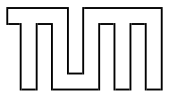
Fortgeschrittene Netzwerk- und Graph-Algorithmen

Prof. Dr. Hanjo Täubig

Lehrstuhl für Effiziente Algorithmen
(Prof. Dr. Ernst W. Mayr)
Institut für Informatik
Technische Universität München

Wintersemester 2010/11



Länge von Slicings

- Länge eines Slicings Z von Graph G kann 1 sein, falls G bipartit ist.
- Allgemein:
  Minimaler Wert für die Länge eines Slicings eines Graphen hängt von der minimalen Zahl k ab, so dass G ein k-partiter Graph ist, also von der sogenannten chromatischen Zahl $\chi(G)$ des Graphen G.

Definition

Ein Graph ist k-partit, wenn die Knotenmenge $V(G)$ in k Mengen $V_1, \ldots, V_k$ partitioniert werden kann, so dass jeder induzierte Graph $G[V_i]$ $(i \in \{1, \ldots, k\})$ keine Kante enthält.

Länge von Slicings

Satz

Für jeden Graph G mit $|E(G)| \geq 1$ gilt:

$$\min\{\ell(Z) : Z \text{ ist Slicing von } G\} = \lceil \log_2 \chi(G) \rceil\}$$

Beweis.

- Sei $Z = (C_1, C_2, \ldots, C_m)$ ein Slicing von G mit $C_i = (A_i, \bar{A}_i)_i$ und $A_i \cup \bar{A}_i = V$.
- Für $v \in V$ und $i \in \{1, \ldots, m\}$ sei $b_i(v) = \begin{cases} 1 & \text{falls } v \in A_i \\ 0 & \text{falls } v \notin A_i \end{cases}$
- Partitioniere Knotenmenge V in die 2^m Teilmengen $V_0, \ldots, V_{2^m-1}$, indem $v \in V$ der Teilmenge V_k mit $k = \sum_{i=1}^{m} b_i(v) \cdot 2^{i-1}$ zugeteilt wird.

Länge von Slicings

Beweis.

- Falls $\{u, w\} \in E$, dann ist $\{u, w\} \in C_i$ für ein i, so dass also Knoten u und w durch Cut C_i getrennt werden.

$\Rightarrow$ $b_i(u) \neq b_i(w)$

$\Rightarrow$ u und w sind in verschiedenen Teilmengen V_k

$\Rightarrow$ Die nichtleeren unter den Teilmengen $V_0, \ldots, V_{2^m-1}$ bilden eine Partition von V in unabhängige Mengen (independent sets), also $2^m \geq \chi(G)$ bzw. $\ell(Z) = m \geq \lceil \log_2 \chi(G) \rceil$.

- Damit wurde erstmal eine untere Schranke gezeigt.
- Als nächstes wird gezeigt, dass der Wert $\lceil \log_2 \chi(G) \rceil$ auch tatsächlich durch ein Slicing Z^* erreicht werden kann.

Länge von Slicings

Beweis.

- Sei $V_0, \ldots, V_{\chi(G)-1}$ eine Partition von V in $\chi(G)$ unabhängige Mengen (independent sets) und sei $m = \lceil \log_2 \chi(G) \rceil$.
- Sei A_i die Vereinigung derjenigen V_k, deren Binärdarstellung den Term 2^{i-1} enthält, also

$$A_i = \bigcup \{ V_k : \lfloor k/2^{i-1} \rfloor \equiv 1 \bmod 2 \} \quad \text{für } i = \{1, \ldots, m\}$$

  und $\bar{A}_i = V \setminus A_i$
- Sei wie zuvor $C_1 = (A_1, \bar{A}_1)$ und $C_i = (A_i, \bar{A}_i)_i = (A_i, \bar{A}_i) - \bigcup_{j=1}^{i-1} C_j$ für $i = 2, \ldots, m$.
- Für jedes $i = 1, 2, \ldots, m$ muss es eine Kante $\{u, w\} \in E$ mit $u \in V_0$, $w \in V_{2^{i-1}}$ geben, da sonst $V_0 \cup V_{2^{i-1}}$ eine unabhängige Menge ist (Widerspruch zur chromatischen Zahl $\chi(G)$)

Länge von Slicings

Beweis.

- Also gilt: $u \in \bar{A}_j$ für $j = 1, \ldots, m$,
  sowie $w \in \bar{A}_j$ für $j = 1, \ldots, i-1$ und $w \in A_i$

$\Rightarrow \{u, w\} \in (A_i, \bar{A}_i)_i = C_i$

$\Rightarrow$ kein C_i ist leer

- Für $\{u, w\} \in E$ gilt außerdem $u \in V_k$ und $w \in V_j$ für ein Paar $k \neq j$.
- Sei i ein Index, so dass die Binärdarstellungen von k und j sich im Term 2^{i-1} unterscheiden.

$\Rightarrow u \in A_i$ und $w \in \bar{A}_i$ oder $u \in \bar{A}_i$ und $w \in A_i$

$\Rightarrow \{u, w\} \in (A_i, \bar{A}_i) = \bigcup_{n=1}^{i} C_n$

$\Rightarrow E \subseteq \bigcup_{i=1}^{m} C_i$ und $Z^* = (C_1, C_2, \ldots, C_m)$ ist Slicing von G mit $\ell(Z^*) = m = \lceil \log_2 \chi(G) \rceil$

Weite von Slicings

Definition

Die Weite eines Slicings Z des Graphen G sei

$$w(Z) = \max\{|C| : C \text{ ist ein Cut des Slicings } Z\}$$

Jeder Cut C von Z mit $|C| = w(Z)$ heißt Wide Cut des Slicings Z.

- Ein MinCut bezieht sich auf den Graphen, während ein Wide Cut sich auf ein bestimmtes Slicing bezieht.

Weite von Slicings

- Da jeder Cut $C_i = (A_i, \bar{A}_i)_i$ eines Slicings Z von G in einem Cut $(A_i, \bar{A}_i)$ von G enthalten ist, und da jedes Slicing mit einem beliebigem Cut von G beginnen kann, muss die maximale Weite eines Cuts gleich der maximalen Anzahl von Kanten in einem beliebigen Cut sein.

Satz

Für jeden Graphen mit $|E(G)| \geq 1$ *gilt:*

$$\max\{w(Z) : \ Z \text{ ist Slicing von } G\} \ = \ \max\{|C| : \ C \text{ ist Cut von } G\}$$

Weite von Slicings

- Die minimale Weite eines Slicings ist ähnlich verwandt zu MinCuts, aber nicht vom ganzen Graphen G.

Satz

Für jeden Graphen mit $|E(G)| \geq 1$ gilt:

$$\min\{w(Z) : Z \text{ ist Slicing von } G\} = \sigma(G)$$

$\Rightarrow$ Dualität zwischen Slicings und Subgraphen:

$$\min_{Z} \max_{C}\{|C| : C \text{ ist Cut des Slicings } Z \text{ von } G\} =$$
$$\max_{G'} \min_{C}\{|C| : C \text{ ist Cut des Teilgraphen } G' \text{ von } G\}$$

Weite von Slicings

Beweis.

(Ungleichung in beiden Richtungen)

$\min\{w(Z) : Z \text{ ist Slicing von } G\} \quad \geq$
$\max\{\lambda(G') : G' \text{ ist Teilgraph von } G\} = \sigma(G)$:

- Sei K ein Cluster von G mit $\lambda(K) = \sigma(K)$.
- Dann gilt für jedes Slicing Z von G, dass es einen ersten Cut C_i gibt, der Knoten von K separiert.
- Dann muss C_i einen Cut für K enthalten, so dass $|C_i| \geq \lambda(K) = \sigma(G)$.
- Also gilt $w(Z) \geq \sigma(Z)$ für jedes Slicing Z.

Weite von Slicings

Beweis.

$\min\{w(Z) : Z \text{ ist Slicing von } G\} \quad \leq$
$\max\{\lambda(G') : G' \text{ ist Teilgraph von } G\} = \sigma(G)$:

- Sei $Z^* = (C_1, \ldots, C_m)$ ein Narrow Slicing von G mit Wide Cut C_i.
- Sei G_i der Teilgraph von G, der durch C_i zertrennt wird.
- Da ein Narrow Slicing nur Minimum Cuts benutzt, folgt $\lambda(G_i) = w(Z^*)$.

Weite von Slicings

- Die Weite eines Slicings muss größer oder gleich der durchschnittlichen Anzahl der Kanten in den Cuts des Slicings sein.
- Aus dem letzten Satz und dem Satz über die maximale Länge eines Slicings folgt damit die folgende untere Schranke für die Stärke $\sigma(G)$:

Folgerung

Für jeden Graphen mit $|E(G)| \geq 1$ *gilt:*

$$\sigma(G) \geq \frac{|E(G)|}{|V(G)|-1} > |E(G)|/|V(G)|$$

Narrow Slicings

- Im Beweis des letzten Satzes war jedes Narrow Slicing ausreichend, um die Ungleichung der zweiten Richtung zu beweisen. Deshalb gelten folgende Korollare:

Folgerung

Für jedes Narrow Slicing Z^* von G gilt: $w(Z^*) = \sigma(G)$.

Folgerung

Jeder Wide Cut eines Narrow Slicings von G ist ein Minimum Cut eines Subclusters von G.

Narrow Slicings und k-Komponenten

Die nächsten zwei Folgerungen zeigen, dass man ein Narrow Slicing benutzen kann, um die k-Komponenten ($k \geq 1$) und die Zusammenhangsfunktion zu berechnen.

Folgerung

Seien $G[A_1]$, $G[A_2]$,...,$G[A_m]$ die Teilgraphen, die durch das Narrow Slicing $Z = (C_1, C_2, \ldots, C_m)$ getrennt werden.
Dann ist jede k-Komponente von G gleich einem $G[A_i]$ für ein $i \in \{1, \ldots, m\}$.
Weiterhin ist ein $G[A_i]$ ($i \in \{1, \ldots, m\}$) genau dann eine k-Komponente von G, wenn

$$A_j \supset A_i \text{ für } j < i \Rightarrow |C_j| < |C_i|.$$

Und solch ein $G[A_i]$ ist genau dann ein Cluster von G, wenn auch gilt

$$A_j \subset A_i \text{ für } j > i \Rightarrow |C_j| \leq |C_i|.$$

Narrow Slicings und k-Komponenten

Beweis.

- Seien $G[A_1], \ldots, G[A_m]$ die Teilgraphen, die durch das Narrow Slicing $Z = (C_1, C_2, \ldots, C_m)$ getrennt werden, so dass $\lambda(G[A_i]) = |C_i|$ $(i \in \{1, \ldots, m\})$.
- Sei H eine k-Komponente von G für ein $k \in \{1, \ldots, \sigma(G)\}$.
- Dann muss der erste Cut C_n aus Z, der Knoten aus H separiert, einen Cut von H enthalten, also $|C_n| = \lambda(G[A_n]) \geq \lambda(H)$.
- Da H zusammenhängend ist, muss H ein Teilgraph von $G[A_n]$ sein, weil C_n der erste Cut ist, der Knoten aus H separiert.

$\Rightarrow$ $G[A_n]$ ist ein $\lambda(H)$-kanten-zusammenhängender Graph $(\lambda(H) \leq \lambda(G[A_n]))$.

- Da H ein maximaler $\lambda(H)$-kanten-zusammenhängender Graph (per Def.) sowie ein Teilgraph von $G[A_n]$ ist, gilt $H = G[A_n]$.

Narrow Slicings und k-Komponenten

Beweis.

- $G[A_i]$ $(i \in \{1, \ldots, m\})$ ist in einer $\lambda(G[A_i])$-Komponente H von G enthalten, und aufgrund des vorangegangenen Arguments gilt $H = G[A_j]$ für ein $j \leq i$.
- Also ist $G[A_i]$ selbst eine $\lambda(G[A_i])$-Komponente von G genau dann, wenn $A_j \supset A_i$ für $j < i$ impliziert, dass $|C_j| < |C_i|$.
- Falls weiterhin $G[A_i]$ eine $\lambda(G[A_i])$-Komponente ist, dann handelt es sich genau dann um ein Cluster, wenn $A_j \subset A_i$ für $j > i$ impliziert, dass $|C_j| \leq |C_i|$.

$\Rightarrow$ Die Teilgraphen, die durch ein beliebiges Narrow Slicing von G zertrennt werden, enthalten alle k-Komponenten von G, und damit alle Cluster von G, die keine isolierten Knoten sind.

Narrow Slicings und Kohäsionsfunktion

Folgerung

Seien $G[A_1]$, $G[A_2]$,..., $G[A_m]$ die Teilgraphen, die durch das Narrow Slicing $Z = (C_1, C_2, \ldots, C_m)$ getrennt werden.
Dann gilt für jedes Graphelement $x \in V(G) \cup E(G)$:

$$h(x) = \begin{cases} 0, \text{ falls } x \text{ ein isolierter Knoten in } G \text{ ist,} \\ \max_i\{|C_i| : x \in A_i \cup E(G[A_i])\}, \text{ sonst} \end{cases}$$

- Wenn $G[A_1], \ldots, G[A_m]$ die Subgraphen sind, die durch ein Narrow Slicing $Z = (C_1, \ldots, C_m)$ von G zertrennt werden, dann müssen nicht alle m Subgraphen $G[A_i]$ k-Komponenten sein.
- Insbesondere, wenn $G[A_i]$ ein Cluster mit $\lambda(G[A_i]) \geq 2$ ist, dann sind mindestens $\lambda(G[A_i]) - 1$ der Teilgraphen $G[A_j]$ mit $j > i$ echte Teilgraphen von $G[A_i]$ und können demzufolge keine k-Komponenten von G für ein $k \geq 1$ sein.

Berechnung der Zusammenhangsfunktion

- Der Algorithmus basiert auf der sequentiellen Bestimmung von globalen Minimum Cuts für höchstens $|V(G)| - 1$ induzierte Teilgraphen von G.

Narrow Slicing Algorithmus

Algorithmus 19 : Berechnung eines Narrow Slicings (Matula)

Input : Graph $G = (V, E)$ mit N Komponenten und mindestens einer Kante
Output : Ein Narrow Slicing Z

Repräsentiere G als Vereinigung seiner Komponenten $G = \bigcup_{j=1}^{N} G[P_{1,j}]$;
$i \leftarrow 1$;
while $i < |V(G)| - N$ **do**
  Wähle $G_i = G[P_{i,k}]$ für ein k, so dass $|P_{i,k}| \geq 2$;
  Finde einen MinCut $C_i = (A, \bar{A})$ von G_i. (Beachte $A \cup \bar{A} = V(G_i)$);
  Definiere neue Komponenten-Knoten-Partition $\{P_{i+1,j}\}$ $(j \in \{1, \ldots, N+i\})$:

$$P_{i+1,j} = \begin{cases} P_{i,j} & \text{für } 1 \leq j < k, \\ A & \text{für } j = k, \\ \bar{A} & \text{für } j = k+1, \\ P_{i,j-1} & \text{für } k+1 < j \leq N+i, \end{cases} \quad \text{mit } G - \bigcup_{n=1}^{i} C_n = \bigcup_{j=1}^{N+i} G[P_{i+1,j}];$$

  $i \leftarrow i + 1$;

$G - \bigcup_{n=1}^{|V(G)|-N} C_n$ besteht jetzt nur noch aus isolierten Knoten und
$Z = (C_1, \ldots, C_{|V(G)|-N})$ ist ein Narrow Slicing;
return Z;

Narrow Slicing Algorithmus

- Ursprünglich wurde von Matula vorgeschlagen, den MinCut mit Hilfe des Cut Trees von Gomory/Hu zu berechnen.

  Hier kann man aber auch andere Algorithmen einsetzen, z.B. den Stoer/Wagner-Algorithmus.

- Der Narrow Slicing Algorithmus bestimmt explizit ein Narrow Slicing $Z = (C_1, \ldots, C_{|V(G)|-N})$, eine Liste $G_1, \ldots, G_{|V(G)|-N}$ von induzierten Graphen, die durch das Slicing zertrennt werden, sowie die durch Z verursachte geschachtelte Sequenz von Komponenten-Knoten-Partitionen.

- Daraus können dann entsprechend den früheren Sätzen sehr einfach die Zusammenhangsfunktion, die k-Komponenten und die Cluster von G bestimmt werden.

Narrow Slicing Algorithmus

- $G_i = G[P_{i,k}]$ hat höchstens $|V(G)| + 2 - N - i$ Knoten.
- Also müssten bei Verwendung der Cut Tree Methode höchstens

$$\sum_{i=1}^{|V(G)|-N} (|V(G)| + 1 - N - i) = \frac{1}{2}(|V(G)| - N)(|V(G)| - N + 1)$$

  Flussprobleme einfacher Art gelöst werden.
- Für einen zusammenhängenden Graphen wären das höchstens $\binom{|V(G)|}{2}$ Flussprobleme.
- Matula gibt die Komplexität grob mit n^4 bis n^5 an.
- Das entspricht auch ungefähr der Größenordnung, die man bei Verwendung des Stoer/Wagner-Algorithmusses errechnen würde ($\mathcal{O}(m_i n_i + n_i^2 \log n_i)$ pro Teilproblem mit n_i Knoten und m_i Kanten, summiert für $n_i = n - 1, \ldots, 1$)