Zusammenhang

Ying Gu

Universität Heidelberg Fakultät für Mathematik und Informatik

Seminar "Analyse von Netzwerken" 10. Nov 2009

Defintionen

Defintion: k-knotenzusammenhängend , $\kappa(G)$

Ein ungerichteter Graph $G = (V, E)$ heißt k-knotenzusammenhängend, falls $|V| > k$ und für jede echte Knotenteilmenge $X \subset V$ mit $|X| < k$ der Graph $G - X$ zusammenhängend ist. Der Knotenzusammenhang $\kappa(G)$ des Graphen G ist die größte natürliche Zahl k, für die G k-knotenzusammenhängend ist.

Defintion: k-kantenzusammenhängend, $\lambda(G)$

Ein ungerichteter Graph $G = (V, E)$ heißt k-kantenzusammenhängend, falls $|V| \geq 2$ und für jede Kantenteilmenge $Y \subseteq E$ mit $|Y| < k$ der Graph $G - Y$ zusammenhängend ist.
Der Kantenzusammenhang $\lambda(G)$ des Graphen G ist die größte natürliche Zahl k, für die G k-kantenzusammenhängend ist.

Definitionen

Defintion: Knoten-Separator, Kanten-Separator

Sei $G = (V, E)$ ein ungerichteter Graph.
Eine Knotenteilmenge $C \subset V$ heißt Knoten-Separator, wenn die Anzahl der Zusammenhangskomponenten in $G - C$ größer als in G ist.
Falls zwei Knoten s und t zwar in G in der gleichen Zusammenhangskomponente sind, aber nicht in $G - C$, dann bezeichnet man C als s-t-Knoten-Separator.
Kanten-Separatoren und s-t-Kanten-Separatoren sind analog definiert.
s-t-Separatoren können auch auf gerichtete Graphen übertragen werden: eine Knoten- bzw. Kantenmenge ist dann ein s-t-Separator, wenn es keinen gerichteten Pfad mehr von s nach t gibt, nachdem die Menge aus dem Graph entfernt wurde.

Definitionen

Defintion: $\kappa_G(s,t)$, $\lambda_G(s,t)$

Für zwei Knoten s und t eines Graphen G ist der lokale (Knoten-)Zusammenhang $\kappa_G(s,t)$ definiert als die minimale Anzahl von Knoten, die entfernt werden müssen, damit es keinen Weg mehr von s nach t gibt.
Für den Fall, dass zwischen s und t eine Kante existiert, können sie nicht durch das Löschen von Knoten separiert werden. Deshalb wird der lokale (Knoten-)Zusammenhang in diesem Fall $\kappa_G(s,t) = n-1$ definiert.
(Anderenfalls wäre höchstens $\kappa_G(s,t) = n-2$ möglich.)
Der lokale Kanten-Zusammenhang $\lambda_G(s,t)$ zweier Knoten s und t ist entsprechend definiert als die minimale Anzahl von Kanten, die entfernt werden müssen, damit es keinen Weg mehr von s nach t gibt.

Defintionen

Definition: Artikulationsknoten, Brücke, Zweifachzusammenhangskomponente, Block

Ein Artikulationsknoten ist ein Knoten, der beim Entfernen aus dem Graphen die Anzahl der Zusammenhangskomponenten erhöht.
Eine Brücke ist eine Kante, die beim Entfernen aus dem Graphen die Anzahl der Zusammenhangskomponenten erhöht.
Eine Zweifachzusammenhangskomponente ist ein maximaler 2-fach (knoten-)zusammenhängender Teilgraph.
Ein Block ist ein maximaler zusammenhängender Teilgraph, der keinen Artikulationsknoten enthält, d.h. die Menge aller Blocks eines Graphen besteht aus den isolierten Knoten, den Brücken, sowie den Zweifachzusammenhangskomponenten.

Definitionen

Defintion: Schnitt (Cut)

Ein Schnitt (A, B) in einem gerichteten oder ungerichteten Graphen G ist eine Partition der Knotenmenge in nichtleere Teilmengen, d.h., $V(G) = A \cup B$ mit $A \cap B = \emptyset, A \neq \emptyset$ und $B \neq \emptyset$.

Defintion: $w(X, Y)$

Für einen ungerichteten Graphen bezeichnet $w(X, Y)$ Gewichtssumme der Kanten von einem Knoten in X zu einem Knoten in Y, wobei $X, Y \subseteq V$ mit $X \cap Y = \emptyset$ (für gerichtete Graphen analog)

Definition: Minimaler Schnitt (Minimum Cut)

Ein Minimaler Schnitt ist ein Schnitt S, sodass für alle anderen Schnitte T gilt, $w(S, V \setminus S) \leq w(T, V \setminus T)$.

Fundamentale Sätze

Satz

Für jeden nicht-trivialen Graphen G gilt:

$$\kappa(G) \leq \lambda(G) \leq \delta(G)$$

Fundamentale Sätze

Satz

Der maximale (Knoten-/Kanten-)Zusammenhang in einem Graphen mit n Knoten und m Kanten ist

$$\begin{cases} \left\lfloor \frac{2m}{n} \right\rfloor & , \quad \text{if } m \geq n-1 \\ 0 & , \quad \text{otherweise.} \end{cases}$$

Der minimale (Knoten-/Kanten-)Zusammenhang in einem Graphen mit n Knoten und m Kanten ist

$$\begin{cases} m - \binom{n-1}{2} & , \quad \text{if } \binom{n-1}{2} < m \leq \binom{n}{2} \\ 0 & , \quad \text{otherweise.} \end{cases}$$

Für jeden Graphen G mit $\delta(G) \geq \left\lfloor \frac{n}{2} \right\rfloor$ *gilt* : $\lambda(G) = \delta(G)$

Fluss-basierte Knotenzusammenhangsalgorithmen

Defintion: Unit Capacity Network, Typ 1, Type 2

- Ein Graph wird als Unit Capacity Network (oder $0-1$ Network) bezeichnet, falls die Kapazität aller Kanten gleich 1 ist.
- Ein Unit Capacity Network ist vom Typ 1, falls es keine parallelen Kanten hat.
- Es ist vom Typ 2, falls für jeden Knoten $v(v \neq s, v \neq t)$ entweder der Eingangsgrad $d^-(v)$ oder der Ausgangsgrad $d^+(v)$ gleich 1 ist.

Fluss-basierte Knotenzusammenhangsalgorithmen

Lemma

- Ein MaxFlow/MinCut kann für ein Unit Capacity Network (mit Dinitz'Algorithmus) in Zeit $\mathcal{O}(m^{3/2})$ berechnet werden.
- Für Unit Capacity Networks vom Typ 1 ist die Zeitkomplexität von Dinitz' Algorithmus $\mathcal{O}(n^{2/3}m)$.
- Für Unit Capacity Networks vom Typ 2 ist die Zeitkomplexität von Dinitz' Algorithmus $\mathcal{O}(n^{1/2}m)$.

Ungerichtete ungewichtete Graphen

- Gegeben: ungerichteter (ungewichteter) Graph $G = (V, E)$ mit n Knoten und m Kanten
- Konstruiere gerichteten Graph $\bar{G} = (\bar{V}, \bar{E})$ mit $|\bar{V}| = 2n$ und $|\bar{E}| = 2m + n$ wie folgt:
  - Ersetze jeden Knoten $v \in V$ durch zwei Knoten $v', v'' \in \bar{V}$, verbunden durch eine (interne) Kante $e_v = (v', v'') \in \bar{E}$.
  - Ersetze jede Kante $e = (u, v) \in E$ durch zwei (externe) Kanten $e' = (u'', v')$ und $e'' = (v'', u') \in \bar{E}$.

Ungerichtete ungewichtete Graphen

- $\kappa(s,t)$ wird nun berechnet als MaxFlow in $\bar{G}$ von Quelle s'' zu Senke t' mit Unit Capacity-Kanten
- Hinweis: $c(e_v) = 1, c(e') = c(e'') = \infty$ führt übrigens zum gleichen Ergebnis.
- Für jedes Paar $v', v'' \in \bar{V}$, das einen internen Knoten $v \in V$ repräsentiert, ist die interne Kante (v', v'') die einzige von v' ausgehende Kante und die einzige eingehende Kante von v''. Der Graph $\bar{G}$ ist also ein UCN vom Typ 2.
- Nach dem Lemma kann die Berechnung des MaxFlow bzw. des lokalen Knotenzusammenhangs in Zeit $\mathcal{O}(\sqrt{n}m)$ erfolgen.

Ungerichtete ungewichtete Graphen

Trivialer Algorithmus zur Bestimmung von $\kappa(G)$:

- Bestimme Minimum aller lokalen Knotenzusammenhangszahlen.
- Für die Endknoten jeder Kante (s, t) in G gilt:

$$\kappa_G(s, t) = n - 1$$

- Anzahl notwendiger MaxFlow-Berechnungen:

$$\frac{n(n-1)}{2} - m$$

Ungerichtete ungewichtete Graphen

Besserer Algorithmus zur Bestimmung von $\kappa(G)$:

- Betrachte minimalen Knoten-Separator $S \subset V$, der eine 'linke' Knotenteilmenge $L \subset V$ von einer 'rechten' Teilmenge $R \subset V$ separiert.
- Man könnte $\kappa(G)$ berechnen, indem man einen Knoten s in einer Teilmenge (L oder R) fixiert und die lokalen Zusammenhangszahlen $\kappa_G(s,t)$ für alle Knoten $t \in V \setminus \{s\}$ berechnet, wobei einer dieser Knoten auf der anderen Seite des Schnitts liegen muss.
- Problem: wie wählt man einen Knoten s, so dass s nicht zu jedem Minimum Vertex Separator gehört?
- Da $\kappa(G) \leq \delta(G)$, könnte man $\delta(G)+1$ Knoten für s versuchen. Einer davon kann nicht Teil aller Minimum Knoten-Separatoren sein.
- Der Algorithmus hat Komplexität

$$\mathcal{O}((\delta+1)\cdot(n-1)\cdot\sqrt{n}m)) = \mathcal{O}(\delta n^{3/2}m)$$

Knotenzusammenhang (Even & Tarjan)

Algorithmus: Knotenzusammenhang κ (Even & Tarjan)

Input: Ungerichteter Graph $G = (V, E)$
Output: $\kappa(G)$
$\kappa_{\min} \leftarrow n - 1$;
$i \leftarrow 1$;
while $i \leq \kappa_{\min}$ **do**
  for $j \leftarrow i + 1$ **to** n **do**
    if $i > \kappa_{\min}$ **then**
      break
    else if $\{v_i, v_j\} \notin E$ **then**
      compute $\kappa_G(v_i, v_j)$ using the MaxFlow algorithm
      $\kappa_{\min} \leftarrow \min\{\kappa_{\min}, \kappa_G(v_i, v_j)\}$
return $\kappa_{\min}$

Knotenzusammenhang (Esfahanian & Hakimi)

Verbesserung von Esfahanian & Hakimi:

Lemma

Wenn ein Knoten v zu allen Knoten-Separatoren minimaler Kardinalität gehört, dann gibt es für jeden Minimum Vertex-Cut S zwei Knoten $l \in L_S$ und $r \in R_S$, die zu v adjazent sind.

Knotenzusammenhang (Esfahanian & Hakimi)

Beweis:

- Annahme: v ist an allen Minimum Vertex-Cutsets beteiligt.
- Betrachte die beiden (getrennten) Teile L und R des Restgraphen, der nach dem Löschen verbleibt.
- Jede der beiden Seiten muss einen Nachbarn von v enthalten, sonst wäre v nicht nötig, um die Teile zu trennen (und die Knotenmenge wäre damit kein minimaler Separator)
- Jede Seite, die mehr als einen Knoten enthält, muss sogar zwei Nachbarn von v enthalten, da man sonst durch Ersetzen von v durch den einzigen Nachbarn einen MinCut ohne v konstruieren könnte (Widerspruch zur Annahme).

□

Algorithmus 11: Knotenzusammenhang κ (Esfahanian & Hakimi)

Input : Ungerichteter Graph $G = (V, E)$
Output : $\kappa(G)$
$\kappa_{\min} \leftarrow n - 1$;
Wähle $v \in V$ mit minimalem Grad, also $d(v) = \delta(G)$;
Seien die Nachbarn $N(v) = \{v_1, v_2, ..., v_\delta\}$;
foreach Nicht-Nachbar $w \in V \setminus (N(v) \cup \{v\})$ **do**
  Berechne $\kappa_G(v, w)$ mit MaxFlow-Prozedur;
  $\kappa_{\min} \leftarrow \min\{\kappa_{\min}, \kappa_G(v, w)\}$;
$i \leftarrow 1$;
while $i \leq \kappa_{\min}$ **do**
  for $j \leftarrow i + 1$ **to** $\delta - 1$ **do**
    if $i \geq \delta - 2$ or $i > \kappa_{\min}$ **then**
      return $\kappa_{\min}$;
    else if $\{v_i, v_j\} \notin E$ **then**
      Berechne $\kappa_G(v_i, v_j)$ mit MaxFlow-Prozedur;
      $\kappa_{\min} \leftarrow \min\{\kappa_{\min}, \kappa_G(v_i, v_j)\}$;
  $i \leq i + 1$;
return $\kappa_{\min}$;

Gesamtkomplexität: $n - \delta - 1 + \kappa(2\delta - \kappa - 3)/2$

Fluss-basierte Kantenzusammenhangsalgorithmen

- Der Kantenzusammenhang λ kann in ungerichteten ungewichteten Graphen ebenfalls mit der MaxFlow-Prozedur gelöst werden.
- Ersetze dafür jede ungerichtete Kante durch zwei antiparallele gerichtete Kanten mit Kapazität 1 und berechne dann den lokalen Zusammenhang zwischen der entsprechenden Quelle s und der Senke t.
- Da das resultierende Netzwerk ein Unit Capacity Network vom Typ 1 ist, ist die Komplexität $\mathcal{O}(m \cdot \min\{m^{1/2}, n^{2/3}\})$.

Fluss-basierte Kantenzusammenhangsalgorithmen

- Ein trivialer Algorithmus könnte einfach mit $n(n-1)/2$ MaxFlow-Aufrufen den lokalen Kantenzusammenhang aller Knotenpaare berechnen.
- Etwas besser wäre es, einen Knoten s festzuhalten und dann für alle anderen Knoten t die lokalen Zusammenhangszahlen $\lambda(s,t)$ zu berechnen.
  Mindestens einer dieser Knoten muss auf der anderen Seite eines MinCuts liegen. Deshalb ist das Minimum aller dieser $(n-1)$ lokalen Zusammenhangszahlen gleich dem (globalen) Kantenzusammenhang λ des Graphen.
  Gesamtkomplexität: $\mathcal{O}(nm \cdot \min\{n^{2/3}, m^{1/2}\})$

Fluss-basierte Kantenzusammenhangsalgorithmen

- Der Algorithmus funktioniert auch, wenn man statt der ganzen Knotenmenge nur eine Teilmenge verwendet, die zwei Knoten s, t enthält, deren lokaler Zusammenhang $\lambda(s,t)$ gleich dem globalen Zusammenhang λ ist.
- Eine solche Teilmenge heißt λ-Covering.

$\Rightarrow$ Versuche, die Kardinalität dieser Knotenmenge zu reduzieren.

Lemma

Sei S ein Minimum Edge-Cut eines Graphen $G = (V, E)$ und sei $L, R \subset V$ eine Partition der Knotenmenge, so dass L and R durch S separiert werden. Wenn $\lambda(G) < \delta(G)$, dann besteht jede Komponente von $G - S$ aus mehr als $\delta(G)$ Knoten, d.h. es gilt $|L| > \delta(G)$ und $|R| > \delta(G)$.

Fluss-basierte Kantenzusammenhangsalgorithmen

Beweis

- Seien $l_1, ..., l_k$ die Elemente von L und sei $E[L] = E(G[L])$ die Menge der durch L induzierten Kanten in G.
- Es gilt:

$$\begin{aligned} \delta(G) \cdot k &\leq \sum_{i=1}^{k} d_G(l_i) \\ &\leq 2 \cdot |E[L]| + |S| \\ &\leq 2 \cdot \frac{k(k-1)}{2} + |S| \\ &< k(k-1) + \delta(G) \end{aligned}$$

- Aus $\delta(G) \cdot (k-1) < k(k-1)$ folgt $|L| = k > 1$ und $|L| = k > \delta(G)$ (sowie $|R| > \delta(G)$).

□

Fluss-basierte Kantenzusammenhangsalgorithmen

Folgerung

Wenn gilt $\lambda(G) < \delta(G)$, dann enthält jede Komponente von $G - S$ einen Knoten, der mit keiner Kante in S inzident ist.

Defintion: Spannbaum

Ein Spannbaum ist ein Teilgraph eines ungerichteten Graphen, der ein Baum ist und alle Knoten dieses Graphen enthält. Spannbäume existieren nur in zusammenhängenden Graphen.

Lemma

Sei $\lambda(G) < \delta(G)$ und sei T ein Spannbaum von G. Dann enthalten alle Komponenten von $G - S$ mindestens einen Knoten, der kein Blatt in T ist. (D.h. die inneren Knoten von T bilden ein λ-Cover.)

Fluss-basierte Kantenzusammenhangsalgorithmen

Beweis:

- Nehmen wir an, dass es nicht so wäre (d.h. alle Knoten in L sind Blätter von T).
- Also für keine Kante von T sind beide Endknoten in L, d.h. $|L| = |S|$.
- Aus der Aussage des vorhergehenden Lemmas (Wenn $\lambda(G) < \delta(G)$, dann besteht jede Komponente von $G - S$ aus mehr als $\delta(G)$ Knoten, d.h. es gilt $|L| > \delta(G)$ und $|R| > \delta(G)$.) folgt sofort, dass $\lambda(G) = |S| = |L| > \delta(G)$
  (Widerspruch zur Annahme).

□

Spannbaum-Berechnung (Esfahanian & Hakimi)

Algorithmus von Esfahanian & Hakimi:

- Berechne Spannbaum des gegebenen Graphen.
- Wähle beliebigen inneren Knoten v des Baums.
- Berechne für jeden anderen Knoten w, der kein Blatt ist, den lokalen Kantenzusammenhang $\lambda(v, w)$.
- Das Minimum dieser Wertemenge, zusammen mit $\delta(G)$, ergibt genau den Kantenzusammenhang $\lambda(G)$.
- Ein Baum mit möglichst vielen Blättern wäre vorteilhaft, aber die Konstruktion eines optimalen Baums ist $\mathcal{NP}$-hart.

Spannbaum-Berechnung (Esfahanian & Hakimi)

Esfahanian & Hakimi:

- Algorithmus zur Berechnung eines Spannbaums T von G, so dass beide Mengen, L und R eines minimalen Kantenseparators mindestens ein Blatt von T enthalten, und nach dem letzten Lemma mindestens einen inneren Knoten.

Spannbaum-Berechnung (Esfahanian & Hakimi)

Algorithmus : Spannbaum-Berechnung (Esfahanian & Hakimi)

Input: Ungerichteter Graph $G = (V, E)$
Output: Spannbaum T mit einem Blatt und einem inneren Knoten in L bzw. R
Wähle $v \in V$;
$T \Leftarrow$ alle Kanten inzident mit v;
while $|E(T)| < n - 1$ **do**
  wähle ein Blatt w in T , so dass für alle Blätter r in T gilt:
  $|N(w) \cap (V - V(T))| \geq |N(r) \cap (V - V(T))|$;
  $T \Leftarrow T \cup \{(w, x) \in E : X \in (V - V(T))\}$
return T;

Kantenzusammenhang λ (Esfahanian & Hakimi)

Algorithmus : Kantenzusammenhang λ (Esfahanian & Hakimi)

Input: Ungerichteter Graph $G = (V, E)$
Output: $\lambda(G)$
Konstruiere einen Spannbaum T (voriger Algorithmus);
Sei P die kleinere der beiden Mengen (entweder die Blätter oder die inneren Knoten von T;
Wähle einen Knoten $u \in P$;
$c \leftarrow \min\{\lambda_G(u, v) : v \in P \setminus \{u\}\}$;
$\lambda \leftarrow \min(\delta(G), c)$;
return λ;

Kantenzusammenhang λ (Esfahanian & Hakimi)

- Da P als kleinere der beiden Mengen (Blätter / innere Knoten) gewählt wird, benötigt der Algorithmus höchstens $n/2$ Berechnungen für lokale Zusammenhangszahlen.

$\Rightarrow$ Gesamtzeitkomplexität: $\mathcal{O}(\lambda mn)$

Kantenzusammenhang λ (Matula)

Defintion

Ein Dominating Set für einen Graphen $G = (V, E)$ ist eine Knotenteilmenge V' von V , so dass jeder Knoten, der nicht in V' ist, mit mindestens einem Knoten in V' über eine Kante verbunden ist.

Lemma

Für den Fall, das gilt $\lambda(G) < \delta(G)$, ist jedes Dominating Set von G auch ein $\lambda - \textit{covering}$ von G.

Kantenzusammenhang λ (Matula)

Verbesserter Algorithmus von Matula:

- Analog zur Spannbaum-Methode, wird λ hier berechnet, indem man
  - ein Dominating Set D von G berechnet,
  - einen beliebigen Knoten $u \in D$ auswählt und
  - den lokalen Kantenzusammenhang $\lambda(u, v)$ für alle anderen Knoten $v \in D \setminus \{u\}$ berechnet.
- Das Minimum der Wertemenge (wieder zusammen mit $\delta(G)$) ist der Kantenzusammenhang.
- Obwohl die Berechnung eines Dominating Sets minimaler Kardinalität $\mathcal{NP}$-hart ist, kann man zeigen, dass der Algorithmus in Zeit $\mathcal{O}(mn)$ läuft, wenn man das Dominating Set wie im folgenden Algorithmus konstruiert.

Kantenzusammenhang λ (Matula)

Algorithmus: Berechnung eines Dominating Sets (Matula)

Input: Ungerichteter Graph $G = (V, E)$
Output: Dominating Set D
Wähle $v \in V$;
$D \leftarrow \{v\}$;
while $V \setminus (D \cup N(D)) \neq \emptyset$ **do**
  Wähle einen Knoten $w \in V \setminus (D \cup N(D))$;

Der Stoer/Wagner-Algorithmus

- 1994 Algorithmus publiziert von M. Stoer und F. Wagner
- benutzt keine MaxFlow-Technik
- sehr einfach
- arbeitet in $n-1$ Phasen
- Phase hat starke Ähnlichkeit zu den Algorithmen von Prim (minimale Spannbäume) bzw. Dijkstra (kürzeste Wege)
- Pro Phase Komplexität $\mathcal{O}(m + n \log n)$
- Gesamtkomplexität $\mathcal{O}(mn + n^2 \log n)$

Der Stoer/Wagner-Algorithmus

Algorithmus: MinCut Berechnung nach Stoer & Wagner

Input: Ungerichteter Graph $G = (V, E)$
Output: Ein MinCut $C_{\min}$ entsprechend $\lambda(G)$
Wähle einen beliebigen Startknoten a;
$C_{\min} \leftarrow$ undefiniert.
while $|V'| > 1$ **do**
  while $A \leftarrow \{a\}$
    Füge zu A den am meisten angebundenen Knoten hinzu;
    Aktualisiere die Kapazitäten zwischen A und den Knoten in $V' \setminus A$;
  C := Schnitt von V', der den zuletzt zu A hinzugefügten Knoten vom Rest des Graphen trennt;
  if $C_{\min}$ = undefiniert or or $w(C) < w(C_{\min})$ **then**
    $C_{\min} \leftarrow C$ Verschmelze die zwei Knoten, die zuletzt zu A hinzugefügt wurden;
return $C_{\min}$;

Der Stoer/Wagner-Algorithmus

- Wahl eines beliebigen Startknotens a
- Algorithmus verwaltet eine Knotenteilmenge A (initialisiert mit a), die immer wieder um einen Knoten erweitert wird, der gerade maximale Anbindung (also Summe der Kantenkapazitäten) an die aktuellen Knoten in A hat.
- Nachdem alle Knoten in A eingefügt wurden, nennt man den Cut, der nur den zuletzt hinzugefügten Knoten t vom Rest abtrennt 'Cut der Phase'.
- Verschmelzung der beiden zuletzt behandelten Knoten s und t:
  - Wenn zwischen s und t eine Kante existiert, wird sie einfach gelöscht.
  - Alte Kanten von einem anderen Knoten $\{x, s\}$ und $\{x, t\}$ werden durch eine neue Kante mit der Summe der alten Gewichte $w(s, x) + w(t, x)$ ersetzt.

Der Stoer/Wagner-Algorithmus

Lemma

Der 'Cut der Phase' ist ein Minimum (s, t)-Cut für die beiden zuletzt in A eingefügten Knoten s und t.

Lemma

Ein 'Cut der Phase' mit minimaler Kantenkapazität ist ein MinCut des gegebenen Graphen.

Der randomisierte MinCut-Algorithmus von Karger

- Gesucht: globaler Minimum Cut in einem Multi-Graphen
- D. Karger (1992): Vorschlag eines sehr einfachen Algorithmus ohne augmentierende Pfade und Flussberechnungen
- Ansatz: randomisierte Suche (Monte-Carlo-Algorithmus), die mit gewisser (Erfolgs-)Wahrscheinlichkeit den Minimum Cut liefert (und mit gewisser (Fehler-)Wahrscheinlichkeit einen beliebigen anderen Cut)

Der randomisierte MinCut-Algorithmus von Karger

- Der Algorithmus wählt eine beliebige Kante $e = \{u, v\}$ von G, und zwar uniform identisch verteilt (d.h. jede Kante wird mit gleicher Wahrscheinlichkeit gezogen).
- Die gewählte Kante wird kontrahiert, d.h. es wird ein Multi-Graph G' erzeugt, bei dem die Knoten u und v zu einem neuen Knoten w verschmolzen sind.
- Kanten zwischen u und v verschwinden.
- Andere Kanten bleiben erhalten. Wenn genau ein Endknoten entweder u oder v ist, dann endet dafür eine neue Kante am (Super-)Knoten w.
- Hinweis: auch wenn G keine Multikanten enthält, kann G' welche enthalten.
- Das Verfahren wird rekursiv auf G' angewendet.

Der randomisierte MinCut-Algorithmus von Karger

- Der Algorithmus endet, wenn nur noch zwei (Super-)Knoten vorhanden sind.
- Jeder der beiden Knoten enthält eine gewisse Menge der Knoten des ursprünglichen Graphen.
- Diese Partition definiert einen Cut, der vom Algorithmus ausgegeben wird.

Lemma

Der Monte-Carlo-Algorithmus von Karger gibt mit Wahrscheinlichkeit $\geq 1/\binom{n}{2}$ den globalen Minimum Cut des Multigraphen zurück.

- Nachdem es exponentiell viele Cuts in G gibt, würde man vermuten, dass die Wahrscheinlichkeit, den globalen MinCut zu finden, exponentiell klein ist.
- Was favorisiert die kleinen Cuts?

Der randomisierte MinCut-Algorithmus von Karger

- Wahrscheinlichkeit, dass der randomisierte MinCut-Algorithmus nicht den Cut (A, B) ausgibt, ist höchstens $1 - 1/\binom{n}{2}$
- Nach $\binom{n}{2}$ Läufen mit unabhängigen Zufallsentscheidungen ist die Wahrscheinlichkeit, dass nie der MinCut gefunden wurde, höchstens

$$\left(1 - 1/\binom{n}{2}\right)^{\binom{n}{2}} \leq \frac{1}{e}$$

- Wenn $c\binom{n}{2} \log n$ Läufe gestartet werden, sinkt die Fehlerwahrscheinlichkeit auf $\leq e^{-c \log n} = 1/n^c$.

Bi-Zusammenhangskomponenten

Defintion: Zweifachzusammenhangskomponenten

Die Zweifachzusammenhangskomponenten oder Blöcke eines Graphen sind die maximalen Teilgraphen, die 2-fach zusammenhängend sind.

Bi-Zusammenhangskomponenten

Modifizierte DFS nach R. E. Tarjan:

- num[v]: DFS-Nummer von v
- low[v]: minimale Nummer num[w] eines Knotens w, der von v aus über beliebig viele (≥ 0) Baumkanten abwärts, evt. gefolgt von einer einzigen Rückwärtskante erreicht werden kann
- low[v]: Minimum von
  - num[v]
  - low[w], wobei w ein Kind von v im DFS-Baum ist
  - num[w], wobei $\{v, w\}$ eine Rückwärtskante ist

Bi-Zusammenhangskomponenten

Lemma

Sei $G = (V, E)$ ein ungerichteter, zusammenhängender Graph und T ein DFS-Baum in G. Ein Knoten $a \in V$ ist genau dann ein Artikulationsknoten, wenn

- a die Wurzel von T ist und mindestens 2 Kinder hat, oder
- a nicht die Wurzel von T ist und es ein Kind b von a mit $\text{low}[b] \geq \text{num}[a]$ gibt.

Bi-Zusammenhangskomponenten

Die Kanten werden auf einem Stack gesammelt und nach der Erkennung eines Artikulationsknotens wird der gesamte Block abgepflükt.

Starke Zusammenhangskomponenten

Definition: Starke Zusammenhangskomponente

Eine Menge $W \subseteq V$ bestimmt eine starke Zusammenhangskomponente, falls zu je zwei Knoten u und $v \in W$ sowohl ein (u, v)- als auch ein (v, u)-Weg in $A(W)$ existieren und W inklusionsmaximal ist.

Starke Zusammenhangskomponenten

Modifizierte DFS nach R. E. Tarjan:

- num[v]: DFS-Nummer von v
- low[v]: minimale Nummer num[w] eines Knotens w, der von v aus über beliebig viele (≥ 0) Baumkanten abwärts, evt. gefolgt von einer einzigen Rückwärtskante oder einer Querkante zu einer Zusammenhangskomponente, deren Wurzel echter Vorfahre von v ist, erreicht werden kann
- low[v]: Minimum von
  - num[v]
  - low[w], wobei w ein Kind von v im DFS-Baum ist
  - num[w], wobei $\{v, w\}$ eine Rückwärtskante ist
  - num[w], wobei $\{v, w\}$ eine Querkante ist und die Wurzel der starken Zusammenhangskomponente von w ist Vorfahre von v
- Ein Knoten v ist genau dann Wurzel einer starken Zusammenhangskomponente, wenn num[v] = low[v].

Literatur

- Frank Kammer and Hanjo Täubig: Network Analysis, Kapitel 7, Connectivity, Springer (2005)
- Dr. Hanjo Täubig: Vorlesungsfolien der Fortgeschrittene Netzwerk- und Graph-Algorithmen. http://www14.informatik.tu-muenchen.de/lehre/2007WS/anga/index.html.de
- Prof. Gerhard Reinelt: Vorlesungsskript der Effiziente Algorithmen 1