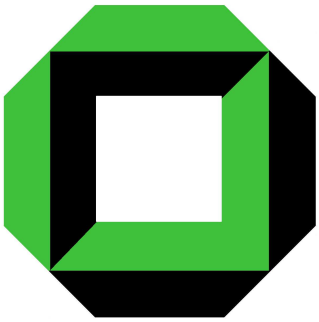
Seminar
Algorithmen für Sensornetzwerke

Clustering in Sensornetzwerken

Marcus Krug



Universität Karlsruhe (TH)
Fakultät für Informatik
Institut für Logik, Komplexität und Deduktionssysteme

Prof. Dr. D. Wagner
Steffen Mecke
Frank Schulz

Wintersemester 2004/2005

Inhaltsverzeichnis

1 Einführung

Die Möglichkeit, immer kleinere und kompaktere Rechner zu konstruieren, hat eine Reihe von neuen Perspektiven eröffnet und eine neue Generation von Rechnern in Aussicht gestellt, deren Stärke nicht länger auf riesigen Speichern und schnellen Prozessoren basiert, sondern vielmehr auf Omnipräsenz und Selbstorganisation. Dabei sollen die gewünschten Dienste nicht von einem einzelnen Rechner geleistet werden, sondern von einer Vielzahl von kleinen und kleinsten Rechnern, die hierzu in die Lage versetzt werden müssen, miteinander kommunizieren und sich organisieren zu können.

Sensornetzwerke bestehen meist aus einer Vielzahl sehr kleiner und nicht sonderlich leistungsfähiger Sensoren, deren Zweck eine möglichst umfangreiche Messung von Daten auf breiter Basis ist, die bisweilen auch deren Vorverarbeitung einschließen kann. Die große Anzahl der Sensoren, die in einem Sensornetzwerk zum Einsatz kommen, sowie deren geringe Größe erlegen den Sensoren eine Reihe von Restriktionen auf. Zum einen werden die Sensoren häufig mit Batterien betrieben, deren Lebensdauer aufgrund ihrer geringen Größe nur sehr kurz sein kann. Zum anderen will man die Kosten für Herstellung und Betrieb der Sensoren möglichst gering halten.

Durch die Fähigkeit zur Kommunikation können die Sensoren große Neztwerke bilden und somit komplexe Vorgänge überwachen.

Prinzipiell gibt es zwei verschiedene Formen der Kommunikation zwischen den Sensoren:

In einem *Single-Hop-Netzwerk* ist es zwei beliebigen Knoten auch über weite Distanzen möglich direkt miteinander zu kommunizieren. Auf diese Art und Weise können die Sensoren schnell und effizient kommunizieren. Der Preis für diesen Vorteil allerdings ist sehr hoch: Die Sensoren müssen mit leistungsstarken Kommunikationsvorrichtungen ausgestattet werden und über komplexe Mechanismen zur Erkennung und Behebung von (Nachrichten-)Kollisionen verfügen. Darüber hinaus schlägt sich der verhältnismäßig hohe Energieverbrauch für Broadcasting in einer kürzeren Lebenszeit des Netzwerkes nieder.

In einem *Multi-Hop-Netzwerk* hingegen können nur nahe beieinander liegende Sensoren direkt miteinander kommunizieren. Die Kommunikation weiter entfernter Knoten erfolgt mittelbar, indem die Nachrichten vermittels Routing von Sensor zu Sensor geschickt werden. Der Aufwand für das Routing hängt von der Größe und der Art des Neztwerkes ab, und schließlich davon, wieviel Information über die Topologie des Netzwerkes in den Sensoren gespeichert werden kann, beziehungsweise davon, ob es sich bei dem Netzwerk um ein statisches oder ein dynamisches handelt. Viele Sensornetzwerke jedoch sind Multi-Hop-Netzwerke.

Der Initialisierung von Sensornetzwerken wurde in vergangenen Arbeiten offenbar nur wenig Aufmerksamkeit geschenkt. Viele Autoren, die sich mit Algorithmen für Sensornetzwerke beschäftigt haben, setzen existierende Kommunikations-Strukturen voraus.

Häufig allerdings gibt es unmittelbar nach dem 'Big Bang' oder der Inbetriebnahme eines Sensornetzwerkes noch keine Struktur, welche Kommunikation überhaupt erst ermöglichen würde. Stattdessen herrscht oft ein quasi-chaotischer Zustand: Die Sensoren sind nicht selten im Ungewissen über ihre Position im Netzwerk, die Anzahl ihrer Nachbarn oder die Verteilung der Sensoren. Darüber hinaus wird die Verständigung der Sensoren untereinander erschwert, wenn die Sensoren asynchron aufwachen. Man stelle sich etwa ein Sensornetzwerk vor, dessen Sensoren aus einem Flugzeug über einem großen Gebiet abgeworfen werden, um geographische Daten zu sammeln. In diesem Falle ist es unmöglich, Position, Anzahl der Nachbarn, aber auch die Gesamtzahl der Sensoren im Netzwerk vorherzubestimmen.

Eine der ersten Aufgaben, die in einem solchen Sensornetz daher zu bewältigen ist, besteht in der Etablierung von Strukturen, die eine Kommunikation zwischen den Sensoren ermöglicht. Solche initialen Strukturen werden auch als Media-Access-Control(MAC)-Layer bezeichnet. Sie dienen gewissermassen als Schnittstelle für komplexere Kommunikations-Algorithmen und -protokolle. Sinnvollerweise sollte der Aufbau dieser Strukturen selbsorganisiert und verteilt erfolgen. Da die Sensoren nur eine begrenzte Lebensdauer haben, ist es darüber hinaus erstrebenswert, diesen Prozess so schnell und effizient wie möglich zu gestalten.

Eine Möglichkeit, ein MAC-Layer vorzubereiten, besteht darin, ein Clustering auf der Menge der Sensoren zu berechnen. Ein solches Clustering kann dann für Routing-Protokolle verwendet werden.

Im Hinblick auf tatsächliche Anwendungen stellt es eine besondere Herausforderung dar, die spezifischen, unmittelbar nach oder bei der Initialisierung eines Sensornetzwerkes herrschenden Umstände zu simulieren und bei der Berechnung zu berücksichtigen.

2 Cluster & Clustering

Ein Cluster (engl.: Traube, Haufen) ist zunächst einmal kein spezifischer Begriff der Informatik und bezeichnet eine meist chaotische oder unstrukturierte Ansammlung von gleichen oder ähnlichen Gegenständen. In der Physik etwa bezeichnet ein (Sternen-) Cluster einen Sternenhaufen, in der Musik hingegen bezeichnet der Begriff des Clusters das gleichzeitige Erklingen mehrerer benachbarter Töne als eine Spezialform des Akkords. Auch die spezifischen Eigenschaften eines Clusters auf einem Sensornetzwerk bedürfen einer speziellen Definition und sind keineswegs als allgemeingültig anzusehen. Sie stehen in einem speziellen Zusammenhang mit der Sichtweise auf die hier betrachteten Problemstellungen.

Ein Cluster auf einem Sensornetzwerk ist in dem hier verwendeten Sinne eine Ansammlung von Sensoren im Kontext der räumlichen Nähe. Jedes Cluster besitzt einen ausgezeichneten Sensor, den sogenannten *Cluster-Sensoren (Cluster-Head)*. Jedes Cluster kann auf diese Weise mit einem Cluster-Sensor identifiziert werden. Umgekehrt kann ein Sensor im räumlichen Kontext mehrerer Cluster-Sensoren liegen, so dass die Gesamtheit der Cluster auf dem Netzwerk eine nicht notwendigerweise disjunkte Überdeckung der Menge der Sensoren bildet. Assoziiert man jedoch jeden Sensor mit genau einem Cluster-Head, so erhält man eine disjunkte Überdeckung der Sensoren mit Clustern. Eine solche Überdeckung entspricht einem *Clustering* in dem hier verwendeten Sinne.

Zunächst ist ein Clustering auf einem Netzwerk also eine (Bi-)Partition der Menge der Sensoren mit einer ausgezeichneten Menge der sogenannten *Cluster-Sensoren*. In Hinblick auf hier betrachteten Anwendungen solcher Clusterings, also etwa die Etablierung von geeigneten Kommunikationsstrukturen, wird von einem Clustering verlangt, dass von je zwei benachbarten Sensoren mindestens einer in der Menge der Cluster-Sensoren liegt. Nur so kann man sicherstellen, dass jeder Sensor durch einen Cluster-Head abgedeckt wird.
Prinzipiell gibt es zwei Varianten eines solchen Clusterings. Die schwächere Variante lässt zu, dass zwei benachbarte Sensoren in der Menge der Cluster-Sensoren enthalten sind. Diese Variante könnte man als *Dominating Set Clustering* bezeichnen, da die Menge der Cluster-Sensoren einem Dominating Set auf der Menge der Sensoren entspricht.
Eine zweite, schärfere Variante fordert, dass von je zwei benachbarten Sensoren lediglich einer in der Menge der Cluster-Sensoren enthalten ist. Diese Variante entspräche einem *Indendent Set Clustering*.
Dabei stellt die Forderung nach Minimalität der Menge der Cluster-Sensoren eine für die genannten Eigenschaften sinnvolle Forderung dar. Insbesondere kann sich die Anzahl der Cluster-Sensoren maßgebend auf die Anzahl der Kollisionen auswirken, die beim Versenden von Nachrichten im Netzwerk auftreten können. Da die Cluster-Sensoren gegenüber den anderen Sensoren im Netzwerk darüber hinaus meist einen höheren Energieverbrauch haben, weil sie zusätzliche Aufgaben erfüllen müssen, beeinflusst die Anzahl der Cluster-Sensoren zudem möglicherweise in direkter Weise die Energie, die im Netzwerk verbraucht wird.

In der vorliegenden Arbeit werde ich zwei Varianten für die Berechnung eines Clusterings auf einem Sensornetzwerk vorstellen, die auf der Abstraktion des *Minimum Dominating Set Clustering (MDS Clustering)* beziehungsweise auf der des *Maximal Independent Set Clustering (MIS Clustering)* beruhen. Die vorgeschlagenen Algorithmen stammen aus der Feder der Autoren Kuhn, Moscibroda und Wattenhofer, die an der ETH Zürich auf diesem Gebiet tätig sind, und wurden in [MOBI04] sowie [MASS04] vorgestellt.

2.1 Minimum Dominating Set Clustering

Grundlage des MDS Clusterings ist das Minimum Dominating Set Problem.

Definition 2.1.1. (Minumum Dominating Set)
Sei G = (V,E) ein Graph mit Knotenmenge V und Kantenmenge E. Ein Minimum Dominating Set (MDS) ist eine minimale Teilmenge $V' \subseteq V$ mit

$$\forall u \in V - V' \; \exists v \in V' \; : \; (u, v) \in E. \tag{1}$$

Minimiert wird die Kardinalität der Menge V'.

Bemerkung 2.1.1. *Das Problem MDS kann polynomial auf eine Instanz von Minimum Set Cover reduziert werden und kann mit einer Approximationsgüte von $1 + \log |V|$ approximiert werden.*

In Analogie lässt sich dieses Problem für ein Sensornetzwerk wie folgt formulieren.

Definition 2.1.2. (Minimum Dominating Set Clustering)
Ein MDS Clustering in einem Sensornetzwerk ist eine minimale Teilmenge von Sensoren (Cluster-Sensoren), so dass jeder Sensor der nicht in der Menge der Cluster-Sensoren enthalten ist, mit (mindestens) einem Cluster-Sensor kommunizieren kann.

2.2 Maximal Independent Set Clustering

Entpsrechend bildet das Maximal Independent Set Problem die Grundlage für das MIS Clustering.

Definition 2.2.1. (Maximal Independent Set)
Sei G = (V,E) ein Graph mit Knotenmenge V und Kantenmenge E. Ein Maximal Independent Set (MIS) ist eine Teilmenge $V' \subseteq V$ mit

$$\forall u \in V - V' \; \exists v \in V' \; : \; (u, v) \in E \tag{2}$$

$$\forall u, v \in V' \; : \; (u, v) \notin E. \tag{3}$$

Ein MIS ist in dem Sinne maximal als die Hinzunahme eines weiteren Knotens zu einer Verletzung der Bedingung (3) führen würde.

Bemerkung 2.2.1. *Ein MIS ist trivialerweise ein Dominating Set, das jedoch im Allgemeinen nicht minimal im Sinne der Definition eines MDS sein muss. Allerdings kann man zeigen, dass ein beliebiges Maximal Independent Set auf einem Unit-Disk Graphen, der noch vorzustellen ist, eine $4\mathcal{O}+1$-Approximation eines Minimum Dominating Set darstellt, wobei $\mathcal{O}$ die Anzahl der Sensoren einer optimalen Lösung bezeichne.*

Definition 2.2.2. (Maximal Independent Set Clustering) *Ein MIS Clustering in einem Sensornetzwerk ist eine Teilmenge von Sensoren (Cluster-Sensoren), so dass jeder Sensor der nicht in der Menge der Cluster-Sensoren enthalten ist, mit (mindestens) einem Cluster-Sensor kommunizieren kann. Zwei Cluster-Sensoren sollen nicht miteinander kommunizieren können.*

Bemerkung 2.2.2. *Definition 2.2.2 impliziert einen Mindestabstand der Cluster-Sensoren in Abhängigkeit der Sendereichweite der verwendeten Sensoren.*

Intuitiv lässt sich vermuten, dass die Berechnung eines MIS aufgrund der Restriktion (3) schwieriger ist als die eines MDS. Tatsächlich schlägt sich diese Vermutung auch in der Konstruktion der hier beschriebenen Algorithmen zur Berechnung der beiden Varianten eines Clusterings nieder.
Bevor ich jedoch auf die Beschreibung der Algorithmen eingehen werde, seien noch einige Worte zu dem als Berechnungsgrundlage dienenden Modell gesagt.

3 Modell

Bei der Modellierung müssen eine Reihe unangehmer Details berücksichtigt werden, die eine verteilte Berechnung eines geeigneten Clusterings erschweren, jedoch in Hinblick auf ein möglichst realitätsnahes Szenario unverzichtbar erscheinen. Im Allgemeinen müssen die Sensoren bei praktischen Anwendungen einer Reihe von Minimalitätsansprüchen genügen, die nicht selten im Widerspruch zu den Voraussetzungen stehen, welche man sich bei der Initialisierung von Netzwerken wünschen würde.
Die Stärke von Sensornetzwerken liegt zu einem großen Teil in der Vielzahl der verwendeten Sensoren. Häufig wird man die Anzahl der zu verwendenden Sensoren maximieren wollen, um etwa ein besonders großes Gebiet abzudecken oder aber durch eine Verdichtung der Messpunkte ein dichteres Netz von Messdaten zu erhalten. Nicht der einzelne Sensor, sondern das Netzwerk als Ganzes bildet die Grundlage für die durchzuführenden Berechnungen. Der Ausfall eines einzelnen Sensors soll im Idealfall das Ergebnis nicht negativ beeinflussen oder leicht behebbar sein. Ein Sensor in einem Sensornetzwerk ist häufig austauschbar. Als Folge sucht man die Herstellungskosten für die Sensoren zu minimieren, so dass die Sensoren spartanisch ausgestattet werden müssen.
Da die Sensoren meist mit Batterien betrieben werden, ist man bestrebt, die Arbeitsweise der Sensoren so energie-effizient wie möglich zu gestalten. Dies kann sogar soweit gehen, dass die Sensoren den grössten Teil ihrer Lebensdauer schlafen und jeweils nur kurz aufwachen, um ihre Aufgaben zu erfüllen. Insbesondere will man häufig eine energieaufwendige Single-Hop-Struktur vermeiden und die Sendereichweite der Sensoren auf ein Minimum beschränken.
Darüber hinaus können einzelne Sensoren im Allgemeinen keine komplexen Probleme lösen und haben oft nur begrenzte Speicher. Da also zum einen die Daten im Netzwerk verteilt sind und die Leistung der Sensoren begrenzt ist, muss man zu verteilten Algorithmen übergehen, die diese Restriktionen überwinden. Die Tatsache jedoch, dass die Sensoren nicht beliebig untereinander kommunizieren können, erschwert die Berechnung mittels verteilten Algorithmen-Modellen. Die Kommunikation wird durch das in Multi-Hop-Netzwerken auftretende Phänomen des Hidden Terminal Problems weiter verschärft.
In einigen Szenarien kann man nicht vermeiden, dass die Sensoren zu unterschiedlichen Zeitpunkten aufwachen. Dies aber bedeutet, dass diese zunächst asynchron arbeiten, solange sie sich nicht synchronisieren, was die Berechnungen weiter erschwert.

3.1 Multi-Hop-Netzwerk

Da die Verwendung von Single-Hop-Strukturen also zusätzliche Ressourcen verbraucht, kommen oft sogenannte Multi-Hop-Lösungen zum Tragen. Die Sensoren solcher Netzwerke verfügen nur über eine beschränkte Sendereichweite, so dass lediglich nahe beieinander liegenden Sensoren miteinander kommunizieren. Um dennoch zu erreichen, dass zwei beliebigen Sensoren miteinander kommunizieren können, müssen die Nachrichten über Vermittlersensoren durch das Netzwerk propagiert werden (Abbildung 1).

3.2 Quasi-Unit-Disk-Graph

Das Netzwerk wird als ungerichteter Graph modelliert. Die Sensoren werden dabei durch Knoten abgebildet. Zwei Knoten sind genau dann mit einer Kante verbunden, wenn sie miteinander kommunizieren können.
Da das zugrundeliegende Netzwerk als Multi-Hop-Netzwerk angenommen wurde, und nur nahe beieinanderliegenden Knoten mit einer Kante verbunden sein können, ergeben sich für den Netzwerkgraphen einige spezielle Eigenschaften. Solche Netzwerk-Graphen können unter dem Begriff des Unit-Disk-Graphen zusammengefasst werden.
Da die Berechnung des Clusterings verteilt vonstatten gehen soll, wird genau genommen nicht ein Graph als Ganzes betrachtet, sondern ledglich die mit je einem Knoten assoziierten Subgraphen.

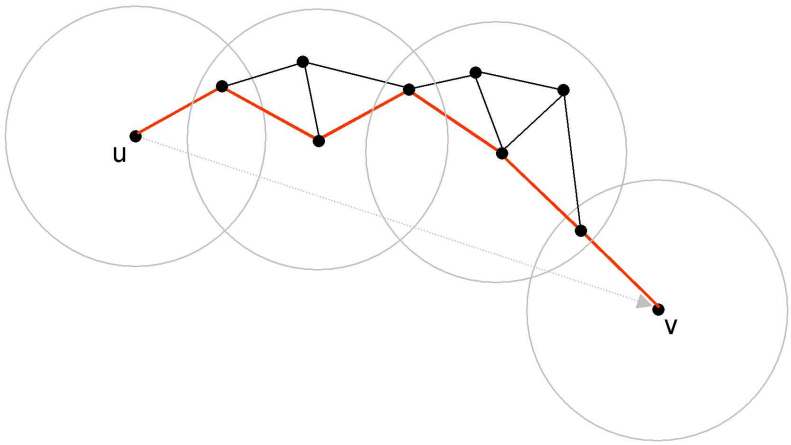


Abbildung 1: Prinzip der Nachrichtenpropagation in Multi-Hop-Netzwerken

Diese zusammenhängenden Graphen umfassen die Menge aller unmittelbaren Nachbarn eines Graphen und bilden den Kommunikationsradius des betrachteten Knotens ab.

Definition 3.2.1. (d-QUDG)
Sei $G = (V, E)$ ein Graph mit Knotenmenge V und Kantenmenge E, sowie $\delta : V \times V \rightarrow \boldsymbol{R}$ die euklidische Abstandsfunktion, die jedem Paar von Knoten seinen euklidischen Abstand zuordnet. Der Graph G heißt Quasi-Unit-Disk-Graph (d-QUDG), wenn

- $uv \in E \Rightarrow \delta(u, v) \leq 1$ *und*
- $\delta(u, v) \leq d \Rightarrow uv \in E$

für $u, v \in V$ gelten.

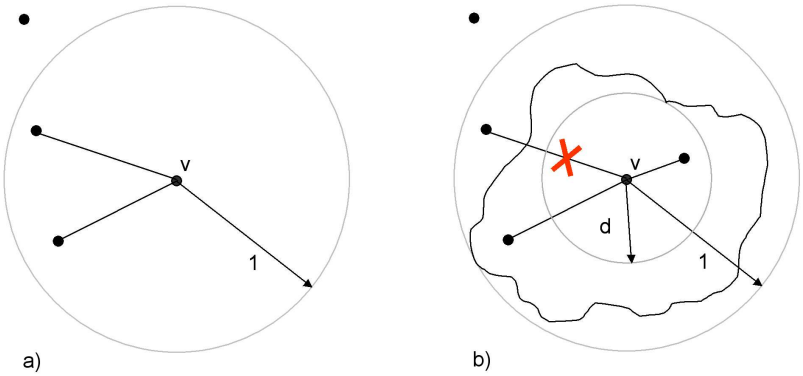


Abbildung 2: a) Unit Disk Graph (1-QUDG) b) d-Quasi-Unit-Disk Graph (d-QUDG)

Die Sendereichweite der Sensoren ist bei dieser Modellierung zur Einheit normiert. Diese ideale Sendereichweite kann in der Praxis jedoch häufig nicht erreicht werden. Hindernisse und andere Störeffekte können die Reichweite der Sensoren einschränken und zu einer Verschlechterung der Kommunikation führen.
Der Parameter $d \leq 1$ modelliert die tatsächliche Reichweite eines Sensors. Im Allgemeinen wird

dieser Parameter vom betrachteten Sensor abhängen und ist unter Umständen schwer zu quantifizieren. Infolgedessen hängen die im Abschnitt Approximation erläuterten Resultate von diesem Parameter ab.

3.3 Kollisions-Erkennung

Für die Modellierung wird angenommen, dass die Sensoren nicht über Mechanismen zur Erkennung von Kollisionen verfügen. Im Falle, dass sich mehrere sendende Sensoren im Empfangsbereich eines empfangsbereiten Sensors befinden, geht die Nachricht schlichtweg verloren. Darüber hinaus verfügt ein sendender Sensor über keinerlei Wissen darüber, ob die gesendeten Pakete ihr Ziel erreicht haben.

3.4 Asynchronität

Weiterhin wird angenommen, dass die Sensoren asynchron aufwachen können und keinen Zugang zu einer globalen Clock haben. Dies hat zur Folge, dass früh aufwachende Sensoren mit einer dünneren Kommunikations-Struktur auskommen müssen. Wenn zu einem frühen Zeitpunkt entschieden werden soll, ob ein bestimmter Sensor zu einem Cluster gehört, sind möglicherweise noch nicht alle Sensoren aufgewacht und die Entscheidung kann nicht unter optimalen Bedingungen getroffen werden.

3.5 Sendekanäle

Die Sensoren verfügen über drei unabhängige Sendekanäle Γ_1, Γ_2 und Γ_3. Dies ist keine echte Einschränkung des Modells, da man zeigen kann, dass eine vergleichbare Approximations-Güte auch mit einem Kanal in polylogarithmischer Zeit erreicht werden kann. Um die drei unabhängigen Kanäle zu simulieren, könnte man etwa ein sogenanntes FDMA (frequency division multiple access) Schema nutzen. Die Verwendung von 3 unabhängigen Kanälen hingegen verbessert die Übersichtlichkeit und Lesbarkeit des Algorithmus und vereinfacht die Analyse.

4 Minimum Dominating Set Clustering

4.1 Algorithmus

Der QUDG Clustering Algorithmus ist ein verteilter Algorithmus, der von jedem Sensor unmittelbar nach dessen Aufwachen durchlaufen wird. Während der Berechnung entscheiden die einzelnen Sensoren probabilistisch nach bestimmten Kriterien, ob sie zur Menge der sogenannten Dominatoren gehören oder nicht.

Die Hauptidee des Algorithmus besteh darin, dass die Sensoren unmittelbar nach ihrem Aufwachen zunächst versuchen herauszufinden ob sich bereits Dominatoren in ihrer unmittelbaren Nachbarschaft befinden. Nach Ablauf einer gewissen Warte- beziehungsweise Suchzeit beginnen sie mit den Sensoren in ihrer unmittelbaren Nachbarschaft um den Dominatorenstatus zu konkurrieren. Dies geschieht, indem sie die Sendewahrscheinlichkeit auf dem Kanal Γ_1 in regelmässigen Abständen verdoppeln, was zu einer exponentiellen Erhöhung der Sendewahrscheinlichkeiten führt. Da die Knoten asynchron aufwachen können, ist dies notwendig, um eine sublineare Laufzeit des Algorithmus garantieren zu können. Die Kanäle Γ_2 beziehungsweise Γ_3 werden dann benötigt, um sicherzustellen, dass die Anzahl neuer Dominatoren in unmittelbarer Nachbarschaft eines bestehenden Dominators hinreichend klein bleibt.

Im Folgenden seien mit n die tatsächliche Anzahl der Sensoren im Netzwerk sowie mit δ der maximale Knotengrad des Netzwerkes bezeichnet. Der Algorithmus erwartet als Parameter die beiden Werte N und Δ, die jeweils eine Approximation für die Werte n beziehungsweise δ darstellen.

Für den Fall, dass kein Wert für Δ angegeben wurde, nimmt der Algorithmus $\Delta = N$ an. Der Parameter N hingegen kann nicht weggelassen werden. Er ist notwendig, um die Laufzeit des Algorithmus zu begrenzen.
Wie sich eine Abweichung der Werte N und Δ von den Werten n und δ auswirkt, soll zu einem späteren Zeitpunkt geklärt werden.

Jeder Sensor führt unmittelbar nach seinem Aufwachen den QUDG Clustering Algorithmus durch, der im Folgenden vorgestellt wird.

QUDG Clustering Algorithm

```
decided ← false
dominator ← false
upon wake-up do
1   if Δ not given as input then
2         Δ ← N;
3   fi
4   for s ← 1 to α · ⌈log² N/(d² log log N)⌉ do
5         if message received then
6           decided ← true;
7         fi
8   od
9   for r ← 0 to ⌈log Δ⌉ do
10        for s ← 1 to α · ⌈log N/d²⌉ do
11          γ1 ← 0; γ2 ← 0; γ3 ← 0;
12          if not decided then
13            γ1 ← 1 w/ probability p ← ηd²2^(−⌈log Δ⌉+r);
14            if γ1 = 1 then
15              dominator ← true;
16            else if message received then
17              decided ← true;
18            fi
19          fi
20          if dominator then
21            γ2 ← 1 w/ probability ηd² log log N/logN;
22            γ3 ← 1 w/ probability ηd² log log N/log²N;
23          fi
24          for c ← 1 to 3 do
25            if γc = 1 then
26              send on channel Γc
27            fi
28          od
29        od
30  od
31  if not decided then
32        dominator ← true;
33        decided ← true;
34  fi
35  if dominator then
36  loop
37        send on Γ2 w/ probability ηd² log log N/logN;
38        send on Γ3 w/ probability ηd² log log N/log²N;
39  end loop
40  fi
```

Der Algorithmus kann – abgesehen von der Initialisierung der booleschen Variablen – in drei Phasen unterteilt werden.

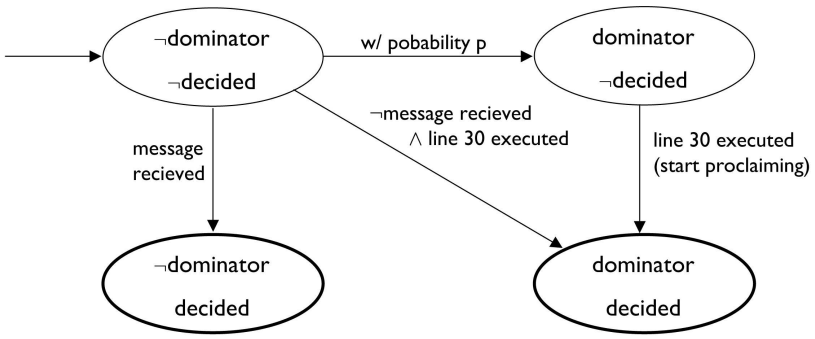


Abbildung 3: Zustandsübergangs-Diagramm des QUDG Clustering Algorithm

In der ersten Phase (Lausch-Phase: Zeilen 1 bis 8) befinden sich die Sensoren in einer Art Lausch-Zustand ($\neg dominator, \neg decided$). Dabei versuchen sie auf allen drei Kommunikationskanälen Nachrichten von den Sensoren in ihrer unmittelbaren Umgebung zu empfangen. Dies dauert genau $\alpha \cdot \lceil \log^2 N/(d^2 \log\log N) \rceil$ Zeiteinheiten. Eine Zeiteinheit entspricht dabei in etwa der Zeit, die ein Sensor benötigt, um ein Nachrichtenpacket von festgelegter Größe zu versenden. Gelingt es einem Sensor, in dieser Phase eine Nachricht zu empfangen, so ist sichergestellt, dass es in seiner unmittelbaren Nachbarschaft (Sendereichweite) bereits einen Dominator gibt. Ein solcher Sensor muss daher nicht mehr um den Dominatorenstatus konkurrieren und kann direkt in den Zustand ($\neg dominator, decided$) übergehen.

Die zweite Phase (Konkurrenzphase: Zeilen 9 bis 30) besteht aus zwei geschachtelten For-Schleifen. Die innere For-Schleife (Zeile 10) dauert $\varphi := \alpha \cdot \lceil \log N/d^2 \rceil$ Zeiteinheiten. Die äußere For-Schleife (Zeile 9) wird $\lceil \log \Delta \rceil + 1$ mal durchlaufen, wobei je ein Durchlauf der äußeren Schleife zur Vereinfachung der Terminologie als *Runde* bezeichnet wird. Eine Runde umfasst also genau φ Zeiteinheiten. Zu Beginn einer Runde entscheidet sich ein Sensor mit der Wahrscheinlichkeit $p = \eta d^2 2^{-\lceil \log \Delta \rceil + r}$ auf dem Kanal Γ_1 zu senden, wobei r der Laufindex für die Anzahl der Runden ist, die ein Sensor durchlaufen muss. Diese Wahrscheinlichkeit beträgt in der ersten Runde $\eta d^2 2^{-\lceil \log \Delta \rceil}$ und wird in jeder folgenden Runde verdoppelt, bis sie schließlich ηd^2 beträgt. Nachdem ein Sensor zum ersten Mal eine Nachricht auf dem Kanal Γ_1 verschickt hat, wird er automatisch zu einem Dominator und geht in den Zustand ($dominator, \neg decided$) über. Der Dominator-Status bleibt dem Knoten erhalten und kann nicht wieder annulliert werden. Ist ein Sensor erst einmal ein Dominator, beginnt er damit auf den Kanälen Γ_2 und Γ_3 mit den Wahrscheinlichkeiten $\eta d^2 \log\log N/\log N$ beziehungsweise $\eta d^2 \log\log N/\log^2 N$ zu senden. Bei Eintritt in die dritte Phase, also nach Ausführung von Zeile 30 des Algorithmus, gehen diese Sensoren in den Zustand ($dominator, decided$) über.

In einer dritten Phase werden vor allem jene Sensoren erfasst, die während der ersten beiden Phasen des Algorithmus weder selbst eine Nachricht gesendet haben - und mithin auch keine Dominatoren sind - noch eine Nachricht von einem Nachbarknoten empfangen haben - etwa weil es keinen Dominator in ihrer Nachbarschaft gibt oder weil ein erfolgreiches Empfangen von solchen Nachrichten durch das Eintreten von Kollisionen verhindert wurde. Solche Sensoren müssen Dominatoren sein, da andernfalls nicht sichergestellt werden kann, dass der Algorithmus ein korrektes Dominating Set berechnet. Sie gehen direkt aus dem Zustand ($\neg dominator, \neg decided$) in den Zustand ($dominator, decided$) über (Abbildung 3).

Von entscheidender Bedeutung für den Erfolg der Berechnung des Algorithmus ist die Summe der Sendewahrscheinlichkeiten auf dem Kanal Γ_1. Ist die Summe der Sendewahrscheinlichkeiten zu niedrig, so machen sich die bestehenden Dominatoren in ihrer Nachbarschaft nicht hinreichend bemerkbar und es kommt zu einer vermehrten Bildung von Dominatoren.

Ist die Summe der Sendewahrscheinlichkeiten zu hoch, kommt es hingegen vermehrt zu Kollisionen, was einen Verlust der von Dominatoren gesendenten Nachrichten bedeutet. Auch dies fürt zu einer vermehrten Bildung von Dominatoren. Der Parameter η quantifiziert die Sendewahrscheinlichkeit der Sensoren.

Der Parameter α hingegen steuert die Länge der Wartephase (Phase 1) des Algorithmus. Auch diese ist von entscheidender Bedeutung für ein Gelingen der Berechnung. Ist die Wartephase am Anfang des Algorithmus zu kurz, so sinkt die Wahrscheinlichkeit, dass die Sensoren eine Nachricht empfangen können. Andererseits wirkt sich eine lange Wartezeit negativ auf die Laufzeit des Algorithmus aus.

Für den Algorithmus wurden die Parameter α und η wie folgt festgelegt:

$$\alpha := \lceil \log^{-1}(753/752) \rceil \quad \eta := 2^{-7}$$

Die Sendewahrscheinlichkeiten auf den Kanälen Γ_2 beziehungsweise Γ_3 wurden so gewählt, dass ein aufwachender Sensor, der bereits durch einen Dominator in seiner Nachbarschaft überdeckt wird, mit hoher Wahrscheinlichkeit eine Nachricht von diesem empfängt. Würde man diese Sendewahrscheinlichkeiten zu groß oder zu klein wählen, könnte die unangenehme Situation eintreten, dass beliebig viele spät aufwachende Sensoren zu Dominatoren werden können, obwohl sie bereits durch einen Dominator abgedeckt sind.

4.2 Analyse

Die Analyse des QUDG-Clustering Algorithmus gliedert sich in zwei Teile. Der erste Teil beschäftigt sich mit Korrektheit und Laufzeit des Algorithmus. In einem zweiten Teil soll gezeigt werden, dass die probabilistischen Annahmen tatsächlich zu einer konstanten Approximation eines Dominating Set führen. Dieser Teil ist erheblich aufwendiger und soll hier nur ansatzweise behandelt werden, um einen Eindruck dafür wiederzugeben, wie eine derartige Verifikation vorgenommen werden kann. Darüber hinaus bildet dieser Teil in leicht modifizierter Weise auch eine Grundlage für die entsprechenden Beweise im Falle des Maximal Independent Set Clustering Algorithmus, der noch vorzustellen ist. Für die hier nicht wiedergegebenen Beweise sei auf [MOBI04] verwiesen.

4.2.1 Korrekheit und Laufzeit

Theorem 4.2.1. *Jeder Sensor entscheidet, ob er ein Dominator ist in*

$$O\left(\frac{\log N}{d^2}\left(\log\Delta + \frac{\log N}{\log\log N}\right)\right)$$

Zeiteinheiten.

Beweis. Die Anzahl der Iterationen in der ersten For-Schleife (Zeilen 4 bis 8) beträgt $\alpha \cdot \lceil \log^2 N/(d^2 \log\log N) \rceil$.
Für die verschachtelten For-Schleifen (Zeilen 9 bis 30) kommen weitere $\lceil \log\Delta \rceil \cdot \alpha \cdot \lceil \log N/d^2 \rceil$ Zeiteinheiten hinzu.
Insgesamt ergeben sich somit

$$\alpha \cdot \left\lceil \frac{\log^2 N}{(d^2 \log\log N)} \right\rceil + \lceil \log\Delta \rceil \cdot \alpha \cdot \left\lceil \frac{\log N}{d^2} \right\rceil = O\left(\frac{\log N}{d^2}\left(\log\Delta + \frac{\log N}{\log\log N}\right)\right)$$

Der Parameter α kann entfallen, da es sich dabei um eine Konstante handelt.

□

Theorem 4.2.2. *Der Algorithmus berechnet ein korrektes Dominating Set.*

Beweis. Sei s ein Sensor, der kein Dominator ist. Nimmt man an, es gäbe keinen Dominator in unmittelbarer Umgebung von s, so hätte der Sensor s während der gesamten Ausführungszeit des Algorithmus kein Signal auf Γ_1 empfangen dürfen. Dies aber hätte zur Folge gehabt, dass die Eigenschaft *decided* von s bei Eintritt in die letzte Phase des Algorithmus (Zeile 31) den Wert **false** gehabt hätte. Somit hätte s also in Zeile 32 zu einem Dominator werden müssen, was im Widerspruch zur Voraussetzung stünde.
Jeder Sensor, der nicht Dominator ist, besitzt also einen Dominator in seiner Umgebung.

□

Bemerkung 4.2.1. *Wählt man d konstant, so lässt sich die gegebene Zeitkomplexität zu*

$$\begin{array}{lll} O\left(\frac{\log^2 N}{\log\log N}\right) & \text{für} & 1 \leq \Delta \leq N^{1/\log\log N} \quad \text{und} \\ O\left(\log N \log \Delta\right) & \text{für} & N^{1/\log\log N} \leq \Delta \leq N \end{array}$$

vereinfachen.

Bemerkung 4.2.2. *Die oberen Schranken N und Δ für n beziehungsweise δ müssen nicht besonders genau sein, um eine gute Zeitkomplexität zu erhalten. Sei $N \leq n^\lambda$ und $\Delta \leq \delta^\lambda$ für ein gegebenenes $\lambda > 1$. Dann beträgt die Zeitkomplexität lediglich*

$$O\left(\frac{\lambda^2 \log N}{d^2}\left(\log \Delta + \frac{\log N}{\log\log N}\right)\right).$$

Gibt man also einem Netzwerk mit $n = 100$ Sensoren eine obere Schranke für von $N = 10.000 = 100^2$ als Parameter, so erhöht sich die Laufzeit des Algorithmus lediglich um den Faktor $4 = 2^2$. Diese Bemerkung könnte für solche Netzwerke von Interesse sein, bei denen die Überlebenschancen der Sensoren bei der Inbetriebnahme gering sind, also etwa wenn Sensoren aus einem Flugzeug abgeworfen werden, und eine nicht unerhebliche Menge von Sensoren beim Aufprall beschädigt werden. Allerdings ist noch zu klären, wie sich ein solcher Ausfall auf die Qualität des zu berechnenden Dominating Set auswirkt.

4.2.2 Approximation

Der Beweis für die Konstantheit der Approximation des QUDG-Clustering Algorithmus ist grob in vier Hauptschritte unterteilt, die hier kurz skizziert werden sollen:

In einem ersten Schritt geht es darum, die Summe der Sendewahrscheinlichkeiten eines Kreises C_i nach oben zu beschränken. Dies ist im Wesentlichen dadurch möglich, dass die Sendewahrscheinlichkeit eines jeden Knotens im Netzwerk durch den Algorithmus beschränkt wird.

Hieraus lässt sich in einem zweiten Schritt eine probabilistische obere Schranke für die Anzahl der Kollisionen gewinnen, die bis zur Terminierung eines bestimmten Kreises auftreten. Da die Sendewahrscheinlichkeiten beschränkt sind, kann man den Erwartungswert für die Anzahl der Kollisionen nach oben durch eine Konstante abschätzen.

In einem dritten Schritt soll die Anzahl der Sensoren beschränkt werden, die bei einer Kollision senden. Auch hierbei handelt es sich wieder um ein probabilistisches Ergebnis. Da die Anzahl der Dominatoren in direkter Weise von der Anzal der Kollisionen und der Anzahl der an einer Kollision beteiligten Sensoren abhängt, kann man nun die Anzahl der Dominatoren abschätzen, die bei einer Kollision hervorgehen können.

Schließlich muss in einem vierten Schritt noch gezeigt werden, dass Sensoren, die zu einem späten Zeitpunkt aufwachen, mit hoher Wahrscheinlichkeit keine Dominatoren mehr werden können, wenn sie bereits im Sendebereich eines Dominators liegen.

Zunächst jedoch noch zwei Lemmata, die im Verlauf der Beweise häufiger Verwendung finden. Lemma 4.2.1 soll kurz erläutert werden, da es sich hierbei um ein Ergebnis handelt, welches aus der gegebenen Problematik hervorgeht, während es sich bei Lemma 4.2.2 um wohlbekannte Abschätzungen handelt.

Lemma 4.2.1. *Seien $p_1, \ldots, p_n$ Wahrscheinlichkeiten mit $p_i \in \left[0, \frac{1}{2}\right]$. Dann gilt*

$$\left(\frac{1}{4}\right)^{\sum_{k=1}^n p_k} \leq \prod_{k=1}^n (1 - p_k) \leq \left(\frac{1}{e}\right)^{\sum_{k=1}^n p_k}.$$

Beweis.

$$(1)\quad \prod_{k=1}^{n}(1-p_k)=\prod_{k=1}^{n}\left((1-p_k)^{\frac{1}{p_k}}\right)^{p_k}\geq\prod_{k=1}^{n}\left(\left(1-\frac{1}{2}\right)^2\right)^{p_k}=\prod_{k=1}^{n}\left(\frac{1}{4}\right)^{p_k}=\left(\frac{1}{4}\right)^{\sum_{k=1}^{n}p_k}$$

$$(2)\quad \prod_{k=1}^{n}(1-p_k)=\prod_{k=1}^{n}\left((1-p_k)^{\frac{1}{p_k}}\right)^{p_k}\leq\prod_{k=1}^{n}\left(\frac{1}{e}\right)^{p_k}=\left(\frac{1}{e}\right)^{\sum_{k=1}^{n}p_k}$$

□

Die maximale Sendewahrscheinlichkeit eines Sensors beträgt $\eta d^2 = d^2/2^7 \ll 1/2$. Somit kann das Lemma auf die Sendewahrscheinlichkeiten der Sensoren angewendet werden.

Lemma 4.2.2. *Für n,t mit $n \geq 1$ und $|t| \leq n$ gilt:*

$$e^t\left(1-\frac{t^2}{n}\right)\leq\left(1+\frac{t}{n}\right)^n\leq e^t.$$

Um den Algorithmus zu analysieren, bedeckt man die Ebene, in welcher die Sensoren liegen, mit imaginären Hexagonen und Kreisen C_i mit Radius $r = d/2$, welche man gemäß Abbildung 4 auf den Hexagonen plaziert. Zu jedem solchen Kreis C_i gehört ein größerer Kreis D_i mit Radius $R = 1 + d/2$. Man sieht, dass jeder Kreis D_i mehrer kleinere Kreise C_j überdeckt beziehungsweise schneidet.
Die Radien der Kreise wurden wie folgt gewählt: Je zwei Sensoren, die sich im selben Kreis C_i befinden, können mit Sicherheit miteinander kommuzieren, da ihr Abstand geringer als d ist. Andererseits kann ein Sensor, der sich ausserhalb des Kreises D_i befindet, mit Sicherheit nicht mit einem Sensor, der sich innerhalb des kleineren Kreises C_i befindet, kommunizieren, und aus ebendiesem Grunde nicht zu Kollisionen im Kreis C_i beitragen, da der Abstand zweier Sensoren mit diesen Eigenschaften größer als 1 ist.

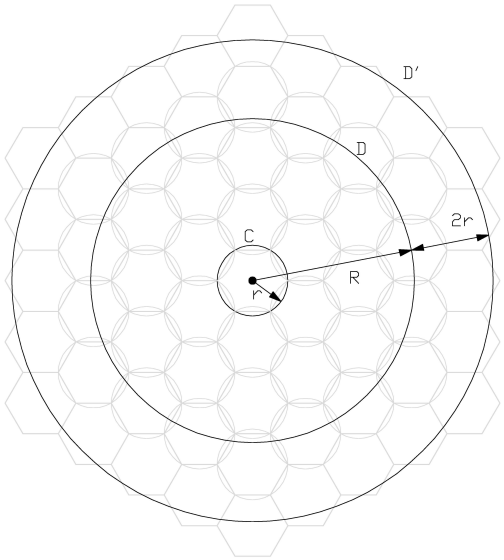


Abbildung 4: Überdeckung der Ebene mit Hexagonen und Kreisen C_i bzw. D_i

Im weiteren Verlauf der Beweise wird kontinuierlich auf indirekte Weise von der Tatsache Gebrauch gemacht wird, dass die Anzahl der Kreise C_j, die durch einen Kreis D_i überdeckt werden, begrenzt ist, falls es eine untere Schranke für den Parameter d gibt.

Lemma 4.2.3. *Der Kreis D_i bedeckt höchstens $5/d^2 + 15/d + 11$ Kreise C_i.*

Beweis. Sei χ die kleinste Anzahl von Kreisen mit Radius R, die man benötigt, um den Kreis D_i zu überdecken. Der Grenzwert für das Verhältnis der Flächen von D_i und den kleineren Kreisen ist kleiner als $\frac{3\sqrt{3}}{2\pi}$. Alle Kreise, die den Kreis D_i schneiden, sind vollkommen im Kreis D_i' mit Radius $R' := R + 2r = 1 + \frac{3d}{2}$. Folglich gilt:

$$\frac{(1 + \frac{3d}{2})}{\chi \cdot (\frac{d}{2})^2 \pi} \geq \frac{3\sqrt{3}}{2\pi}.$$

Dies ist gleichbedeutent mit:

$$\chi \leq \frac{3\sqrt{3}}{2\pi} \cdot \left(\frac{1}{d^2} + \frac{3}{d} + \frac{9}{4} \right).$$

□

In Kapitel 4.1 wurde gezeigt, dass die Qualität der oberen Schranken N und Δ für n beziehungsweise δ in direkter Weise die Laufzeit des Algorithmus beeinflusst. Das folgende Lemma zeigt, dass die Approximationsrate durch schlechte obere Schranken für n und δ hingegen nicht negativ beeinflusst wird.

Lemma 4.2.4. *Die Anzahl der zu erwartenden Dominatoren im Falle $n < N$ oder $\delta < \Delta$ ist nicht größer als im Falle $n = N$ beziehungsweise $\delta = \Delta$.*

Beweis. Annahme: Die Anzahl der Dominatoren im Falle $n_1 < N$ ist echt größer als die Anzahl der Dominatoren im Falle $n_2 = N$. Da über das Aufwachverhalten der Sensoren keine Annahmen gemacht wurden, sind die beiden Fälle nicht zu unterscheiden, wenn im zweiten Fall $n_2 - n_1$ Sensoren niemals aufwachen. Ein ähnliches Argument kann für den Fall $\delta_1 < \Delta$ beziehungsweise $\delta_2 = \Delta$ herangezogen werden.

□

Betrachten wir ein gegebenes Sensornetzwerk S mit n Sensoren. Simuliert man den Algorithmus auf S also mit $N > n$, so ist die Anzahl der Dominatoren D nicht größer als im Falle $N = n$. Folglich ist das Verhältnis D/n nicht schlechter - wenngleich die Berechnung eine längere Zeitspanne in Anspruch nimmt.

Da es im Folgenden um eine Analyse der Approximationsgüte gehen wird, seien daher $n = N$ und $\delta = \Delta$ nicht weiter unterschieden. Des Weiteren sei die Dauer, die ein Sensor für das Versenden einer Nachricht benötigt, im Folgenden als Zeitfenster oder Zeiteinheit bezeichnet.

Um eine obere Schranke für die Summe der Sendewahrscheinlichkeiten angeben zu können ist man an solchen Zeitfenstern interessiert, in denen genau ein Sensor in einer gewissen Umgebung sendet. In einem solchen Zeitfenser können alle Knoten in der unmittlebaren Nachbarschaft des sendenen Knotens die gesendete Nachricht empfangen.
Da die Sensoren zu unterschiedlichen Zeitpunkten aufwachen können, interessiert man sich nicht nur für den ersten solchen Zeitpunkt, sondern auch für alle folgenden.

Definition 4.2.1. *Betrachten wir den Kreis C_i. Sei t ein Zeitfenster, in welchem eine Nachricht von einem Knoten $v \in C_i$ auf Kanal Γ_1 gesendet und ohne Kollision von allen anderen Knoten in C_i empfangen wird. Dann klärt sich der Kreis C_i im Zeitfenster t. Sei t_0 das erste solche Zeitfenster. Dann terminiert der Kreis C_i im Zeitfenster t_0. Für alle folgenden Zeitfenster $t \geq t_0$ heißt C_i terminiert.*

Unmittelbar nachdem sich ein Kreis C_i geklärt hat, haben alle bereits erwachten Sensoren ihre endgültige Rolle im Netzwerk übernommen. Der erste Zeitpunkt, zu dem sich ein gegebener Kreis C_i klärt ist somit der erste Zeitpunkt, bei welchem alle Rollen in C_i auf wohldefinierte Weise verteilt sind. Ein solcher Kreis heißt dann terminiert. Sobald nun ein weiterer Sensor aufwacht, ist diese Eigenschaft zunächst hinfällig, bis sich der Kreis erneut klärt.

Darüber hinaus betrachtet man solche Zeitfenster, in denen die Summe der Sendewahrscheinlichkeiten einen bestimmten Grenzwert überschreitet.

Definition 4.2.2. *Sei $p_k(t)$ die Sendewahrscheinlichkeit des Knotens k auf Kanal Γ_1 im Zeitfenster t. Das Zeitfenster t_i^j sei das Zeitfenster, in welchem die Summe der Sendewahrscheinlichkeiten im Kreis C_i zum j-ten mal den Grenzwert von ηd^2 überschreitet. Dann gilt formal in jedem solchem Zeitfenster t_i^j, dass*

$$\sum_{k \in C_i} p_k(t_i^j - 1) < \eta d^2 \text{ und } \sum_{k \in C_i} p_k(t_i^j) \geq \eta d^2.$$

Der Grenzwert ηd^2 entspricht der Summe der Sendewahrscheinlichkeiten, falls alle Sensoren mit der initialen Sendewahrscheinlichkeit senden, beziehungsweise der Sendewahrscheinlichkeit, mit der ein Sensor sendet, welcher kurz davor steht die Konkurrenz-Phase des Algorithmus (zweite Phase) zu verlassen.

Lemma 4.2.5 liefert zunächst eine obere Schranke für die Summe der Sendewahrscheinlichkeiten, in einer Runde, die einem Zeitfenster folgt, in welchem der Grenzwert von ηd^2 überschritten wurde. Eine Runde entspricht dabei $\varphi = \alpha \cdot \lceil \log n/d^2 \rceil$ Zeiteinheiten beziehungsweise einem Durchlauf der äußeren For-Schleife.

Lemma 4.2.5. *Sei $\varphi = \alpha \cdot \lceil \log n/d^2 \rceil$. Dann gilt für alle Zeitfenster $t' \in \left[t_i^j, t_i^j + \varphi - 1\right]$, dass die Summe der Sendewahrscheinlichkeiten durch*

$$\sum_{k \in C_i} p_k \leq 3\eta d^2$$

beschränkt wird.

Beweis. Gemäß der Definition von t_i^j ist die Summe der Sendewahrscheinlichkeiten $\sum_{k \in C_i} p_k$ zum Zeitpunkt $t_i^j - 1$ echt kleiner als ηd^2. Alle Sensoren, die zum Zeitpunkt t_i^j aktiv sind, werden ihre Sendewahrscheinlichkeit p_k während der nächsten φ Zeiteinheiten genau einmal verdoppeln. Hinzu kommt, dass maximal δ Sensoren während dieses Zeitraumes aufwachen und mit einer initialen Sendewahrscheinlichkeit von $\eta d^2 2^{-\log \Delta} = \eta d^2/\Delta$ im gegebenen Zeitraum senden. Somit ergibt sich im Intervall $t' \in \left[t_i^j, t_i^j + \varphi - 1\right]$

$$\sum_{k \in C_i} p_k \leq 2\eta d^2 + \sum_{k=1}^{\delta} \frac{\eta d^2}{2^{\log \Delta}} \leq 2\eta d^2 + \frac{\delta \eta d^2}{\Delta} \leq 3\eta d^2$$

da $\delta/\Delta \leq 1$ gilt.

□

Nun gilt es eine obere Schranke für einen beliebigen Zeitpunkt während der Ausführung des Algorithmus anzugeben.

Lemma 4.2.6. *Die Summe der Sendewahrscheinlichkeiten aller Knoten im Kreis C_i wird durch*

$$\sum_{k \in C_i} p_k \leq 3\eta d^2$$

mit einer Wahrscheinlichkeit von mindestens $1 - o(\frac{1}{n^2})$ beschränkt. Mit einer Wahrscheinlichkeit von mindestens $1 - o(\frac{1}{n})$ ist dies eine obere Schranke für alle C_i in G.

Beweis. Sei wieder $\varphi = \alpha \cdot \lceil \log n/d^2 \rceil$. Der Beweis durchläuft per Induktion alle Zeitfenster t_i^j in aufsteigender Reihenfolge. Sei $t' := t_i^0$ das erste solche Zeitfenster im Netzwerk. Nach Lemma 4.2.5 wird die Summe der Sendewahrscheinlichkeiten im Zeitraum $[t', t' + \varphi - 1]$ durch $3\eta d^2$ nach oben beschränkt. Die Aussage des Lemmas ist sicher richtig, wenn man zeigen kann, dass sich entweder der Kreis C_i in dem gegebenen Intervall klärt oder die Summe der Sendewahrscheinlichkeiten mit hoher Wahrscheinlichkeit unter ηd^2 sinkt.

Klärt sich nämlich der Kreis, so gibt es einen Sensor in C_i der eine Nachricht sendet, welche von allen anderen Sensoren in C_i empfangen werden kann. Da etablierte Dominatoren nur auf den Kanälen Γ_2 und Γ_3 senden, und Sensoren, die keine Dominatoren sind und eine Nachricht empfangen haben, ebenfalls nicht mehr auf dem Kanal Γ_1 senden, sinkt die Summe der Sendewahrscheinlichkeiten in diesem Fall sogar kurzfristig auf Null. Im weiteren Verlauf des Zeitintervalls können maximal δ neue Sensoren aufwachen und mit einer initialen Sendewahrscheinlichkeit von insgesamt ηd^2 senden.

Wenn die Summe der Sendewahrscheinlichkeiten nicht unter ηd^2 zurückfällt, so gilt im Zeitraum $[t', t' + \varphi - 1]$:

$$\eta d^2 \leq \sum_{k \in C_i} p_k \leq 3\eta d^2 : \text{in } C_i \ (1)$$

$$0 \leq \sum_{k \in C_j} p_k \leq 3\eta d^2 : \text{in } C_j \in D_j,\ i \neq j \ (2)$$

Die zweite Ungleichung gilt, weil es sich bei t' um den ersten Zeitpunkt handelt, bei welchem in einem Kreis C_i die Summe der Sendewahrscheinlichkeiten den Grenzwert ηd^2 überschreitet. Daher ist in jedem $C_j \in D_i$ die Summe der Sendewahrscheinlichkeiten im Intervall $[t', t' + \varphi - 1]$ nicht größer als $3\eta d^2$. Andernfalls gäbe es einen Kreis C_j, der vor dem Kreis C_i den Grenzwert überschritten hätte, was im Widerspruch zur Annahme stünde.

Die Wahrscheinlichkeit P_0, dass kein Sensor in einem benachbarten Kreis $C_j \in D_i$ mit $i \neq j$ sendet, beträgt:

$$\begin{aligned}
P_0 &= \prod_{C_j \in D_i} \prod_{k \in C_j} (1 - p_k) \\
&\overset{\text{Lemma 4.2.1}}{\geq} \prod_{C_j \in D_i} \left(\frac{1}{4}\right)^{\sum_{k \in C_j} p_k} \\
&\overset{\text{Lemma 4.2.6}}{\geq} \prod_{C_j \in D_i} \left(\frac{1}{4}\right)^{3\eta d^2} \\
&\overset{\text{Lemma4.2.3}}{\geq} \left[\left(\frac{1}{4}\right)^{3\eta d^2}\right]^{\frac{5}{d^2} + \frac{15}{d} + 11} \\
&\geq \left(\frac{1}{4}\right)^{\eta(15 + 45d + 33d^2)} > \left(\frac{1}{4}\right)^{\frac{3}{4}}
\end{aligned}$$

Sei P_{suc} die Wahrscheinlichkeit, dass genau ein Sensor in C_i sendet.

$$\begin{aligned}
P_{suc} &= \sum_{k \in C_i} \left(p_k \cdot \prod_{l \in C_i, l \neq k} (1 - p_l) \right) \\
&\geq \sum_{k \in C_i} p_k \cdot \prod_{l \in C_i} (1 - p_l) \\
&\overset{\text{Lemma 4.2.1}}{\geq} \sum_{k \in C_i} p_k \cdot \left(\frac{1}{4}\right)^{\sum_{k \in C_i} p_k} \\
&\geq \eta d^2 \cdot \left(\frac{1}{4}\right)^{\eta d^2}.
\end{aligned}$$

Die Wahrscheinlichkeit P_c, dass genau ein Sensor in C_i und kein weiterer in D_i sendet, beträgt somit

$$P_c = P_0 \cdot P_{suc} \geq \eta d^2 \left(\frac{1}{4}\right)^{\eta d^2 + \frac{3}{4}}.$$

Die Wahrscheinlichkeit P_c ist eine untere Schranke für die Wahrscheinlichkeit, dass sich der Kreis C_i zu einem Zeitpunkt $t \in [t', t' + \varphi - 1]$ klärt.

Bezeichne $k_1 := \eta \left(\frac{1}{4}\right)^{\frac{3}{4}}$ und $k_2 := \left(\frac{1}{4}\right)^{\eta d^2}$. Die Wahrscheinlichkeit $\overline{P_{term}}$, dass sich der Kreis C_i während eines gesamten Intervalls der Länge φ nicht klärt, läßt sich also schreiben als

$$\begin{aligned}
\overline{P_{term}} &\leq \left(1 - \eta d^2 \left(\frac{1}{4}\right)^{\eta d^2 + \frac{3}{4}}\right)^{\alpha \cdot \lceil \log n / d^2 \rceil} \\
&\leq \left[\left(1 - \eta d^2 \left(\frac{1}{4}\right)^{\eta d^2} \cdot \left(\frac{1}{4}\right)^{\frac{3}{4}}\right)^{\frac{1}{d^2}}\right]^{\alpha \log n} \\
&= \left[\left(1 - k_1 d^2 k_2\right)^{\frac{1}{d^2}}\right]^{\alpha \log n} \\
&= \left[\left(1 + \frac{-k_1 k_2}{\frac{1}{d^2}}\right)^{\frac{1}{d^2}}\right]^{\alpha \log n} \\
&\overset{\text{Lemma 4.2.2}}{\leq} \left(e^{-k_1 k_2}\right)^{\alpha \log n} \\
&= e^{-\alpha k_1 k_2 \log n} = n^{-\alpha k_1 k_2 / \ln n} \in o(n^{-2})
\end{aligned}$$

Es kann somit gezeigt werden, dass entweder die Summe der Sendewahrscheinlichkeiten in einem Kreis C_i und einem Intervall $[t', t' + \varphi - 1]$ der Länge φ wieder unter ηd^2 zurückfällt oder der Kreis C_i sich in dem gegebenen Intervall klärt (†).
Bisher wurde jedoch lediglich gezeigt, dass dies für das erste t_i^j gilt. Per Induktion werden nun die weiteren t_i^j betrachtetet. Nach Induktionsannahme sind alle vorangegangenen Zeitfenster dieser Art bereits betrachtet und gemäß (†) eingeordnet.
Alle bereits betrachteten Zeitfenster $t_{i'}^{j'}$ haben sich also bereits geklärt oder die Summe der Sendewahrscheinlichkeiten ist unter den Grenzwert von ηd^2 zurückgefallen. Unmittelbar nach der Klärung eines Kreises C_i beträgt die Summe der Sendewahrscheinlichkeiten in diesem Kreis maximal ηd^2. Dies ist damit zu erklären, dass in diesem Fall alle Sensoren bis auf einen ihre Sende-Aktivität auf dem Kanal Γ_1 einstellen und somit nur noch ein Sensor auf Γ_1 sendet. Die Sendewahrscheinlichkeit für einen Sensor ist aber durch den Algorithmus auf ηd^2 in der letzten Runde beschränkt.
Gemäß Lemma 4.2.5 ist die Summe der Sendewahrscheinlichkeiten in den umliegenden Kreisen im Intervall $[t_i^j, t_i^j + \varphi - 1]$ durch $3\eta d^2$ beschränkt. Andernfalls wäre ein Kreis, für den diese Eigenschaft nicht zutrifft, bereits vor dem aktuell betrachteten Kreis betrachtet worden.
Die Ungleichungen (1) und (2) gelten daher mit hoher Wahrscheinlichkeit und die Berechnung für den Induktionsschritt entspricht der Berechnung für den Induktionsanfang. Somit gilt $\overline{P_{term}} \in o(\frac{1}{n^2})$ in jedem Induktionsschritt.
Da die Anzahl der Sensoren im Neztwerk und damit die Anzal der nicht leeren Kreise C_i durch n beschränkt ist, ist auch die Anzahl der Induktionsschritte durch n beschränkt.
Die Wahrscheinlichkeit, dass die Aussage des Lemmas zutrifft, beträgt somit mindestens

$$\left(1 - o\left(\frac{1}{n^2}\right)\right)^n \geq 1 - o\left(\frac{1}{n}\right).$$

□

Durch Zuhilfenahme von Lemma 4.2.6 kann man nun den Erwartungswert für die Anzahl der Dominatoren in einem Kreis C_i berechnen. Die Berechnung gliedert sich in zwei Teile.
Zunächst wird der Erwartungswert für die Anzahl der Dominatoren in einem Kreis vor seiner

Terminierung berechnet. In einem zweiten Schritt wird der Erwartungswert für die Anzahl der Dominatoren in einem Kreis nach seiner Terminierung angegeben.
Bevor jedoch diese Erwartungswerte berechnet werden können, muss man eine Aussage über die Anzahl der Kollisionen und die Anzahl der an einer Kollision beteiligten Sensoren machen.
Bei einer Kollision können sich rein theoretisch beliebig viele Dominatoren herausbilden. Nehmen wir als eine Art Worst-Case-Szenario an, alle Sensoren im Netzwerk wachten zur gleichen Zeit auf und begännen gleichzeitig mit der Ausführung des Algorithmus. Nehmen wir weiter an, dass alle Sensoren sich im ersten Durchlauf der Inneren Schleife für den Dominatoren-Status entschieden. Dann käme es bei dem Versuch auf dem Kanal Γ_1 zu senden bei jedem Sensor zu einer Kollision. An einer solchen Kollision wären jeweils alle Nachbar-Knoten des betrachteten Sensors beteiligt.
Allerdings ist dieses Szenario selbst im Fall, dass alle Sensoren zur gleichen Zeit aufwachen und zur gleichen Zeit mit der Ausführung des Algorithmus beginnen mit einer Wahrscheinlichkeit von $(\frac{\eta d^2}{\Delta})^n$ sehr unwahrscheinlich.
Tritt hingegen nie eine Kollision auf, so gibt es in jeder Umgebung nur einen Dominator, da alle Sensoren in einer betrachteten Umgebung unmittelbar nach der Herausbildung eines Dominators von dessen Existenz wissen.
Die eigentliche Herausforderung beim Entwurf des Algorithmus bestand somit in der Aufgabe, die Wahrscheinlichkeiten so zu konstruieren, dass einerseits die Sendewahrscheinlichkeiten groß genug sind um sublineare Laufzeit zu erreichen, dass andererseits aber die Summe der Sendewahrscheinlichkeiten so klein ist, dass eine zu große Anzahl von Kollisionen vermieden werden kann.

Lemma 4.2.7. *Sei C die Anzahl der Kollisionen in einem gegebenen Zeitfenster t und einem Kreis C_i (mehr als ein Knoten sendet). Der Erwartungswert für die Anzahl der Kollisionen in einem Kreis C_i vor seiner Terminierung ist $E[C] < 5$. Darüber hinaus ist $C < 6$ mit einer Wahrscheinlichkeit von mindestens $1 - o(\frac{1}{n^2})$.*

Beweis. In diesem Beweis wird lediglich der Kanal Γ_1 betrachtet. Wir nehmen an, dass der Kreis C_i noch nicht terminiert ist. Seien die Ereignisse A, X, Y, Z gegeben durch

A : Genau ein Sensor in D_i sendet.
X : Mehr als ein Sensor in C_i sendet.
Y : Mindestens ein Sensor in C_i sendet.
Z : Es gibt einen sendenden Sensor in $D_i - C_i$.

Für den Beweis werden nur solche Runden betrachtet, in denen mindestens ein Sensor in C_i sendet. (Wenn kein Sensor sendet, kann auch kein neuer Dominator hervorgehen). Von Interesse ist nun die bedingte Wahrscheinlichkeit $P[A|Y]$, dass genau ein Sensor in D_i sendet, und dieser Sensor in C_i liegt. Dabei kann benutzt werden, dass $P[Y|X] = 1$ gilt, sowie dass Y und Z unabhängige Ereignisse sind.

$$\begin{aligned}
P[A|Y] &= P[\overline{X}|Y] \cdot P[\overline{Z}|Y] \\
&= P[\overline{X}|Y] \cdot P[\overline{Z}] \\
&= (1 - P[X|Y]) \cdot (1 - P[Z]) \\
&= \left(1 - \frac{P[X] \cdot P[Y|X]}{P[Y]}\right) \cdot (1 - P[Z]) \\
&= \left(1 - \frac{P[X]}{P[Y]}\right) \cdot (1 - P[Z]).
\end{aligned}$$

Weiter ist

$$\begin{aligned} P[X] &= 1 - \prod_{k \in C_i} (1 - p_k) - \sum_{k \in C_i} \left(p_k \prod_{l \in C_i, l \neq k} (1 - p_l) \right) \\ &\leq 1 - \left(\frac{1}{4}\right)^{\sum_{k \in C_i} p_k} - \sum_{k \in C_i} p_k \cdot \left(\frac{1}{4}\right)^{\sum_{k \in C_i} p_k} \\ &= 1 - \left(1 + \sum_{k \in C_i} p_k\right) \left(\frac{1}{4}\right)^{\sum_{k \in C_i}} \end{aligned}$$

$$\begin{aligned} P[Y] &= 1 - \prod_{k \in C_i} (1 - p_k) \geq 1 - \left(\frac{1}{e}\right)^{\sum_{k \in C_i} p_k} \\ P[Z] &= 1 - \prod_{C_j \in D_i - C_i} \prod_{k \in C_j} (1 - p_k) \leq 1 - \left(\frac{1}{4}\right)^{\frac{3}{4}} \\ P[A|Y] &= \left(1 - \frac{P[X]}{P[Y]}\right) \cdot (1 - P[Z]) \\ &\geq \left(1 - \frac{1 - (1 + 3\eta d^2) \left(\frac{1}{4}\right)^{3\eta d^2}}{1 - \left(\frac{1}{e}\right)^{3\eta d^2}}\right) \left(\frac{1}{4}\right)^{\frac{3}{4}} \end{aligned}$$

Somit ergibt sich

$$\begin{aligned} P[A|Y] &= \left(1 - \frac{P[X]}{P[Y]}\right) \cdot (1 - P[Z]) \\ &\geq \left(1 - \frac{1 - (1 + 3\eta d^2)(\frac{1}{4})^{3\eta d^2}}{1 - (\frac{1}{e})^{3\eta d^2}}\right) \left(\frac{1}{4}\right)^{\frac{3}{4}} \end{aligned}$$

Da dieser Ausdruck für $d = 1$ minimiert wird, gilt $P[A|Y] \geq 0.211$, wenn man den angegeben Wert für η einsetzt. Die Wahrscheinlichkeit dafür, dass der Kreis C_i terminiert, wenn ein Sensor in C_i sendet, beträgt also $P[A|Y]$. Da dieser Wert konstant ist, lässt sich die Anzahl der erwarteten Dominatoren in einem Kreis C_i vor seiner Terminierung als geometrische Verteilung modellieren und es gilt

$$E[C] = \frac{1 - P[A|Y]}{P[A|Y]} \leq \frac{1}{P[A|Y]} \leq 5.$$

Ferner gilt

$$P[C \geq 6 \log n] = (1 - P[A|Y])^{6 \log n} \in O(n^{-1}).$$

□

Insgesamt lässt sich also feststellen, dass der Erwartungswert für die Anzahl der Dominatoren in einem Kreis C_i vor seiner Terminierung mit einem konstanten Wert abgeschätzt werden kann. Das folgende Lemma zeigt, dass auch die zu erwartende Anzahl von Dominatoren konstant ist, welche sich während einer Kollision herausbilden.

Lemma 4.2.8. *Sei D die Anzahl der Knoten in einem Kreis C_i, die in einem gegebenen Zeitfenster senden, und sei Φ das Ereignis einer Kollision. Dann ist der Erwartungswert für die Anzahl der (gleichzeitig) sendenden Knoten im Falle einer Kollision $E[C|\Phi] \in O(1)$. Darüber hinaus ist die Wahrscheinlichkeit $P[D < 3 \log n / \log \log n | \Phi]$ sehr hoch.*

Beweis. siehe [MOBI04]

□

Lemma 4.2.9. *Sei A die Anzahl der neuen Dominatoren im Kreis C_i, die nach dessen Terminierung zur Menge der Dominatoren hinzukommen. Dann ist $A \in O(1)$ mit hoher Wahrscheinlichkeit.*

Beweis. siehe [MOBI04]

□

Theorem 4.2.3. *Der Erwartungswert für die Anzahl der Dominatoren im Kreis C_i $E[D]$ liegt in $O(1)$.*

Beweis. Für den Beweis sei der Kreis C_i betrachtet. Nach Lemma 4.2.7 ist die erwartete Anzahl der Dominatoren in C_i vor seiner Terminierung kleiner als 5. Lemma 4.2.8 besagt, dass die erwartete Anzahl neuer Dominatoren im Falle einer Kollision nicht größer ist als 3. Da C und $D|\Phi$ unabhängig sind, kann man die erwartete Anzahl der Dominatoren in C_i vor seiner Terminierung als

$$E[D] = E[C] \cdot E[D|\Phi] \leq 15 \in O(1)$$

berechnen.

□

Der Wert von Theorem 4.2.3 bildet eine probabilistische obere Schranke und konnte in Simulationen des Algorithmus deutlich unterboten werden.

Theorem 4.2.4. *Der Algorithmus berechnet ein korrektes DOMINATING SET in*

$$O\left(\frac{\log N}{d^2}\left(\log \Delta + \frac{\log N}{\log \log N}\right)\right)$$

Zeiteinheiten und erreicht dabei eine relative zu erwartende Gütegarantie in $O(1/d^2)$.

Beweis. Die Aussagen über Laufzeit und Korrektheit folgen aus den Theoremen 4.1.1 und 4.1.2. Theorem 4.2.3 beschränkt die Anzahl der zu erwartenden Dominatoren durch eine Konstante. Eine optimale Lösung muss in jedem D_i mindestens einen Sensor als Dominator wählen. Da D_i nach Lemma 4.2.3 maximal $5/d^2 + 15/d + 11$ Kreise C_i überdeckt, folgt die Behauptung.

□

4.3 Simulationsergebnisse

In diesem Kapitel sollen einige Simulationsergebnisse des QUDG Clustering Algorithm vorgestellt werden. Hierzu wurde ein Testszenario entworfen, in welchem n Sensoren zufällig über ein Quadrat der Größe 5×5 verteilt wurden. Der Senderadius der Sensoren betrug 1. Auf ein Testszenario, in welchem die Sensoren einen echten d-Quasi-Unit-Disk Graphen bilden, wurde verzichtet, da nicht klar ist, wie Kanten behandelt werden sollen, deren Existenz unspezifiziert ist. Gemeint sind damit solche Kanten, deren Länge kleiner als 1 aber größer als d ist. In den vorgestellten Beweisen wird keine Annahme über das Verhalten solcher Kanten gemacht. Denkbar wären Szenarien, in welchen auf solchen Kanten mit einer gewissen Wahrscheinlichkeit gesendet werden kann, aber auch solche, in welchen die Kommunikation nur anfänglich oder gar nicht funktioniert.
Um also konsistente Ergebnisse erzielen zu können, wurde ein Unit-Disk Graph ($d = 1$) gewählt. Der Parameter η wurde leicht relaxiert mit $\eta = 2^{-6}$ implementiert.
Der Parameter α wurde gemäß der in Pseudocode vorliegenden Implementierung des QUDG Clustering Algorithmus implementiert. Je größer man α wählt, desto länger wird die Laufzeit des Algorithmus. Andererseits erhöht eine zu niedrige Wahl des Parameters die Wahrscheinlichkeit mehrer Dominatoren in einer gegebenen Umgebung.

Ein weiterer Parameter p beschreibt das Aufwach-Verhalten der Sensoren: Sei s die Anzahl der schlafenden Sensoren zum Zeitpunkt t. Dann wacht jeder dieser schlafenden Sensoren zum Zeitpunkt $t + 1$ mit der Wahrscheinlichkeit $\frac{np}{s}$ auf. Dies führt zu einer gleichmäßigen Verteilung der

aufwachenden Sensoren in den ersten p^{-1} Zeiteinheiten.
Im Falle $p = 1$ wachen alle Sensoren sofort auf und es ergibt sich ein synchrones Aufwach-Verhalten der Sensoren. Ist der Parameter p sehr klein, so wachen die Sensoren über einen großen Zeitraum verteilt auf.
Für die Simulation wurde weiterhin $N = \Delta = n$ angenommen.

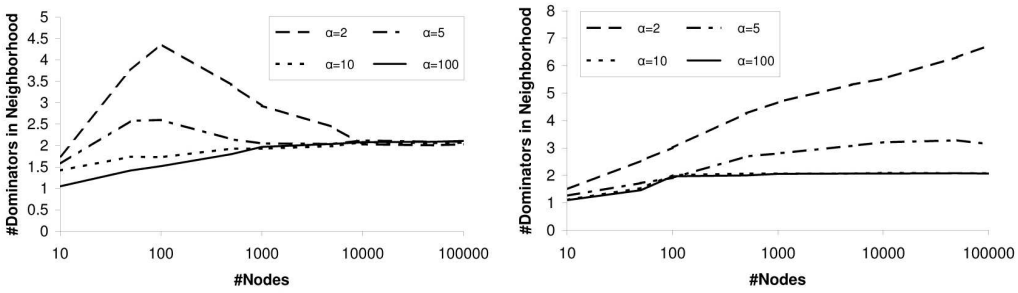


Abbildung 5: a) Synchrones Aufwachen ($p = 1$, links); b) Asynchrones Aufwachen ($p = 10^{-5}$, rechts)

In Abbildung 5 wurde die Anzahl der Dominatoren in Relation zur Anzahl der Sensoren im Netzwerk gesetzt. In beiden Graphiken wurde auf der y-Achse die durchschnittliche Anzahl von Dominatoren in der Umgebung eines Sensors abgetragen.
Abbildung 5 a) zeigt den Fall des synchronen Aufwachens aller Sensoren, Abbildung 5 b) hingegen den Fall des asynchronen Aufwachens. Beide Graphiken zeigen verschiedene Ergebnisse in Abhängigkeit des Parameters α. Für einen hinreichend groß gewählten Parameter α konvergiert die durchschnittliche Anzahl der Dominatoren in der Umgebung eines Sensors gegen 2. Ein solches Dominating Set weist bereits eine hervorragenden Qualität auf. Wählt man $\alpha \geq 10$, so sind die Unterschiede in Bezug auf die Qualität des Dominating Set selbst im Falle des asynchonen Aufwachens der Sensoren nicht mehr allzu groß.
Für den Beweis wurde der Parameter α mit einem Wert über 500 angenommen, um die gewünschten Ergebnisse mit einer großen Wahrscheinlichkeit nachzuweisen. Die Ergebnisse scheinen allerdings nahezulegen, dass man α durchaus auch kleiner wählen kann, ohne wirkliche Abstriche bei der Qualität des Dominating Set hinnehmen zu müssen. Allerdings darft nicht vergessen werden, dass die Unsicherheit, die der Parameter d bei der Modellierung einbringt, in der Simulation nicht berücksichtigt wurde. Nach Theorem 4.2.4 wirkt sich die Wahl des Parameters nämlich durchaus auf die Qualität des Dominating Set aus.
Da die Laufzeit des Algorithmus durch die Wahl von α direkt beeinflusst wird, ist man bestrebt, α so klein wie möglich zu wählen. Wählt man α allerdings zu klein, so kann es zu einer unkontrollierten Vermehrung der Dominatoren kommen, wie man Abbildung 5 b) entnehmen kann. Dies hat vor allem zwei Gründe: Zum einen steigt die Anzahl der Kollisionen, da die Sensoren ihre Sendewahrscheinlichkeit zu schnell verdoppeln. Zum anderen wird die Lausch-Phase zu Beginn des Algorithmus zu kurz, so dass die Wahrscheinlichkeit sinkt, dass Sensoren, die bereits durch Dominatoren abgedeckt sind, diese auch bemerken.

Abbildung 6 setzt nun die durchschnittliche Anzahl der Dominatoren in der Umgebung eines Sensors in Relation zum Faktor α und soll der Frage nachgehen wie klein man α tatsächlich wählen kann, ohne allzu große Einbußen bei der Qualität des Dominating Set hinnehmen zu müssen. Beide Graphiken zeigen unterschiedliche Ergebnisse in Abhängigkeit der Anzahl n der Sensoren im Netzwerk.
Dabei wird deutlich, dass die durchschnittliche Anzahl der Sensoren im Bereich von $\alpha = 10$ einen akzeptablen Wert von etwa 2 erreicht, diesen bei größerem α jedoch kaum noch verringern kann. Im Fall des synchonen Aufwachens (Abbildung 6 a)) liegt dieser Wert sogar teilweise deutlich unter 10.
Die Ergebnisse legen nahe, dass es ausreicht $\alpha = 10$ zu wählen, falls $d = 1$ gilt.

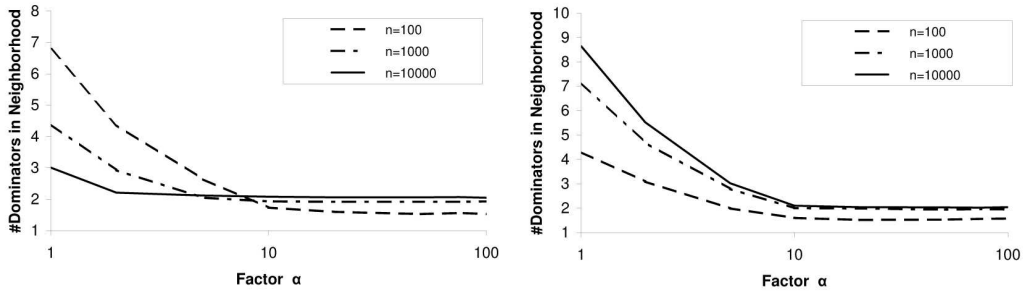


Abbildung 6: a) Synchrones Aufwachen ($p = 1$, links); b) Asynchrones Aufwachen ($p = 10^{-5}$, rechts)

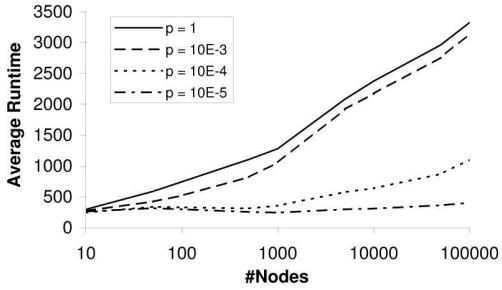


Abbildung 7: Laufzeit ($\alpha = 10$)

Auf der y-Achse von Abbildung 7 wurde die durchschnittliche Anzahl der Zeiteinheiten abgetragen, welche zwischen dem Zeitpunkt des Aufwachen und dem des Sich-Entscheidens eines Sensors vergehen. Diese wurde in Relation zur Anzahl der Sensoren im Netzwerk gesetzt. Es sei darauf hingewiesen, dass die x-Achse logarithmisch skaliert ist.
Für große p (geringe Asynchronität) wächst die Laufzeit etwa asymptotisch in $O(\log^2 n)$. Für kleine p (größere Asynchronität) hingegen ist die Laufzeit weit kürzer und verlängert sich für $n \in [10...100000]$ nicht signifikant.
Während der ersten $\alpha\lceil \log^2 n/(d^2 \log\log n)\rceil$ Zeiteinheiten befinden sich die Sensoren in einem Lauschzustand, in welchem sie nicht senden. Im Falle geringer Asynchronität befinden sich fast alle Sensoren gleichzeitig in diesem Zustand, was zu einer Erhöhung der Laufzeit führt, da dieser Zeitraum nicht effektiv genutzt wird und ein entsprechender Aufwand zu einem späteren Zeitpunkt geleistet werden muss. Im Falle hoher Asynchronität hingegen wachen die meisten Sensoren erst auf, wenn sich bereits Dominatoren in ihrer Umgebung etabliert haben. Nach Lemma 4.2.9 empfangen solche Sensoren mit hoher Wahrscheinlichkeit eine Nachricht eines Dominators und entscheiden sich somit relativ früh kein Dominator zu werden. Die hohe Laufzeit des Algorithmus im synchronen Falle lässt sich also direkt aus der Funktionsweise des Algorithmus ableiten.

Zusammenfassend lässt sich sagen, dass man den Parameter α (zumindest im Falle $d = 1$) kleiner wählen lässt als im Kapitel Analyse vorgeschlagen wurde. Die Simulation legt nahe, dass im Falle $d = 1$ ein $\alpha \approx 10$ ausreicht, und dass im Durchschnitt etwa 2 Dominatoren in der Umgebung eines Sensors erreicht werden. Die durchschnittliche Laufzeit des Algorithmus liegt in $O(\log^2 n)$, wenn man ein synchrones Aufwach-Verhalten zugrundelegt, jedoch deutlich darunter, wenn das Aufwach-Verhalten eine größere Asynchronität aufweist.

5 Maximal Independent Set Clustering

Eine weitere Möglichkeit eine Clustering auf einem Netzwerk(graphen) zu erzeugen besteht in der Berechnung einer maximalen unabhängigen Menge (Maximal Independent Set (MIS)).

Eine unabhängige Menge (Independent Set (IS)) eines Graphen $G = (V, E)$ ist eine Menge $V' \subset V$ von unabhängigen Knoten, so dass für alle $u, v \in S$ gilt $vu \notin E$. Ein maximales Independent Set (MIS) ist eine solche unabhängige Menge mit der zusätzlichen Eigenschaft, dass jeder Knoten, der nicht zur unabhängigen Menge V' gehört, einen Nachbarknoten in V' hat.
Wählt man die Clusterheads für das zu berechnende Clustering als eine solche maximale unabhängige Menge, so erhält man im Grunde wieder ein Dominating Set mit der zusätzlichen Eigenschaft, dass je zwei Clusterheads nicht in unmittelbarer Nachbarschaft, und folglich nicht in gegenseitiger Sendereichweite liegen dürfen.
Für das Neztwerk hat dies den Vorteil, dass sich je zwei Clusterheads beim Senden einer Nachricht nicht gegenseitig stören können.

5.1 Algorithmus

Der folgende Algorithmus basiert auf einem ähnlichen Muster wie der QUDG Clustering Algorithm. Es handelt sich auch hier um einen probabilistischen, verteilten Algorithmus.

MIS Clustering Algorithm

state ← uncovered;
excited ← **false**;
upon wake-up **do:**
1 **for** $j \leftarrow 1$ to $2\delta \cdot \lceil \log^3 \hat{n} / \log\log \hat{n} \rceil$ **do**
2 wait();
3 **od**
4 counter ← 0;
5 **for** $j \leftarrow \lceil \log \hat{n} \rceil$ to 0 by -1 **do**
6 $p \leftarrow 1/\left(2^{j+\beta}\right)$;
7 **for** $i \leftarrow 1$ to $\gamma \cdot \lceil \log \hat{n} \rceil$**do**
8 $b \leftarrow \left\{ \begin{array}{ll} 1 & \text{w/ probability } p \\ 0 & \text{w/ probability } 1-p \end{array} \right\}$
9 **if** b = 1 **then**
10 **send**() on Γ_1;
11 **start candidacy**();
12 **stop executing main-loop**;
13 **fi**
14 **od**
15 **od**

Candidacy Phase():
16 **loop**
17 $b \leftarrow \left\{ \begin{array}{ll} 1 & \text{w/ probability } q \\ 0 & \text{w/ probability } 1-q \end{array} \right.$
18 **if** $b = 1$ **then**
19 excited ← **true**
20 **send**(counter) on Γ_2;
21 **fi**
22 **if** excited **then**
23 counter ← counter +1;
24 **fi**
25 **if** counter $= \delta \cdot \lceil \log^3 \hat{n} / \log\log \hat{n} \rceil$ **then**
26 state ← MIS;
27 **send** on Γ_3 w/ probability 1/6 forever;
28 **fi**
29 **end loop**

Receive Triggers:
(Only executed if the nodes does not send a message in the time-slot)
upon receiving msg on Γ_1 **do:**
30 **if not** candidate then
31 restart main-loop at line 1;
32 **fi**

upon reveiving msg (c') on Γ_2 **do:**
33 $\Delta c \leftarrow |c' - \text{counter}|$;
34 **if** candidate **and** $\Delta c \leq 8 \log \hat{n}$ **then**
35 counter $\leftarrow -8\lceil 8 \log \hat{n} \rceil$;
36 **fi**

upon receiving msg on Γ_3 **do:**
37 state $\leftarrow$ covered;
38 **terminate**();

Der MIS-Clustering Algorithmus ist in drei Teile untergliedert. Zunäst führt jeder Knoten die *main*-Schleife aus. Sobald ein Sensor seine erste Nachricht sendet, wird die Ausführung der *main*-Schleife abgebrochen und der entsprechende Sensor ruft die *start candidacy()*-Routine auf, welche dann bis zur Terminierung des Sensors ausgeführt wird.
Parallel hierzu stehen die Sensoren auf allen drei Kanälen auf Empfang. Auch hier sei angenommen, dass eine Nachricht nur dann empfangen werden kann, wenn kein weiterer Sensor im entsprechenden Zeitintervall ebenfalls sendet.
Der Algorithmus besteht aus zwei Haupt-Phasen. In der ersten Phase, der *main*-Schleife, werden zufällig Kandidaten ausgewählt, die in der *candidacy*-Phase des Algorithmus dann um die Aufnahme in die unabhängige Menge konkurrieren.
Wenn ein Sensor seine erste Nachricht auf Γ_1 sendet, wird er automatisch zu einem solchen Kandidaten (Zeilen 10,11). Eine Haupt-Aufgabe der *main*-Phase besteht darin, die Anzahl der Sensoren zu beschränken, die gleichzeitig in der *candidacy*-Phase um eine Aufnahme in die unabhängige Menge konkurrieren. Somit kann sichergestellt werden, dass die Auswahl der Sensoren, die in die Menge der unabhängigen Knoten aufgenommen werden, nicht zu lange dauert.
Die Auswahl der Sensoren in der *candidacy*-Phase geschieht ausschließlich auf dem Kanal Γ_2. Der Kanal Γ_3 hingegen ist ausschließlich für jene Sensoren vorgesehen, die bereits in die unabhängige Menge aufgenommen wurden.
Aufgrund des im Modell berücksichtigten asynchronenen Aufwachens der Sensoren, sind die *candidacy*-Phasen verschiedener Sensoren nicht notwendigerweise synchronisiert. Es ist sogar weit wahrscheinlicher, dass die *candidacy*-Phasen zweier Sensoren unsynchronisiert sind.
Darüber hinaus hat ein Sensor, der noch keine Nachricht von einem seiner Nachbarn empfangen hat, kein Wissen darüber, ob andere Sensoren bereits die *main*- oder *candidacy*-Phase ausführen.
Wie bereits bei der Berechnung einer minimalen überdeckenden Menge (MDS) besteht auch in diesem Fall die Herausforderung zu einem wesentlichen Teil darin, den Mangel an Wissen, welches für den Aufbau einer unabhängigen Menge benötigt wird, durch ein probabilistisches Vorgehen zu kompensieren.
Genauer funktioniert der Algorithmus wie folgt:
Unmittelbar nach ihrem Aufwachen führen die Sensoren die *main*-Schleife aus. Zunächst warten die Sensoren darauf, dass sie auf einem der drei Kommunikationskanäle Nachrichten empfangen, ohne dabei selbst sendend aktiv zu sein. Diese Wartephase verfolgt zwei Ziele: Zum einen können diese Sensoren die Kandidaten, welche in der *candidacy*-Phase um die Aufnahme in das MIS konkurrieren nicht durch Interferenzen stören. Zum anderen haben die Sensoren so die Möglichkeit Wissen über die Sensoren in ihrer Nachbarschaft zu akkumulieren. Empfängt ein Sensor in dieser Warte-Phase etwa eine Nachricht auf Γ_3, so gibt es in seiner Umgebung offenbar bereits einen Sensor im MIS, und der Sensor kann die Ausführung des Algorithmus einstellen (Zeile 38).
Der Hauptteil des Algorithmus, der in Zeile 5 beginnt, ist in Runden organisiert, die jeweils $\gamma \cdot \lceil \log \hat{n} \rceil$ Zeiteinheiten dauern. Auch hier entspricht eine Zeiteinheit in etwa der Zeit, die ein Sensor zum

Versenden einer Nachricht benötigt.
In der *main*-Phase des Algorithmus sendet jeder Sensor auf Γ_1 mit einer gewissen Wahrscheinlichkeit p. Diese beträgt zu Beginn des Algorithmus $2^{-\log \hat{n}+\beta}$ und wird in jeder Runde verdoppelt. Somit ergibt sich in der letzten Runde ein Wahrscheinlichkeit von $2^{-\beta}$. Mit jeder Runde, die ein Sensor durchläuft, erhöht sich also die Wahrscheinlichkeit, dass er auf Γ_1 sendet und damit die Wahrscheinlichkeit zu einem Kandidaten zu werden, exponentiell. Gleichzeigtig aber erhöht sich die Wahrscheinlichkeit, dass er Signale von umliegenden Sensoren empfängt, sofern solche existieren.
Empfängt ein solcher Sensor in der *main*-Phase des Algorithmus ein Signal auf Γ_1, so gibt es bereits einen Kandidaten in seiner Umgebung. Der Sensor bricht daraufhin die Berechnung ab und beginnt von neuem in Zeile 1 des Algorithmus.
Dieser Mechanismus erlaubt es, eine obere Schranke für die Anzahl der Sensoren anzugeben, die gleichzeitig Kandidaten sein können. Ferner kann man zeigen, dass bei jedem Neustart dieser Art ein Sensor in der Umgebung des zurückgesetzten Sensors existiert, welcher innerhalb der erforderlichen Zeitschranken in das MIS aufgenommen werden wird.
Wartende Sensoren werden inaktiv genannt, solche, die den Hauptteil des Algorithmus ausführen aktiv.
Die *candidacy*-Phase funktioniert nun wie folgt:
Jeder Kandidat sendet in einem gegebenen Zeitfenster mit einer gewissen Wahrscheinlichkeit q auf Γ_2. Nachdem ein Sensor seine erste Nachricht auf Γ_2 gesendet hat, wird er *excited* und beginnt damit einen Zähler in jedem Zeitfenster zu erhöhen. Dieser Zähler wird dann an jede Nachricht, die dieser Sensor versendet angehängt.
Ein Sensor, welcher eine solche Nachricht mit angehängtem Zähler empfängt, vergleicht den Zähler der Nachricht mit seinem eigenen. Falls sich der eigene Zähler vom Zähler des sendenden Knotens um weniger als $8\lceil \log \hat{n} \rceil$ unterscheidet, wird der eigene Zähler zurückgesetzt. Diese verhindert, dass zwei benachbarte Sensoren kurz hintereinander in das MIS aufgenommen werden. Interessanterweise verhindert dieser Zählervergleich auch kaskadierende Rücksetzungen.
Erreicht der Zähler eines Sensors den Grenzwert von $\delta \cdot \lceil \log^3 \hat{n} / \log\log \hat{n} \rceil$, so wird dieser in das MIS aufgenommen und beginnt unmittelbar damit auf dem Kanal Γ_3 mit konstanter Wahrscheinlichkeit zu senden.
Die Zähler je zweier Sensoren in einer Umgebung können den Grenzwert nicht innerhalb von $8 \log \hat{n}$ Zeiteinheiten erreichen. Somit hat der Sensor, der als erster in das MIS aufgenommen wird, etwas Zeit seine Umgebung davon zu informieren.
Die Parameter q und β werden wie folgt definiert:

$$q = \log\log \hat{n} / \log^2 \hat{n} \qquad \beta = 6.$$

Der Parameter q muss dabei zwei Aspekte erfüllen. Einerseits muss q groß genug sein, damit überhaupt ein Sensor innerhalb der erwünschten Laufzeit des Algorithmus in das MIS aufgenommen wird. Andererseits stellt ein kleiner Wert von q sicher, dass keine benachbarten Sensoren in das MIS aufgenommen werden.

Der Parameter β maximiert die Wahrscheinlichkeit einer erfolgreichen Berechnung.
Die Parameter δ und γ können dazu verwendet werden Laufzeit und Erfolgswahrscheinlichkeit der Berechnung zu beeinflussen. Kleine Werte von δ und γ verkürzen zwar die Laufzeit, vermindern aber auch die Wahrscheinlichkeit einer erfolgreichen Berechnung, während große Werte der beiden Parameter die Erfolgswahrscheinlichkeit zulasten einer längeren Laufzeit erhöhen.

6 Zusammenfassung

Die vorgestellten Algorithmen der Autoren Kuhn, Mosibroda und Wattenhofer stellen sich der Frage, wie man auf effiziente Weise ein Clustering in einem quasi-chaotischen Multi-Hop-Netzwerk mit verteilten Algorithmen berechnen kann. Dabei sollen die spezifischen, unmittelbar nach der Inbetriebnahme eines unstrukturierten Netzwerkes geltenden Rahmenbedingungen möglichst realitätsnah modelliert werden. Insbesondere fließen in die Modellierung nicht nur häufig auftretende

Problemstellungen wie asynchrones Aufwach-Verhalten, unscharfe Sendereichweiten und mangelnde Kollisionserkennungs- und -behebungsmaßnahmen der Sensoren ein, die Sensoren sollen vielmehr nur ein Mindestmaß an Rahmenbedingungen erfüllen müssen, um für die Berechnung geeignet zu sein. So stellt die einzige Anforderung der Autoren an die Sensoren die Fähigkeit zur Kommunikation und die Speicherung zweier oberer Schranken für die Gesamtzahl N der Sensoren im Netzwerk und die Anzahl Δ der Nachbarsensoren dar. Man kann zeigen, dass eine schlechte obere Schranke Δ die Laufzeit des Algorithmus nur mäßig beeinflusst, so dass man durch die Annahme $\Delta = N$ selbst letztere Anforderung einsparen könnte.
Die Autoren schlagen zwei probabilistische, verteilte Algorithmen zur Berechnung eines Clusterings unter den gegebenen Rahmenbedingungen vor, die auf der Abstraktion eines Minimum Dominating Set beziehungsweise auf der eines Maximal Independent Set beruhen. Beide Algorithmen haben im Wesentlichen polylogarithmische Laufzeit und erreichen im Erwartungswert eine konstante Approximation eines MDS beziehungsweise MIS.
Beide Algorithmen kompensieren den Mangel an Wissen der Sensoren über die Eigenschaften des Netzwerkes durch einen probabilistischen Ansatz: Mit zunehmenden Wahrscheinlichkeiten entscheiden sich die Sensoren quasi spontan, ob sie zu einer der ausgezeichneten Mengen gehören wollen. Anschließend werden die so getroffenen Entscheidungen mittels Nachrichten dieser Sensoren in ihrer Umgebung geflutet. Die Nachrichten werden mit gewissen konstanten Wahrscheinlichkeiten verschickt. Aufwachende Sensoren lauschen zunächst eine gewisse Zeit lang, ohne selbst tätig zu werden auf solche Nachrichten, bevor auch sie damit beginnen, sich an dem Wettlauf um den Dominatorenstatus zu beteiligen.
Von entscheidender Bedeutung für den Erfolg der Berechnung ist die richtige Wahl der Wahrscheinlichkeiten. Wählt man die Wahrscheinlichkeiten, mit denen die Sensoren in eine der ausgezeichneten Mengen ausgewählt werden zu klein, steigt die Laufzeit des Algorithmus, wählt man sie hingegen zu groß, besteht die Gefahr, dass beliebig viele Sensoren zu Dominatoren werden können. Auch die Wahl der Wahrscheinlichkeiten für das Versenden der Nachrichten seitens der Dominatoren muss mit Bedacht durchgeführt werden. Sind diese zu groß, kommt es vermehrt zu Kollisionen und die Mehrzahl der Nachrichten geht verloren, sind sie allerdings zu klein, sinkt die Wahrscheinlichkeit, eine hinreichend große Zahl von Nachbarsensoren zu erreichen.
Aber auch die Dauer der Wartephase beeinflusst das Ergebnis der Berechnung. Eine kurze Wartephase wirkt sich positiv auf die Laufzeit des Algorithmus aus, verschlechtert aber die Wahrscheinlichkeit, dass aufwachende Sensoren die Nachrichten eventuell bereits in ihrer Umgebung existierender Sensoren empfangen.
Obwohl die Algorithmen in dem vorgeschlagenen Setting von jedem Sensor nur einmal während der gesamten Lebensdauer des Netzwerkes durchgeführt werden muss, ist die Laufzeit des Algorithmus von entscheidender Bedeutung. Die Berechnung eines solchen Clusterings ist insbesondere in großen Netzwerken eine nicht zu unterschätzende Aufgabe. Da die Kapazität der häufig batteriebetriebenen Sensoren stark begrenzt ist, ist man bestrebt, die Initialisierungsphase so kurz wie möglich zu gestalten, um den Nutzen der Sensoren, deren eigentliche Aufgabe in anderem besteht, zu maximieren.

Die vorgeschlagenen Algorithmen eignen sich im Wesentlichen für die Initialisierung von statischen Netzwerken. Dynamische Netzwerke müssen ihr Clustering ständig neu berechnen. Hierfür allerdings bieten die beiden Algorithmen zu wenig Flexibilität, da ein einmal anerkannter Dominatorenstatus nicht mehr infrage gestellt wird. Unterstellt man allerdings ein asynchrones Aufwach-Verhalten der Sensoren, so entsteht auch in statischen Sensornetzwerken eine gewissen Dynamik bei der Initialisierung.
Insbesondere könnte es etwa vorkommen, dass früh aufwachende Sensoren schlechter für das Clustering geeignet sind als Sensoren die erst noch aufwachen müssen. Man könnte sich also fragen, ob im Hinblick auf diese Form der Dynamik eine flexiblere Handhabung des Dominatoren-Status nicht bessere Ergebnisse erzielt werden können, wenn man also zulässt, dass während der Initialisierung des Netzwerkes schlecht geeignete Dominatoren wieder degradiert werden können.

Ungeklärt jedoch bleibt auch eine Frage, die sich erst nach der Initialisierung stellt. Offenbar streben die Autoren ein Szenario an, in welchem theoretisch jeder beliebige Sensor zu einem Dominator werden kann. Dies aber ist nur sinnvoll, wenn alle Sensoren gleicher Bauart sind. Dominatoren jedoch haben durch das Versenden von Nachrichten einen erheblich höheren Energieverbrauch als

Nicht-Dominatoren, die sich nach Empfang einer Nachricht quasi abschalten können. Dominatoren nämlich senden mit gewissen Wahrscheinlichkeiten in einer Endlosschleife. Selbst wenn man einem Vorschlag der Autoren folgen und Sleep-Listen-Phasen einrichten würde, wäre der stark erhöhte Energieverbrauch von Dominatoren nicht zu leugnen. Dass bei einem batteriebetriebenen Betrieb der Sensoren aber gerade dieser Backbone des Netzwerkes als erstes wegbricht, ist fatal.
Zum einen verschärft dies die Forderung nach einer möglichst kurzen Berechnung des Clusterings, zum anderen wirft dies die Frag auf, ob nicht auch hier eine fexiblere Handhabung des Dominatorenstatus bessere Ergebnisse erzielen könnte. So könnte bei Ausfall eines Dominators ein Nicht-Dominator an dessen Stelle treten.

Die vorliegenden Algorithmen stellen aus meiner Sicht dennoch eine solide Grundlage für die Berechnung eines Clusterings auf einem Multi-Hop-Netzwerk dar.

Literaturverzeichnis

[MOBI04] Fabian Kuhn, Thomas Moscibroda, Roger Wattenhofer, „Initializing Newly Deployed Ad Hoc and Sensor Networks“, *10th Annual International Conference on Mobile Computing and Networking (MOBICOM), Philadelphia, USA*, 2004.

[ESA04] Fabian Kuhn, Thomas Moscibroda, Roger Wattenhofer, „Radio Network Clustering from Scratch“, *12th Annual European Symposium on Algorithms (ESA), Bergen, Norway*, 2004.

[MASS04] Thomas Moscibroda, Roger Wattenhofer, „Efficient Computation of Maximal Independent Sets in Unstructured Multi-Hop Radio Networks“, *1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), Fort Lauderdale, Florida, USA*, 2004.