

Seminararbeit zum Thema: "Does Topology Control Reduce Interference?"

Paul Köhler

Wintersemester 2007/2008

"Does Topology Control Reduce Interference?" von Martin Burkhart, Pascal von Rickenbach, Roger Wattenhofer, Aaron Zollinger des Department of Computer Science an der ETH Zurich veröffentlichten dieses Paper auf der MobiHoc '04 in Roppongi, Japan. Die Arbeit widerlegt die fälschliche Annahme, dass ein Topologiekontrollalgorithmus lediglich darauf achten muss, dass der resultierende Graph Knoten niedrigen Grads erzeugen muss, um die Interferenzen zu senken. Dazu wird zunächst der Begriff der "Interferenz" exakt bestimmt und damit ein Optimierungsproblem definiert.

1 Einführung

In Drahtlosen Ad-Hoc Netzwerken ist es das Ziel aller, das Netzwerk so lange wie möglich erhalten zu können. Da der Stromverbrauch jedoch quadratisch mit der Entfernung wächst ist es sinnvoll lange Entfernungen durch mehrere kurze zu ersetzen, die dann eine kürzere Reichweite haben, was dazu führt, dass Kollisionen vermieden werden und der Energieverbrauch durch Einsparung von erneuten Übertragungen gesenkt werden kann. Reduziert man jedoch den Graphen zu sehr geht dies auf Kosten der Konnektivität, denn es können so z.T. sehr lange Wege entstehen, es kann sogar zu einem komplett unverbundenen Graphen führen.

Ein Topologiekontrollalgorithmus hat also die Aufgabe einen sinnvollen Konsens zwischen Stromersparnis und möglichst minimaler Interferenz auf der einen und der Konnektivität und Spanneigenschaft des Netzwerks auf der anderen Seite zu finden.

Bisherige Ergebnisse auf dem Gebiet der Topologiekontrolle meinten Interferenzen durch niedriggradige Graphen in den Griff bekommen zu haben dies wird in diesem Artikel widerlegt. Eine intuitive Definition der "Interferenz" soll mit einer Reihe von wichtigen zu erfüllenden Eigenschaften der entstehenden Topologie, den Grundstein für die Lösung des Problems zur "perfekten" Algorithmusfindung legen.

Allen Anforderungen entsprechend wird dann im Laufe dieser Arbeit der LLISE Algorithmus entwickelt und ausgewertet werden.

2 Bisherige Arbeiten

Nachdem frühe Arbeiten von einer randomisierten Verteilung aller Knoten auf einer euklidischen Ebene ausgingen, wurden später Ansätze aus der rechnerrelevanten Geometrie, wie der Delaunay Triangulation, MST, Relative Neighborhood oder auch dem Gabriel Graph, verfolgt. Da einige Algorithmen aber nicht lokal berechenbar sind (z.B. MST, Delaunay Tri.) rückten nun neue Methoden in den Vordergrund, wie z.B. der CBTC-Algorithmus.

Ein weiterer Ansatz zur Findung einer geeigneten Topologie ist es, Knoten zu gruppieren (Cluster), um dann nicht nur die Sendestärke zu verringern, sondern auch rechenintensive Aufgaben auf das

Netzwerkcluster zu verteilen.

Für die weitere Betrachtung definieren wir Graph $G = (V, E)$ mit $V \in \mathbb{R}$ und $E \in V^2$. Mit Knoten = V(ertice)'s und Verbindungen = E(dge)'s. Um "Kommunikation" zwischen zwei Knoten (u,v) zu schaffen, muss es sich um ungerichtete Graphen handeln, welche auch nur zustande kommen können, wenn beide diese Sendeleistung überhaupt erbringen können. Es wird folgendes definiert:

$$Cov(e) := |\{w \in V | w \text{ ist abgedeckt durch } D(u, |u, v|)\} \cup \{w \in V | w \text{ ist abgedeckt durch } D(v, |v, u|)\}|.$$

und

$$D(u, r) | u \in V, r \in \mathbb{R}, \text{ Kreisfläche mit } u \text{ Mittelpunkt und } r \text{ Radius}$$

Nun stellen wir eine Definition zur Interferenzmessung auf, welche auf dem soeben definierten Modell beruht:

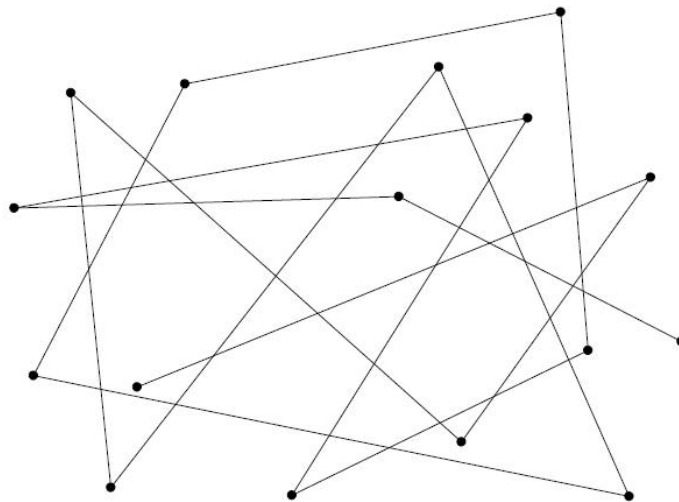
$$I(G) := \max_{e \in E} Cov(e)$$

Diese Definition reicht aber nicht aus, da man dieses Problem ganz einfach optimal lösen kann indem man die Übertragungsstärke auf Null setzt. Deswegen benötigen wir noch weitere Nebenbedingungen:

- Konnektivität (wenn $e=(u,v)$ in G , dann auch in G')
- Spann-Eigenschaft (kürzeste Wege in G nur konstant größer in G')
- Planarität (keine sich überkreuzenden Verbindungen)

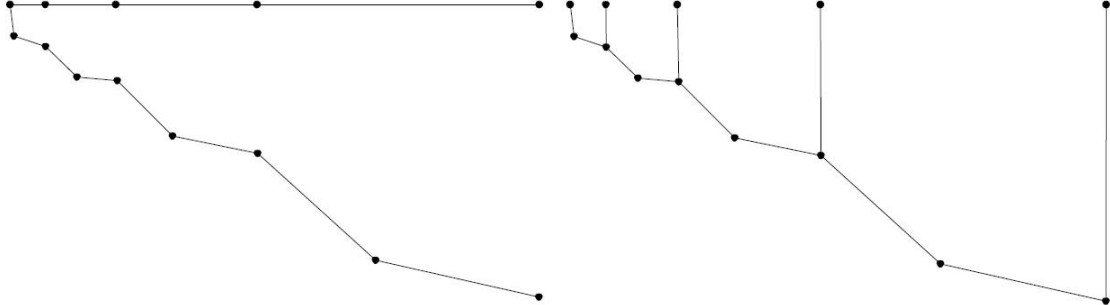
3 Interferenz in bekannten Topologien

Dieses Kapitel wird zeigen, dass niedriger oder allgemein beschränkter Graphen- bzw. Knotengrad nicht unbedingt zu niedriger Interferenz führt und auch bei dem Vergleich mit dem Optimum eher schlecht abschneiden kann.



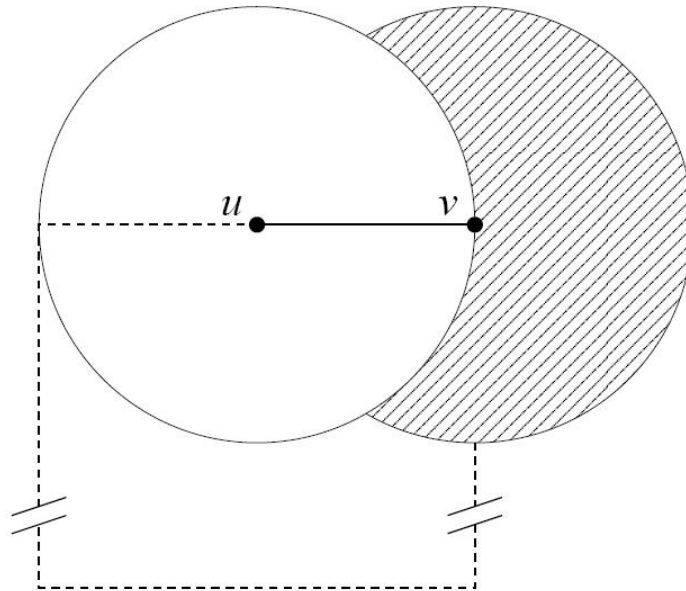
Dieses Bild (s.o.) zeigt einen Graphen mit Grad 2 aber einer Interferenz von fast n (Anzahl der Knoten). Desweiteren ergeben sich durch die Nebenbedingungen Situationen, wie die exponentiell wachsende Knotenkette, deren Interferenz in $\Omega(n)$ liegt. Alle bis dato bekannten Topologiekontrollsysteme verfolgten den Sinn des "Nearest Neighbor" - Algorithmuses. Dieser führt jedoch bei der Topologiefindung nicht zum Optimum, was im Nachfolgenden gezeigt wird.

Theorem 1. *No currently proposed topology control algorithm establishing only symmetric connections - required to maintain connectivity of the given network - is guaranteed to yield a nontrivial interference approximation of the optimum solution. In particular, interference of any proposed topology can be $\Omega(n)$ times larger than the interference of the optimum connected topology, where n is the total number of network nodes.*



Der Beweis für Theorem 1 wird nun bildlich veranschaulicht, durch die oben zu sehenden Graphen. Da wir von lokalen Algorithmen ausgehen und jeder Knoten den gesamten Graph nicht sieht wird er sich mit dem ihm am nächsten liegenden Knoten verbinden, was wie wir leicht ersehen können im Endeffekt die Interferenz auf $\Omega(n) = \Omega(|V|)$ treibt (siehe links). Auf der rechten Seite wiederum ist deutlich zu ersehen, dass das Optimum bei $\Omega(1)$ liegt und somit die Interferenz fast zu vernachlässigen. Genau hier liegt die Problematik der lokalen Algorithmen. Auch sieht man hier, dass der Knotengrad nichts über weniger Interferenz aussagt, da hier bei der Interferenz-Optimalen Topologie einige Knoten einen Grad von 3 haben im Gegensatz zu der $\Omega(n)$ interferenzierten Topologie mit kompletten Knotengrad 2.

Theorem 2. *For the requirement of maintaining connectivity of the given network, there exists a class of graphs for which there is no local algorithm that approximates optimum interference.*



Auch hier schauen wir uns für die Beweisführung des zweiten Theorems ein Bild (siehe oben) an. Wir definieren nun, dass alle weiteren Knoten des Netzwerks in dem schattierten Bereich befinden so läge

hier auch die Interferenzklasse bei $\Omega(n)$. Zusätzlich befinden sich Knoten auf der gestrichelten Linie, welche auch u und v Verbinden würden durch einen jedoch etwas längeren Weg. Da der Algorithmus lokal ist, er jedoch die Konnektivität bewahren muss, wird dieser bei einer genügend langen Kette die Verbindung zwischen u und v herstellen, die dann in der Interferenzklasse $\Omega(n)$ statt ohne der Kante (u,v) in $O(1)$ wäre.

4 Niedrig interferenzierte Topologien

Low Interference Forest Establisher (LIFE)

Input: a set of nodes V , each $v \in V$ having attributed a maximum transmission radius r_v^{max}

- 1: $E =$ all eligible edges (u, v) ($r_u^{max} \geq |u, v|$ and $r_v^{max} \geq |u, v|$) (* unprocessed edges *)
- 2: $E_{LIFE} = \emptyset$
- 3: $G_{LIFE} = (V, E_{LIFE})$
- 4: **while** $E \neq \emptyset$ **do**
- 5: $e = (u, v) \in E$ with minimum coverage
- 6: **if** u, v are not connected in G_{LIFE} **then**
- 7: $E_{LIFE} = E_{LIFE} \cup \{e\}$
- 8: **end if**
- 9: $E = E \setminus \{e\}$
- 10: **end while**

Output: Graph G_{LIFE}

Theorem 3. *The forest constructed by LIFE is a Minimum Interference Forest.*

LIFE wendet sozusagen Kruskals Algorithmus auf Interferenzen an. Es sortiert Kanten nach deren Coverage und fügt sie der Reihe nach von niedrig nach hoch hinzu, wenn nicht bereits ein Weg zwischen den zwei Knoten existiert. Da nun aber mehrere Bedingungen neben der Konnektivität wichtig sind entstand LISE mit denselben Eigenschaften von LIFE jedoch mit der Eigenschaft ein t-Spanner zu sein.

Low Interference Spanner Establisher (LISE)

Input: a set of nodes V , each $v \in V$ having attributed a maximum transmission radius r_v^{max}

- 1: $E =$ all eligible edges (u, v) ($r_u^{max} \geq |u, v|$ and $r_v^{max} \geq |u, v|$) (* unprocessed edges *)
- 2: $E_{LISE} = \emptyset$
- 3: $G_{LISE} = (V, E_{LISE})$
- 4: **while** $E \neq \emptyset$ **do**
- 5: $e = (u, v) \in E$ with maximum coverage
- 6: **while** $|p^*(u, v) \text{ in } G_{LISE}| > t|u, v|$ **do**
- 7: $f =$ edge $\in E$ with minimum coverage
- 8: move all edges $\in E$ with coverage $Cov(f)$ to E_{LISE}
- 9: **end while**
- 10: $E = E \setminus \{e\}$
- 11: **end while**

Output: Graph G_{LISE}

Theorem 4. *The graph $G_{LISE} = (V, E_{LISE})$ constructed by LISE from a given network $G = (V, E)$ is an interference-optimal t-spanner of G .*

Um die t -Spann-Eigenschaft von G_{LISE} zu beweisen reicht es zu zeigen, dass der kürzeste Weg zwischen u und v in G_{LISE} nicht länger ist als $t|u,v|$ des Originalgraphen. Dies gilt, da man für jeden Pfad des Originalpfades $p(u,v)$, der höchstens t Mal so lang sein darf wie der des resultierenden Graphen, durch einfaches Ersetzen jeder einzelnen Kante aus $p(u,v)$ durch den $p'(u,v)$. Somit wäre das Problem auf $t(u,v) \in G \geq (u,v) \in G'$ reduziert. Ist eine Kante in E und in E_{LISE} , so ist die Lösung trivial. Ist eine Kante jedoch in E aber nicht in E_{LISE} , kann sie aufgrund der *if-Bedingung* in Zeile 6 des Codes nicht länger als t Mal sein. Somit erfüllt G_{LISE} die t -Spanneigenschaft.

Die optimale Interferenz von LISE zeigen wir durch einen Widerspruchsbeweis:

Sei G_{LISE} ein nicht optimal interferenzierter Graph mit der t -Spann-Eigenschaft und $G^* = (V, E^*)$ ein optimal interferenzierter mit t -Spann-Eigenschaft aus $G(V, E)$. Somit gilt $I(G_{LISE}) > I(G^*)$ somit gilt $\forall e^* \in E^* Cov(e^*) < I(G_{LISE})$. Auch ist E^* nichttriviale Teilmenge von E_{LISE} . Sei T die Menge der Kanten, wobei $t \in T$, so sei $Cov(t) = I(G_{LISE})$ und $\tilde{G} = (V, \tilde{E})$ mit $\tilde{E} = E_{LISE} \setminus T$. \tilde{G} erfüllt die t -Spann-Eigenschaft, da $E^* \subset \tilde{E}$ und $I(\tilde{G}) \leq I(G_{LISE} - 1)$ gilt. Weil T in E_{LISE} wegen Zeile 8 im Code sein kann, existiert die Kante $(u,v) \in E$, die in Zeile 5 im Code gewählt wurde und für die kein Pfad $p(u,v)$ in \tilde{G} existiert mit $|p| \leq t|u,v|$. Somit erfüllt \tilde{G} und damit auch G^* die t -Spann-Eigenschaft nicht, was zum Widerspruch führt.

LLISE

- 1: collect $(\frac{t}{2})$ -neighborhood $G_N = (V_N, E_N)$ of $G = (V, E)$
- 2: $E' = \emptyset$
- 3: $G' = (V_N, E')$
- 4: **repeat**
- 5: $f = \text{edge} \in E_N$ with minimum coverage
- 6: move all edges $\in E_N$ with coverage $Cov(f)$ to E'
- 7: $p = \text{shortestPath}(u - v)$ in G'
- 8: **until** $|p| \leq t|u,v|$
- 9: inform all edges on p to remain in the resulting topology.

Note: $G_{LL} = (V, E_{LL})$ consists of all edges eventually informed to remain in the resulting topology.

Dieser Algorithmus geht nach drei Schritten vor:

1. Erforsche die $(\frac{t}{2})$ -neighborhood
2. Berechne minimale Interferenz für e
3. Informiere alle Kanten auf dem Pfad in der Topologie zu bleiben

Theorem 5. *The Graph $G_{LL} = (V, E_{LL})$ constructed by LLISE from a given network $G=(V,E)$ is an interference minimal t -spanner of G .*

Die t -Spann-Eigenschaft wird wie bei LISE bewiesen. Um die Interferenzoptimalität zu beweisen reicht es aus zu zeigen, dass jeder berechnete 'Spann-Pfad' für jede Kante $e = (u,v) \in G$ von LLISE interferenzoptimal berechnet wird, wobei die Interferenz der Pfades die Interferenz der Kante mit der größten coverage ist. Der Grund ist, dass nur die auf dem Pfad liegenden Kanten in die spätere Topologie übernommen werden. Ist G_{LL} nicht optimal, so ist mindestens einer der 'Spannpfade' auch nicht optimal.

Nun schauen wir uns den Algorithmus für eine beispielhaft Kante $e=(u,v)$ an. Angefangen mit $E' = \emptyset$, werden in Zeile 6 des Codes Kanten von E fortlaufend in E' eingefügt, bis der 'Spannpfad' p von u nach v in Zeile 8 gefunden wird. Da LLISE die in aufsteigender Interferenz in E' einfügt und p der erste 'Spannpfad' ist, so ist p ein Interferenz-Optimaler t -Spann-Pfad. Die $(\frac{t}{2})$ -neighborhood fügt alle 'Spannpfade' von u nach v ein und somit auch den Interferenz-Optimalen. So ist es LLISE nicht möglich den Interferenz-Optimalen t -Spann-Pfad, wegen des lokalen Wissens über G nicht zu sehen. In Folge dessen ist p der Interferenz-Optimale t -Spann-Pfad von e .

5 Interferenz Average-Case Graphen

In diesem Kapitel wird die Kompetenz der Algorithmen an einem zufällig verteilten Netzwerk fester Größe und jeweiliger Dichte getestet und ausgewertet.

5.1 Konnektivitätssparende Topologien

Bei der Simulation (sprich einer randomisierten Verteilung) von rund 1000 Netzwerken pro Dichte wurden nun die Ergebnisse in einem Diagramm verdeutlicht. Bis zu einer Dichte (wobei hier Dichte "Einheiten pro Feld fester Größe heißt") von fünf Knoten auf einem Feld nehmen sich die einzelnen Graphen nicht allzu viel. Darüber hinaus jedoch ist abzusehen, dass in jedem Fall ein Netzwerk "LIFE" - gesteuert, um ein großes Maß effektiver ist als jeder andere (hier Gabriel- und Relative Neighborhood - Graph). Die Interferenz bleibt ab einer fünfer Dichte sogar fast konstant.

5.2 Niedrig interferenzierte Spanner

Bei diesem Test ging es wiederum um den Vergleich mehrerer Topologien, diesmal wurde jedoch mit LISE die Spann-Eigenschaft erhalten und der Streckfaktor variiert, um diesen dann mit "LIFE" zu vergleichen, also der Interferenz minimalsten Lösung, jedoch wie bereits beschrieben ohne die Spann-Eigenschaft zu erfüllen.

Die durch unsere Algorithmen resultierenden Graphen können sich in zweierlei Hinsicht sehen lassen, nicht nur, das bereits LISE mit Streckfaktor 2 den Relative Neighborhood Graphen in der Interferenz unterbot, auch ein 10 fach gestreckter LISE konnte fast die idealwerte von LIFE erreichen.

6 Zusammenfassung

In dieser Arbeit wurde in vielerlei Hinsicht wiederlegt, dass die bis dato bekannten Topologien, einen falschen bzw. nicht ausreichend präzise definierten Ansatz verfolgten, um konkret Interferenzen zu senken. Zudem konnte für einige spezielle Aufgaben, Algorithmen liefern, die bewiesenermaßen optimale Ergebnisse lieferten.