TU Wien
Institut für Logic and Computation
Forschungsbereich Algorithms and Complexity

186.866 Algorithmen und Datenstrukturen VU

Übungsblatt 6

PDF erstellt am: 14. Mai 2024

Deadline für dieses Übungsblatt ist **Montag, 3.6.2024, 20:00 Uhr**. Damit Sie für diese Übung Aufgaben anerkannt bekommen können, gehen Sie folgendermaßen vor:

1. Öffnen Sie den TUWEL-Kurs der Lehrveranstaltung *186.866 Algorithmen und Datenstrukturen (VU 5.5)* und navigieren Sie zum Abschnitt *Übungsblätter*.
2. Teilen Sie uns mit, welche Aufgaben Sie gelöst haben **und** welche gelösten Aufgaben Sie gegebenenfalls in der Übungseinheit präsentieren können. Gehen Sie dabei folgendermaßen vor:
   - Laden Sie Ihre Lösungen in einem einzigen PDF-Dokument in TUWEL hoch.
     Link *Hochladen Lösungen Übungsblatt 6*
     Button *Abgabe hinzufügen* bzw. *Abgabe bearbeiten*
     PDF-Datei mit Lösungen hochladen und *Änderungen sichern*.
   - Kreuzen Sie an, welche Aufgaben Sie gegebenenfalls in der Übung präsentieren können. Die Lösungen der angekreuzten Aufgaben müssen im hochgeladenen PDF enthalten sein.
     Link *Ankreuzen Übungsblatt 6*
     Aufgaben entsprechend anhaken und *Änderungen speichern*.

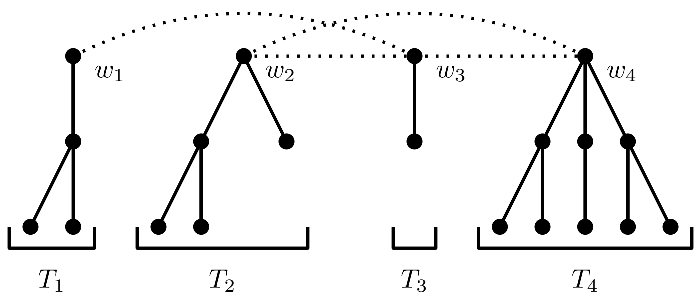
Bitte beachten Sie:

- Bis zur Deadline können Sie sowohl Ihr hochgeladenes PDF, als auch Ihre angekreuzten Aufgaben beliebig oft überschreiben. Sollte kurz vor der Deadline etwas schief gehen (Ausfall TUWEL, Internet, Scanner, etc.) und Sie die Endversion mit allen gelösten Aufgaben nicht mehr hochladen können, haben Sie zumindest Ihre Lösungen teilweise schon hochgeladen und angekreuzt. Nach der Deadline ist keine Veränderung mehr möglich. Die Deadline ist strikt – es werden ausnahmslos keine Nachabgabeversuche (z.B. per E-Mail) akzeptiert.
- Sie können Ihre Lösungen entweder direkt in einem Textverarbeitungsprogramm erstellen und hochladen, oder aber auch gut leserliche Scans bzw. Fotos von handschriftlichen Ausarbeitungen hochladen (beachten Sie die maximale Dateigröße).
- Beachten Sie die Richtlinien für das An- und Aberkennen von Aufgaben. Details dazu finden Sie in den Folien der Vorbesprechung.

Aufgabe 1. Wir betrachten die Klasse $\mathcal{X}$ aller zusammenhängenden Graphen G, die durch folgende Konstruktion entstehen können. Sei $k \in \mathbb{N}$, seien $T_1 = (V_1, E_1), \ldots, T_k = (V_k, E_k)$ Bäume mit Wurzeln $w_1, \ldots, w_k$ und sei $E_w \subseteq \{w_1, \ldots, w_k\} \times \{w_1, \ldots, w_k\}$. Nehmen Sie an, dass $V_i \cap V_j = \emptyset$ für $i \neq j$ gilt. Dann ist G die Vereinigung über die Bäume, bei denen die Wurzeln wie von E_w beschrieben miteinander verbunden sind. Formal ist $G = (V, E)$ definiert durch:

$$V = \bigcup_{1 \leq i \leq k} V_i \quad \text{und} \quad E = E_w \cup \bigcup_{1 \leq i \leq k} E_i.$$

Hier ist ein Beispiel für $k = 4$. Die Kanten in E_w werden gepunktet dargestellt.



Erinnern Sie sich an das im Allgemeinen NP-schwere Problem 3-Coloring. Entwerfen und beschreiben Sie einen polynomiellen Algorithmus, der das Problem für den Spezialfall löst, dass G Teil dieser Graphklasse $\mathcal{X}$ ist. Nehmen Sie dabei an, dass $k \in \mathbb{N}$ **konstant** ist und dass die Knoten $w_1, \ldots, w_k$ explizit gegeben sind. Begründen Sie kurz die Korrektheit Ihres Algorithmus und erläutern Sie dessen Laufzeit.

Aufgabe 2. Wir definieren zwei Probleme: Für das Problem CYCLE PAIR besteht die Eingabe aus einem Graphen G und einer Zahl $\ell \in \mathbb{N}$ und es soll entschieden werden, ob es zwei knoten-disjunkte Kreise in G gibt, die jeweils Länge genau ℓ haben. Für das zweite Problem, nehmen Sie $k \in \mathbb{N}$ als konstant an. Dann ist k-CYCLE das Problem für einen Graphen G zu entscheiden, ob G einen Kreis der Länge genau k enthält. Nehmen Sie an, dass $\mathrm{P} \neq \mathrm{NP}$ gilt.

(a) Zeigen Sie, dass CYCLE PAIR NP-schwer ist, indem Sie von einem verwandten, aus der Vorlesung bekannten Problem reduzieren.

(b) Entwerfen und beschreiben Sie einen polynomiellen Algorithmus für k-CYCLE. Begründen Sie kurz dessen Korrektheit und polynomielle Laufzeit.

(c) Begründen Sie, ob 3-COLOR $\leq_P$ CYCLE PAIR gilt.

(d) Begründen Sie, ob 3-COLOR $\leq_P$ k-CYCLE gilt.

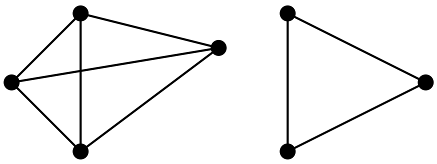
Hinweis: Für die Teilaufgaben (c) und (d) müssen Sie keine Reduktionen angeben.

Aufgabe 3. Wenden Sie den Branch-and-Bound Algorithmus aus der Vorlesung auf die unten angegebene Instanz des Rucksackproblems an. Stellen Sie den Ablauf des Algorithmus als Baum dar. Geben Sie für jeden Schritt die obere Schranke U', die untere Schranke L', sowie eine passende Auswahl von Gegenständen mit Gesamtwert L' an. Verwenden Sie die „Best-First" Heuristik für die Auswahl von Teilproblemen.

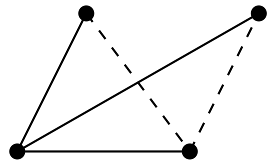
Gegenstand	1	2	3	4	5
Gewicht g_i	6	4	4	6	8
Wert w_i	18	6	8	11	17

Rucksackkapazität: 20

Aufgabe 4. Entwerfen und beschreiben Sie einen Branch-and-Bound Algorithmus für das Optimierungsproblem Cluster Editing, welches wie unten definiert ist. Dabei ist ein *Clustergraph* ein Graph, bei dem jede Zusammenhangskomponente eine Clique ist. Man kann leicht zeigen, dass ein Graph G ein Clustergraph ist, genau dann wenn G keinen Pfad mit drei Knoten als induzierten Teilgraphen enthält. Hier ist ein Beispiel:



Clustergraph



Kein Clustergraph mit einem gestrichelt hervorgehobenem induzierten Pfad mit drei Knoten.

Cluster Editing

Eingabe: Ein (schlichter) Graph G.

Aufgabe: Kanten zu G hinzufügen oder aus G entfernen sodass der resultierende Graph ein Clustergraph ist.

Optimierungsziel: Die Anzahl der hinzugefügten und entfernten Kanten minimieren.

Gehen Sie in Ihrer Lösung insbesondere auf nachfolgende Punkte ein. Pseudocode ist nicht notwendig.

(a) Wie repräsentieren Sie ein Teilproblem?

(b) Wie erfolgt das Branching, d.h. das Aufteilen eines (Teil-)Problems in weitere Teilprobleme?

(c) Beschreiben Sie eine sinnvolle Möglichkeit für die Auswahl des nächsten Teilproblems. Nach welcher Strategie gehen Sie dabei vor?

In dieser Aufgabe können Sie sich auf die Teilprobleme und das Branching konzentrieren. Es ist nicht nötig auf obere und untere Schranken einzugehen.

Aufgabe 5. Im nächsten Schritt sollen Sie Ihren Branch-and-Bound Algorithmus für Cluster Editing aus Aufgabe 4 durch geeignete Heuristiken zur Berechnung von unteren und oberen Schranken verfeinern.

Sei G ein beliebiger schlichter Graph. Betrachten Sie dazu die folgenden beiden Punkte:

(a) Was ist eine geeignete **obere** Schranke für den optimalen Lösungswert wenn G die Eingabe ist?

(b) Was ist eine geeignete **untere** Schranke für den optimalen Lösungswert wenn G die Eingabe ist? Geben sie eine untere Schranke analog zur Vorgehensweise zum Vertex Cover-Problem aus der Vorlesung.

Bestimmen Sie möglichst gute, nicht-triviale untere und obere Schranken. Beschreiben Sie, wie diese berechnet werden können, und stellen Sie sicher, dass sie effizient berechenbar sind.
