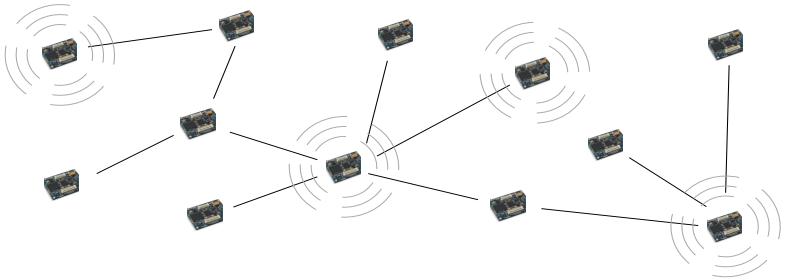
Algorithmen für Ad-hoc- und Sensornetze

VL 09 – Clustering (1. Teil)

Markus Völker | 20. Juni 2012 (Version 2)

Institut für Theoretische Informatik - Lehrstuhl für Algorithmik (Prof. Wagner)



Überblick

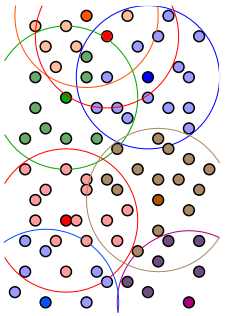
- Was ist Clustering und wofür brauche ich das?
  - Selbstorganisation und Aufgabenteilung in WSN
  - Formalisierung: Dominating Sets
- Minimum Dominating Sets in allgemeinen Graphen
  - Optimale Approximation (ist das gut genug?)
- Minim**um** Dominating Sets in Sensornetzen: UDGs und mehr
  - Welche Eigenschaften/Modelle können uns helfen?
  - Minim**um** Dominating Sets und Maxim**al** Independent Sets
  - Algorithmen für Maximal Independent Sets

Motivation: Selbstorganisation

Sensornetze können sehr viel dichter sein als benötigt. Nicht immer müssen alle Knoten messen, Nachrichten weiterleiten, wach sein...

- Selbstorganisation weist Knoten Aufgaben zu

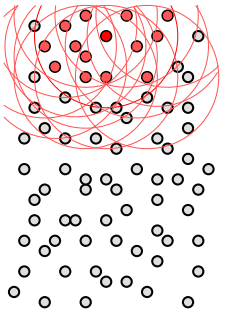
Beim *Clustering* werden unter den Knoten *Clusterheads* ausgewählt, die bestimmte Aufgaben für eine Menge umgebender Knoten übernehmen.



Beispiel: Broadcast/Fluten

Broadcast durch Fluten bisher:

- *jeder* Knoten, der eine Nachricht zum ersten Mal hört, wiederholt sie.



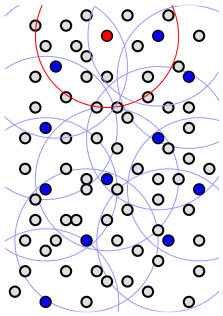
Beispiel: Broadcast/Fluten

Broadcast durch Fluten bisher:

- *jeder* Knoten, der eine Nachricht zum ersten Mal hört, wiederholt sie.

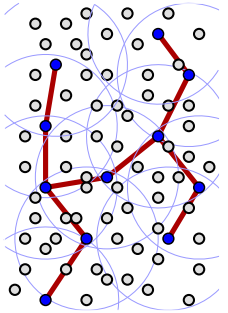
Was, wenn wir Knoten als *Gateways* auszeichnen?

- *nur Gateways*, die eine Nachricht zum ersten Mal hören, wiederholen sie.
- das spart sicher viel Energie!
- was müssen wir fordern, damit unsere Nachricht überall ankommt?



Wann geht Fluten mit Gateways gut?

1. Jeder Knoten muss ein Gateway erreichen können
   - jeder Knoten hat ein Gateway in der Nachbarschaft
2. Fluten innerhalb der Gateways muss alle Gateways erreichen
   - die Gateways *induzieren* einen zusammenhängenden Graphen („Backbone“)
3. jeder Knoten muss ein Gateway hören
   - jeder Knoten hat ein Gateway in der Nachbarschaft (hatten wir schon!)



(Connected) Dominating Set

Definition: Dominating Set

Sei $G = (V, E)$ ein Graph und $U \subset V$ eine Menge von Knoten.

- Ein Knoten $v \in V$ ist von U *dominiert*, wenn er einen Nachbarn $u \in U$ hat
- U heißt *Dominating Set*, wenn jedes $v \in V \setminus U$ von U dominiert ist.
- Ein Dominating Set U in G heißt *Connected Dominating Set*, wenn es zwischen je zwei Knoten in U einen Pfad gibt, der nur Knoten aus U verwendet.

- DS: Jeder Nicht-Clusterhead $v \in V \setminus U$ hat einen Clusterhead $u \in U$ in der 1-hop-Nachbarschaft
- CDS: zusätzlich hängen Clusterheads zusammen

Anwendungen und Ziele

- Broadcast/Flooding/Routing (CDS)
  - jede Kommunikation wird einfacher, wenn man nur auf dem Backbone routen muss!
  - Knoten adressieren sich zusätzlich mit zugehörigem Clusterhead, und nur Clusterheads müssen wissen, wie sie sich untereinander erreichen!
- lokale Aufgabenverteilung/Sensoreinsatzplanung (DS?)
  - Clusterheads lösen Aufgabenzuordnung lokal
- Scheduling/Nutzung des drahlosen Kanals (DS?)
  - Bsp: Clusterheads verabreden konfliktfreies Nutzung des Kanals und „teilen“ verwaltete Slots den Knoten im Cluster zu

Wir interessieren uns vor allem für Lösungen, die *verteilt und schnell kleine Dominating Sets* liefern!
(Exakt/Approximation? $o(n)$, $O(\sqrt{n})$, $O(\log n)$, $\Theta(1)$ Schritte?)

Minimum Dominating Set (MDS)

Minimum Dominating Set

Gegeben: Graph $G = (V, E)$.

Gesucht: Dominating Set $U \subset V$ *minimaler* Kardinalität[a].

[a] engl: *minimum*: minimale Kardinalität, *minimal*: inklusionsminimal

- Problem ist NP-schwer

$\Rightarrow$ Wir suchen gute Approximation

  - (so wenig aktive Knoten wie möglich)

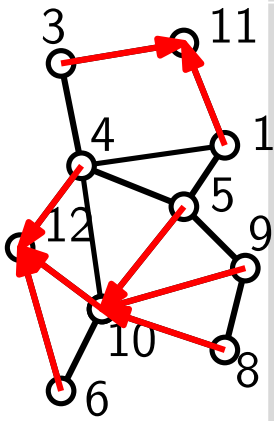
- Modelle sind entscheidend! Wir werden
  - allgemeine Graphen
  - positionsbewusste Unit-Disk-Graphen
  - Verallgemeinerungen von Unit-Disk-Graphen (BIGs)

  betrachten.
- (Zu „Minimum Connected DS“ kommen wir später)

Ein ganz schneller DS-Algorithmus

Größte-ID-DS

1. initial sind alle Knoten weiß
2. jeder Knoten sendet seine ID an alle Nachb.
3. jeder Knoten färbt Nachbarn mit höchster ID schwarz, oder sich selbst, wenn es keinen mit höherer ID gibt

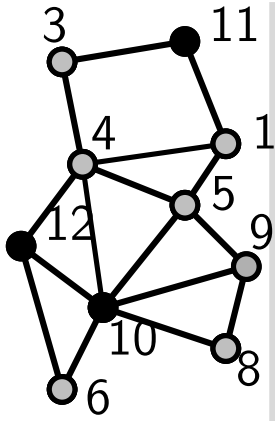


Ein ganz schneller DS-Algorithmus

Größte-ID-DS

1. initial sind alle Knoten weiß
2. jeder Knoten sendet seine ID an alle Nachb.
3. jeder Knoten färbt Nachbarn mit höchster ID schwarz, oder sich selbst, wenn es keinen mit höherer ID gibt

- schwarze Knoten sind DS (klar)
- Größte-ID terminiert nach 2 Runden!
- Approximationsfaktor?

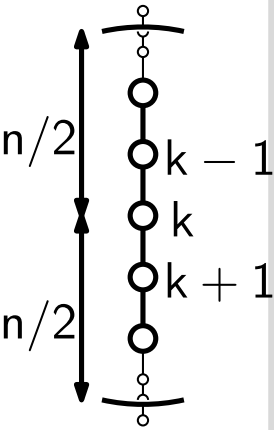


Ein ganz schneller DS-Algorithmus

Größte-ID-DS

1. initial sind alle Knoten weiß
2. jeder Knoten sendet seine ID an alle Nachb.
3. jeder Knoten färbt Nachbarn mit höchster ID schwarz, oder sich selbst, wenn es keinen mit höherer ID gibt

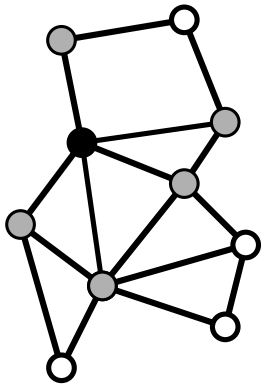
- schwarze Knoten sind DS (klar)
- Größte-ID terminiert nach 2 Runden!
- Approximationsfaktor? $\Omega(n)$!
  - Pfad, jeder sieht die vorigen/letzten $n/2$
  - Optimum: nur mittleren Knoten wählen
  - Aber jeder Knoten $k \geq n/2$ ist größter Nachbar von Knoten an Stelle $k - n/2$!



Schlechtes Greedy-DS

Schlechtes Greedy-DS

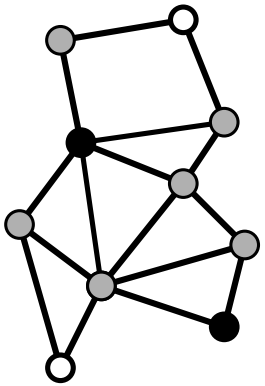
1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle einen weißen Knoten u, färbe ihn schwarz und alle weißen Nachbarn von u grau
3. Gib die schwarzen Knoten zurück



Schlechtes Greedy-DS

Schlechtes Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle einen weißen Knoten u, färbe ihn schwarz und alle weißen Nachbarn von u grau
3. Gib die schwarzen Knoten zurück



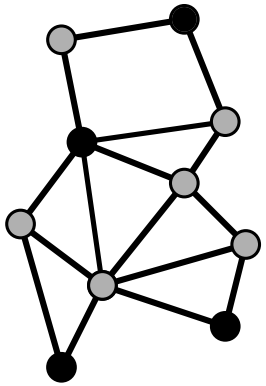
Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (10/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Schlechtes Greedy-DS

Schlechtes Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle einen weißen Knoten u, färbe ihn schwarz und alle weißen Nachbarn von u grau
3. Gib die schwarzen Knoten zurück

- schwarze Knoten sind DS (klar, oder?)
- Approximationsfaktor?



Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (10/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Schlechtes Greedy-DS

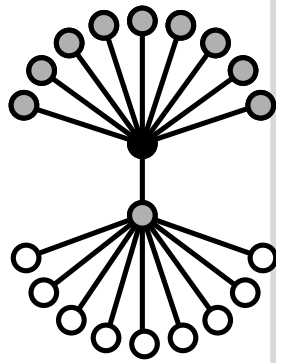
Schlechtes Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle einen weißen Knoten u, färbe ihn schwarz und alle weißen Nachbarn von u grau
3. Gib die schwarzen Knoten zurück

- schwarze Knoten sind DS (klar, oder?)
- Approximationsfaktor? $\Theta(n)$!

$\Rightarrow o(n)$-Approximation geht nur, wenn benachbarte Knoten im DS erlaubt sind!

- Nachher werde ich das Gegenteil behaupten ;)



Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (10/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

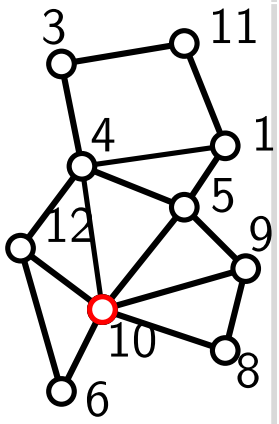
Gutes Greedy-DS

k-Hop-Nachbarschaft

Im folgenden bezeichnen wir die k-Hop-Nachbarschaft eines Knotens u als $N^k(u) := \{v : d_G(u,v) \leq k\}$

Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle Knoten u mit maximaler Anzahl weißer Knoten in $N^1(u)$, färbe ihn schwarz und alle weißen Nachbarn u grau
3. Gib die schwarzen Knoten zurück



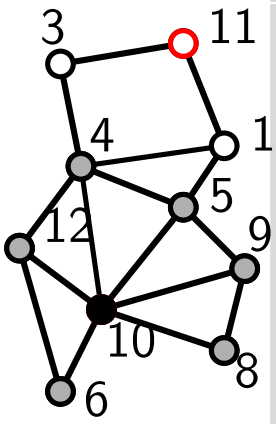
Gutes Greedy-DS

k-Hop-Nachbarschaft

Im folgenden bezeichnen wir die k-Hop-Nachbarschaft eines Knotens u als $N^k(u) := \{v : d_G(u,v) \leq k\}$

Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle Knoten u mit maximaler Anzahl weißer Knoten in $N^1(u)$, färbe ihn schwarz und alle weißen Nachbarn u grau
3. Gib die schwarzen Knoten zurück



Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (11/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

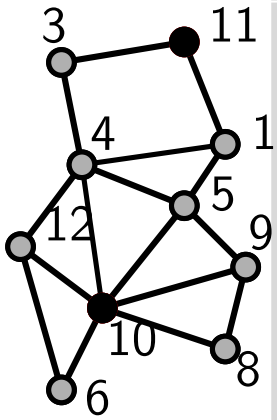
Gutes Greedy-DS

k-Hop-Nachbarschaft

Im folgenden bezeichnen wir die k-Hop-Nachbarschaft eines Knotens u als $N^k(u) := \{v : d_G(u,v) \leq k\}$

Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle Knoten u mit maximaler Anzahl weißer Knoten in $N^1(u)$, färbe ihn schwarz und alle weißen Nachbarn u grau
3. Gib die schwarzen Knoten zurück



Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (11/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Gutes Greedy-DS

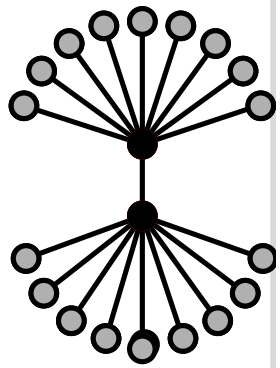
k-Hop-Nachbarschaft

Im folgenden bezeichnen wir die k-Hop-Nachbarschaft eines Knotens u als $N^k(u) := \{v : d_G(u,v) \leq k\}$

Greedy-DS

1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle Knoten u mit maximaler Anzahl weißer Knoten in $N^1(u)$, färbe ihn schwarz und alle weißen Nachbarn u grau
3. Gib die schwarzen Knoten zurück

- DS ist klar! Approximation?



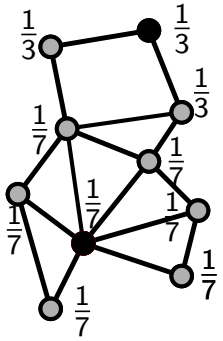
Markus Völker – Algorithmen für Ad-hoc- und Sensornetze
VL 09 – Clustering (1. Teil) (11/31)
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Satz & Beweis, Teil 1: Cover-Kosten

Satz

$|\mathrm{DS}_{\mathrm{Greedy}}| \leq (1 + \ln \Delta)|\mathrm{DS}_{\mathrm{OPT}}|$ $\quad (\Delta := \max_{v \in V} \deg v)$

- Wir legen Kosten auf markierte Knoten um!
  - in jedem Schritt wird *ein* Knoten schwarz (das sind die Kosten)
  - die Kosten legen wir um auf alle Knoten, die in diesem Schritt von weiß zu grau/schwarz gefärbt werden
    - werden 7 weiße Knoten gefärbt, bekommt jeder Kosten 1/7
  - jeder Knoten bekommt nur einmal Kosten auferlegt
  - Zu zeigen: Insgesamt sind die Kosten aller Knoten zusammen unter $(1 + \ln \Delta) \cdot |DS_{\mathrm{OPT}}|$



Beweis, Teil 2

Lemma (Beweis kommt gleich)

Sei $v \in V$ ein beliebiger Knoten. Die Kosten innerhalb von $N^1(v)$ betragen höchstens $1 + \ln(\deg v)$.

- Das reicht: Wir können dann ein beliebiges minimales $\mathrm{DS}_{\mathrm{OPT}}$ fixieren und die Kosten der Nachbarschaften aufsummieren:

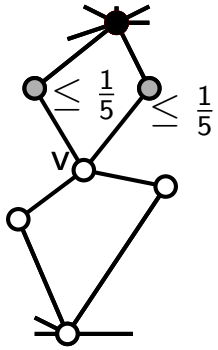
$$\begin{aligned} |\mathrm{DS}_{\mathrm{Greedy}}| &\overset{\text{Def.}}{\leq} \sum_{v \in V} \text{Kosten}(v) \\ &\overset{\text{DS}}{\leq} \sum_{u \in \mathrm{DS}_{\mathrm{OPT}}} \sum_{v \in N^1(u)} \text{Kosten}(v) \\ &\overset{\text{Lemma}}{\leq} \sum_{u \in \mathrm{DS}_{\mathrm{OPT}}} (1 + \ln(\deg u)) \\ &\leq |\mathrm{DS}_{\mathrm{OPT}}| \, (1 + \ln \Delta) \end{aligned}$$

Beweis des Lemmas

Lemma

Sei $v \in V$ ein beliebiger Knoten. Die Kosten innerhalb von $N^1(v)$ betragen höchstens $1 + \ln(\deg v)$.

- Der Trick: Wird ein Knoten $w \in N^1(v)$ gefärbt, wenn noch k Knoten in $N^1(v)$ weiß sind, dann bekommt w höchstens Kosten $1/k$ angerechnet.
  - wer immer schwarz gemacht wurde, hat mindestens k weiße Knoten in seiner Nachbarschaft!
  - sonst hätte eher v ausgewählt werden müssen!



Beweis des Lemmas

Lemma

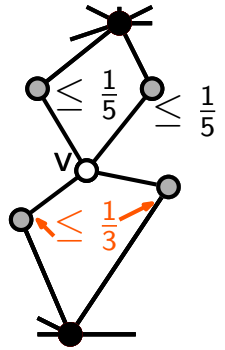
Sei $v \in V$ ein beliebiger Knoten. Die Kosten innerhalb von $N^1(v)$ betragen höchstens $1 + \ln(\deg v)$.

- Also: alle Knoten in $N^1(v)$, die in einer Runde gefärbt werden, bekommen maximal Kosten $1/k$, wenn zu Beginn der Runde k Knoten in $N^1(v)$ weiß sind.

$\Rightarrow$ Im schlimmsten Fall werden alle Knoten in $N^1(v)$ in unterschiedlichen Runden abgedeckt!

  - $\frac{1}{5} + \frac{1}{5} + \frac{1}{3} + \frac{1}{3} + \frac{1}{1} < \frac{1}{5} + \frac{1}{4} + \frac{1}{3} + \frac{1}{2} + \frac{1}{1}$

- Kosten in $N^1(v)$: $\frac{1}{\deg(v)+1} + \frac{1}{\deg(u)} + \cdots + \frac{1}{2} + \frac{1}{1}$
- $= H(\deg(v) + 1) \leq 1 + \ln(\deg(v))$



Greedy-MDS

Satz

$|DS_{Greedy}| \leq (1 + \ln \Delta)|DS_{OPT}|$ $(\Delta := \max_{v \in V} \deg v)$

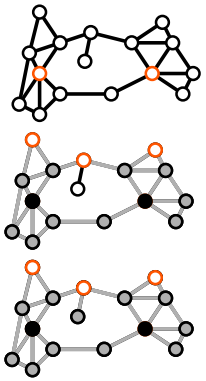
- das haben wir bewiesen
  - bei jeder Auswahl eines schwarzen Knotens haben wir Kosten 1 verteilt
  - in jeder Nachbarschaft eines Knotens v liegen maximal Kosten $1 + \ln(\deg(v))$
  - die Nachbarschaften eines MDS decken alle Knoten ab!
- Man kann zeigen, dass in Polynomialzeit keine Approximation $o(\log \Delta)$ möglich ist, für $P \neq NP$

$\Rightarrow$ Greedy-MDS ist im wesentlichen optimal.

- geht Greedy-MDS auch verteilt und schnell?

Die gute Nachricht!

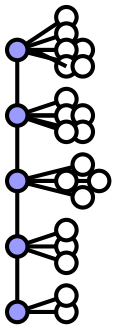
Greedy-MDS lässt sich sehr gut verteilen!



- Drei Schritte pro Runde! Jeder Knoten u
  1. zählt weiße Knoten in Nachbarschaft
  2. tauscht Zähler mit allen Knoten $v \in N^2(u)$ aus (ignoriere Kanten zu grauen Knoten)
  3. enthält kein $N^1(v)$ eines solchen v mehr weiße Knoten als $N^1(u)$, färbt sich u schwarz und teilt ungefärbten Nachbarn mit, dass sie jetzt grau sind.
  4. bei Gleichstand „gewinnt“ höhere ID
- Das ergibt dasselbe Ergebnis wie Greedy-MDS:
  - die Anzahl der weißen Knoten in $N^1(u)$ kann sich nur verringern, wenn ein Knoten aus $N^2(u)$ schwarz gefärbt wird (das geht aber nicht!)

Die schlechte Nachricht.

Greedy-MDS benötigt im worst-case $\Theta(\sqrt{n})$ Schritte!



- Greedy-MDS wählt genau die blauen Knoten, und zwar einen nach dem anderen!
- bei k blauen Knoten gibt es $\approx k^2/2$ Knoten insgesamt
- das kann auch verteilt nicht schneller gehen, weil jeder Knoten sicher ist, dass er in seiner Umgebung nicht der nächste Knoten ist, bis er tatsächlich ausgewählt wird.

Diskussion: MDS in allgemeinen Graphen

Wir können MDS in allgemeinen Graphen sehr leicht approximieren (sogar verteilt), aber nicht *lokal*. Dafür können wir sehr lokal ein DS bestimmen, aber das kann beliebig groß sein.

- Größte-ID-DS: 2 Runden, aber Approximation in $\Theta(n)$
- Gutes Greedy-DS: optimale Approximation, aber $\Theta(\sqrt{n})$ Schritte

Geht beides gleichzeitig?

- Ja, aber nur mit *richtig* schweren Geschützen!
  - Kuhn, Moscibroda, 2006
  - approximiert MDS in k Runden erwartet mit Faktor $O\left((\Delta+1)^{c/\sqrt{k}} \log \Delta\right)$ (verteilte LP-Approximation)
  - kaum geeignet für Sensornetze
- Bessere Frage: Können wir *in Sensornetzen* schnell und verteilt gut approximieren?

UDG mit Koordinaten: Grid-MDS

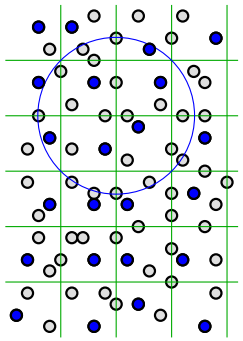
Satz

Es gibt einen positionsbewussten Algorithmus, der in UDG in einer Runde eine konstante Approximation für MDS berechnet.

- teile Ebene in Gitterzellen mit Diagonale R (R ist die Reichweite)

$\Rightarrow$ jeder Knoten ist zu jedem anderen in seiner Zelle benachbart

- in jeder Zelle wird der Knoten schwarz, der der Zellenmitte am nächsten ist
- Fertig, das ist DS, und enthält maximal 16 mal so viele Knoten wie ein DS_{OPT}:
  - Der Umkreis eines Knotens aus DS_{OPT} schneidet maximal 16 Zellen und in jeder liegt maximal ein ausgewählter Knoten



Ein Blick auf Modelle

Schnelle Approximation hängt ganz wesentlich vom Modell für unsere Graphen ab, was ist sinnvoll?

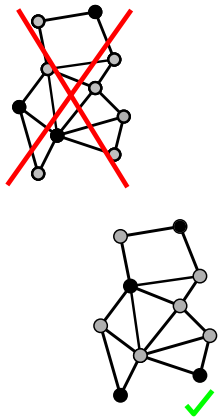
- UDG mit bekannten Positionen ist schon extrem optimistisch
  - wenigstens machen wir keinen echten Fehler, wenn Knoten sich nicht hören, sondern bekommen immer noch ein DS!
- was haben wir sonst noch?
  - nur UDG? QUDG? Noch was allgemeineres?
  - was wollen wir denn überhaupt ausnutzen?

Unabh. Mengen (Independent Sets)

Definition: Unabhängige Knoten

Sei $G = (V, E)$ ein Graph. Eine Knotenmenge $I \subset V$ heißt *unabhängig*, wenn jede Kante $e \in E$ höchstens einen Endpunkt in I hat.

- d.h. $v \in I \Rightarrow$ kein Nachbar von $v \in I$



Maximal Independent Sets (MIS)

Maxim*al* Independent Set

Gegeben: Graph $G = (V, E)$.

Gesucht: *Inklusionsmaximales* Independent Set $I \subset V$. [a]

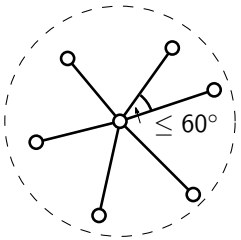
[a] engl: *maximum*: maximale Kardinalität, *maximal*: inklusionsmaximal

- ein großer Unterschied:
  - Kardinalitätsmaximale unabhängige Menge
    - engl: Maxim**um** Independent Set
    - NP-schwer zu finden, hilft uns nicht
  - Inklusionsmaximale unabhängige Menge
    - engl: Maxim**al** Independent Set (MIS)
    - wird uns noch weiter beschäftigen!
    - Maximale Mengen findet man greedy!
    - solange es geht, füge Knoten hinzu!

Das Sternlemma...

Sternlemma

Sei $G = (V, E)$ ein Unit-Disk-Graph, $I \subset V$ eine unabhängige Menge und $v \in V$ ein beliebiger Knoten. Die 1-Hop-Nachbarschaft $N^1(v)$ enthält höchstens 5 Knoten aus I.



- Unter sechs Nachbarn eines Knotens müssen mindestens zwei zueinander benachbart sein.
  - das Argument ist nicht sooo neu:
  - bei 6 oder mehr Nachbarn muss es ein Paar geben, das einen Winkel $\leq 60°$ einschließt.

  $\Rightarrow$ die müssen sich dann sehen!

...und sein Nutzen

Satz

Sei G ein Unit-Disk-Graph und I eine maximale unabhängige Menge. I ist ein Dominating Set und $|I| \leq 5 \cdot |\mathrm{DS}_{\mathrm{OPT}}|$.

- I ist DS:
  - Annahme: es gibt einen von I nicht-dominierten Knoten v

  $\Rightarrow$ Dann ist kein Nachbar von v in I

  $\Rightarrow$ $I \cup \{v\}$ ist unabhängig, I kein MIS!
- $|I| \leq 5 \cdot |\mathrm{DS}_{\mathrm{OPT}}|$
  - wenn wir über Knoten in $\mathrm{DS}_{\mathrm{OPT}}$ die Nachbarn in I zählen, zählen wir jeden Knoten aus I mindestens einmal.
  - das sind höchstens $5 \cdot |\mathrm{DS}_{\mathrm{OPT}}|$ (Sternlemma!)

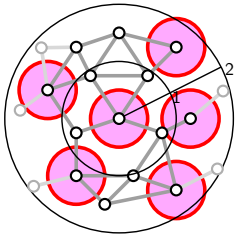
Verallgemeinerung: Bounded Independence

Definition: Bounded Independence Graph (BIG)

Ein Graph G hat beschränkte Unabhängigkeit, wenn es eine Konstante c gibt, so dass in jeder r-Hop-Nachbarschaft maximal $O(r^c)$ unabhängige Knoten liegen. Wir nennen G einen BIG.

- Unit-Disk-Graphen haben beschränkte Unabhängigkeit
  - jede r-Hop-Nachbarschaft liegt in einem Kreis C_r mit Radius r und Fläche πr^2.
  - ist I ein Independent Set, berühren sich Kreise mit Radius $1/2$ um die $i \in I$ nicht.
  - Mindestens ein Drittel jedes solchen Kreises liegt innerhalb von C_r, das ist eine Fläche von $\frac{\pi}{12}$.

  $\Rightarrow$ jedes Independent Set hat höchstens $\pi r^2 / \frac{\pi}{12} = 12r^2$ Knoten innerhalb einer r-Hop-Nachbarschaft

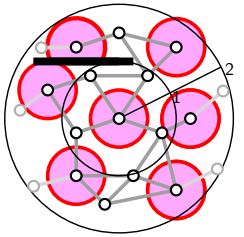


Verallgemeinerung: Bounded Independence

Definition: Bounded Independence Graph (BIG)

Ein Graph G hat beschränkte Unabhängigkeit, wenn es eine Konstante c gibt, so dass in jeder r-Hop-Nachbarschaft maximal $O(r^c)$ unabhängige Knoten liegen. Wir nennen G einen BIG.

- Unit-Disk-Graphen haben beschränkte Unabhängigkeit
- Quasi-Unit-Disk-Graphen haben beschränkte Unabhängigkeit (warum?)
- Beschränkte Unabhängigkeit bildet noch viel komplexere Fälle ab, z.B. QUDG mit konstanter Anzahl an Hindernissen
  - Konstante c charakterisiert dann die Komplexität



Zusammenhänge

- Inklusionsmaximale Independent Sets sind konstante Approximation für (kardinalitäts-)Minimale Dominating Sets in jedem Graphen mit beschränkter Unabhängigkeit (BIG)
  - Sternlemma ist nur Sonderfall für UDGs

Knobelaufgabe

Gegeben ein MIS I in einem Graphen mit beschränkter Unabhängigkeit, wie finde ich ein *Connected* Dominating Set mit $O(|I|)$ Knoten?

Jetzt

Wie berechnet man schnell ein MIS?

Berechnung MIS

Beobachtung

Beim Greedy-Algorithmus für MDS mussten wir Knoten geschickt wählen, weil wir eine *kardinalitätsminimale* dominierende Menge gesucht haben. Jetzt tut es eine *inklusionsmaximale* unabhängige Menge.

- *Jetzt können wir Knoten willkürlich auswählen!*

Greedy-MIS

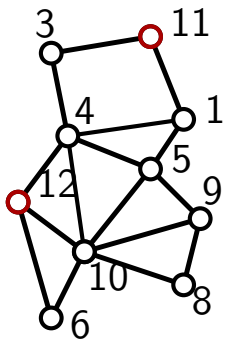
1. initial sind alle Knoten weiß
2. solange es weiße Knoten gibt, wähle einen weißen Knoten u, färbe ihn schwarz und alle weißen Nachbarn von u grau
3. Gib die schwarzen Knoten zurück

jedes MIS ist DS und approximiert MDS in Graphen mit beschränkter Unabhängigkeit konstant!

Verteilter Greedy-MIS-Algorithmus

Greedy-MIS verteilt

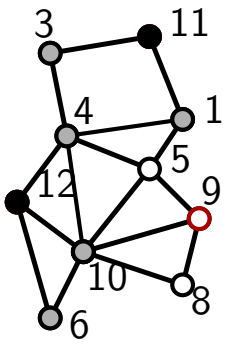
1. initial sind alle Knoten weiß
2. jeder Knoten wartet, bis alle Nachbarn mit höherer ID eine Farbe haben
3. ist er dann noch weiß, färbt er sich schwarz und alle Nachbarn weiß



Verteilter Greedy-MIS-Algorithmus

Greedy-MIS verteilt

1. initial sind alle Knoten weiß
2. jeder Knoten wartet, bis alle Nachbarn mit höherer ID eine Farbe haben
3. ist er dann noch weiß, färbt er sich schwarz und alle Nachbarn weiß

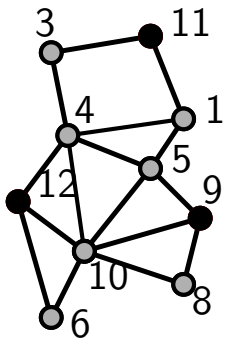


Verteilter Greedy-MIS-Algorithmus

Greedy-MIS verteilt

1. initial sind alle Knoten weiß
2. jeder Knoten wartet, bis alle Nachbarn mit höherer ID eine Farbe haben
3. ist er dann noch weiß, färbt er sich schwarz und alle Nachbarn weiß

- das berechnet ein MIS (klar?) und

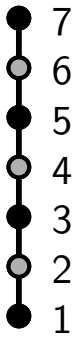


Verteilter Greedy-MIS-Algorithmus

Greedy-MIS verteilt

1. initial sind alle Knoten weiß
2. jeder Knoten wartet, bis alle Nachbarn mit höherer ID eine Farbe haben
3. ist er dann noch weiß, färbt er sich schwarz und alle Nachbarn weiß

- das berechnet ein MIS (klar?) und
- kann $\Theta(n)$ Schritte dauern.
  - sind die IDs in einem Pfad geordnet, werden in jedem Schritt immer nur zwei Knoten eingefärbt, einer schwarz, und einer grau



Wir könnten wieder zufällige IDs betrachten, aber mit Randomisierung geht noch was viel besseres!
⇒ Details nächste Woche...

(Vorläufiges) Wrap-Up *Clustering*

- *Clustern*: Clusterheads auswählen, so dass jeder Knoten einen CH in seiner Nachbarschaft hat
  - das ist ein Dominating Set
    - manchmal sollte es zusätzlich verbunden sein (CDS)
  - Kardinalitätsminimale DS (MDS) sind schwer zu berechnen
  - Schnelle Approximation ist in allgem. Graphen sehr komplex
- In Modellen für Sensornetze ist das leichter!
  - Wenn UDG/QUDG zu streng sind — BIG ist eigentlich immer gerechtfertigt!
  - In BIGs sind inklusionsmaximale Independent Sets (MIS) schon konstante Approximationen für MDS
- *Nächste Woche:*
  - MIS lassen sich in $O(\log(n))$ Runden in *jedem* Graphen verteilt berechnen!

Literatur

1. V. Chvátal: *A greedy heuristic for the set covering problems*. In: Operations Research 4(3), Seiten 233–235, 1979
2. S. Schmid, R. Wattenhofer: *Algorithmic Models for Sensor Networks*. In: 14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), 2006