Algorithmen für Ad-hoc- und Sensornetze

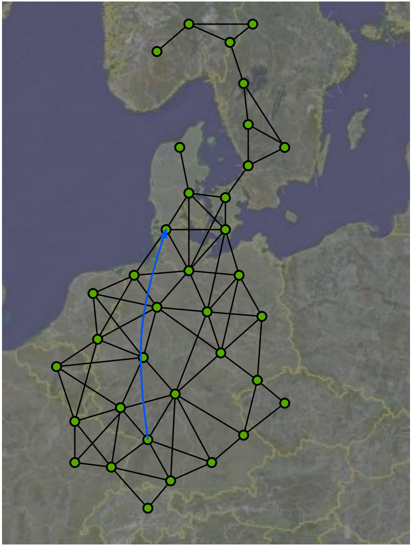
VL 02 – Georouting

Dr. rer. nat. Bastian Katz

Lehrstuhl für Algorithmik I
Institut für theoretische Informatik
Universität Karlsruhe (TH)
Karlsruher Institut für Technologie

29. April 2009

(Version 3 vom 30. April 2009)



Geographisches Routing

Geographisches Routing

- Knoten kennen ihre Geokoordinaten (und die ihrer Nachbarn)
- Pakete enthalten Start- und Zielkoordinaten
- Vollkommen reaktiv: keine Routingtabellen

- Geographisches Routing ist realistisch
  - Knotenpositionen bekannt durch GPS/Galileo oder Lokalisierungsverfahren
  - Zielpositionen bekannt
    - wenn statt Labels Positionen verwendet werden (statisch)
    - wenn Informationen an Zielkoordinaten geroutet werden
    - Beispiel: Location Services (VL03)

Positionsbewusstsein

Definition

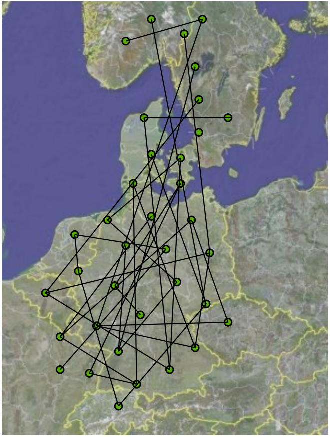
Eine Einbettung eines Graphen $G = (V, E)$ ist eine Abbildung $\mathbf{p} : V \to \mathbb{R}^d$.

- Sensornetze bringen ihre Einbettung gleich mit!
- zunächst nur in der Ebene: $d = 2$

Definition

Ein verteilter Algorithmus auf einem eingebetteten Graphen heißt *positionsbewusst*, wenn jeder Knoten v seine Position $\mathbf{p}(v)$ und die seiner Nachbarn kennt.

- zunächst nur statisch: $\mathbf{p}$ ändert sich nicht



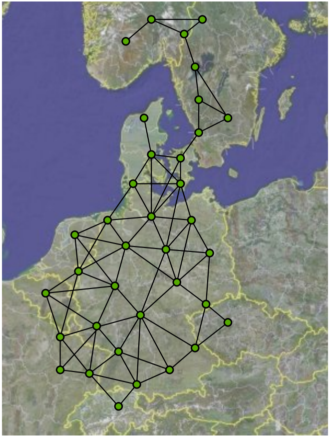
Was sind Koordinaten wert, wenn die Vernetzung nichts mit ihnen zu tun hat?

Erinnerung

- Signal fällt mit Entfernung ab
- Empfänger braucht gewisse Signalstärke

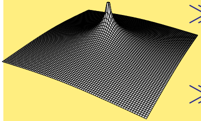
Verbindungen

beliebige Graphen ??? ???



Was sind Koordinaten wert, wenn die Vernetzung nichts mit ihnen zu tun hat?

Erinnerung



- Signal fällt mit Entfernung ab
- Empfänger braucht gewisse Signalstärke

Verbindungen

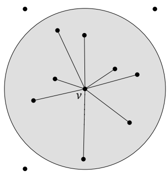
beliebige Graphen ??? UDG

Unit-Disk-Graph

Definition *Unit-Disk-Graph*

Ein Graph heißt *Unit-Disk-Graph (UDG)*, wenn es eine Einbettung **p** in die Ebene gibt, so dass jeder Knoten genau mit allen Knoten in Abstand ≤ 1 verbunden ist, d. h.

$$\{u, v\} \in E \Leftrightarrow |\mathbf{p}(u) - \mathbf{p}(v)| \leq 1$$



- Achtung:
  - UDG zu sein ist eine kombinatorische Eigenschaft
  - Ein Unit-Disk-Graph bleibt ein Unit-Disk-Graph, auch wenn man ihn anders einbettet!
  - Zu wissen, dass ein Graph ein UDG ist, heißt nicht, eine entsprechende Einbettung zu kennen
- Wir gehen von entsprechender Einbettung aus!

Unit-Disk-Graph-Modell

Definition *Unit-Disk-Graph-Modell*

Im *Unit-Disk-Graph-Modell* belegen die Knotenpositionen $\mathbf{p}$ die Eigenschaft des Kommunikationsgraphen, ein Unit-Disk-Graph zu sein, d. h. es gibt einen Kommunikationsradius R, so dass

$$\{u, v\} \in E \Leftrightarrow |\mathbf{p}(u) - \mathbf{p}(v)| \leq R$$

- ≫ beschreibt zu gegebenen Positionen genau, wie Kommunikationsgraph aussieht
- ≫ unrealistisch, aber gut zu analysieren, fangen wir damit an!

Georouting

Gegeben Eingebetteter Unit-Disk-Graph $G = (V, E)$, zwei Knoten $s, t \in V$

Gesucht Positionsbewusste Routingstrategie, um ein Paket, das die Position $\mathbf{p}(t)$ enthält, von s nach t zu bringen.

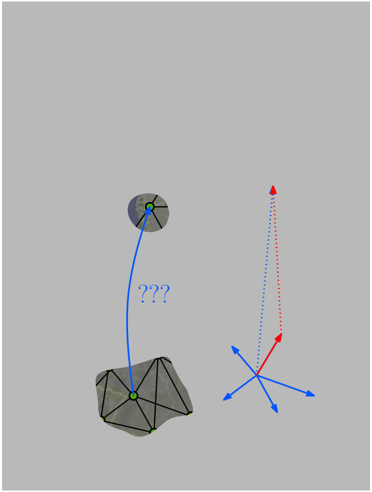
≫ Es ist erlaubt, „etwas“ Routinginformationen im Paket zu speichern (neben der Zielposition)

Greedy Routing

Greedy Routing

Leite Paket an den Nachbarn, der dem Ziel am nächsten ist.

Was, wenn es nicht weitergeht?

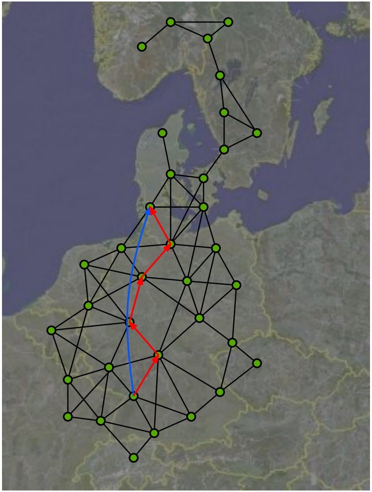


Greedy Routing

Greedy Routing

Leite Paket an den Nachbarn, der dem Ziel am nächsten ist.

Was, wenn es nicht weitergeht?

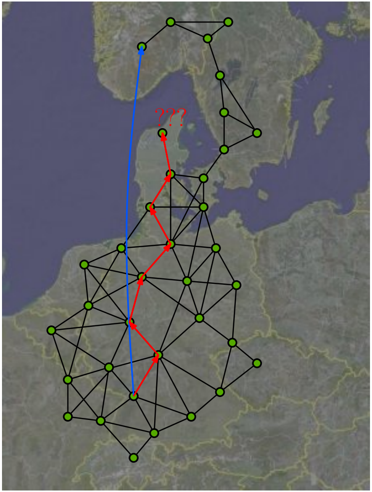


Greedy Routing

Greedy Routing

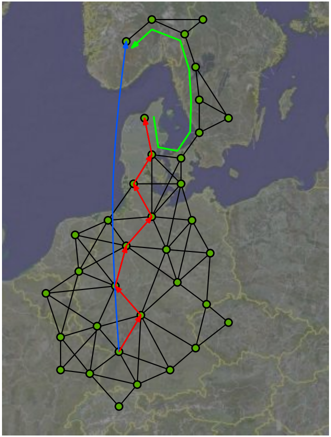
Leite Paket an den Nachbarn, der dem Ziel am nächsten ist.

Was, wenn es nicht weitergeht?



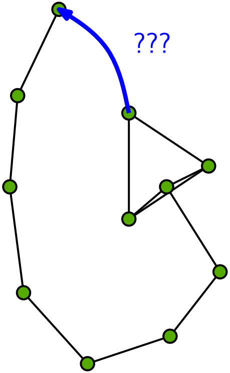
Alte Bekannte

Idee: Irgendwas mit Rechte-Hand-Regel?



Alte Bekannte

Idee: Irgendwas mit
Rechte-Hand-Regel?



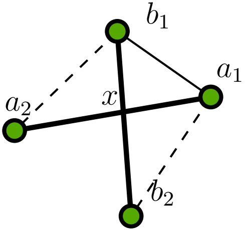
Was haben Irrgärten, was
UDG nicht haben?

Planare Subgraphen

Lemma

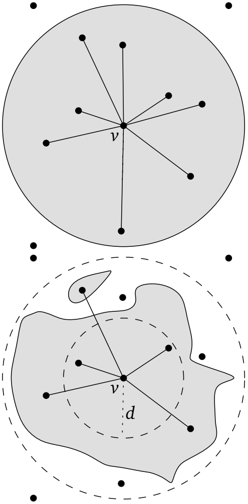
Jeder eingebettete zusammenhängende Unit-Disk-Graph hat einen zusammenhängenden kreuzungsfreien Teilgraphen[a].

[a] hier immer: geradlinig gezeichnet



- Betrachte Kreuzung x von $\{a_1, a_2\}$ und $\{b_1, b_2\}$
- ObdA $|a_1 - x| \leq 0.5$ und $|b_1 - x| \leq 0.5$
  $\Rightarrow$ $|b_1 - a_1| \leq 1$
- $|a_1 - b_2| + |a_2 - b_1| \leq 2$
  - $|a_1 - b_2| \leq 1$ oder $|a_2 - b_1| \leq 1$
- eine der kreuzenden Kanten kann entfernt werden

Einschub: d-QUDG



Definition *d-Quasi-Unit-Disk-Graph*

Ein Graph heißt *d-Unit-Disk-Graph (d-QUDG)* für $d \leq 1$, wenn es eine Einbettung $\mathbf{p}$ in die Ebene gibt, so dass jeder Knoten nur mit Knoten in Abstand ≤ 1 verbunden ist, darunter mit allen in Abstand $\leq d$, d. h.

$$\{u, v\} \in E \Rightarrow |\mathbf{p}(u) - \mathbf{p}(v)| \leq 1$$

und

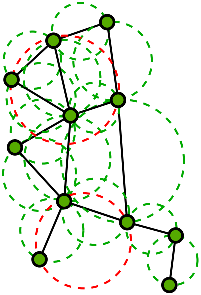
$$\{u, v\} \notin E \Rightarrow |\mathbf{p}(u) - \mathbf{p}(v)| > d$$

Das Lemma gilt für alle $d > 1/\sqrt{2}$ (o. Bew.)

Gabriel Graphs (weil's so schön ist)

Definition

Sei P eine Menge von Punkten in der Ebene. Der *Gabriel Graph* enthält ein Punktepaar genau dann als Kante, wenn der kleinste Kreis, der beide Punkte enthält, keine weiteren Punkte enthält.

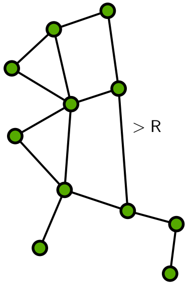


- Gabriel Graph ist planar, aber kann „lange" Kanten enthalten
  - schränke UDG auf GG-Kanten ein!
  - $UDG \cap GG$ ist lokal entscheidbar
- $UDG \cap GG$ ist planar
- $UDG \cap GG$ ist zshg. $\Leftrightarrow$ UDG ist zshg.
- Beweise übernächste Woche!
- QUDG: Cross-Link-Detection-Protokolle!

Gabriel Graphs (weil's so schön ist)

Definition

Sei P eine Menge von Punkten in der Ebene. Der *Gabriel Graph* enthält ein Punktepaar genau dann als Kante, wenn der kleinste Kreis, der beide Punkte enthält, keine weiteren Punkte enthält.

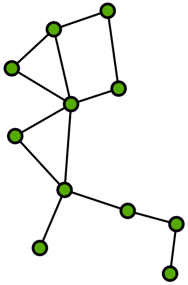


- Gabriel Graph ist planar, aber kann „lange" Kanten enthalten
  - schränke UDG auf *GG*-Kanten ein!
  - *UDG* ∩ *GG* ist lokal entscheidbar
    - beide Enden "sehen" störende Knoten
- *UDG* ∩ *GG* ist planar
- *UDG* ∩ *GG* ist zshg. ⇔ UDG ist zshg.
- Beweise übernächste Woche!
- QUDG: Cross-Link-Detection-Protokolle!

Gabriel Graphs (weil's so schön ist)

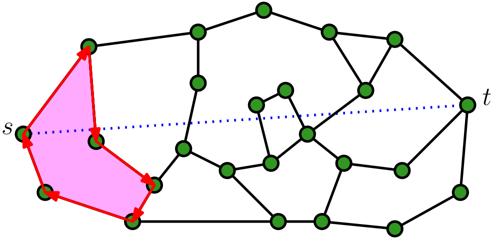
Definition

Sei P eine Menge von Punkten in der Ebene. Der *Gabriel Graph* enthält ein Punktepaar genau dann als Kante, wenn der kleinste Kreis, der beide Punkte enthält, keine weiteren Punkte enthält.



- Gabriel Graph ist planar, aber kann „lange“ Kanten enthalten
  - schränke UDG auf GG-Kanten ein!
  - $UDG \cap GG$ ist lokal entscheidbar
    - beide Enden “sehen“ störende Knoten
- $UDG \cap GG$ ist planar
- $UDG \cap GG$ ist zshg. $\Leftrightarrow$ UDG ist zshg.
- Beweise übernächste Woche!
- QUDG: Cross-Link-Detection-Protokolle!

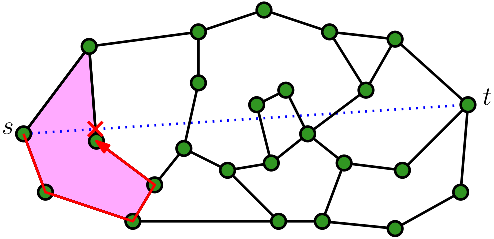
Facettenrouting (FR)



Facettenrouting [Kranakis et. al. '99]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

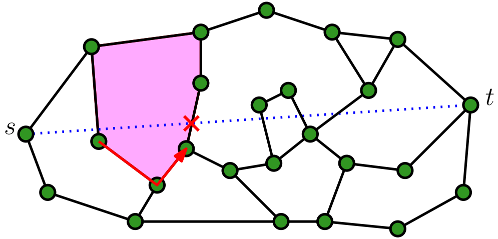
Facettenrouting (FR)



Facettenrouting [Kranakis et. al. '99]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

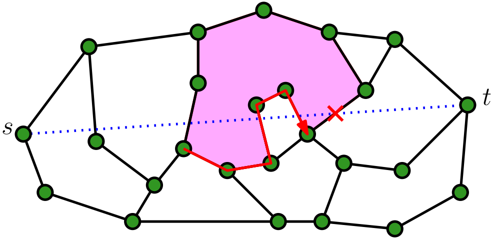
Facettenrouting (FR)



Facettenrouting [Kranakis et. al. '99]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

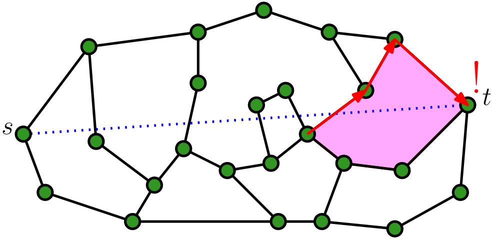
Facettenrouting (FR)



Facettenrouting [Kranakis et. al. '99]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

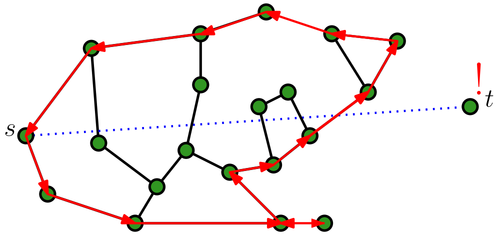
Facettenrouting (FR)



Facettenrouting [Kranakis et. al. '99]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

Facettenrouting (FR)

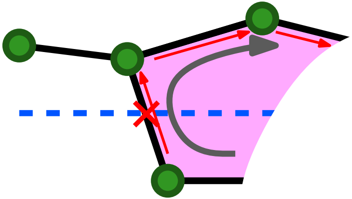


Fehlerbehandlung

- Was, wenn kein besserer Schnitt gefunden wird?
- Schicke Paket mit Fehlermeldung an *s* zurück
- einfach wieder per Facettenrouting

Regeln des Facettenroutings

- Routinginformation komplett in Nachricht enthalten
  - Start- und Zielkoordinaten
  - bester Übergangsknoten zur nächsten Facette
- Vollständig lokal
  - Positionen der Nachbarknoten reichen aus
  - Facettenerkundung nur implizit durch Linke-Hand-Regel



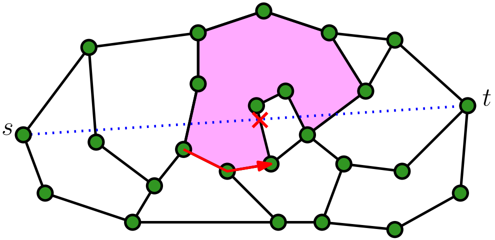
Korrektheitsbeweis und Laufzeit

Satz

Facettenrouting findet in $O(n)$ Schritten zum Ziel oder erkennt fehlenden Zusammenhang.

- Beweis:
  - Jede Facette wird maximal einmal behandelt
    - Facetten ohne Schnitt mit $\overrightarrow{st}$ fallen weg
    - sortiere andere Facetten nach letztem Schnitt mit $\overrightarrow{st}$
    - Facetten werden nur aufsteigend besucht!
  - jede Kante wird maximal viermal genutzt
    - zweimal in jeder anliegenden Facette (Erkundung, Rückkehr zum besten Schnitt)
  - Euler: Planare Graphen haben maximal 3n-6 Kanten
  - Fehlermeldung kann Aufwand verdoppeln

Vereinfachung Facettenrouting

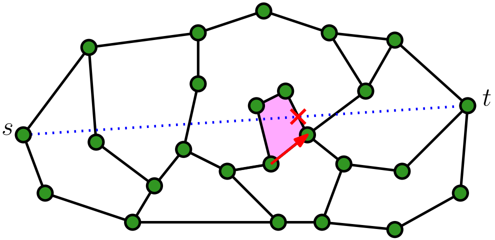


Vereinfachtes Facettenrouting

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Sobald besserer Schnitt mit $\overrightarrow{st}$ gefunden ist
  - Wechsle zur angrenzenden Facette
- beende, wenn Facette keinen besseren Schnitt enthält

- Korrekt, aber asymptotisch nicht besser! (ohne Beweis)

Vereinfachung Facettenrouting



Vereinfachtes Facettenrouting

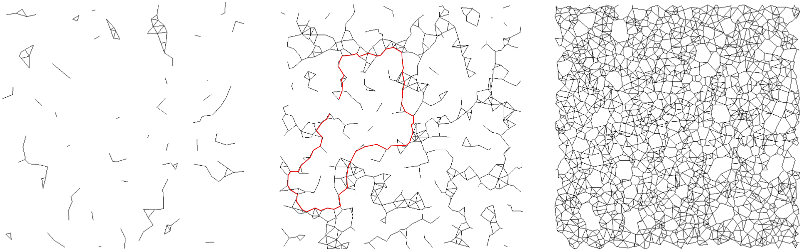
- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Sobald besserer Schnitt mit $\overrightarrow{st}$ gefunden ist
  - Wechsle zur angrenzenden Facette
- beende, wenn Facette keinen besseren Schnitt enthält

- Korrekt, aber asymptotisch nicht besser! (ohne Beweis)

Bessere Analyse

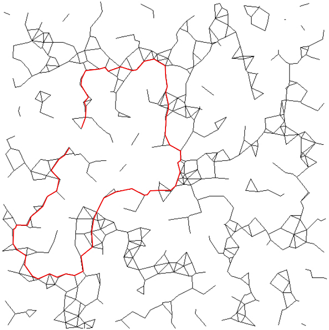
Facettenrouting findet in $O(n)$ Schritten zum Ziel oder erkennt fehlenden Zusammenhang. Was, wenn Start und Ziel dicht beieinander liegen?

- Nicht immer gibt es einen kurzen Weg!



kein Zusammenhang	interessant	greedy ausreichend

Bessere Analyse



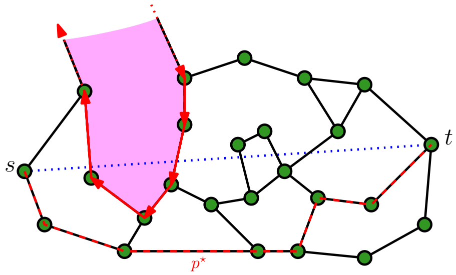
- Kosten des kürzesten Weges $c(p^\star)$
- Euklidischer Abstand $|\mathbf{p}(s) - \mathbf{p}(t)|$
- $c(p^\star) \gg |\mathbf{p}(s) - \mathbf{p}(t)|$

Kann man wenigsten so routen, dass man nicht beliebig schlechter als der kürzeste Weg ist?

Bessere Analyse? Unmöglich!

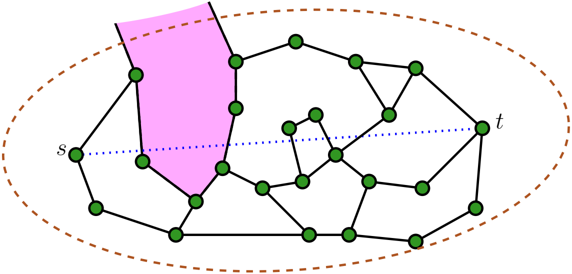
Problem

Facettenrouting lässt keine Abschätzung der Weglänge in Abhängigkeit vom kürzesten Weg zu!



Gibt's denn ein besseren Routingalgorithmus?

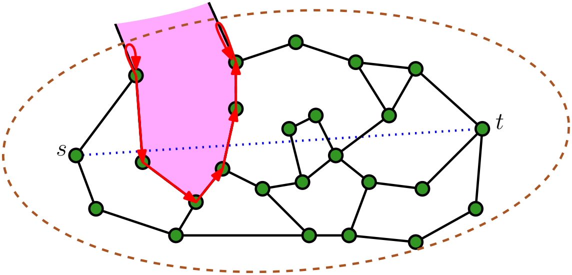
Beschränktes Facettenrouting (BFR)



Lemma

Ein Weg p zwischen s und t mit Länge liegt vollständig innerhalb der Ellipse mit Foci s, t und Hauptachse $c(p)$. Diese Ellipse ist Menge aller Punkte x mit $|\mathbf{p}(x) - \mathbf{p}(s)| + |\mathbf{p}(x) - \mathbf{p}(t)| \leq c(p)$.

Beschränktes Facettenrouting (BFR)



Beschränktes Facettenrouting (BFR)

Ist die Länge $c(p^\star)$ des kürzesten Weges bekannt, beschränke Facettenrouting durch Ellipse mit Foci s und t und Länge der Hauptachse $c(p^\star)$.

The secret ingredient: $\Omega(1)$-Modell

Definition: $\Omega(1)$-Modell

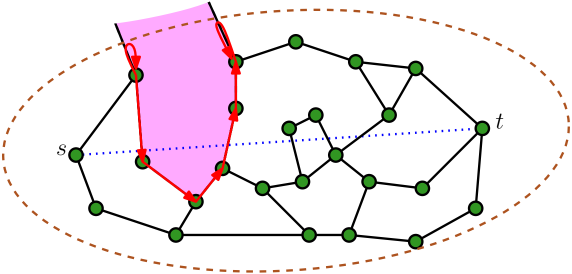
Im $\Omega(1)$-Modell ist der minimale Abstand zwischen zwei Knoten durch eine Konstante d_0 nach unten beschränkt.

Lemma

Im $\Omega(1)$-Modell besucht das beschränkte Facettenrouting nur $O(c(p^\star)^2)$ Knoten.

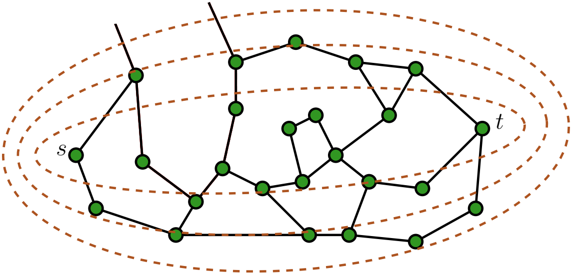
- Beweis:
  - Ellipse ist vollständig in Kreis mit Durchmesser $c(p^\star)$ um Mitte von s und t enthalten.
  - In Kreis mit Durchmesser d passen nur $O((d/d_0)^2)$ Punkte mit paarweisem Abstand d_0
  - d_0 konstant $\Rightarrow O(c(p^\star)^2)$ Knoten im Kreis!

Beschränktes Facettenrouting (BFR)



Woher sollte man wissen, wie lang der kürzeste Weg zwischen Start- und Zielknoten ist? Was macht man, wenn man es nicht weiß?

Adaptives Facettenrouting (AFR)



Adaptives Facettenrouting (AFR)

Ist die Weglänge $c(p^{\star})$ zum Ziel nicht bekannt, führe BFR mit $c_0 = 2 \cdot |st|$ durch. Schlägt das Routing fehl, starte BFR mit $c_1 = 2 \cdot c_0,\ c_2 = 2 \cdot c_1 \ \ldots (c_i = 2^i \cdot c_0)$

Adaptives Facettenrouting – Analyse

Satz

Im $\Omega(1)$-Modell erreicht das Adaptive Facettenrouting das Ziel mit Kosten in $O\left(c^2(p^\star)\right)$, wenn die optimale Route Kosten $c(p^\star)$ hat.

- Beweis
  - spätestens in Runde k erfolgreich mit $c_{k-1} \leq c(p^\star) \leq c_k$
  - $\mathrm{cost}_{\mathrm{AFR}} = \sum_{i=0}^{k} c_i^2 = \sum_{i=0}^{k} (2^i c_0)^2 = \sum_{i=0}^{k} 4^i c_0^2$
  - $= \frac{4^{k+1}-1}{3} c_0^2 < \frac{16}{3}(2^{k-1} c_0)^2 = \frac{16}{3} c_{k-1}^2 < \frac{16}{3} c^2(p^\star)$

Adaptives Facettenrouting – Analyse

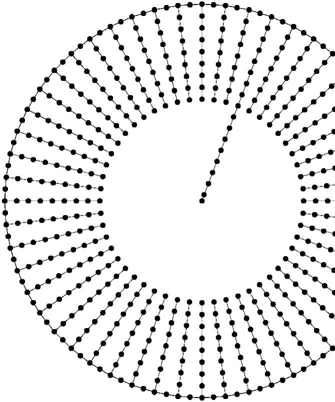
Bemerkung

Fehlender Zusammenhang wird in $O(n' \log n')$ Schritten erkannt, wenn n' die Anzahl der Knoten in der Zhgs.-Komponente von s ist.

- Entferntester Knoten ist maximal n' Funkradien von s entfernt
- Nach $O(\log n')$ Ellipsenverdoppelungen alle Knoten enthalten
  - Erkennung: Paket „stößt“ nicht mehr gegen die Ellipse
- $\Rightarrow O(\log n')$ BFR-Phasen mit je maximal $O(n')$ Schritten

$O(c(p^\star)^2)$ ist worst-case-optimal

- c Pfade vom Rand Richtung Mitte
  - jeder hat Länge $\Theta(c)$
- Ziel: Knoten in der Mitte
- Start: Beliebiger Knoten auf dem Ring
- Ohne Routinginformationen könnte jeder Pfad zur Mitte führen
- Verfahren muss $\Omega(c)$ Pfade testen.
- Bester Weg hat Länge $O(c)$
- Kosten: $\Omega(c^2)$ statt $O(c)$



Satz

AFR ist (asymptotisch) worst-case-optimal.

Schön, aber etwas langsam

FR ⚭ **Ellipt. Suchraum**

Schnell, aber sehr neugierig

BFR ⚭ **Adaptiver Suchraum**

Praktischer Jüngling

Greedy

AFR

Theoretische Schönheit

Greedy AFR (GAFR)

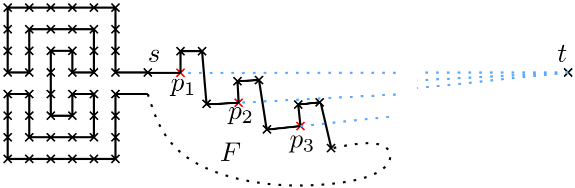
- Eigentlich ist Facettenrouting immer nur „Plan B“
  - Greedy Routing schneller, weniger fehleranfällig
  - Greedy Routing kann alle Verbindungen nutzen
- Verbinde Greedy Routing und Facettenrouting
  1. Route greedy, bis Paket an Knoten p steckenbleibt
  2. Starte adaptives Facettenrouting für Start p, Ziel t
  3. Stoppe, sobald *eine Facette* passiert ist (
  4. Gehe zurück zu 1.

Satz

Greedy AFR ist nicht asymptotisch worst-case-optimal.

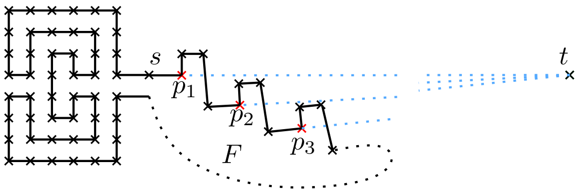
Suboptimalität Greedy+AFR

- Erinnerung:
  - AFR zerfällt in BFR-Schritte verschiedener Ellipsen.
  - In jedem BFR-Durchgang wird jede Facette nur einmal besucht
  - Aufwand linear in der Anzahl der Knoten (der Fläche d. Ellipse)
- Greedy-Schritte zerstören diese Eigenschaft!
- Beweisskizze (betrachte „passende“ Ellipse):
  - Instanz mit Kosten $\in \Theta(c_k^3)$:



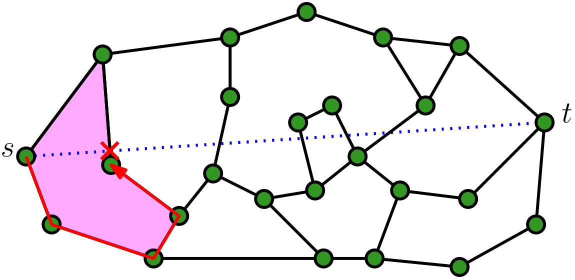
  - Face Routing startet an p_1, p_2, p_3 ($\Theta(c_k)$ mal)
  - Facette F enthält $\Theta(c_k^2)$ Knoten.

Zurück zum Facettenrouting



- Was läuft falsch?
  - Selbst „bester“ Schnittpunkt von $\overrightarrow{p_1 t}$ ($\overrightarrow{p_i t}$) mit Facettenrand kein guter Startpunkt für Greedy-Routing.
  - Greedy-Routing trifft immer wieder auf dieselbe Facette.
  - Facettenrouting erkundet immer wieder dieselbe Facette.
- Facettenrouting lässt sich so anpassen, dass das nicht passiert.

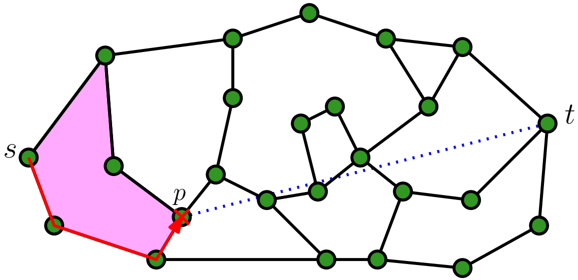
Optimiertes Facettenrouting (OFR)



Herkömmliches Facettenrouting

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zu Schnitt mit $\overrightarrow{st}$ zurück, der t am nächsten ist
- Wechsle zur angrenzenden Facette

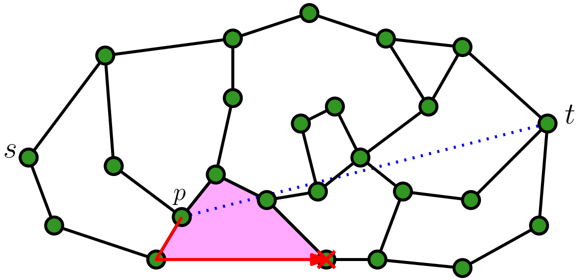
Optimiertes Facettenrouting (OFR)



Optimiertes Facettenrouting [Wattenhofer et al. '03]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zum Punkt zurück, der t am nächsten ist siehe Update
- Wechsle zur angrenzenden Facette

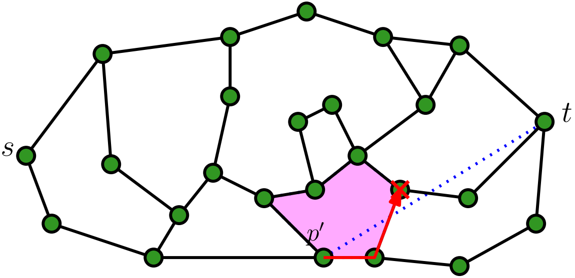
Optimiertes Facettenrouting (OFR)



Optimiertes Facettenrouting [Wattenhofer et al. '03]

- ≫ Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- ≫ Kehre zum Punkt zurück, der t am nächsten ist siehe Update
- ≫ Wechsle zur angrenzenden Facette

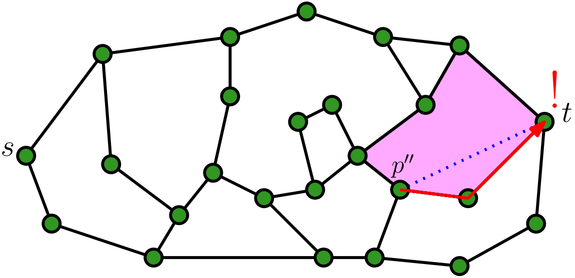
Optimiertes Facettenrouting (OFR)



Optimiertes Facettenrouting [Wattenhofer et al. '03]

- ≫ Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- ≫ Kehre zum Punkt zurück, der t am nächsten ist siehe Update
- ≫ Wechsle zur angrenzenden Facette

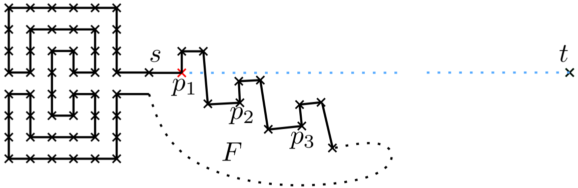
Optimiertes Facettenrouting (OFR)



Optimiertes Facettenrouting [Wattenhofer et al. '03]

- Erkunde Facette, in die $\overrightarrow{st}$ zeigt
- Kehre zum Punkt zurück, der t am nächsten ist siehe Update
- Wechsle zur angrenzenden Facette

Greedy+Optimiertes Facettenrouting (GOFR)



- Greedy Routing stockt bei p_1
- Facettenrouting endet nicht in Schnitt von $\overrightarrow{p_1 t}$ mit Facettenrand, sondern bei p_t
- Greedy Routing kommt nicht wieder an den Rand von F.

Optimiertes adaptives FR (OAFR)

Satz

Für das optimierte Facettenrouting gelten alle Aussagen bis auf die Suboptimalität in Verbindung mit Greedy-Routing analog.

- Optimiertes FR findet in $O(n)$ Schritten zum Ziel.
- Beschränktes optimiertes Facettenrouting mit geschätzter Pfadlänge c findet in c^2 Schritten zum Ziel oder erkennt fehlenden Zusammenhang.
- Adaptives optimiertes FR findet in $c^2(p^\star)$ zum Ziel.
- Adaptives optimiertes FR erkennt fehlenden Zusammenhang in $O(n' \log n')$ Schritten.

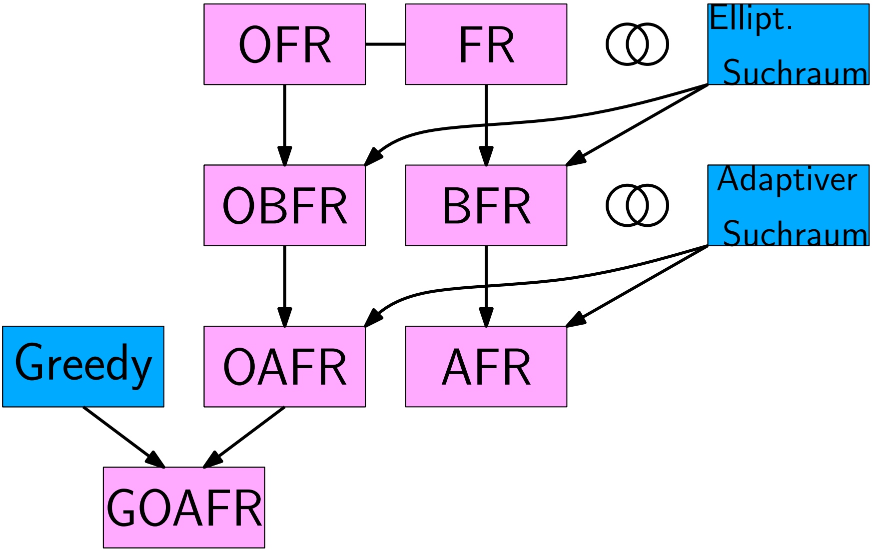
Greedy + Optimiertes adaptives FR (GOAFR)

Greedy + Optimiertes adaptives Facettenrouting

1. Route greedy, bis Paket an Knoten p steckenbleibt
2. Starte optimiertes, adaptives FR für Start p, Ziel t
3. Stoppe, sobald *eine Facette* passiert ist
4. Gehe zurück zu 1.

Satz

GOAFR ist asymptotisch worst-case-optimal. (Ohne Beweis)



OFR
FR
Ellipt. Suchraum
OBFR
BFR
Adaptiver Suchraum
Greedy
OAFR
AFR
GOAFR

Was mitnehmen?

- Neue Modelle!
  - UDG: stark idealisiert, aber sehr mächtig
  - QUDG: etwas allgemeiner, immer noch idealisiert
  - $\Omega(1)$-Modell: vernünftige Annahme zur Analyse!
- Eine große Hochzeit aus guten Ideen
  - Greedy Routing
  - Facettenrouting dem Haus „Planare Graphenalgorithmen“
    - bewegte Familiengeschichte!
  - (eigentlich erst möglich durch Planarisierungsalgorithmen)

Denkanstoß

?

Können wir irgendwas davon noch in 3D verwenden?

- Modelle?
- Greedy Routing?
- Facettenrouting?
- Schranken?

Literatur

1. E. Kranakis, H. Singh, J. Urrutia: *Compass Routing on geometric networks*. In Proceedings of the 11th Canadian Conference on Computational Geometry, 1999
2. P. Santi: *Topology Control in Wireless Ad Hoc and Sensor Networks*. Wiley, 2005
3. F. Kuhn, R. Wattenhofer, Y. Zhang, A. Zollinger: *Geometric ad-hoc routing: Of theory and practice*. In: Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing (PODC'03)