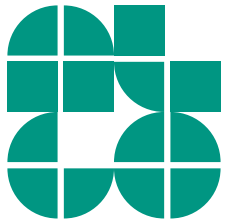
Algorithmen zur Visualisierung von Graphen

Übung 7: Kreuzungen in Lagenlayouts & Metro Maps

INSTITUT FÜR THEORETISCHE INFORMATIK · FAKULTÄT FÜR INFORMATIK

Tamara Mchedlidze · **Martin Nöllenburg**

05.02.2014



Aufgabe 1 – Fehlstände Zählen

(a) Sei $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ eine Permutation. Ein Paar (i, j) mit $1 \leq i < j \leq n$ heißt Inversion, wenn $\pi(i) > \pi(j)$. Entwerfen Sie einen Algorithmus, der die Anzahl der Inversionen einer Permutation von n Elementen in $O(n \log n)$ Zeit berechnet.
Hinweis: Denken Sie an Sortieralgorithmen wie Mergesort.

Aufgabe 1 – Fehlstände Zählen

(a) Sei $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ eine Permutation. Ein Paar (i, j) mit $1 \leq i < j \leq n$ heißt Inversion, wenn $\pi(i) > \pi(j)$. Entwerfen Sie einen Algorithmus, der die Anzahl der Inversionen einer Permutation von n Elementen in $O(n \log n)$ Zeit berechnet.
Hinweis: Denken Sie an Sortieralgorithmen wie Mergesort.

(b) Gegeben sei ein einfacher, bipartiter Graph $G = (V, E)$, dessen Knoten gemäß der Bipartition auf zwei parallele Geraden verteilt sind. Die Knoten seien disjunkt und die Kanten geradlinig gezeichnet. Entwerfen Sie einen Algorithmus, der die Anzahl der Kreuzungen in $O(|E| \log |V|)$ Zeit bestimmt. Begründen Sie, warum es nicht möglich ist, in dieser worst-case-Laufzeit alle Kreuzungen (Paare betroffener Kanten) *auszugeben*.

Aufgabe 1 – Lösungsskizze

- modifiziere divide & conquer mergesort-Algorithmus
- zähle beim Sortieren die übersprungenen Elemente mit – genau das sind die Inversionen

- zwei Kanten (u, v) und (w, z) schneiden sich gdw. $u < w$ und $v > z$
- bezeichne Knoten auf Lage 1 als $a, b, c, \ldots$ und Knoten auf Lage 2 als $1, 2, 3, \ldots$
- bilde Folge der Kanten aus Sicht von Lage 2, d.h. $a1, c1, d1, a2, b2, b3, d3, e3, \ldots$
- wende Algorithmus aus (a) an und sortiere lexikographisch
- leicht zu sehen, dass es Graphen mit $O(|E|^2)$ Kreuzungen gibt

Aufgabe 2 – Baryzenter Heuristik

Karlsruhe Institute of Technology

Zeigen Sie, dass die Baryzenter-Heuristik zur einseitigen Kreuzungsreduktion die optimale Lösung liefert, falls diese keine Kreuzung enthält.

Aufgabe 2 – Baryzenter Heuristik

Karlsruhe Institute of Technology

Zeigen Sie, dass die Baryzenter-Heuristik zur einseitigen Kreuzungsreduktion die optimale Lösung liefert, falls diese keine Kreuzung enthält.

Lösungsskizze

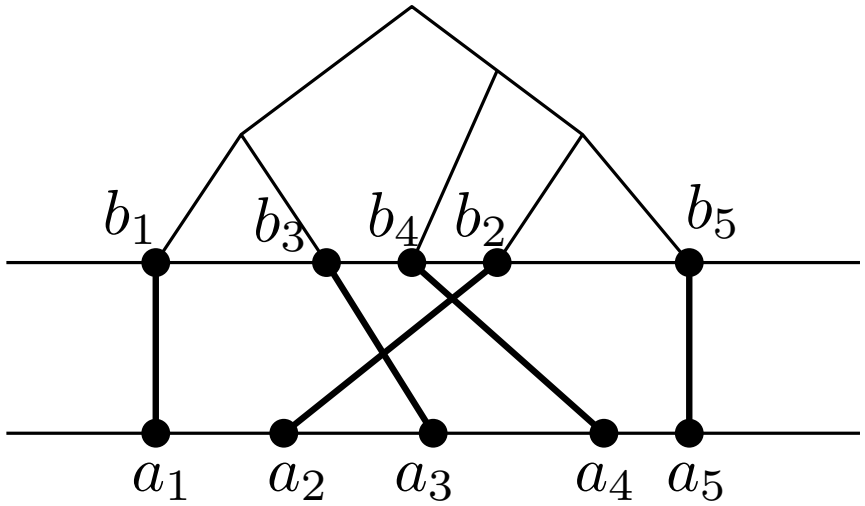
- kreuzungsfrei gdw. alle Nachbarn auf fester Lage konsekutiv sind
- Nachbarn bilden im Inneren disjunkte Intervalle, also getrennte Schwerpunkte
- Schwerpunkte nur identisch bei Grad-1 Knoten $\rightarrow$ Reihenfolge beliebig

Aufgabe 3 – Tanglegram Layout

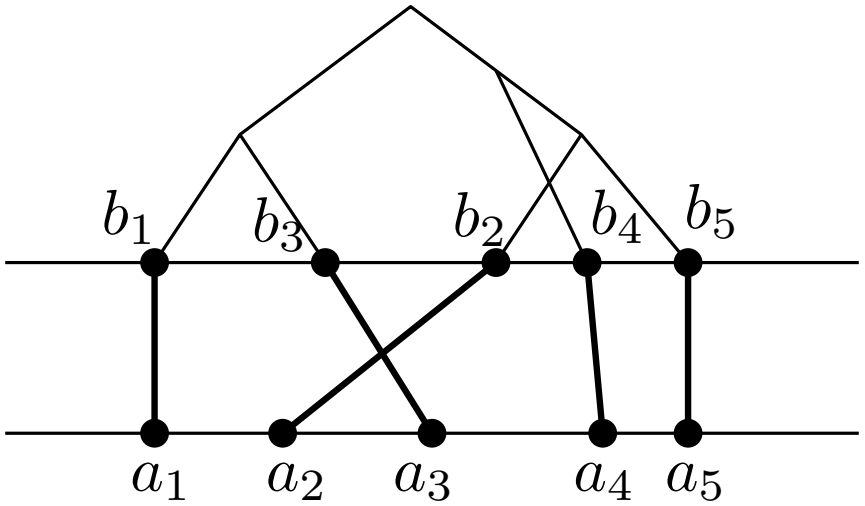
One Tree Crossing Minimization (OTCM): Gegeben sei ein perfektes Matching M auf $2n$ Knoten $A = \{a_1, \ldots, a_n\}$ und $B = \{b_1, \ldots, b_n\}$, mit Matchingkanten $E = \{(a_i, b_i) \mid i = 1, \ldots, n\}$. Bestimme für eine feste Permutation π_1 der Menge A eine Permutation π_2 der Menge B, die die Anzahl der Kreuzungen von M minimiert, wobei A und B geordnet nach π_1 und π_2 auf zwei horizontalen Geraden platziert sind. Dabei muss π_2 *kompatibel* zu einem Binärbaum T mit Blattmenge B sein.

Die Permutation π_2 heißt dabei *kompatibel* zu T, wenn eine planare Zeichnung von T in der Halbebene $y \geq 0$ existiert, die die Blätter in der Reihenfolge π_2 auf der x-Achse platziert und alle inneren Knoten im Bereich $y > 0$.

Aufgabe 3 – Tanglegram Layout



kompatibel



nicht kompatibel

Aufgabe 3 – Tanglegram Layout

(a) Zeigen Sie, dass das Problem OTCM in polynomieller Zeit gelöst werden kann. Welchen asymptotischen Zeitaufwand hat Ihr Algorithmus?

Hinweis: Verwenden Sie dynamische Programmierung.

(b) Angenommen beide Permutationen π_1 und π_2 sind variabel und jeweils durch Kompatibilität mit zwei Binärbäumen T_1 und T_2 auf den Blättern A bzw. B eingeschränkt. Zeigen Sie, dass in polynomieller Zeit entschieden werden kann, ob das Matching kreuzungsfrei gezeichnet werden kann.

Hinweis: Verwenden Sie, dass für einen gerichteten Graphen mit einer einzigen Quelle in linearer Zeit geprüft werden kann, ob er aufwärtsplanar ist.

Aufgabe 3 – Lösungsskizze

- definiere Tabelle C indiziert durch Baumknoten mit $C[v] =$ min. # Kreuzungen des Teilbaums T_v
- initialisiere $C[v] = 0$ für Blätter
- sonst: $C[v] = C[u] + C[w] + \min\{cr(u,w), cr(w,u)\}$, wobei u und w die Kinder von v sind und $cr(u,w)$ die Anzahl Kreuzungen von Matchingkanten von T_u mit Matchingkanten von T_v falls u links von v liegt
- Zeitaufwand $O(n^2)$ bei Berechnung der cr-Werte in jeweils $O(n)$
- Berechnung auch in $O(n \log n)$ möglich

(s. [Fernau, Kaufmann, Poths, Comparing Trees via Crossing Minimization, 2010])

- erzeuge DAG aus T_1 und T_2 gerichtet von der Wurzel von T_1 zur Wurzel von T_2

Aufgabe 4 – Metro Maps

(a) Beim Zeichnen von Metro Maps können neben Knickzahl, Kantenlänge und Erhaltung der relativen Lage andere Optimierungskriterien relevant sein. Modifizieren Sie das in der Vorlesung vorgestellte ILP so, dass zusätzlich der Umfang der Bounding-Box bzw. die Breite bei festgelegter Höhe minimiert wird. Warum ist die Fläche der Zeichnung kein geeignetes Optimierungskriterium?

Aufgabe 4 – Metro Maps

(b) Zur Minimierung der Knickkosten entlang von Metrolinien wurde folgende Kostenfunktion definiert:

$$\mathrm{cost}_{\mathrm{bend}} = \sum_{L \in \mathcal{L}} \sum_{uv, vw \in L} \mathrm{bend}(u, v, w).$$

Dabei steigt die Kostenfunktion $\mathrm{bend}(u, v, w)$ mit steigendem Knickwinkel zwischen den Kanten uv und vw linear an $(45° : 1, 90° : 2, 135° : 3)$. Überlegen Sie sich, wie man diese Kostenfunktion im ILP-Modell umsetzen kann. Sie können dazu z.B. die für jede Kante uv definierte Richtungsvariable $\mathrm{dir}(u, v)$ verwenden. Wie lässt sich eine beliebige monoton steigende Kostenfunktion modellieren?

Aufgabe 4 – Metro Maps

(c) Zur Minimierung der Kantenlänge wurde in der Vorlesung die folgende Kostenfunktion definiert.

$$\text{cost}_{\text{len}} = \sum_{\{u,v\} \in E} \text{len}(\{u, v\})$$

Geben Sie lineare Nebenbedingungen an, die sicherstellen, dass die Variable $\text{len}(\{u, v\})$ der Länge der Kante $\{u, v\}$ bezüglich der Normen L_1, L_∞ und L_2 entspricht. Verwenden sie dazu nur die Positionen $(x(u), y(u))$ und $(x(v), y(v))$ von u bzw. v.

Aufgabe 4 – Lösungsskizze

- führe neue Variablen für Höhe und Breite des Layouts ein
- Minimierung der Fläche würde quadratische Zielfunktion erfordern

- für $\mathrm{cost}_{\mathrm{bend}}$ s. Paper [NW11]

- $-\operatorname{len}(u,v) \leq x(u) - x(v) \leq \operatorname{len}(u,v)$ und
  $-\operatorname{len}(u,v) \leq y(u) - y(v) \leq \operatorname{len}(u,v)$
- wie vorher, aber zwei separate Variablen für Breite und Höhe $\rightarrow \operatorname{len}(u,v) = \mathsf{width}(u,v) + \mathsf{height}(u,v)$
- Problem Euklidischer Abstand nicht linear; aber: hier nur acht Richtungen und diagonaler Abstand ist $\sqrt{2}/2(\mathsf{width}(u,v) + \mathsf{height}(u,v))$