Zusammenfasung Lokale Netzstrukturen

Carsten Hartenfels, Marco Brack, turbopope

2016-02-23

Fragen und Unsicherheiten

Diese Sachen kommen in den Klausurvorbereitungsfragen vor oder wurden in der freymütigen Zusammenfassung in der letzten Vorlesung genannt, sind aber nicht in der Zusammefassung beantwortet.

- Was sind $O-$, Θ- und Ω-Spanner?
- Wofür sind die Tower-Konstruktionen gut?

Grundlagen

Netzstrukturen

Gewöhnlicher Weise sind mobile Endgeräte mit einer **infrastructure**-Verbindung an ein größeres, hierarchiches Netzwerk angebunden. Es gibt jedoch auch so genannte **ad-hoc**-Netze, bei denen mehrere Endgeräte ein kleines, abgeschlossenes Netzwerk mit einander bilden. Es wird unterschieden zwischen **single-hop ad-hoc**-Netzwerken, bei denen alle Geräte direkt mit einander verbunden sind, und **multi-hop ad-hoc**-Netzwerken, bei denen eine komplexere Gliederung vorliegt und in denen Geräte nicht unbedingt *direkt* mit einander verbunden sein müssen.

Ein **Wireless Sensor Network (WSN)** ist ein solches ad-hoc-Netzwerk mit vielen kleinen Sensor-Knoten und wird zum Beispiel zur Erkennung von Waldbränden verwendet.

Ein **Wireless Sensor Actuator Network (WSAN)** erweitert dieses Konzept um Knoten, die nicht nur messen, sondern auch direkt Aktionen ausführen können, zum Beispiel um bei großen Gebäuden automatisch die Fenster zu schließen (es gibt hier anscheinend keine tollen Beispiele).

In **Roboternetzen** können sich die Knoten zusätzlich selbstständig bewegen. Beispiel: Drohnen überwachen ein kontaminiertes Gebiet.

Schwierigkeiten in Drahtlosnetzwerken

Ein charakteristisches Problem in Drahtlosnetzwerken ist die **Mehrwegeausbreitung** der übertragenen Signale. Diese kann durch eine Vielzahl von Ursachen hervorgerufen werden, die häufigste davon Reflektion des Signals an Oberflächen und damit Überlagerung des **Line of Sight (LOS)**-Signals mit phasenverschobenen Signalen. Dies führt zum typischen **Fading**-Verhalten von Drahtlosnetzwerken.

Zusätzlich können andere Übertragungen im Gebiet das Signal überlagern. Um solche großen **Kollisionsdomänen** zu vermeiden sind **multi-hop**-Netzwerke mit geringer Übertragungsreichweite pro Knoten sinnvoll.

Weiterhin schränkt die begrenzte **Batteriekapazität** die mögliche Sendeleistung ein. Um eine höhere **Energieeffizienz** zu erreichen können **Schlaf-Wach-Zyklen (sleep-cycles)** für die Übertragungskomponenten genutzt werden. Des weiteren können manche Berechnungen auf den Sensordaten bereits vor dem Weitersenden von Netzwerkknoten übernommen werden (**in-network-processing**).

Modellbildung

Pfadverlust

Auf Grund der räumlichen Ausbreitung des Signals nimmt dessen Stärke exponentiell ab. Der Zusammenhang zwischen Sendeleistung und Empfangsleistung ist:

$$S_{RX} = a \cdot \frac{P_{TX}}{d^{\alpha}}$$

Wobei S_{RX} die Empfangsleistung, a ein Übertragungsfaktor, P_{TX} die Übertragungsleistung, d die Distanz und α die **Path-Loss Exponent** ist (*Nicht sicher ob korrekt*). Dieser **Pfadverlust** kann umgeformt in Dezibel ausgedrückt werden durch:

$$PL(d) = 10 \cdot \alpha \cdot log_{10}(d) + X_{\delta}$$

Wobei X_{δ} eine gaußsche Zufalls-Variable mit Varianz δ ist. Alternativ kann die Verteilung der Zufallsvariable durch zwei Fading-Modelle beschrieben werden:

- **Rayleigh-Fading**: Keine Line of Sight. X ist exponentiell verteilt
- **Ricean-Fading**: Dominante Line of Sight. X ist ricean-verteilt

Vereinfacht kann die benötigte Leistung für eine Übertragung $f(d)$ angegeben werden durch:

$$f(d) = a \cdot d^{\alpha} + b$$

Mit Distanz d, Übertragungsfaktor a, Path-Loss Exponent α und additiver Konstante b (*???*).

Graphenmodelle

Der **Unit-Disk-Graph (UDG)** beschreibt einen Graphen auf einer Menge Knoten, bei dem jeder Knoten mit jedem Knoten verbunden ist, der innerhalb seines **Unit-Disk-Radius** R liegt. Dies modelliert die Übertragungsreichweite jedes Knoten. Der **Quasi-Unit-Disk-Graph (QUDG)** erweitert den UDG um eine minimal-Radius (im QUDG heißen die Parameter r_{max} und r_{min}).

Topologiekontrolle

Durch die limitierte Übertragungsreichweite hat ein drahtloses Netzwerk bereits eine implizite Struktur. Um eine Netztwerkstruktur noch weiter zu vereinfachen und somit das **Routing** von Nachrichten zu erleichtern und die Energieeffizienz zu erhöhen, können verschiedene **Topologiekontrollen** zum Einsatz kommen. Die wichtigsten, in dieser Vorlesung teilweise detailliert besprochenen, sind:

- **Neighbor elimination** (Kanten streichen)
- **Backbone construction**
- **Virtual overlays**
- **Relocation**

Ein **lokaler Algorithmus** (für eine Topologiekontrolle) erreicht ein netzwerkweites Ziel mit (Knoten-)lokalen Entscheidungen, während ein **globaler Algorithmus** Wissen über den Zustand des gesamten Netzwerkes mit einbezieht. Lokale Algorithmen haben folgende Vorteile:

- Resourcen sparen
- Beliebig große Netzwerke ermöglichen
- Besser mit dynamischen Änderungen klar kommen
- Arbeiten besser wenn keine gesamte Sicht auf das Netzwerk verfügbar ist

Datenkommunikation

Datenkommunikation bezeichnet die Problematik, eine Nachricht möglichst effizient durch ein nicht oder nur tweilweise bekanntes Netzwerk zu schicken.

Graphen

Ein **Graph** ist definiert als: $G = (V, E)$, wobei $v \in V$ **Knoten** und $(u, v) \in E$ **Kanten** sind.

Die Kanten sind Teilmenge aller Knotenverbindungen: $E \subseteq V \times V$. Falls $E = V \times V$, dann ist G ein **vollständiger Graph**.

Ein **ungerichteter Graph** hat Verbindungen, die immer in beide Richtungen verlaufen: $uv \in G \Leftrightarrow vu \in G$.

Seien $G = (V_G, E_G)$ und $H = (V_H, E_H)$ Graphen.

Falls $V_G \subseteq V_H$ und $E_G \subseteq E_H$, dann ist G ein **Subgraph** von H: $G \subseteq H$. Im Umkehrschluss ist H ein **Supergraph** von G: $H \supseteq G$

Klausurrelevant: wenn G ein Spanner, dann H auch ein Spanner. *TODO: warum? Wird freymütig abgefragt.*

Der **Schnitt** der Graphen ist definiert als: $G \cap H = (V_G \cup V_H, E_G \cup E_H)$.

Ein **Pfad** ist eine Reihe von Knoten, die mit Kanten verbunden sind: $p = (v_1 ... v_n), v_i v_{i+1} \in E$.

G ist **verbunden**, wenn man zwei beliebige Knoten mit einem Pfad verbinden kann.

Als **k-hop-Nachbarn** $N_k(v)$ von Knoten v werden die Knoten bezeichnet, die k oder weniger Schritte von v entfernt sind. Ohne explizites k gilt $N(v) = N_1(v)$.

Knoten u und v **sehen einander**, wenn sie Nachbarn voneinander sind: $u \in N(v) \wedge v \in N(w)$.

Graphentypen

Gewichteter Graph

Zusätzlich zum Graphen gibt es noch eine Funktion f, die das Gewicht einer Kante berechnet: $(G, f) = ((V, E), f)$ mit $f : E \to \mathbb{R}$.

Bei einem gewichteten Graphen kann man die **Pfadlänge** eines Pfades berechnen, indem man alle Kantengewichte im Pfad zusammenzählt: $\hat{f} = \sum_{i=1}^{n-1} f(v_i v_{i+1})$

Euklidischer Graph

Ein Graph im euklidischen Raum (normale Geometrie).

Die Gewichtung ist die euklidische Distanz zwischen zwei Punkten $p = (p_1...p_n)$ und $q = (q_1...q_n)$ im n-dimensionalen Raum: $\|pq\| = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$

Topologischer Graph

Laut Dausi ein gewichteter Graph, bei dem die Kantengewichtfunktion aber nicht nur die Distanz zwischen den Punkten ist.

Topologiekontrolle

Eine **Topologiekontrolle** ist eine Abbildung $\tau : C \to D$ eines Graphen C auf einen Graphen D. D ist ein Subgraph von C. Eine Topologiekontrolle kann Kanten und Knoten entfernen.

Die **k-lokale Sicht** $G[v, k]$ eines Knotens v ist ein Subgraph von G, bestehend aus v, seinen k-hop Nachbarn und den Pfaden zu diesen. Vereinfachung: $G[v]$ ist die 1-lokale-Sicht von v.

Eine **k-lokale Topologiekontrolle** ist eine Topologiekontrolle, die nur die k-lokale Sicht eines Knotens betrachtet, um Entscheidungen zu treffen.

UDG (Unit-Disk Graph)

Ein UDG hat einen Radius r. Mit diesem Radius zieht man einen Kreis um eine Knotenmenge V. $UDG(V)$ macht dann eine Kante zwischen allen Punkten, die in einander Kreisen liegen.

QUDG (Quasi-Unit-Disk Graph)

In der Realität kann man nicht einfach um z.B. einen WLAN-Router einen Kreis ziehen und sagen, dass alles im Kreis sich zum Router verbinden kann. Stattdessen nimmt man zwei Radien r_{min} und r_{max} ($r_{min} \leq r_{max}$) und zieht damit zwei Kreise um eine Knotenmenge V. Der innere Kreis garantiert eine Verbindung, dahinter nur vielleicht und hinter dem äußeren Kreis gibt es keine Verbindung mehr.

Besonders interessant ist wenn $\frac{r_{max}}{r_{min}} \leq \sqrt{2}$, aber warum das so ist weiß keiner.

2D-Geometrie

Der **Bisektor** zweier Punkte u und v ist die Menge aller Punkte, die zu u und v den gleichen Abstand haben: $B(u,v) = bisektor(u,v) = \{x : \|xu\| = \|xv\|\}$

Die **geschlossene Halbebene** zweier Punkte u und v ist die Menge aller Punkte, die näher zu u liegen als zu v: $H(u,v) = halfPlane(u,v) = \{x : \|xu\| \leq \|xv\|\}$

Der **Kreis** um einen Punkt u mit dem Radius r ist die Menge aller Punkte, die zu u den Abstand r haben: $C_r(u) = circle(u,r) = \{x : \|xu\| \leq r\}$. Weitere relevante Kreise sind der Kreis, der die Punkte u, v und w umschließt $C(u,v,w) = circle(u,v,w)$ und der kleinste Kreis, der die Punkte u und v umschließt $C(u,v) = circle(u,v)$.

Des Weiteren wird der **Winkel** zwischen zwei Geraden uv und vw wie folgt notiert: $\measuredangle uvw = angle(u,v,w)$

Spanner

Man nehme zwei beliebige Knoten $p, q \in G$. Dann setzt man den tatsächlich kürzesten Pfad zwischen den beiden Knoten mit der Pfadlänge der Knoten aus G ins Verhältnis. Wenn man dieses Verhältnis auf einen Stretch-Faktor („Umständlichkeit") c beschränken kann (O, Θ, Ω), dann ist G ein c-Spanner.

Man kann auch die Pfade in zwei Graphen ins Verhältnis setzen und so Spanner über Umwege definieren. Spanner sind nützlich, weil man damit trotz nur eingeschränktem Wissen sich dem kürzesten Pfad annähren kann.

In einem topologischen, gewichteten Graphen G ist der **kürzeste Pfad** p_{min} zwischen zwei Knoten u und v der Pfad, für den die Pfadlänge minimal ist. Er wird wie folgt notiert: $p_{min} = \Gamma_G(u,v)$

In einem euklidischen Graphen ist das Gewicht der Kanten durch die Distanz der Knoten gegeben und der kürzeste Pfad p_{min} wird notiert mit $p_{min} = \Pi_G(u,v)$.

Ein **euklidischer c-Spanner** ist ein Graph, bei dem für alle Knoten das Verhältnis zwischen kürzester Pfadlänge und tatsächlicher euklidischer Distanz den **stretch-factor** c nicht überschreitet. Formell:

$$\forall u, v \in G : \frac{\|\Pi_G(u,v)\|}{\|uv\|} \leq c$$

Sei H ein euklidischer Graph und G ein Subgraph von H. Nun wird das Verhältnis zwischen der kürzesten Pfadlänge des Subgraphen und der des ursprünglichen Graphen gemessen. G ist ein **euklidischer Graph-Spanner**, wenn dieser stretch-factor c nie überschreitet:

$$\forall u, v \in G : \frac{\|\Pi_H(u,v)\|}{\|\Pi_G(u,v)\|} \leq c$$

Für einem topologischen Graphen gilt das gleiche, aber es werden die topologischen Pfadlängen betrachtet:

$$\forall u, v \in G : \frac{|\Gamma_H(u,v)|}{|\Gamma_G(u,v)|} \leq c$$

Wichtig für die ganzen Teile die folgen: wenn $F \subseteq G \subseteq H$ und F ein t-Spanner von H, dann ist auch G ein t-Spanner. Damit kann man folgende Ungleichung bauen:

$$\underbrace{UDel(V)}_{UDG-Spanner} \subseteq LDel^{(k)}(V) \subseteq ... \subseteq LDel^{(2)}(V) \subseteq PlDel(V) \subseteq LDel^{(1)}(V)$$

Planare Graphen

Graphentheoretisch: Graph G ist planar, wenn G auf die Ebene gezeichnet werden kann, sodass sich keine Kanten schneiden.

Definition hier: Graph G ist als Zeichnung auf der Ebene gegeben. Gesucht ist ein Teilgraph H, der keine schneidenden Kanten enthält. Dieser Graph H wird dann als planar bezeichnet.

EMST (Euclidean Minimal Spanning Tree)

Das euklidische Gewicht eines euklidischen Graphen ist

$$\|G\| = \sum_{uv \in G} \|uv\|$$

Der **EMST** über eine Punktemenge V ist der zusammenhängende Graph, für den $\|V\|$ minimal ist. Ist $UDG(V)$ verbunden, so ist $EMST(V) \subseteq UDG(V)$ (*Beweis ausgelassen*). EMST ist jedoch keine k-lokale Topologiekontrolle (*Beweis ausgelassen*).

Der EMST über V lässt sich über den Algorithmus von Kruskal herstellen: Zeichne in jedem Schritt die kurzmöglichste Kante, die keinen Zyklus erzeugen würde, bis der Graph zusammenhängend ist.

LMST (Local EMST)

Der **LMST** ist eine Graphenstruktur, die „nahe am“ EMST ist. Gegeben sei ein Graph G mit zwei Knoten u und v und der Kante uv. Wendet man LMST als Topologiekontrolle an, bleibt uv im Graphen, wenn:

$uv \in LMST(G) \Leftrightarrow uv \in EMST(N(u)) \wedge uv \in EMST(N(v))$

Das heißt: Wenn uv und vu im EMST über die Nachbarn des jeweils anderen Knotens enthalten sind, bleibt uv bestehen (hier etwas kompliziertere Definition um ungerichtete Graphen mit einzubeziehen).

(*Problem mit gleichen Kantenlängen ausgelassen.*)

Der LMST ist eine 2-lokale Topologiekontrolle (*Beweis ausgelassen*).

Ist $UDG(V)$ verbunden, so ist $EMST(V) \subseteq LMST(V)$ (*Beweis ausgelassen*).

Der LMST hat maximalen Grad 6. Wikipedia.

RNG (Relativer Nachbarschaftsgraph)

V sei eine Menge von Punkten, die zu einem Graphen verbunden werden sollen.

$$uv \in RNG(V) \Leftrightarrow \|uv\| \leq max\{\|uw\|, \|vw\|\} \quad \forall w \in V \backslash \{u, v\}$$

Alternativ: Sei Radius $r = \|uv\|$. Sei c_u die Menge der Punkte im Kreis um u mit Radius r: $c_u = circleSet(u, r)$ und c_v das Gleiche für v. Dann ist

$$uv \in RNG(V) \Leftrightarrow \nexists w \in V : w \in c_u \cap c_v$$

Das heißt: Die Schnittmenge (**Lune**) der Kreise um u und v mit mit deren Abstand als Radius enthält keine weiteren Knoten.

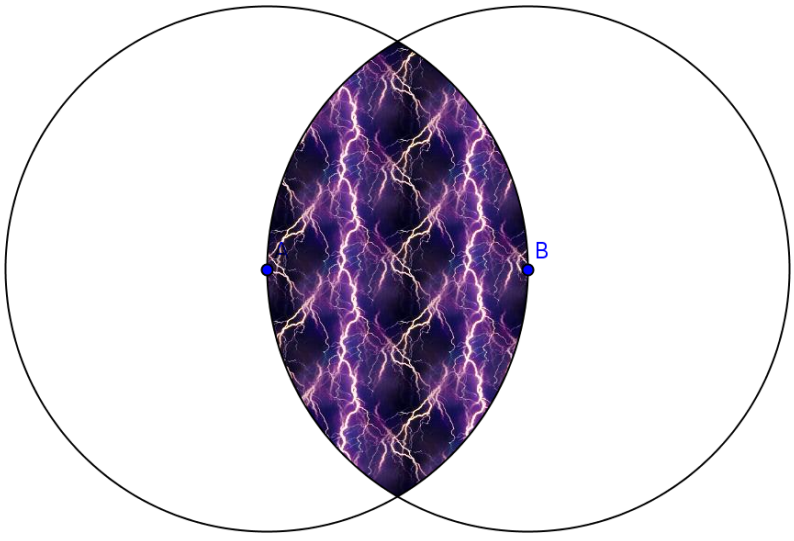

Figure 1: RNG – Im Blitzgewitter dürfen keine Punkte sein.

Der **Unit- Relativer Nachbarschaftsgraph (URNG)** ist die Schnittmenge aus RNG und UDG:

$$URNG(V) = RNG(V) \cap UDG(V)$$

Der URNG ist eine 1-lokale Graphkonstruktion und $LMST(V) \subseteq URNG(V)$ (*Beweis ausgelassen*).

Sei V eine Punktemenge mit n Punkten. Dann ist

$$\frac{\|\Pi_{URNG}(u,v)\|}{\|\Pi_{UDG}(u,v)\|} \leq n-1$$

(*??? Keine Ahnung was das soll, Beweis ausgelassen*)

GG (Gabriel-Graph)

Man malt einen Kreis um jedes Knotenpaar, sodass der Mittelpunkt des Kreises genau zwischen den beiden Punkten liegt. Der $GG(V)$ macht eine Kante zwischen alle Knoten, bei denen kein anderer Knoten im Knotenpaarkreis liegt.

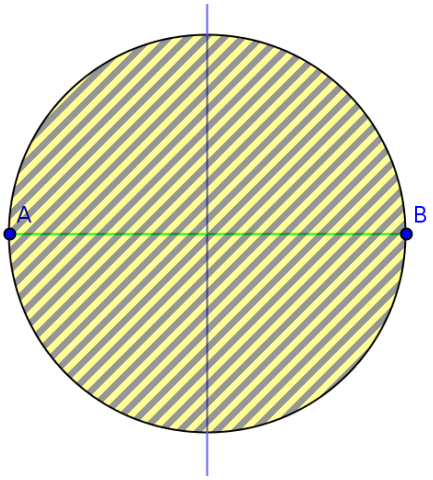

Figure 2: GG – Im Bienenbereich dürfen keine Punkte sein

Der **Unit-Gabriel-Graph** $UGG(V) = GG(V) \cap UDG(V)$ ist eine lokale Konstruktion.

Voronoi-Diagramm

Gegeben sein eine Punktemenge S. Die **Voronoi-Region** um einen Punkt $u \in S$ wird konstruiert, indem man die Halbebene zu jedem anderen Punkt in S zeichnet. Alles was auf der Halbebene von u liegt, ist Teil der Voronoi-Region.

Formal:

$$VR_S(u) = \bigcap_{v \in S \setminus \{u\}} H(u, v)$$

Beispiel:

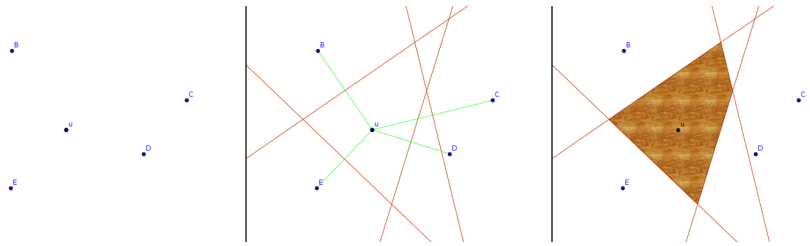

Figure 3: Voronoi-Region mit rustikaler Holzmaserung

Das komplette **Voronoi-Diagramm** $VD(S)$ erhält man, wenn man die Regionen aller Punkte vereinigt: $VD(S) = \{VR_S(v) \mid v \in S\}$

Kanten, die mehr als eine Region delimitieren, sind die **Voronoi-Kanten**. Sie können Geraden, Halbgeraden oder Strecken sein.

Die Endpunkte der Kanten sind die **Voronoi-Knoten**. Sie haben mindestens 3 ausgehende Kanten. Wenn sie genau 3 Kanten haben, sind sie **nicht-degeneriert** und wenn sie mehr als 3 Kanten haben, sind sie **degeneriert**. Wenn ein Voronoi-Diagramm einen degenerierten Knoten hat, dann ist das ganze Diagramm degeneriert.

Das ganze kann man mit diesem Simulator ausprobieren. Beispielergebnis (farbige Gebiete sind die Voronoi-Regionen, deren Grenzen die Voronoi-Kanten und die Enden der Kanten die Voronoi-Knoten):

Delaunay-Triangulierung

Drei mögliche Definitionen für $Del(S)$:

1. Verbinden aller Punkte in S, deren Voronoi-Region eine gemeinsame Kante haben.

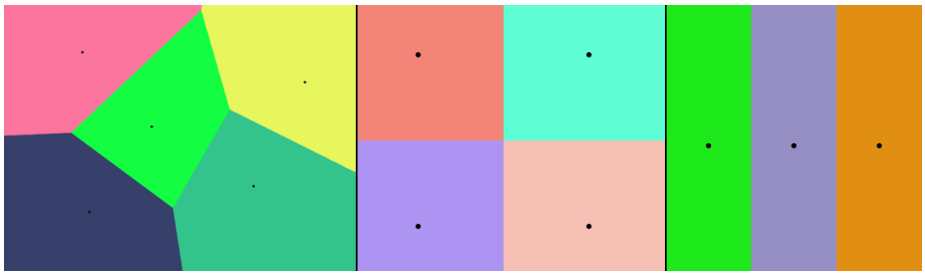

Figure 4: Voronai-Diagramme: normal, degenerierter Krüppel, Geraden

2. Verbinde $u, v, w \in S$, wenn ihr Kreis keine weiteren Punkte enthält.
3. Verbinde $u, v \in S$, wenn es einen Kreis gibt, der die beiden Punkte tangiert, aber keine anderen Punkte beinhaltet. (*Hinweis: Der Kreis muss nicht minimal sein, das wäre dann GG.*)

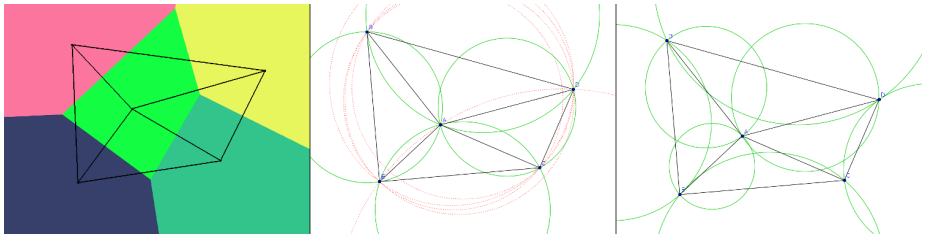

Figure 5: Delaunay-Definitionen

Die drei Definitionen sind in gewissen Fällen nicht äquivalent. Darum macht man zwei vereinfachende Annahmen über S, womit diese Fälle ausgeschlossen werden:

1. **Nicht-Cozirkularität**: es gibt keine vier Punkte, die auf einem Kreis liegen $\Leftrightarrow$ kein Voronai-Knoten ist degeneriert.
2. **Nicht-Colinearität**: es gibt keine drei Punkte, die auf einer Geraden liegen $\Leftrightarrow$ keine Voronai-Kante ist eine Gerade.

Delauny-Triangulierung ist planar und keine lokale Topologie.

UDel (Unit-Disk-Delaunay-Triangulierung)

$UDel = Del \cap UDG$. Keine lokale Topologie, aber wichtiges Vehikel um Spanning Properties von k-lokalen Graphen zeigen zu können.

*TODO: Grundidee Keil-Gutwin-Beweis, Kreissortierung über Θ

$UDel$ ist UDG-Spanner.

DT-Pfade (Delauney-Triangulierungspfade)

TODO: Dobkin Beweis (versteh ich nicht), Trick mit kürzestem Pfad wird evtl. abgefragt*

Ein **direkter DT-Pfad** von u nach v mit $u, v \in S$ schreitet durch die Knoten von benachbarten Voronai-Regionen direkt von u nach v. Wenn alle Knoten des Pfades auf einer Seite der Geraden durch u und v liegen, dann ist der Pfad **einseitig**.

Ein einseitiger Pfad hat eine nach oben beschränkte Pfadlänge von $\frac{\pi}{2}\|uv\|$. Er besucht nur Kanten aus $UDG(S)$.

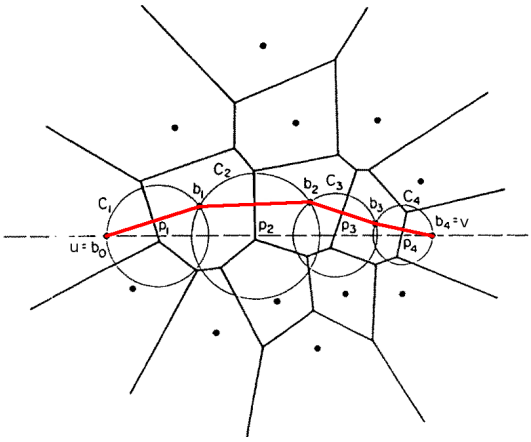

Figure 6: Einseitiger direkter DT-Pfad, aus dem Dobkin-Beweis-PDF, modifiziert

LDel (Lokale Delaunay-Triangulierung)

Man schränkt Del auf die k-Hop-Nachbarn ein. So erhält man $LDel$.

Ein Dreieck mit den Punkten $u, v, w \in S$ ist ein k-lokales Dreieck, wenn sie nach $UDG(S)$ verbunden sind und $C(u, v, w)$ keine weiteren Punkte enthält.

$LDel^{(k)}(V)$ besteht aus allen k-lokalen Dreiecken und allen Kanten aus UGG (damit der Graph zusammenhängt).

$UDel(S) \subseteq LDel^{(k)}(S) \Rightarrow LDel^{(k)}(S)$ ist ein UDG-Spanner. $LDel^{(1)}(V)$ ist nicht unbedingt planar.

PlDel (Planare Delaunay-Triangulierung)

TODO, niedrige Priorität, wird nicht unbedingt in der Klausur drauf eingegangen

RDG (Restricted Delaunay Graph)

$RDG(V)$ für $UDG(V)$

$$uv \in RDG(V) \Leftrightarrow uv \in UDG(V) \wedge \forall w \in N(u) \cap N(v) : uv \in Del(N(w))$$

Eine Kante uv zwischen u und v aus V ist Teil des RDG über V genau dann, wenn uv im UDG über V ist und uv in der Delauney-Triangulierung der Nachbarn von w ist, für jeden Punkt w in der gemeinsamen Nachbarschaft von u und v.

Es gilt $UDel(V) \subseteq RDG(V)$.

PDT (Partielle Delaunay-Triangulierung)

Wir nehmen uns den $UDG(S)$ mit Radius r. Die $PLT(S)$ verbindet alle Punkte $u, v \in S$ für die gilt:

1. $\|uv\| < r$
2. Und eins der folgenden Optionen:
   - $uv \in GG(S) \cap UDG(S)$
   - $C(u, v, w) \cap N(u) \setminus \{u, v, w\} = \emptyset$ und $sin(\alpha) \geq \frac{\|uv\|}{r}$, wobei $w \in S \setminus \{u, v\}$ der winkelmaximierende Knoten

PDL ist ein UDG-Spanner und ein UDel-Spanner. Er ist identisch mit dem $PuDel$.

PuDel (Partielle ungerichtete Delauney-Triangulierung)

TODO, beim Beweis verstehen wie „somit: PuDel ist blabla – UDG – Spanner" funktioniert, Gleichheit PuDel und PDT wird er nicht so genau fragen

Irgendwie Teilschritte in UDel und für jeden Teilschritt zeigen, dass dort ein PuDel-Weg drin ist.

$PuDel$ und PDT sind gleich, wird aber in der Klausur nicht so genau gefragt.

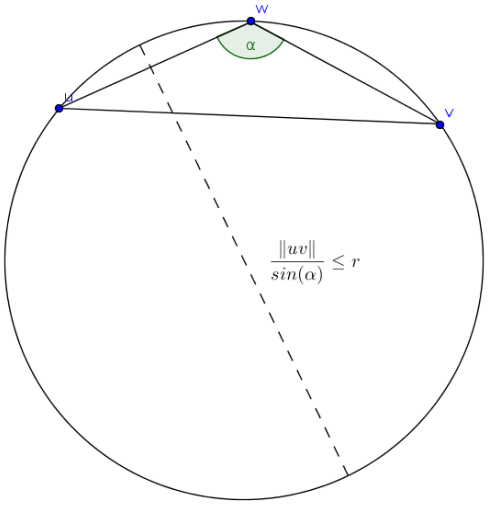

Figure 7: PDT visualisiert

Das ganze weitere Graphengedöns...

...steht in den Folien. Wichtig sind:

- Ideen des Chu-Lin/Edmunds-Algorithmus
- Backbone
  - Dominating Set
  - Grundidee des Verhaltens (Lightweight Construction)
  - Verbesserungen
  - Erweiterungen „schon noch drauf haben"

Den Rest nicht mehr.

Übersicht

Name	Akronym	k-lokal	planar	Supergraphen	t-Spanner
Unit Disk Graph	**UDG**	-	nein		
Quasi Unit Disk Graph	**QUDG**	-	nein	UDG	
Euclidean Minimal Spanning Tree	**EMST**	nein	ja	UDG, LMST, Yao	
Local EMST	**LMST**	2	ja	URNG	

Name	Akronym	k-lokal	planar	Supergraphen	t-Spanner
Relativer Nachbarschaftsgraph	**RNG**	nein	ja		
Unit RNG	**URNG**	1	ja	UGG	Θ-RNG
Gabriel Graph	**GG**	nein	ja		$\Omega(\sqrt{n})$-UDG
Unit GG	**UGG**	1	ja		
Delauney Triangulierung	**Del**	nein	ja		2.24
Unit Del	**UDel**	nein	ja	LDel, RDG	UDG
1-Lokale Del	**LDel**^{1}	1	nein	PlDel (für $LDel^1$)	$\frac{1+\sqrt{5}}{2}\pi$-UDG
k-Lokale Del	**LDel**^{k}	k	ja	Udel	$\frac{1+\sqrt{5}}{2}\pi$-UDG
Planarized LDel	**PlDel**	?	ja	$LDel^2$	
Restricted Delauney Graph	**RDG**	2	ja		UDG
Partielle Del	**PDT**	1	ja		$\frac{\pi}{2}$-Udel, $\frac{1+\sqrt{5}}{2}\pi\frac{\pi}{2}$-UD
Yao-Graph	**Yao**	1	nein		

Datenkommunikation

Das gezielte versenden von Nachrichten (**Routing**) in einem WSN muss viele Einschränkungen und Hindernisse überwinden, allen voran die limitierte Energiekapazität und damit verbunden die limitierte Sendereichweite. Dennoch möchten wir ein Routing mit geringem Aufwand, garantierter Zustellung, zyklenfreiheit und guter Qualität der Pfade.

Errichbarkeit eines Knotens soll **garantierte Zustellung** gewährleisten. Dazu nehmen wir vereinfachend an, dass jede Nachrichtenübertragung zwischen zwei Knoten erfolgreich ist (**ideal MAC layer**).

Flooding

Flooding ist das simpelste Routingverfahren. Jeder Knoten sendet die Nachricht an jeden seiner Nachbarn weiter. Dass hier Optimierungspotential besteht ist offensichtlich. Im Allgemeinen wollen wir lieber **single path Strategien**, also Routing-Strategien, bei denen es nur einen einzelnen Pfad vom Start zum Ziel gibt.

Globale vs. Lokale Routing-Strategien

Globale Routing-Strategien benutzen Wissen über den gesamten Graphen. Sie können **proaktiv** oder **reaktiv** sein und finden den kürzesten Pfad. Zustellung, zyklenfreiheit und Pfadqualität sind gewährleistet. Allerdings sind sie sehr aufwendig, wenn sie überhaupt möglich sind.

Lokale Routing-Strategien benutzen nur ihre 1- (oder k-)hop Nachbar Informationen und merken sich abgehandelten Traffic nicht. Sie sind weniger aufwändig, skalieren gut und Netzwerk-Änderungen betreffen nur Teilbereiche. Allerdings sind Zustellung, zyklenfreiheit und Pfadqualtät hier etwas schwieriger.

Localized Geographic Greedy Packet Routing

Wir gehen davon aus dass jeder Knoten seine Position kennt und dass der Start-Knoten, der die Nachricht versenden möchte, die Position des Ziels kennt (*nicht ganz klar wie*).

Beim **Greedy Packet Forwarding** wählt jeder Knoten den "besten" Nachbarn als nächsten Hop als Paket. Der "beste" Nachbar wird anhand einer **Metrik** bestimmt. Es gibt verschiedene Metriken. Andere Single-Path Strategien sind **MFR** (*auf den Folien nicht näher beschrieben*) und **DIR** (*nicht verständlich beschrieben*). Zusätzlich kann eine geeignete Topologiekontrolle das Routing verbessern.

Die einfachste Metrik ist die Distanz zum Ziel (**Geo-Routing**). Greedy Geo-Routing ist zyklenfrei.

Basierend auf der Formel für den **Pfadverlust** $u(d) = d^a + c$ (*???*) kann eine andere Metrik definiert werden, die als nächsten Hop den Knoten nimmt, für den direkte Pfadverlust und der geschätzte weitere Pfadverlust zusammen minimal ist.

Eine Metrik basierend auf dem **Cost over Progress Modell** minimiert Verhältnis zwischen den Kosten, zu einem Knoten zu springen, und dem Fortschritt, der mit dem Sprung gemacht werden würde. Dabei ist die Kostenfunktion beliebig.

Um zu vermeiden, dass durch wiederholte Benutzung eines Pfades die beteiligten Knoten schnell ihre Energie verlieren, kann eine Metrik definiert werden, die die **Restenergie** der Knoten berücksichtigt. Diese Metrik kann auch mit einer **Energiekosten**-Metrik kombiniert werden.

Um die Energienutzung noch weiter zu balancieren kann der Nächste Hop auch (ein Stück weit) **per Zufall** gewählt werden. Es wird eine gewisse Vorauswahl getroffen um sicherzustellen, dass der Hop dem Ziel auch (etwas) näher kommt, und dann wird per Zufall ein Knoten ausgewählt. Die Zufallsverteilung kann weiterhin durch eine andere Metrik gewichtet werden.

TODO: Vorteile, Nachteile, Problem mit Namen nennen

Beacon-Less-Routing

Bisher gingen wir davon aus, dass jeder Knoten seine Nachbarn kennt. Um dies zu vermeiden kann **BLR** benutzt werden:

1. Der Startknoten kennt seine Nachbarn nicht
2. Sendet blind einen Broadcast an die umliegenden Knoten
3. Die Knoten antworten nicht sofort
4. Jeder Knoten berechnet für sich selbst die Metrik
5. Je besser die Metrik, desto früher antworten die Knoten

Der Fehler von Greedy Routing

Ein "dummes" Greedy-Routing zieht keine Rückwärts-Hops in Betracht, obwohl diese zu einem insgesamt besseren Pfad führen könnten. Des Weiteren kann es in "Sackgassen" (im folgenden **konkave Knoten**) laufen und stecken bleiben.

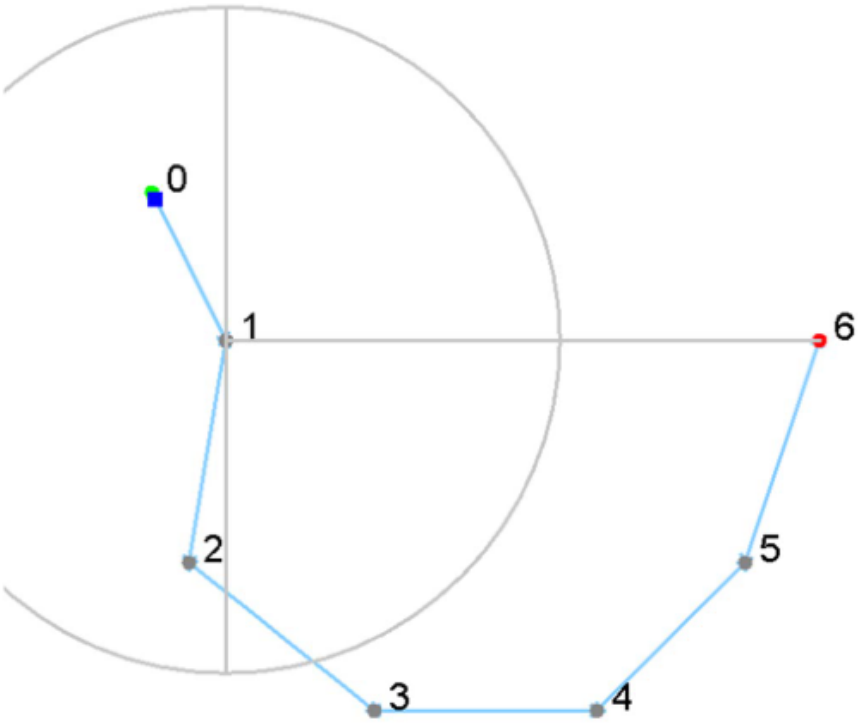

Figure 8: Greedy-Routing kommt nicht von 1 zu 6, da es nicht zu 2 springt

Um trotzdem Greedy-Routing so lange wie möglich zu benutzen, kann man im Falle eines solchen Routing-Fehlers eine **Greedy-Recovery** einschalten. Dafür müssen allerdings bereits gemachte Routing-Entscheidungen gespeichert werden, entweder in den Knoten oder in der Nachricht selbst.

Flooding-Recovery

Konkave Knoten senden die Nachricht an alle Nachbarn. Bekommen sie die selbe Nachricht noch einmal, ignorieren sie sie, um Nachrichtenschleifen zu vermeiden.

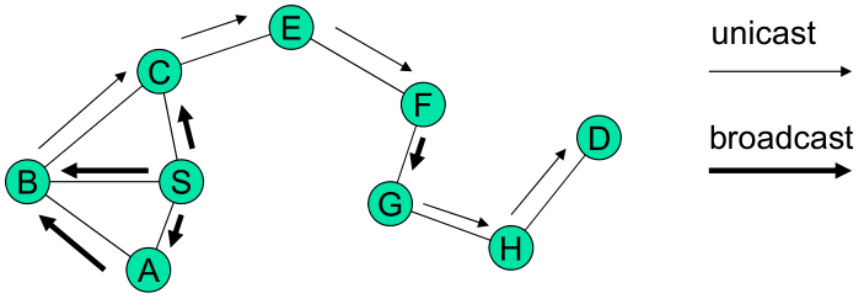

Figure 9: Nachricht von S nach D mit Flooding-Recovery

Verbesserung: **Component-Routing**. Der konkave Knoten flooded das vielversprechenste Subset seiner Nachbarn.

DFS-Based Recovery

Zur Recovery wird eine Tiefensuche (**Depth First Search, DFS**) vom konkaven Knoten gestartet. Die Kotenauswahl kann hierbei wieder auf einer der oben beschriebenen Metriken beruhen.

Face-Routing

Face-Routing unterteilt einen planaren Graphen in **Faces** und besucht dann eine Sequenz von Faces bis zum Ziel. Dazu läuft es die Grenze der Faces mit oder gegen den Uhrzeigersinn entlang (**left/right hand rule**). Der Vorteil von Face-Routing ist, dass es keine Routing-Informationen speichern muss. Es kann als Recovery-Strategie für Greedy-Routing benutzt werden.

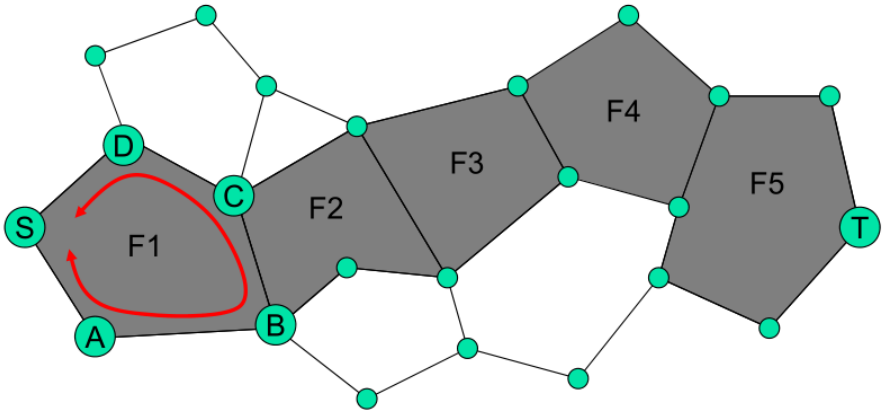

Figure 10: Faces Beispiel

Greedy-Face-Greedy (GFG)

Es wird eine Gerade vom Start zum Zielknoten gezogen. Diese durchläuft zwangsläufig eine Menge Faces, entlang der dann traverisiert werden soll. Der Algorithmus läuft die Grenze eines Faces entlang, bis sie die Gerade schneidet. Dann wechselt er zum nächsten Face.

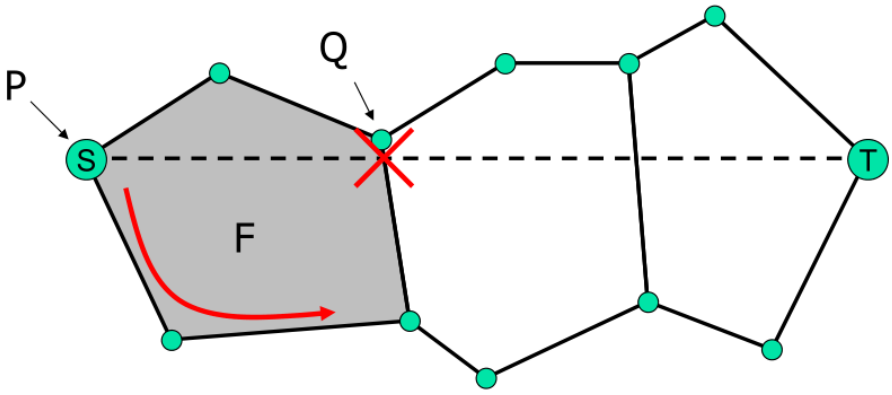

Figure 11: Beispiel Face-Routing: 1. Schritt

Greedy Other Adaptive Face Routing (GOAFR)

Ähnlich wie GFG, aber hier wird jedes Face erstmal komplett abgelaufen, und dann der Knoten ausgewählt, der am nächsten am Ziel liegt. Von diesem wird dann die Gerade neu gezogen und ab diesem traversiert.

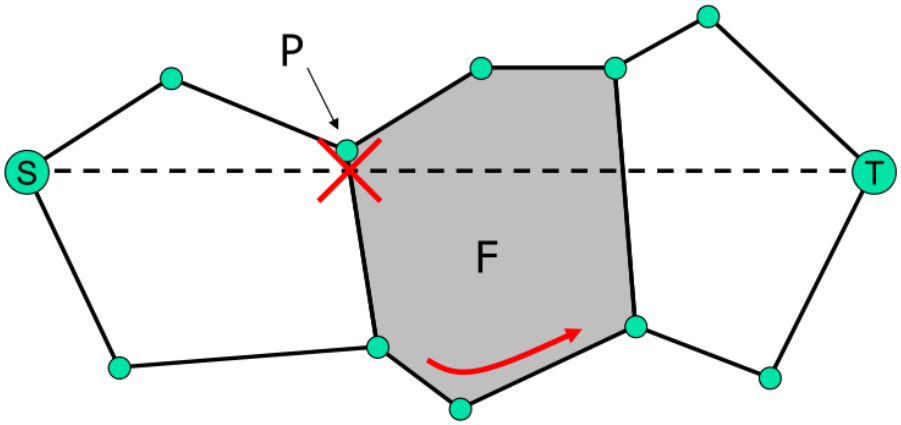

Figure 12: Beispiel Face-Routing: 2. Schritt

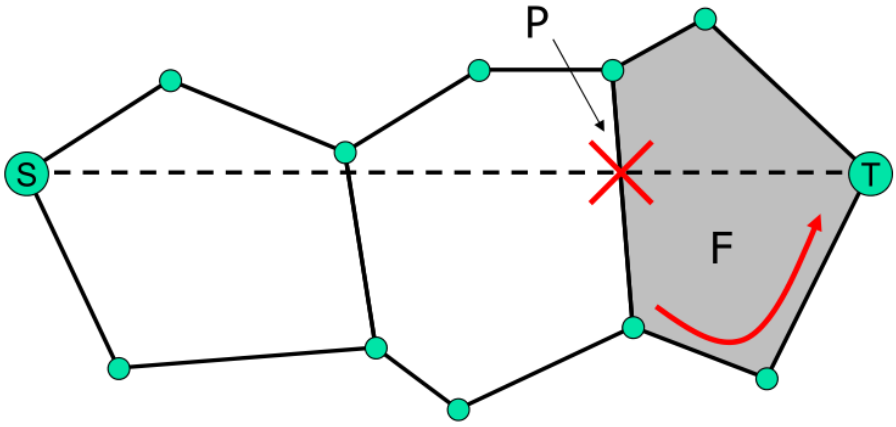

Figure 13: Beispiel Face-Routing: 3. Schritt

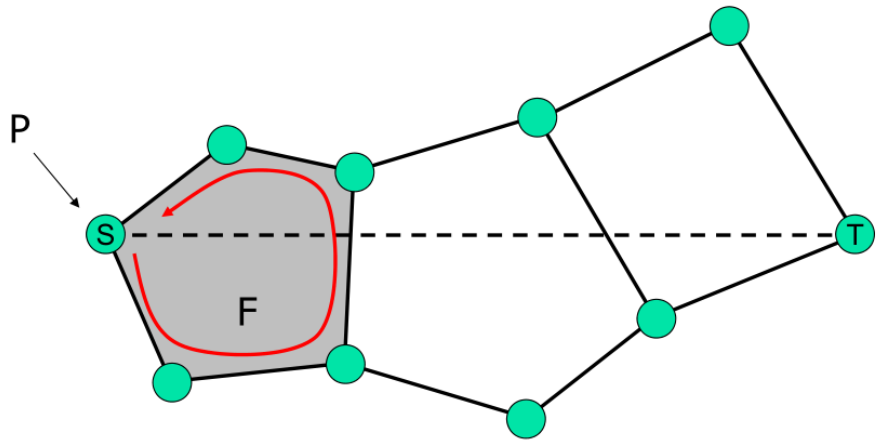

Figure 14: Beispiel GOAFR: 1. Schritt

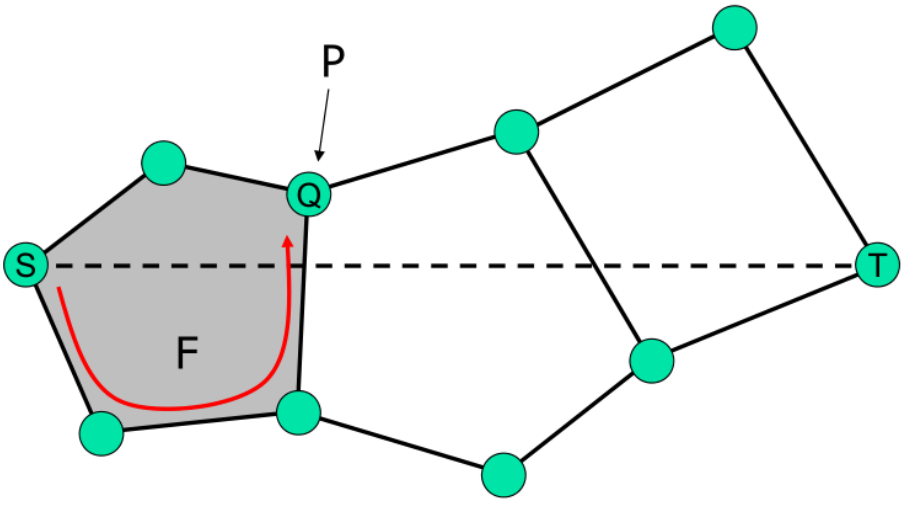

Figure 15: Beispiel GOAFR: 2. Schritt

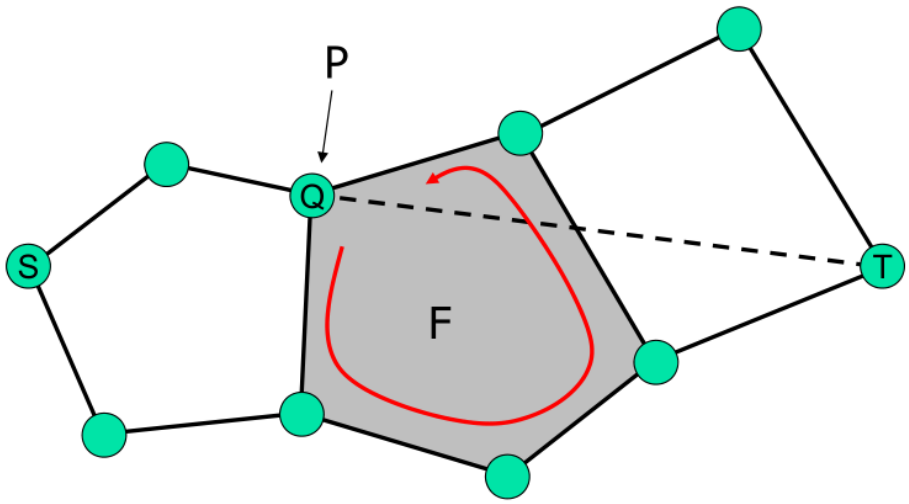

Figure 16: Beispiel GOAFR: 3. Schritt

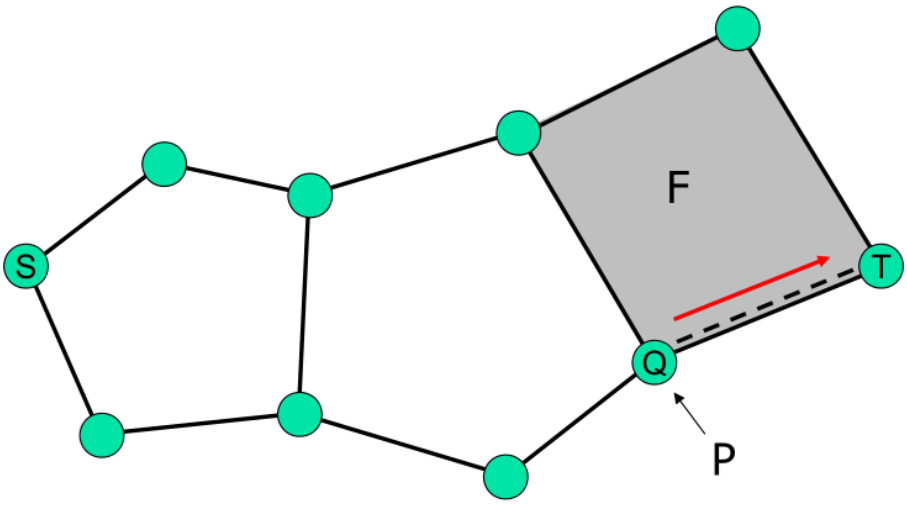

Figure 17: Beispiel GOAFR: 4. Schritt

Geographical Cluster Based Routing

Beim **Geographical Cluster Based Routing** wird ein virtuelles Overlay über den Graphen gelegt, um die Knoten zu Clustern/Faces zusammenzufassen. Cluster sind mit einander verbunden, wenn mindestens ein verbundenes Knotenpaar zwischen ihnen gibt. Das Routing folgt dann den Clustern.

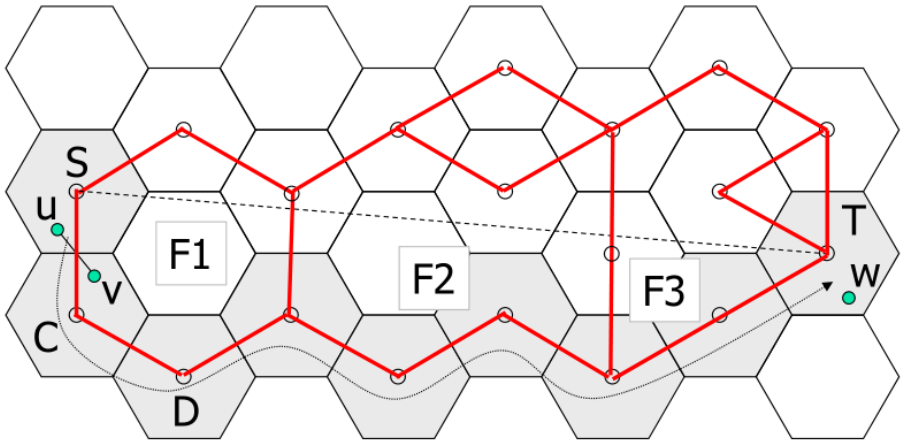

Figure 18: Geographical Cluster Based Routing

Soll der **Overlay-Graph** jedoch auf jeden Fall verbunden sein, müssen Kanten zwischen nicht direkt benachbarten Clustern zugelassen werden, was jedoch wieder die Planarität verletzen könnte.

Ein mögliche Lösung ist der **Purged Aggregated Gabriel Graph**:

1. Erstelle Graphen über Knoten mit UDG
2. Wende GG auf diesen UDG an
3. Aggregiere Cluster durch Overlay

An dieser Stelle kann es jedoch noch irreguläre Schnitte geben:

Um diese los zu werden entfernt man im Beispiel oben die Kante AC und fügt eine **implizite kante** BC ein. *Leider wird nirgends erklärt was das soll und warum man das machen kann.*

Localized Multicasting

Multicasting sendet Nachrichten von einem an mehrere Knoten, aber nicht an alle. Man möchte eine Nachricht so weit wie möglich über den selben Pfad schicken und erst “auf den letzten Meter” splitten. **Lokales geographisches**

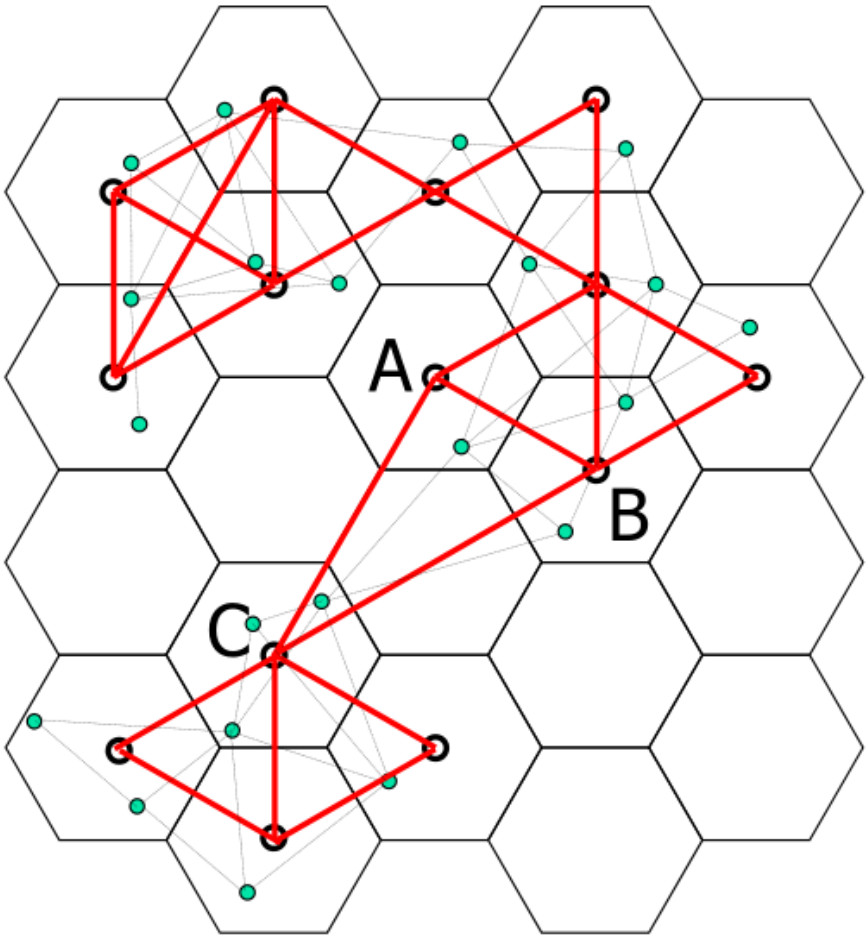

Figure 19: Verbundenheit und Planarität in Overlay-Graphen

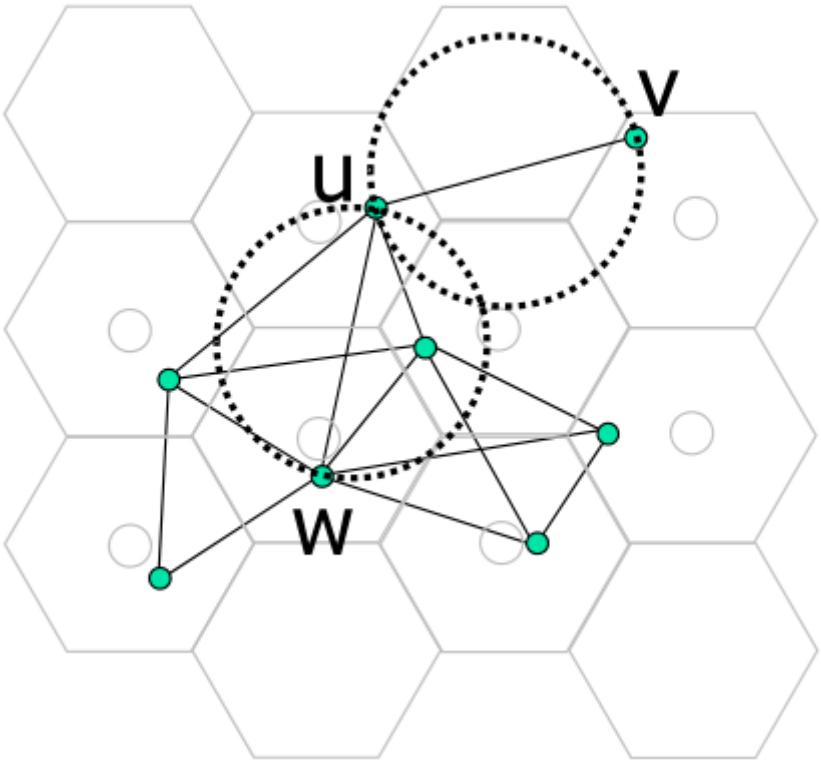

Figure 20: Aggregated Gabriel Graph: Schritt 2

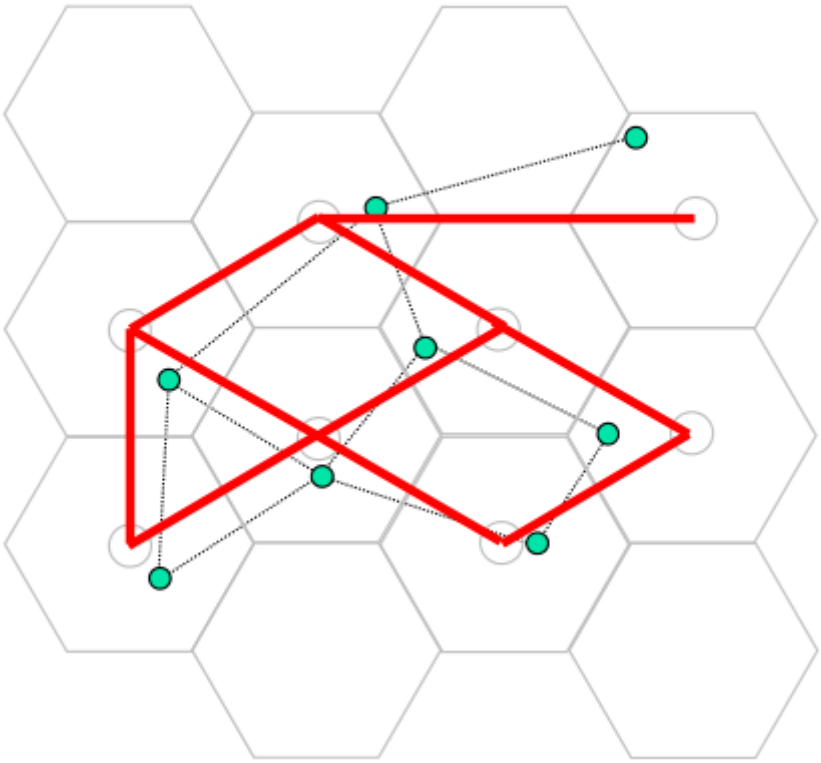

Figure 21: Aggregated Gabriel Graph: Schritt 3

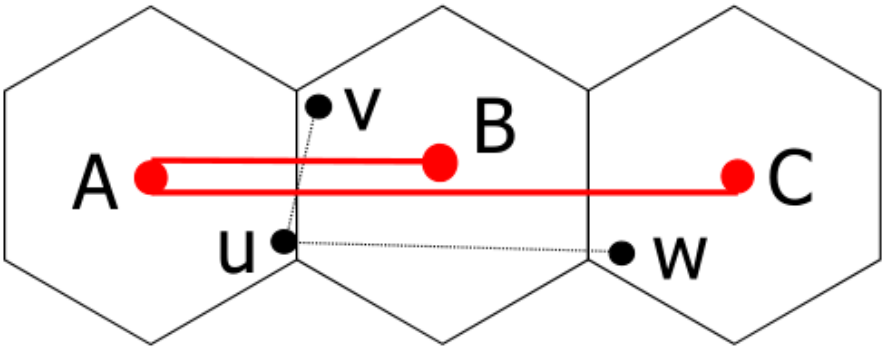

Figure 22: Irregulärer Schnitt in Aggregated Gabriel Graph

Multicasting versucht dieses Problem in einem geographischen Netzwerk mit lokalen Entscheidungen zu lösen.

MSTEAM

MSTEAM steht anscheinend für **EMST Backbone Assisted Localized Routing** *auch wenn das irgendwie keinen Sinn macht.*

Jeder Knoten bildet mit EMST einen virtuellen **Backbone-Graphen** über sich und den Zielknoten. Dann schickt er die Nachricht an die Knoten, die die nächsten Hops im EMST "am besten" (entsprechend einer der üblichen Metriken) approximieren. Der virtuelle Backbone-Graph kann natürlich nur gebildet werden, wenn die geographischen Positionen der Zielknoten bekannt sind.

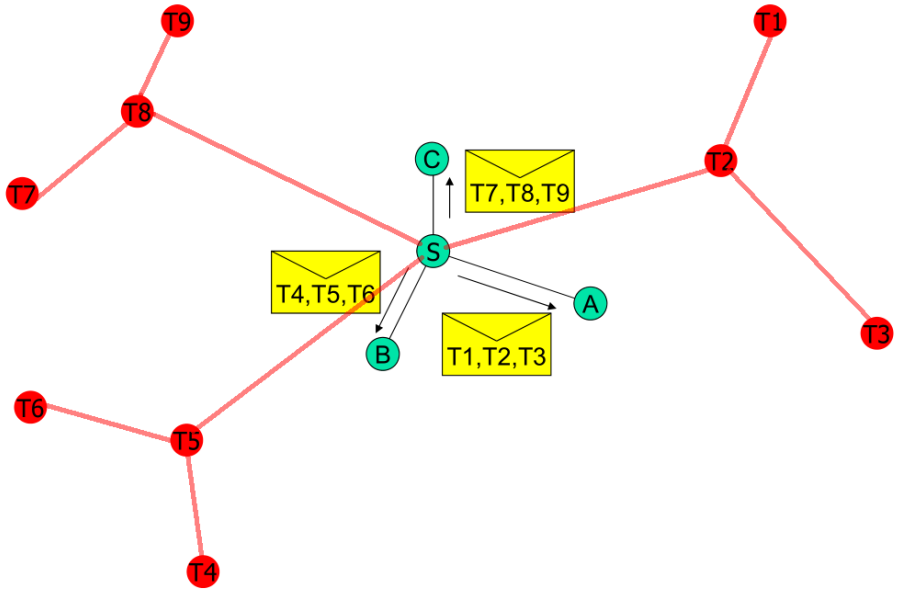

Figure 23: MSTEAM von S nach T1 bis T9, erster Hop. Rot: Virtueller EMST Backbone

MFACE

MFACE wird leider nicht verständlich beschrieben, hier nur eine unsichere Interpretation.

MFACE benutzt die Kanten des virtuellen Backbone-Graphen als die Geraden, die die Faces schneiden, entlang derer dann zu den Zielknoten traverisiert wird.

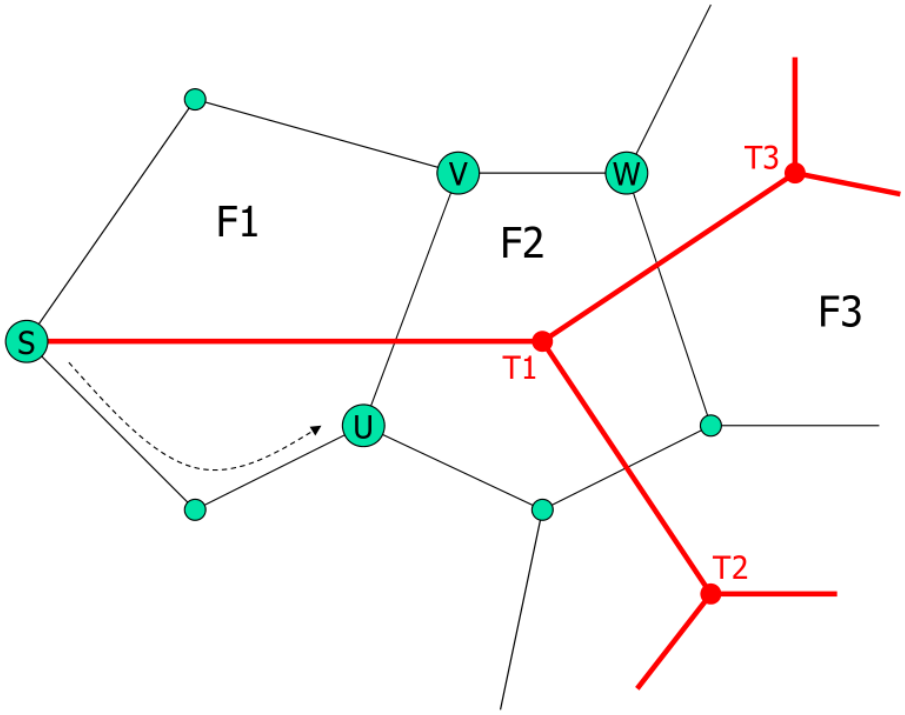

Figure 24: MFACE Ausgangssituation

Reaktive Topologiekontrolle

Verwendet man die PDT um eine Nachricht zu versenden, müssen auf dem Weg der Nachricht durch das Netz viele weitere Nachrichten ausgetauscht werden, um die PDT zu bilden. Besser wäre wenn nur die PDT-Nachbarn antworten (**reagieren**) und damit der PDT-graph direkt konstruiert wäre.

Zunächst eine alternative Definition der PDT:

$$uv \in PDT \Leftrightarrow \forall w \in circle(u,v) : circle(u,v) \cap \overline{H^w(u,v)} = \emptyset \wedge sin(\alpha) \geq \frac{\|uv\|}{r}$$

Mit

- $\overline{H^w(u,v)}$ Halbebene von u und v die nicht w enthält
- α Winkel zwischen u und v über w ($\measuredangle(u,w,v)$)
- r Unit Disk Radius

In Worten: Die Halbebene gegenüber w muss leer sein und der Winkel muss "ziemlich spitz sein".

Reactive PDT

Reactive PDT (RPDT) für Knoten u:

1. Sende Request to Send (RTS) mit Position blind an alle Nachbarn (Broadcast)
2. Setze eigenen Timer auf das Maximum ($t(u) = t_{max}$)
3. Alle anderen Knoten (v) empfangen den Broadcast:
    1. Initialisiere Menge bekannter Knoten ($S(v) = \emptyset$)
    2. Speichere Maximalwinkel ($\alpha_{max}(v) = \pi/2$)
    3. Setze Timer entsprechend des normalisierten Abstands ($t(v) = \frac{\|uv\|}{r} \cdot t_{max}$)
        - Bei Timerablauf: Sende Clear to Send (CTS) mit Position an alle Nachbarn
        - Bei fremden CTS (von z): Füge z zu $S(v)$ hinzu und prüfe ob uv bezüglich z die PDT-Bedingung verletzt.
            - Ja: Lösche Timer (wird kein CTS senden)
            - Nein: Teste ob $\alpha = \measuredangle(u,v,z) > \alpha_{max}$. Wenn ja aktualisiere den Maximalwinkel ($\alpha_{max} = \alpha$) und aktualisiere den Timer ($t(v) = \frac{\|uv\|}{sin(\alpha)} * \frac{1}{r} * t_{max}$)

4. Empfange CTS von Nachbarn bis der Timer abgelaufen ist. Alle Nachbarn die geantwortet haben sind PDT-Nachbarn.

TLDR: Bei CTS-Empfang passen die anderen Knoten ihre Timer so geschickt an, dass nur die PDT-Nachbarn ein CTS senden. Der RTS-Knoten kennt nach Ablauf seines Timers alle PDT-Nachbarn.

BFP-GG

BFP-GG funktioniert ähnlich wie reactive PDT, erzeugt aber einen Gabriel Graphen. Hier kann es allerdings passieren, dass einige nicht-GG Kanten erzeugt werden (*die Details sind kompliziert*) sodass es anschließend eine Protestphase geben muss, in der diese wieder entfernt werden.

O_k-Reaktivität

Nachrichtenkomplexität bezeichnet die Speicher-Komplexität der Nachrichten, also der Anzahl von Bits, die übertragen werden müssen (In O-Notation).

Eine lokale Topologiekontrolle ist O_k**-reaktiv**, wenn zur Konstruktion:

- ... kein Knoten seine Nachbarn kennen muss
- ... ein Knoten zu Beginn einen Broadcast sendet
- ... die Nachrichtenkomplexität auf $O(beteiligteKnoten \cdot log_2(n))$ beschränkt ist

n bezeichnet hier die Anzahl Knoten im Ausgangsgraphen und $beteiligteKnoten$ die Anzahl der Knoten, die bei dem Konstruktionsverfahren der Topologiekontrolle beteiligt sind (*Stark vereinfacht*).

Analog dazu ist eine lokale Topologiekontrolle Ω_k**-reaktiv**, wenn zur Konstruktion die Nachrichtenkomplexität auf $O(beteiligteKnoten \cdot log_2(n))$ *nach unten* beschränkt ist.

Übersicht der Reaktivität reaktiver Verfahren

Algorithmus	Topologie	O_1-Reaktiv	Ω_1-Reaktiv
(GDBF)	GG	Nein	Ja
BFP-GG	GG	Nein	Ja
(BFP-RNG)	RNG	Ja	Nein

Algorithmus	Topologie	O_1-Reaktiv	Ω_1-Reaktiv
(BFP-CNG)	(CNG)	Ja	Nein
RPDT	PDT	Ja	Nein

Eingeklammertes wurde nicht behandelt.

Redundanz und Koexistenz

Redundanz und **Koexistenz** sind Netzwerkeigenschaften, die oft von realen Netzwerken erfüllt werden und auch vom UDG erfüllt werden.

Redundanz

Im Falle eines Schnittes ist mindestens ein Endpunkt mit allen anderen verbunden.

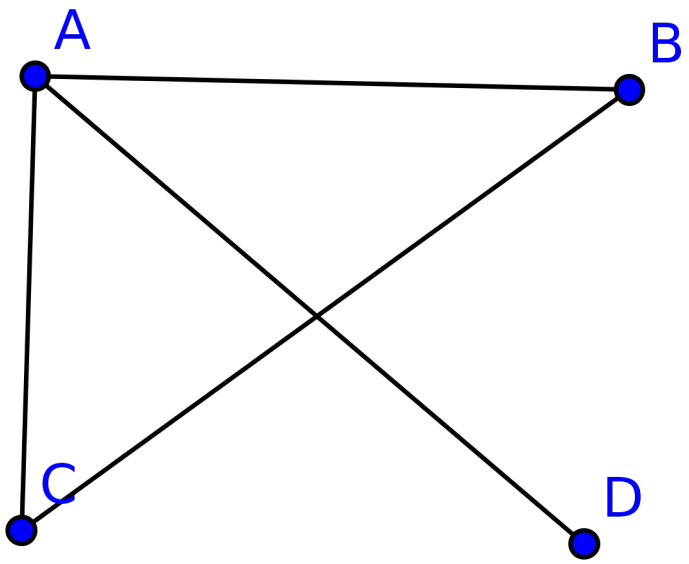

Figure 25: Redundanzeigenschaft

Schwache Redundanz

???

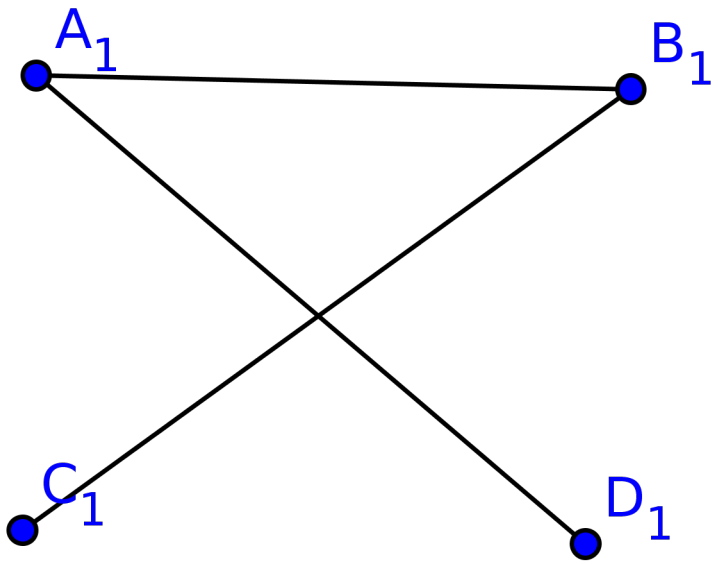

Figure 26: Schwache Redundanzeigenschaft

Koexistenz

Sind 3 Knoten mit einander verbunden ist auch jeder Knoten innerhalb des Dreiecks mit allen Punkten verbunden.

Schwache Koexistenz

Sind 3 Knoten mit einander verbunden ist auch jeder Knoten innerhalb des Dreiecks mit mindestens einem der Punkte verbunden.

Klausurfragen

1. Was versteht man unter einem Ad Hoc Netzwerk? Welche Ausprägungen gibt es? Wie unterscheiden sich diese?

2ez

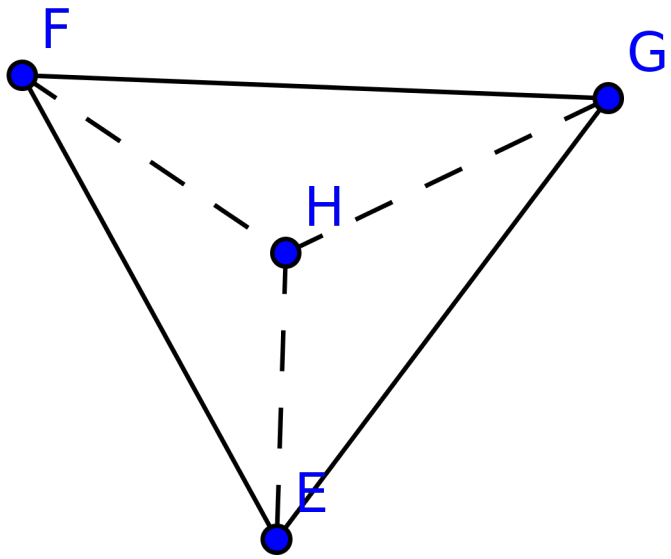

Figure 27: Koexistenzeigenschaft

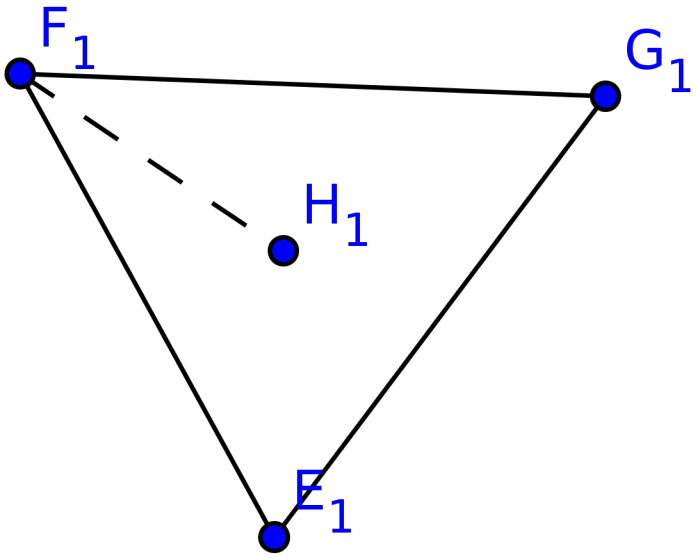

Figure 28: Schwache Koexistenzeigenschaft

2. Was versteht man unter Single-Hop bzw. Multi-Hop Netzen? Welche Vorteile bieten Multi-Hop gegenüber Single-Hop Netzen?

Single-Hop hat direkte Verbindungen, Multi-Hop geht um Ecken. Multi-Hop hat eine potentiell einfachere Graphstruktur, weniger Energieverbrauch (weil Kürzere Strecken), weniger Channels/Kabel.

3. Was versteht man unter dem Begriff Pfadverlust?

Signalverlust in Dezibel wenn es ankommt.

4. Welche vereinfachten Graphmodelle gibt es und wie sind diese definiert? Wieso wird bei QUDG typischerweise angenommen, dass $\frac{r_{max}}{r_{min}} \leq sqrt2$ gilt?

UDG, QUDG? Und keine Ahnung.

5. Was versteht man unter einem lokalen Algorithmus und was ist der Unterschied zu einem globalen Algorithmus? Welchen Preis bezahlt man für die Lokalität?

Lokale Algorithmen beachten nur die (k-)Nachbarn eines Knotens bei der Berechnung, wobei globale Algorithmen den ganzen Graph kennen und damit Entscheidungen treffen. Lokalität ist weniger optimal.

6. Was versteht man unter Topologiekontrolle? Wie ist k-lokale Topologiekontrolle definiert? Wie ist die lokale Sicht eines Knotens auf einen Graphen definiert und was ist damit gemeint?

Graph zu Subgraph. Eine k-lokale Topologiekontrolle betrachtet nur die k-Nachbarn jedes Knotens. Die lokale Sicht eines Knotens sind diese Nachbarn und die Pfade dahin.

7. Was ist ein Euklidischer Spanner bzw. Euklidischer Graph Spanner bzw. Topologischer Graph Spanner? Wie sind diese definiert?

Ein Spanner ist ein Subgraph mit limitierten stretch-Faktoren. Der stretch-Faktor ist das Verhältnis zwischen Pfadlänge im Subgraphen und dem ursprünglichen Graphen. Ein Euklidischer Spanner vergleicht mit dem vollständigen Euklidischen Graphen. Der Euklidische Graph Spanner vergleicht mit einem beliebigen Euklidischen Graphen. Der Topologische Graph Spanner macht das gleiche, aber für topologische Graphen.

8. Ist ein Euklidischer Spanner automatisch auch immer ein Topologischer Spanner? Ist ein Topologischer Spanner automatisch immer ein Euklidischer Spanner? Begründen Sie Ihre Antworten. Welche Eigenschaften muss ein Graph bspw. erfüllen damit jeder Euklidische Spanner auch automatisch ein Topologischer Spanner ist?

Ja. Nein. Weil die Graphen selbst sich auch so verhalten.

9. Was ist ein Euklidischer Minimaler Spannbaum (EMST)? Ist EMST eine k-lokale Graphstruktur? Wieso nicht? Welche vergleichbare Graphstruktur kann lokal konstruiert werden? Wie kann LMST lokal konstruiert werden? Welche Subgraphrelation erfüllen EMST und LMST? Ist LMST Knotengradbeschränkt?

Es ist der optimalste Spanner in einem euklidischen Graphen. Nein. Weil es in der VL so steht. LMST. Mit Kruskarl. $EMST \subseteq LMST$ falls UDG verbunden. Ja, auf 6.

10. Wie sind RNG, URNG, GG und UGG definiert? Welche Eigenschaften haben diese Graphen bzgl. Gradbeschränkung, Zusammenhang, Planarität, k-lokale Konstruierbarkeit und welche Subgraphrelation erfüllen sie? Wie beweist man das GG planar ist?

Schau in die Zusammenfassung und lern es auswendig.

11. Wie ist die Hop- bzw- Euklidische Spanning Ratio von GG bzw. RNG? Nennen Sie die Obere- und Untere-Schranke und die entsprechenden Beweisideen.

Das hat der Frey gesagt es kommt nicht dran.

12. Wie ist die Delaunay Triangulierung (DT) definiert und welcher Zusammenhang besteht zu Voronoi Diagrammen? Unter welchen Annahmen ist die DT wohldefiniert? Was ist eine konvexe Menge und was die konvexe Hülle einer Punktmenge?

Kante existiert wenn sich Voronoi-Regionen eine Voronoi-Kante teilen. Nicht-Cozirkularität und Nicht-Colinearität. Unsicher.

13. Wie kann man beweisen, dass DT ein Euklidischer Spanner für den vollständigen Euklidischen Graphen ist? Was ist der DT-Pfad und welche Rolle spielt er im Spanner Beweis von Dobkin?

Mit dem Dobkin-Beweis. Direkter pfad durch Voronoi-Regionen und keine Ahnung.

14. Was ist die Unit Delaunay Triangulierung (UDT) und ist diese lokal konstruierbar? Wieso?

Nope, weil Del ist auch nicht lokal.

15. Welche lokalen Konstruktionen für planare Spanner gibt es? Wie ist $LDel^{(k)}$ definiert? Ist LDel^{(1)} planar? Wie kann man $LDel^{(1)}$Ä planarisieren?

GG, RNG, LDel usw.. Del auf k-hop-Nachbarschaft. Mit PlDel.

16. Erläutern Sie einen 2-lokalen Algorithmus zur Konstruktion von LDel (2).

Alle 2-lokalen Dreiecke aus S finden, deren Kreis drumrum keine weiteren Punkte enthält. Das Ergebnis davon mit UGG ∪nifizieren.

17. Kann man planare Spanner auch 1-lokal konstruieren? Nennen Sie zwei 1-lokale Spanner Konstruktionen. Wie sind diese definiert?

Ja. UGG und URNG. UDG+GG und UDG+RNG.

18. Wie ist der Yao-Graph definiert? Welche Subgraphrelation erfüllt er? Ist der Yao-Graph planar und wieso? Was ist eine Cone-Based Topology? Wie kann lokal eine Symmetrische Subtopologie konstruiert werden?

Kuchen. kA. Nein weil Vorlesung. Empfangsreichweite so einstellen, dass in jedem Cone ein Knoten ist. Exkursion.

19. Nennen Sie Beispiele für Topologiekontrollmechanismen die in gerichteten Netzgraphen unzusammenhängende Topologien konstruieren. Wie ist DRNG definiert?

DRNG und DLSS.

20. Erläutern Sie das Prinzip des Chu-Liu/Edmonds Algorithmus zur Konstruktion von Directive Local Spanning Subgraphs (DLSS). Welche Subgraphrelation erfüllen DRNG und DLSS?

Sammelt Kanten mit geringen Gewichten und ersetzt Zyklen durch virtuelle Knoten. DLSS ist Subgraph von DRNG.

21. Was versteht man unter einem k-vertex connected network? Wie kann man mit lokalen Regeln ein k-vertex connected network konstruieren?

Jeder Knoten hat mindestens k Nachbarn. Yao Graph mit 6 cones.

22. Was versteht man unter einem Backbone? Welchen Zweck erfüllen diese? Wie ist ein Dominating Set (DS) bzw. ein Connected Dominating Set (CDS) definiert?

Jeder Knoten ist entweder ein Backbone oder Nachbar eines Backbones. Damit nur Backbones schwere Arbeit machen müssen. Menge von Backbones bzw. verbundene Menge von Backbones.

23. Was versteht man unter dem Begriff Clustering? Welche Rollen nehmen Knoten beim Clustering ein? Erläutern Sie das Funktionsprinzip des Verteilten Clustering Algorithmus aus der VL. Erläutern Sie das lokale Verfahren zur Konstruktion von CDS mittels 2D-Grid Clustering. Erläutern Sie die Funktionsweise der Lightweight CDS Construction nach Wu und Li.

Fasst Knoten logisch zusammen. Cluster-Head, Cluster-Member oder Gluster-Gateway. Cluster entscheiden der Rehe nach ob sie CH oder CM sind und broadcasten ihre Entscheidung. 2D-Grid, Knoten mit höchster ID in Kästchen wird dominating. Wi Tu Lo.

24. Was versteht man unter guaranteed delivery unicast, multicast, ..? Welche Annahmen müssen hierzu getroffen werden und warum?

Garantierte Zustellung. Casts gehen an verschiedene Mengen von Knoten. Graph connected und sinnvolles Routing-Verfahren.

25. Erläutern Sie das Prinzip von Greedy Packet Forwarding. Garantiert dieses Verfahren garantierte Nachrichtenauslieferung? Erläutern Sie anhand eines Beispiels.

Nachricht wird an nächstbesten Knoten weitergeschickt. Nein, weil Sackgassen.

26. Welche Routing Metriken zur Anwendung beim Greedy Routing kennen Sie? Erläutern Sie in Kürze das Cost Over Progress Framework.

Distanz, Verbleibende Batterie, Hops, Zufall, Cost-over-Progress Framework. Das setzt Kosten für nächsten Hop und geschätzte weitere Kosten in Vergleich zu geschätzten aktuellen verbleibenden Kosten.

27. Erläutern Sie das Prinzip von FACE-Routing sowie von Greedy-Face-Greedy (GFG) Routing.

Die Nachricht wandert entlang von Faces zum Ziel. GFG zieht Linie von Start zu Ziel und wechselt Face wenn nächste Kante die Gerade schneidet.

28. Erläutern Sie das Prinzip des Algorithmus Beaconless Forwarder Planarization (BFP). Ist dieser Algorithmus O-reaktiv? Benennen Sie ein Gegenbeispiel.

Knoten antworten mit geschicktem Timing auf RTS mit CTS. Kommt drauf an, BFP-GG ist O-Reaktiv, BFP-RNG nicht.

29. Erläutern Sie das Prinzip des Algorithmus reactive-PDT. Ist dieser Algorithmus O-reaktiv? Wieso?

S.o.. Ja, weil die Nachrichtenkomplexität ist beschränkt auf $O(beteiligteKnoten \cdot log_2(n))$.

30. Wann ist ein lokaler Algorithmus zur Topologiekontrolle Ω-reaktiv?

Wenn die Nachrichtenkomplexität nach unten beschränkt ist auf s.o..

Marco

- Was sind die einfachsten Graphmodelle? Was sind UDG und QUDG?
- Was ist eine lokale Topologiekontrolle? Wie ist sie definiert?
- Was sind Spanner? Wie sind sie definiert?
- „Was gibt es so für Topologiekontrollen?" Wie sind sie definiert? Wie stehen sie in Relation mit einander?
  - Ich hab RNG, GG, URNG und UGG genannt und musste sie dann beschreiben
- Wie kann man im Allgemeinen beweisen, dass ein Graph Subgraph eines anderen ist
  - Ich hab im Endeffekt nur graphisch gezeigt wie der GG im RNG enthalten ist
- Was ist UDel? Warum ist es nicht lokal? Zeige ein Beispiel an dem man sieht, dass UDel nicht lokal ist. Was ist Delauney-Triangulierung? Wie sind die 3 Definitionen? Sind sie gleich? Was sind Cozirkularität und Kolinearität? Warum ist UDel trotzdem nützlich? Warum kann man Spanning-Aussagen über einen Graphen auf dessen Supergraphen übertragen?

Dausi

- UDG, QUDG
- PDT
- Beweisideen zu PDT und UDel
- Topologiekontrolle
- Routing

Carsten

- UDG, QUDG
- Erkläre lokale Topologiekontrolle
  - Schreibe diese Formel da hin: $\tau : \mathcal{C} \to \mathcal{D}, \forall G \in \mathcal{C} \forall v \in \tau(G) \exists u \in G : \tau(G)[v] = tau(G[u,k])[v]$
  - Warum ist das so umständlich? $\to$ Weil Geoclustering, dieses Raster über die Knotenmenge, imaginäre Kanten und so. Keine Ahnung, aber die Begründung hat ihm gereicht.
- Schreib mal ein paar lokale Graphen hin
  - Antwort mit Graphen die man gut kann, bei mir $URNG, UGG, PDT, LDel^{(k)}$
- Zeige, das $RNG \subseteq GG$
  - Man malt die RNG-Kreise, sagt die Schnittmenge ist Lune und malt da den GG-Kreis rein $q.e.d.$
- Erkläre Spanner