bestehen darin, dass die Prozessoren sehr einfach, die Speicherausstattung klein, und die Energieversorgung gering ist. Zusätzlich kann nicht jedes Gerät mit jedem anderen beliebig Daten austauschen, sondern nur mit solchen, die es mittels seiner Funkschnittstelle erreichen kann. Dadurch spielt die geometrische Verteilung eine entscheidende Rolle.

Diese Arbeit umfasst neben Einführung und Überblick in das Thema sowie einer Voruntersuchung drei wesentliche Teile: Zuerst untersuchen wir das grundlegende Problem, wie

Sensorknoten ein Bewusstsein für ihre Position erlangen können. Motiviert daraus, dass das Problem, Koordinaten für ein globales Koordinatensystem zu bestimmen, in fast allen Varianten NP-schwer ist, wird ein vollkommen neuer Ansatz skizziert, bei dem das Netzwerk selbständig geometrische Cluster bildet und einen abstrakten Graphen konstruiert, der die Topologie des zugrunde liegenden Gebiets sehr genau widerspiegelt. Das sich daraus ergebende Positionsbewusstsein ist für einige Anwendungen dem klassischen euklidischen Ansatz deutlich überlegen.

Der zweite Teil widmet sich einem Flussproblem für Sensornetzwerke, das klassische dynamische Flüsse um Batteriebeschränkungen erweitert. Gesucht ist ein Fluss, der für gegebenen Zeithorizont die Datenmenge maximiert, die von einer Quelle zur Senke geschickt werden kann. Dieses Problem wird auch im zentralisierten Modell untersucht, da keine Vorarbeiten existieren. Wir beweisen Komplexitäten von Problemvarianten und entwickeln Approximationsschemata.

Der dritte Teil stellt den Netzwerksimulator Shawn vor. Da der Benutzer zwischen verschiedenen geometrischen Kommunikationsmodellen wählen kann und das Speichermodell für den daraus resultierenden Graphen an den verfügbaren Speicher sowie an Simulationsparameter wie eventuell mögliche Mobilität der Knoten anpassen kann, ist Shawn hochflexibel und gleichzeitig deutlich schneller als vergleichbare Simulationsumgebungen.

2 Topologisches Lokationsbewusstsein

Eines der wichtigsten Problemfelder bei Sensornetzen ist *Lokalisierung*, die Bestimmung der Position einzelner Knoten. Viele Anwendungen gehen davon aus, dass eine unbekannte Zahl von Sensorknoten in einem Gebiet ausgebracht werden, über das den Knoten keine Information zur Verfügung steht. Wenn die Knoten starten, kennen sie weder ihre eigene Position noch die der anderen Knoten. Nicht lokalisierte Sensornetze sind jedoch extrem eingeschränkt:

- Ein isolierter Sensorwert ist ohne zugehörige Position kaum zu verwerten.
- Die Kommunikation zwischen nicht direkt benachbarten Knoten ist deutlich erschwert, wenn die Position des Ziels nicht bekannt ist.

Ein naheliegender Ansatz der Lokalisierung besteht darin, die Koordinaten jedes Knotens in einem Koordinatensystem zu bestimmen. Dabei kann es sich um ein extern vorgegebenes, etwa dem Gradnetz auf der Erdkugel, oder um ein nur lokal gültiges System handeln. Sollten alle Knoten mit Positionierungsgeräten wie GPS-Empfängern ausgestattet sein, ist dieser Ansatz leicht umzusetzen. Dieser Fall ist jedoch eher selten, da solche Geräte teuer sind und nur im Freien funktionieren.

Daher wurden Probleme studiert, die Koordinaten aller Knoten aus den Positionen von einigen oder keinen Knoten zu berechnen, unter Zuhilfenahme von Abstandsschätzungen, Nachbarschaftsbeziehungen oder gemessenen Winkeln. Diese Ansätze sind oft dadurch

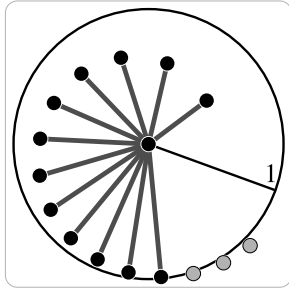
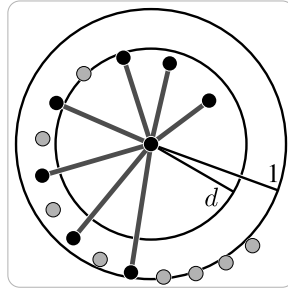
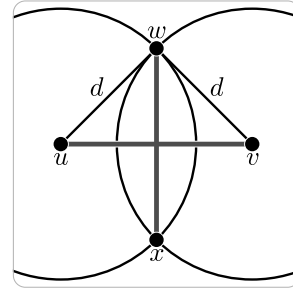


Abbildung 1: Unit Disk Graph (UDG).

Abbildung 2: Quasi Unit Disk Graph (d -QUDG).Abbildung 3: Kreuzungseigenschaft bei d -QUDG mit $d \geq 1/\sqrt{2}$.

motiviert, dass es trivial ist, ein relatives Koordinatensystem zu bestimmen, wenn die Knoten in allgemeiner Lage sind und exakte Abstände für alle Knotenpaare vorliegen. Unglücklicherweise ist diese Problem nur in diesem Fall trivial, jede praktisch relevante Variante ist NP-schwer, beispielsweise wenn exakte Abstände bekannt sind, aber nur für räumlich nahe Knotenpaare [AGY04], wenn exakte Winkel für nahe Knoten bekannt sind [BGJ05], oder wenn Abstände und Winkel naher Knoten bekannt sind, aber einem beliebig kleinen Fehler unterliegen [BGMS06]. In Experimenten zeigte sich, dass praktisch eingesetzte Lokalisierungsheuristiken schon bei vermeintlich einfachen Netzwerken Fehler produzieren [FKB⁺05].

Ein weiteres Problem besteht in der Ausnutzung der Lokationsinformation in Anwendungen wie beispielsweise Routing. Wenn für eine Nachricht nur die Zielposition bekannt ist, nicht aber die Knotenpositionen auf einem Weg dorthin, müssen aufwändige Protokolle zur Wegfindung eingesetzt werden, da die verfügbare Information keine Aussage über die Netzwerktopologie zwischen Quelle und Ziel zulässt, insbesondere über die Existenz und Lage von Löchern.

Zusammengenommen motiviert dies die Frage, inwieweit Positionsbewusstsein immer koordinatenbasiert sein muss. Im Folgenden wird ein alternativer Ansatz skizziert. Wir konzentrieren uns dabei auf Szenarien, in denen das Netzwerk eine komplexe Topologie mit vielen Löchern aufweist, also auf den für klassische Routingverfahren schwierigsten Fall. Wir nehmen an, dass das Netzwerk ein d -Quasi Unit Disk Graph [BFN01] mit $1/\sqrt{2} \leq d \leq 1$ ist. Dabei handelt es sich um eine Verallgemeinerung des klassischen Unit-Disk-Graph-Modells, in dem zwei Knoten genau dann kommunizieren können, wenn sie innerhalb eines bestimmten Abstands liegen, siehe Abbildung 1. Ein d -QUDG mit Parameter $0 < d \leq 1$ berücksichtigt, dass in der Realität Kommunikationsbereiche keine perfekten Kreise sind. In diesem Modell können Knoten immer kommunizieren, wenn ihr Abstand höchstens d beträgt, und garantiert nicht, wenn der Abstand über 1 ist. Für Abstände zwischen d und 1 wird nichts vorausgesetzt. QUDGs mit $d \geq 1/\sqrt{2}$ haben eine Eigenschaft, die wir für unser Positionsbewusstsein ausnutzen, siehe Abbildung 3: Die geometrische Kante xw zwischen zwei benachbarten Knoten kann keine andere Kante uv kreuzen, ohne dass die beiden Knotenpaare zusammenhängen.

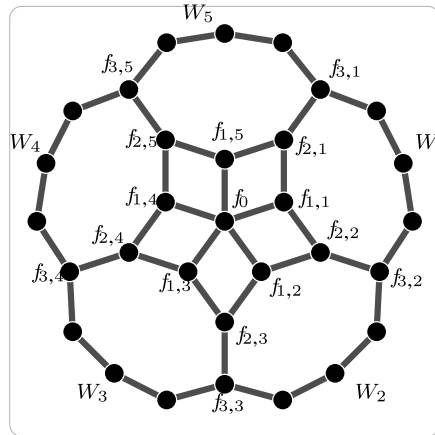


Abbildung 4: Eine 5-Flower.

Wir gehen davon aus, dass jeder Knoten seine direkten Nachbarn und die indirekten Nachbarn über konstant viele Zwischenknoten kennt, also den lokal induzierten Subgraphen. Kein Knoten hat Zugriff auf Koordinaten, Abstände oder Winkel im Netz.

Unser Verfahren zur Positionsbestimmung verwendet zwei Phasen: Die Erkennung des Randes und die Konstruktion von geometrischen Clustern. Eine anschauliche Visualisierung des Verfahrens, allerdings mit einer früheren Form der zweiten Phase, findet sich in unserem Video [FK06a].

2.1 Phase 1: Randerkennung

In der ersten Phase sollen die Knoten erkannt werden, die am Rand des Gebietes liegen. Dadurch wird eine Form von Topologiebewusstsein geschaffen, da die Randknoten für die übrigen Knoten „aussen“ liegen. Zuerst sucht das Netzwerk nach induzierten Subgraphen, für die topologische Aussagen beweisbar sind. Eine solche Struktur, eine *5-Flower*, ist in Abbildung 4 dargestellt.

Man kann zeigen, dass in jeder d -QUDG-Einbettung dieses Graphen der Knoten f_0 im Inneren des durch die Aussenknoten $(f_{3,1}, W_1, f_{3,2}, \dots, W_5)$ definierten Polygons liegt. Damit liefert eine erkannte Flower einen lokalen Randbegriff, da die Aussenknoten den Rand des Subgraphen bilden. Die Erkennung von Flowers lässt sich effektiv durchführen, da die gesamte Struktur in der 5-Nachbarschaft von f_0 liegt, und somit jeder Knoten selbständig überprüfen kann, ob er der zentrale Knoten einer Flower ist.

Nachdem die Suche nach Flowers abgeschlossen ist, werden sie zu lokalen berandeten Zonen. Jeder Knoten speichert, ob er Rand einer Zone, innerhalb einer Zone oder noch unerschlossen ist. Im nächsten Schritt wird iterativ versucht, neues Gebiet zu erschliessen. Dazu suchen die Randknoten mittels einer verteilten DFS-Variante einen Kreis im

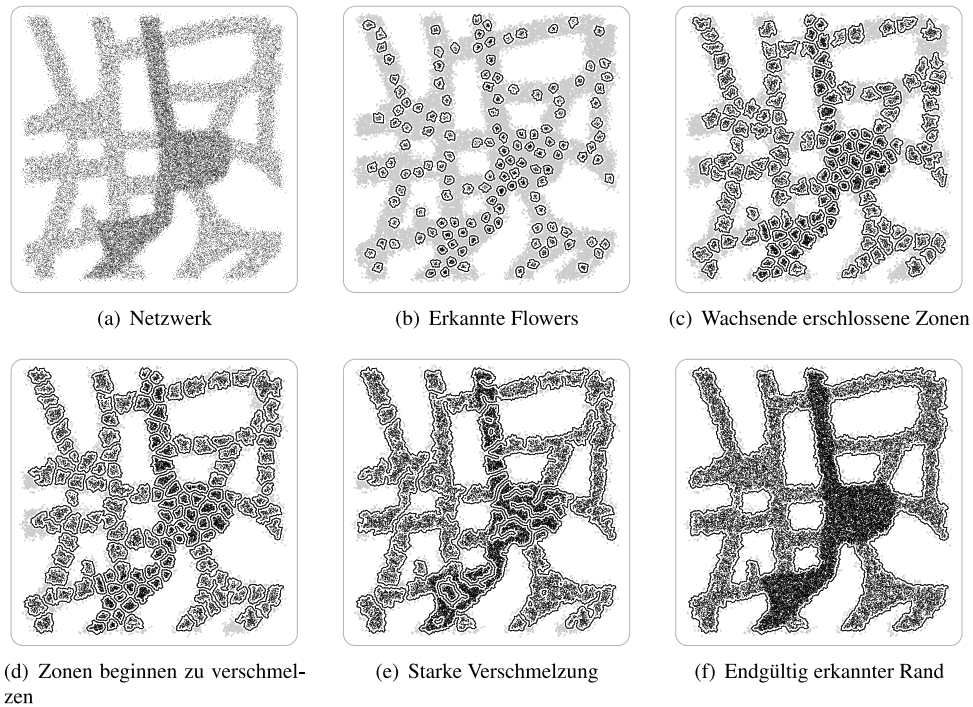


Abbildung 5: Zwischenschritte und Ergebnis der Randerkennung.

Graphen, der eine Zone erweitert oder mehrere Zonen verbindet. Dabei wird jedesmal die Anzahl unerschlossener Knoten verringert. Wenn keine Erweiterungen mehr möglich sind, endet die erste Phase.

Eine entscheidende Eigenschaft dieses Verfahrens besteht darin, dass die identifizierten Ränder beweisbar den Rand der erschlossenen Knoten darstellen. In Netzwerken mit Bereichen geringer Dichte ist ein Rand nicht klar definierbar. Dies stellt für den Algorithmus jedoch keine Hürde dar, da er einen möglichst grossen Teil des Netzwerks findet, für den er beweisbar den Rand identifizieren kann. Diese Fähigkeit unterscheidet ihn von neueren Verfahren, die durch unsere Arbeiten motiviert wurden [FK06b, WGM06], da diese in ungünstigen Netzwerken fehlerhafte Resultate produzieren.

Während wir die Korrektheit des Ergebnisses beweisen, lässt sich keine Aussage über die Grösse des erschlossenen Gebiets machen. Wir griffen auf Simulationen zurück, um Verhalten und Stabilität des Verfahrens zu untersuchen. Es zeigte sich, dass es in Netzwerken ab einem durchschnittlichen Knotengrad von ca. 12 stabil funktioniert. Ein Beispiel ist in Abbildung 5 dargestellt. Das Netzwerk in 5(a) besteht aus 50 000 Knoten. Es hat zwei Bereiche mit verschiedenen Dichten (20 bzw. 30) und als zusätzliche Schwierigkeit unscharfe Ränder. Es ist klar zu erkennen, dass am Ende nahezu die ganze Fläche erschlossen und der Rand exakt erkannt ist.

2.2 Phase 2: Clusterbildung

In der zweiten Phase wird der erkannte Rand verwendet, um Cluster zu bilden und einen Graphen zu konstruieren, der die Topologie des Netzwerks widerspiegelt. Dazu greifen wir auf eine rein geometrische Konstruktion zurück, die ein Gebiet $A \subset \mathbb{R}^2$ zerlegt. Wir beweisen Korrektheit und geometrische Eigenschaften. Für das Sensornetzwerk finden wir diskrete Entsprechungen und zeigen mittels Simulationen, dass die erzeugten Cluster der geometrischen Erwartung entsprechen.

Unser Verfahren basiert auf der Medial Axis, die ein geometrisches Gebiet in einen geometrischen Graphen überführt [CCM97]. Jeder inklusionsmaximale Kreis in A steuert seinen Mittelpunkt zur Medial Axis $MA(A)$ bei. Dabei entstehen geometrische Kanten, deren Punkte zu Kreisen gehören, die genau zwei Randstücke berühren. Die Endpunkte der Kanten liegen an Knoten, die entweder genau eine oder mindestens drei Randberührungen aufweisen. Dabei verlaufen die Kanten genau in der Mitte strassenförmiger Teile von A , und die Knoten mit mindestens drei Berührungen liegen auf den Kreuzungen. Wir definieren eine Zerlegung von A , bei der jedem Knoten von $MA(A)$ ein konvexer Kreuzungsknoten zugeordnet wird. Die verbleibenden Zusammenhangskomponenten von A sind dann tunnelförmige Gebiete mit höchstens zwei Randsegmenten, die entweder zwei Kreuzungsknoten verbinden, oder an genau eine Kreuzung anschliessen. Diese werden zu Strassenclustern. Schliesslich definieren wir einen Clustergraphen, dessen Knoten den Clustern entsprechen und dessen Adjazenzen den Berührungen von Clustern entsprechen.

Im diskreten Fall identifizieren wir Knoten, deren kürzester Abstand zum Rand im Kommunikationsgraphen von mehreren voneinander entfernten Randknoten angenommen wird. Diese Knoten entsprechen der Medial Axis, siehe Abbildung 6(a). Danach werden Kreuzungsknoten und Strassencluster in einer diskreten Entsprechung zum geometrischen Fall konstruiert (6(b)). Schliesslich speichert jeder Knoten den Clustergraphen durch ein Broadcasting der Clusternachbarschaften.

Der Clustergraph bildet das neue Lokationsbewusstsein, da er genau die Topologie des Netzes abbildet und jeder Knoten über seine eigene Clustermitgliedschaft weiss, wo im Netzwerk er sich befindet. Zudem ist der Graph so klein, dass er in jedem Knoten gespeichert werden kann und damit globales Wissen überall lokal verfügbar macht. Man kann den Graphen danach verwenden, um beispielsweise lokal Routingpfade festzulegen, die die globale Topologie berücksichtigen. Solche Pfade werden im Clustergraph beschrieben, das lokale Routing innerhalb der Cluster lässt sich aufgrund des bekannten Aufbaus trivial mit lokalen Entscheidungen umsetzen.

3 Dynamische Netzwerkflüsse

Im zweiten Teil analysieren wir ein dynamisches Flussproblem, das *Energy-Constrained Dynamic Flow Problem* (ECDP). Es motiviert sich darin, die Kanten des Clustergraphen mit Gewichten anzureichern, wie etwa Kapazität, Restenergie oder Latenzzeit. Gleichzeitig ist es aber ein zuvor nicht untersuchtes Problem, das bei Sensornetzen natürlich auftritt

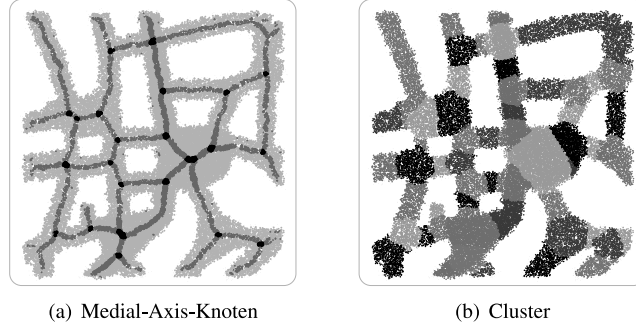


Abbildung 6: Ergebnisse der zweiten Phase.

und daher auch als isoliertes Problem relevant ist.

Wir gehen von einem ungerichteten Graphen $G = (V, E)$ aus, mit einer Quelle s und einer Senke t . Jede Kante e hat eine Transitzeit $\tau_e \in \mathbb{N}$ und eine Kapazität $u_e \geq 0$. Die Transitzeit gibt dabei an, wie lange nach dem Senden von Daten über e diese vom Zielknoten weitergesendet werden können. Für Sensornetze gilt typischerweise $\tau_e = 1 \forall e \in E$, diesen Fall bezeichnen wir als 1-ECDF-Problem, im Gegensatz zu τ -ECDF, worin τ frei ist.

Weiterhin hat jeder Knoten v eine Batterie mit Restenergie $C_v \in \mathbb{R}$. Jedesmal, wenn ein Datum der Grösse 1 über einen Pfad $P \ni v$ geschickt wird, sinkt C_v um $c_{v,P}^*$. Dadurch wird der Energieverbrauch für das Empfangen und Weitersenden modelliert. Wir gehen davon aus, dass diese Kosten kantenabhängig sind, das Empfangen von Daten über e kostet c_e^r , das Senden c_e^s . Für einen inneren Knoten v_i von P ist also $c_{v_i,P}^* = c_{v_{i-1}v_i}^r + c_{v_iv_{i+1}}^s$. Ein Knoten, dessen Energie verbraucht ist, scheidet aus und kann nicht mehr kommunizieren.

Wir suchen den maximalen fraktionalen Fluss, der in einem vorgegebenen Zeithorizont T von s nach t geschickt werden kann. Formal lautet das Problem folgendermaßen:

$$\max \sum_{P \in \mathcal{P}} \sum_{\theta=0}^{T-\tau(P)} x_P(\theta) \quad (1)$$

$$\text{s.t.} \quad \sum_{\substack{P \ni e: \\ 0 \leq \theta - \tau_e(P) \leq T - \tau(P)}} x_P(\theta - \tau_e(P)) \leq u_e \quad \forall e \in E, \theta = 0, \dots, T \quad (2)$$

$$\sum_{P \ni v} \sum_{\theta=0}^{T-\tau(P)} c_{v,P}^* x_P(\theta) \leq C_v \quad \forall v \in V \quad (3)$$

$$x_P(\theta) \geq 0 \quad \forall P \in \mathcal{P}, \theta = 0, \dots, T - \tau(P) \quad (4)$$

Darin bezeichnet $\mathcal{P}$ die Menge aller s - t -Pfade in G , $\tau(P)$ die Gesamttransitzeit eines Pfads, und $\tau_e(P)$ die Zeit, nach der ein Fluß über P die Kante $e \in P$ erreicht. Die Variable $x_P(\theta)$ modelliert die Flußmenge, die zum Zeitpunkt θ bei s startet und entlang P geroutet wird.

Ohne die Batteriebeschränkungen (3) entspricht dieses Problem dem *Maximum Dynamic Flow*, für das seit langer Zeit polynomielle Algorithmen bekannt sind [FF58].

Wir analysieren zunächst die Komplexität des Problems und zeigen, dass τ -ECDF und die ganzzahlige Variante von 1-ECDF NP-schwer sind. Desweiteren beweisen wir, dass es Instanzen für τ -ECDF gibt, in denen eine Optimallösung immer exponentiell viele Routingpfade benutzt. Ein kantenbasiertes, polynomielles Kodierungsschema für Lösungen ist nicht bekannt.

Im nächsten Schritt untersuchen wir *temporally repeated flows*, also Lösungen, bei denen ein Pfad nur einen Flußwert hat, den er zu jedem zulässigen Zeitpunkt verwendet. Wir zeigen, dass sich solche Lösungen in polynomieller Zeit finden lassen und beschränken den Gap zu 1-ECDF. Dies führt zu einem zentralisierten FPAS. Da es auf der Ellipsoidmethode basiert, ist es nicht direkt verteilt implementierbar. Dafür muss eine weitere Approximation eingesetzt werden. Sowohl die Formulierung (1)–(4) als auch die entsprechende Formulierung für temporally repeated flows sind fraktionale Packungsprobleme, die mit dem Algorithmus von Garg und Könemann [GK98] approximiert werden können. Die dafür nötige duale Separation reduziert sich dabei auf eine kürzeste-Wege-Suche in G mit dualen Gewichten als Kosten, die verteilt schnell lösbar ist. Insgesamt ergibt sich dadurch ein verteiltes FPAS für 1-ECDF.

4 Simulation

Um verteilte Algorithmen simulieren zu können, benötigt man einen passenden Simulator. Für Sensornetze gab es vormals ausschließlich zwei Typen: Die einen konzentrieren sich auf die möglichst exakte Simulation physikalischer Effekte und sämtlicher Protokollschichten. Diese sind prinzipbedingt sehr langsam und nicht in der Lage, grosse Netzwerke wie das in Abbildung 5(a) dargestellte in akzeptabler Zeit zu simulieren. Die anderen sind im Wesentlichen Graphenbibliotheken mit einfachen Kommunikationsmodulen, ohne die Möglichkeit, Effekte der Realität abzubilden.

Shawn¹ verbindet beide Arten. Wir entwickelten Shawn mit zwei Zielsetzungen:

1. Es werden die Effekte der Realität simuliert, nicht die Realität selbst.
2. Jeder Teil der Simulation lässt sich deaktivieren oder durch ein triviales Modell ersetzen, um Zeit zu gewinnen.

Shawn verwendet für jeden relevanten Teil der Simulation objektorientierte Interfaces, die von jeweils verschieden detaillierten Klassen implementiert werden. Dadurch kann der Benutzer festlegen, welcher Teil der Simulation gerade wichtig ist und alle anderen Teile durch triviale, aber schnelle Implementierungen ersetzen. Gleichzeitig unterstützt Shawn die Entwicklung von Algorithmen in jedem Schritt. Anfänglich kann man zentralisierte Algorithmen entwickeln, die Shawn als Graphenbibliothek verwenden. Da man in

¹<http://shawn.sourceforge.net>

der Simulation jederzeit zwischen zentralisiertem und verteiltem Modus wechseln kann, kann man seinen Algorithmus schrittweise in verteilte Algorithmen auf erst hohem, später niedrigem Abstraktionsgrad umsetzen und schließlich ein vollständig verteiltes Protokoll erhalten.

Shawn erfüllt die Anforderungen theoretischer wie praktischer Forschung gleichermaßen. Dies lässt sich damit belegen, dass die beiden EU-Forschungsprojekte FRONTS und WISEBED Shawn hauptsächlich einsetzen werden, sowie damit, dass die Coalesenses GmbH Sensorknoten und zugehörige APIs entwickelt, bei denen dieselbe Software in Shawn simuliert als auch in tatsächlicher Hardware ausgeführt werden kann.

5 Conclusion

Sensornetze sind ein hochaktuelles Forschungsgebiet, in dem noch viele Fragen offen sind. Wir haben gezeigt, wie das vermeintlich schwere Lokalisierungsproblem dadurch umgangen werden kann, dass man einen vollständig neuen Begriff von Lokationsbewusstsein einführt. Danach legten wir dar, wie die Berücksichtigung der in Sensornetzen unumgänglichen Energiebeschränkungen aus einem komplexitätstheoretisch einfachen Flußproblem ein NP-schweres mit einigen beantworteten und vielen offenen Fragen macht. Zuletzt diskutierten wir den Simulator Shawn, der theoretische und praktische Anforderungen verbindet und dadurch eine grosse Zahl von Anwendungen ermöglicht, und beispielsweise nicht die Algorithmen des ersten Teils simulierte, sondern auch visualisierte und damit das Video [FK06a] ermöglichte.

Literatur

- [AGY04] J. Aspnes, D.K. Goldenberg und Y.R. Yang. On the computational complexity of sensor network localization. In *Proc. 1st Intl. Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS'04)*, Seiten 32–44, 2004.
- [BFN01] L. Barrière, P. Fraigniaud und L. Narayanan. Robust position-based routing in wireless Ad Hoc networks with unstable transmission ranges. In *Proc. 5th Intl. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (Dial-M'01)*, Seiten 19–27. ACM Press, 2001.
- [BGJ05] J. Bruck, J. Gao und A. Jiang. Localization and routing in sensor networks by local angle information. In *Proc. 6th ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'05)*, Seiten 181–192. ACM Press, 2005.
- [BGMS06] A. Basu, J. Gao, J.S.B. Mitchell und G. Sabhnani. Distributed localization using noisy distance and angle information. In *Proc. 7th ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'06)*, Seiten 262–273. ACM Press, 2006.
- [CCM97] H.I. Choi, S.W. Choi und H.P. Moon. Mathematical theory of medial axis transform. *Pacific Journal of Mathematics*, 181(1):57–88, 1997.

- [FF58] L.R. Ford und D.R. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6:419–433, 1958.
- [FK06a] S.P. Fekete und A. Kröller. Geometry-based reasoning for a large sensor network. In *Proc. 22th Annual Symposium on Computational Geometry (SCG'06)*, Seiten 475–476, 2006. Video available at <http://compgeom.poly.edu/acmvideos/socg06video/index.html>.
- [FK06b] S. Funke und Ch. Klein. Hole detection or: how much geometry hides in connectivity? In *Proc. 22th Annual Symposium on Computational Geometry (SCG'06)*, Seiten 377–385, 2006.
- [FKB⁺05] S.P. Fekete, A. Kröller, C. Buschmann, S. Fischer und D. Pfisterer. Koordinatenfreies Lokationsbewusstsein. *it - Information Technology*, 47:70–78, 2005.
- [GK98] N. Garg und J. Könemann. Faster and Simpler Algorithms for Multicommodity Flow and other Fractional Packing Problems. In *Proc. 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, Seite 300. IEEE Computer Society Press, 1998.
- [KKP99] J.M. Kahn, R.H. Katz und K.S.J. Pister. Next century challenges: mobile networking for “Smart Dust”. In *Proc. 5th Annual Intl. Conference on Mobile Computing and Networking (MOBICOM'99)*, Seiten 271–278. ACM Press, 1999.
- [WGM06] Y. Wang, J. Gao und J.S.B. Mitchell. Boundary recognition in sensor networks by topological methods. In *Proc. 12th Annual Intl. Conference on Mobile Computing and Networking (MOBICOM'06)*, Seiten 122–133. ACM Press, 2006.



Alexander Kröller studierte bis 2003 Mathematik an der TU Berlin. Seine Diplomarbeit schrieb er unter Prof. Dr. Martin Grötschel am Zuse-Institut Berlin und bei der Atesio GmbH. Darin untersuchte er Netzwerkplanungsprobleme für Mobilfunkkommunikationsnetze. Von 2003 bis 2007 war er Doktorand bei Prof. Dr. Sándor Fekete an der TU Braunschweig, zunächst am Institut für Mathematische Optimierung, später am Institut für Betriebssysteme und Rechnerverbund. Seine im Rahmen des DFG-Schwerpunktprogramms 1126 „Algorithmik großer und komplexer Netzwerke“ geförderte Forschung beschäftigte sich hauptsächlich mit Sensornetzen und verteilten Algorithmen.