Implementierung der Cut & Count-Technik am Beispiel Steiner tree

Levin von Hollen, Tilman Beck

{stu127560-, stu127568-}@informatik.uni-kiel.de

Christian-Albrechts Universität Kiel

3. November 2016

Überblick

Cut & Count-Technik

- Technik um connectivity-type Probleme mithilfe von Randomisierung zu lösen(Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michał Pilipczuk, Johan van Rooij, Jakub Onufry Wojtaszczyk)
- angewendet auf viele verschiedene Probleme (z.B. Longest Path, Steiner Tree, Feedback Vertex Set, uvm.)
- Randomisierung durch Isolation-Lemma
- als Ergebnis ein einseitiger Monte-Carlo-Algorithmus mit Laufzeit $c^{tw(G)}|V|^{\mathcal{O}(1)}$

Cut & Count-Technik

Theorem

There exist Monte-Carlo algorithms that given a tree decomposition of the (underlying undirected graph of the) input graph of width t solve the following problems:

- Steiner Tree in $3^t|V|^{\mathcal{O}(1)}$
- Feedback Vertex Set in $3^t|V|^{\mathcal{O}(1)}$
- ...

The algorithms cannot give false positives and may give false negatives with probability at most $1/2$.

Steiner Tree

Problem

Input: An undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$ and an integer k.
Question: Is there a set $X \subseteq V$ of cardinality k such that $T \subseteq X$ and $G[X]$ is connected?

Cut (1)

- definiere Gewichtsfunktion $\omega : V \to \{1, \dots, N\}$
- sei $\mathcal{R}_W$ die Menge aller Teilmengen von X aus V mit $T \subseteq X$, $\omega(X) = W$ und $|X| = k$
- sei $\mathcal{S}_W = \{X \in \mathcal{R}_W | G[X] \text{ ist zusammenhängend}\}$
- $\cup_W \mathcal{S}_W$ ist die Menge der Lösungen
- gibt es ein W für das die Menge nichtleer ist, so gibt der Algorithmus eine positive Antwort

Cut (2) - wozu ω ?

- Gewichtsfunktion $\omega : V \to \{1, \ldots, N\}$
- der Algorithmus (ohne ω) ist korrekt, sofern es **genau eine** oder **keine** Lösung gibt
- mithilfe des Isolation Lemma kann mit hoher Wahrscheinlichkeit eine große Lösungsmenge auf eine einzige reduziert werden

Isolation Lemma

Let $\mathcal{F} \subseteq 2^U$ be a set family over a universe U with $|\mathcal{F}| > 0$. For each $u \in U$, choose a weight $\omega(u) \in 1, 2, ..., N$ uniformly and independently at random. Then

$$prob[\omega \text{ isolates } \mathcal{F}] \geq 1 - \frac{|U|}{N}$$

Cut (3)

- einen beliebigen Terminalknoten $v \in T$ als v_1 festlegen
- sei $\mathcal{C}_W$ die Menge aller Subgraphen, die einen konsistenten Cut $(X, (X_1, X_2))$ bilden, wobei $X \in \mathcal{R}_W$ und $v_1 \in X_1$

Lemma 3.3

Let $G = (V, E)$ be a graph and let X be a subset of vertices such that $v_1 \in X \subseteq V$. The number of consistently cut subgraphs $(X, (X_1, X_2))$ such that $v_1 \in X_1$ is equal to $2^{cc(G[X])-1}$.

Count (1)

- aus Lemma 3.3 ist bekannt: $|\mathcal{C}| = \sum_{X \in \mathcal{R}} 2^{cc(G[X])-1}$
- wir legen W fest und ignorieren die Indices:
  $|\mathcal{C}| \equiv |\{X \in \mathcal{R} | cc(G[X]) = 1\}| = |\mathcal{S}|$

Lemma 3.4

Let G, ω, $\mathcal{C}_W$ and $\mathcal{S}_W$ be as defined above. Then for every W, $|\mathcal{S}_W| \equiv |\mathcal{C}_W|$.

Einschub: Tree Decomposition (1)

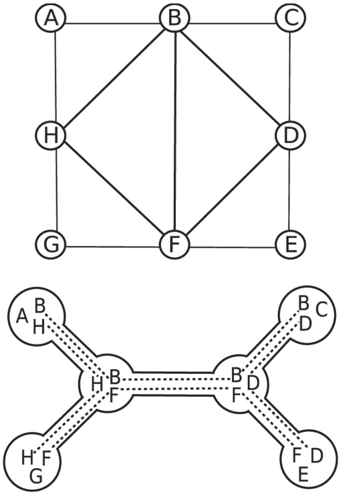
Tree Decomposition

A tree decomposition of a graph G is a tree $\mathbb{T}$ in which each vertex $x \in \mathbb{T}$ has an assigned set of vertices $B_x \subseteq V$ (called a bag) such that $\cup_{x \in \mathbb{T}} B_x = V$ with the following properties:

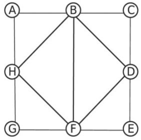
- for any $uv \in E$, there exists an $x \in \mathbb{T}$ such that $u, v \in B_x$
- if $v \in B_x$ and $v \in B_y$, then $v \in B_z$ for all z on the path from x to y in $\mathbb{T}$

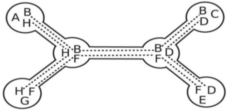
Kann in polynomieller Zeit berechnet werden (Beweis: Ton Kloks. Treewidth, Computations and Approximations, volume 842 of Lecture Notes in Computer Science. Springer, 1994)

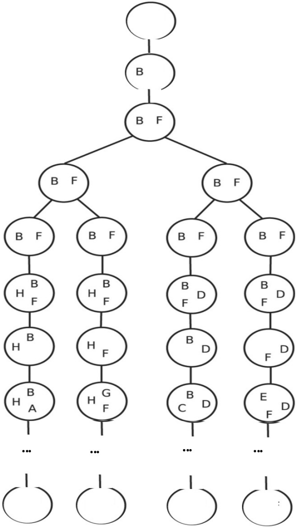
Einschub: Tree Decomposition (2)



Einschub: Nice Tree Decomposition (NTD)







Count (2)

- $|\mathcal{C}_W|$ modulo 2 kann mit dynamischen Programm berechnet werden
- für jeden Bag $x \in \mathbb{T}$, integers $0 \leq i \leq k, 0 \leq w \leq kN$ und Färbung $s \in \{0, 1_1, 1_2\}^{B_x}$ definiere:
  - $\mathcal{R}_x(i, w) = \{X \subseteq V_x | (T \cap V_x) \subseteq X \wedge |X| = i \wedge \omega(X) = w\}$
  - $\mathcal{C}(i, w) = \{(X, (X_1, X_2)) | X \in \mathcal{R}_x(i, w) \wedge (X, (X_1, X_2))$ is a consistently cut subgraph of $G_x \wedge (v_1 \in V_x \Rightarrow v_1 \in X_1\}$
  - $\mathcal{A}_x(i, w, s) = |\{(X, (X_1, X_2)) \in \mathcal{C}_x(i, w) | (s(v) = 1_j \Rightarrow v \in X_j) \wedge (s(v) = 0 \Rightarrow v \notin X\}|$

Count (3)

Färbung $s \in \{0, 1_1, 1_2\}^{B_x}$

- $s[v] = 0 \Rightarrow v \notin X$
- $s[v] = 1_1 \Rightarrow v \in X_1$
- $s[v] = 1_2 \Rightarrow v \in X_2$
- $\mathcal{A}_x(i, w, s)$ zählt die Elemente $\mathcal{C}_x(i, w)$, die entsprechend gefärbt sind
- $\sum\limits_{s \in \{0, 1_1, 1_2\}^{B_x}} A_x(i, w, s) = |C_x(i, w)|$
- für die Lösung nur $A_r(k, W, \emptyset) = |C_r(k, W)| = |C_W|$ interessant (für alle W)
- ausgehend von Wurzel-Knoten rekursiver Abstieg zu den Blatt-Knoten der NTD

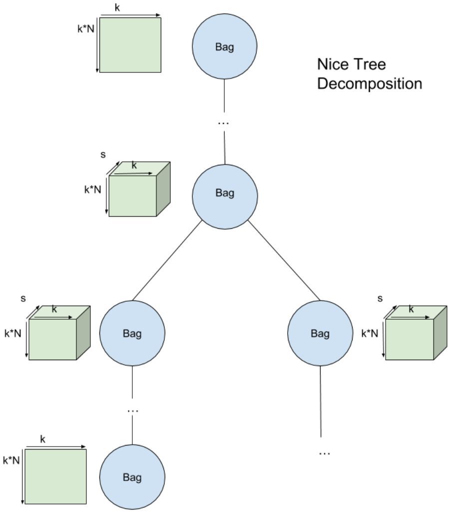
Count (3)

Theorem (Lemma 3.5)

Given $G = (V, E)$, $T \subseteq V$, an integer $k, \omega : V \to \{1, \ldots, N\}$ and a nice tree decomposition $\mathbb{T}$, there exists an algorithm that can determine $|\mathcal{C}_W|$ modulo 2 for every $0 \leq W \leq kN$ in $3^t N^2 |V|^{\mathcal{O}(1)}$ (t is the treewidth of the NTD)

Implementierung

- Inorder-Traversierung der NTD
- Berechnung einer Datenmatrix basierend auf dynamischen Programm und Kind-Knoten
- Wurzel- und alle Blattknoten enthalten 2D Datenmatrix



dynamisches Programm

Berechnungen

- **Leaf bag**:
  - $A_x = (0, 0, \emptyset) = 1$
- **Introduce vertex v**:
  - $A_x = (i, w, s[v \to 0]) = [v \notin T] A_y(i, w, s)$
  - $A_x = (i, w, s[v \to 1_1]) = A_y(i-1, w - w(v), s)$
  - $A_x = (i, w, s[v \to 1_2]) = [v \neq v_1] A_y(i-1, w - w(v), s)$
- **Introduce edge uv**
  - $A_x(i, w, s) = [s(u) = 0 \vee s(v) = 0 \vee s(u) = s(v)] A_y(i, w, s)$
- **Forget vertex v**
  - $A_x(i, w, s) = \sum_{\alpha \in 0, \mathbf{1_1}, \mathbf{1_2}} A_x(i, w, s[v \to \alpha])$
- **Join bag**
  - $A_x(i, w, s) = \sum_{i_1 + i_2 = i + |s^{-1}(1_1, 1_2)|} \sum_{w_1 + w_2 = w + w(s^{-1}(1_1, 1_2))} A_y(i_1, w_1, s) A_z(i_2, w_2, s)$

Multiple Columns

Heading

1. Statement
2. Explanation
3. Example

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Table

Treatments	Response 1	Response 2
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Tabelle: Table caption

Theorem

Theorem (Mass–energy equivalence)

$E = mc^2$

Verbatim

Example (Theorem Slide Code)

```
\begin{frame}
\frametitle{Theorem}
\begin{theorem}[Mass--energy equivalence]
$E = mc^2$
\end{theorem}
\end{frame}
```

Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.

Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References

John Smith (2012)
Title of the publication
Journal Name 12(3), 45 – 678.

Fragen?