TU Wien
Institut für Logic and Computation
Forschungsbereich Algorithms and Complexity

186.866 Algorithmen und Datenstrukturen VU

Übungsblatt 8

PDF erstellt am: 6. Juni 2024

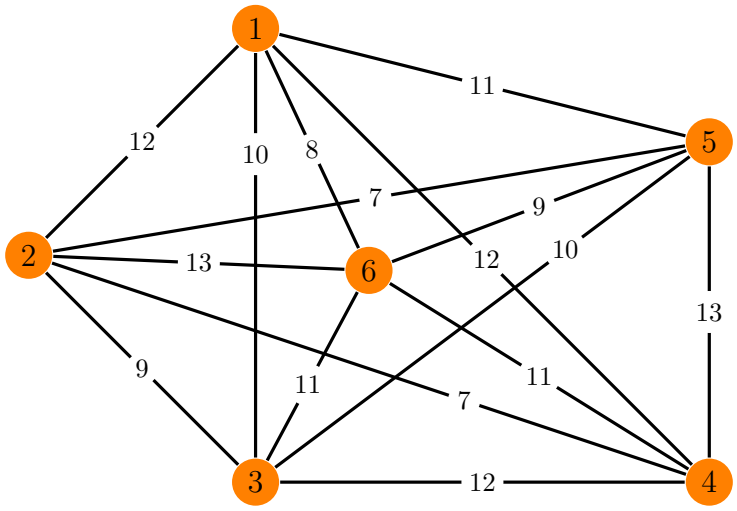
Deadline für dieses Übungsblatt ist **Montag, 17.6.2024, 20:00 Uhr**. Damit Sie für diese Übung Aufgaben anerkannt bekommen können, gehen Sie folgendermaßen vor:

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 8*
     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 8*
     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. Gegeben sei eine Instanz des symmetrischen Traveling Salesperson Problems (TSP) in Form des folgenden vollständigen Graphen G mit Knoten $V = \{1, 2, 3, 4, 5, 6\}$ und Kanten E, denen die eingezeichneten Gewichte zugeordnet sind:



(a) Wenden Sie auf diese Instanz *zweimal* die in der Vorlesung gezeigte Spanning-Tree-Heuristik an, wobei Sie unterschiedliche Eulerkreise verwenden, sodass zwei unterschiedliche Touren erzeugt werden. Beschreiben Sie die einzelnen Schritte im Detail.

(b) Versuchen Sie eine optimale Tour durch „Hinsehen“ zu finden. Geben Sie die Tour und deren Länge an.

(c) Wie viele Touren müssten Sie im Allgemeinen mindestens für das symmetrische TSP mit n Städten und hier konkret in einem vollständigen Enumerations-Verfahren durchprobieren, um eine bewiesen optimale Lösung zu finden?

Hinweis: $n!$ kann durch Ausnutzung von Symmetrien verbessert werden.

(d) Ist die gegebene Instanz metrisch? Was bedeutet das im Allgemeinen für die Gütegarantie der Spanning-Tree-Heuristik? Welche konkreten Approximationsgüten haben Ihre beiden in Unteraufgabe (a) gefundenen Touren?

Aufgabe 2. Betrachten Sie das folgende Optimierungsproblem:

> Sie wollen eine Arbeitsgruppe aus Experten bilden die eine Menge $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_m\}$ von Qualifikationen abdeckt und haben eine Menge an verfügbaren Experten $\mathcal{E} = \{E_1, E_2, \ldots, E_n\}$. Jeder dieser Experten $E \in \mathcal{E}$ hat eine nicht-leere Menge $Q_E \subseteq \mathcal{Q}$ von Qualifikationen. Für jede Qualifikation sind maximal 3 Experten verfügbar. Aus Kostengründen wollen Sie die Arbeitsgruppen so klein wie möglich halten.

Orientieren Sie sich an den Ideen des Algorithmus `Approx-Vertex-Cover` aus der Vorlesung und entwerfen Sie einen polynomiellen Approximationsalgorithmus mit Gütegarantie 3 für dieses Problem.

(a) Beschreiben Sie den Algorithmus.

(b) Erläutern Sie, weshalb die Approximationsgüte gilt.

(c) Geben Sie ein Beispiel an, bei dem sich die optimale Lösung und die Approximation um den Faktor 3 unterscheiden.

(d) Analysieren Sie die asymptotische Laufzeit Ihres Algorithmus.

Freiwillige Zusatzaufgabe: Können Sie die Approximationsgüte auch noch garantieren, wenn es eine Qualifikation (z.B. *Teamchef*) gibt, für die 9 159 993 Experten qualifiziert sind und es für alle anderen Qualifikation immer noch maximal 3 Experten gibt? Begründen Sie Ihre Antwort kurz.

Aufgabe 3. Betrachten Sie das aus der Vorlesung bekannte MAX-CUT Problem, bei dem für einen gewichteten Graphen ein maximaler Schnitt berechnet werden soll.

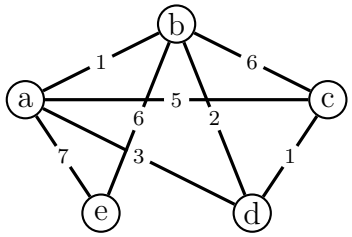
MAX-CUT: Gegeben sei ein ungerichteter Graph $G = (V, E)$ mit positiven ganzzahligen Kantengewichten w_{uv} für alle Kanten $(u, v) \in E$. Finde einen Schnitt, d.h. eine Partition der Knoten (A, B), sodass das Gesamtgewicht $w(A, B)$ von Kanten, die Knoten in den unterschiedlichen Partitionen verbinden, maximiert wird.

$$w(A, B) := \sum_{u \in A,\ v \in B} w_{uv}$$

Betrachten Sie nun folgenden Greedy Algorithmus für MAX-CUT.

Function
GREEDY-MAX-CUT($G = (V, E)$)
  $A \leftarrow \emptyset$, $B \leftarrow \emptyset$
  for $v \in V$
    $a \leftarrow \sum_{u \in A} w_{uv}$
    $b \leftarrow \sum_{u \in B} w_{uv}$
    if $a \leq b$ **then**
      $A \leftarrow A \cup \{v\}$
    else
      $B \leftarrow B \cup \{v\}$
  return (A, B)

Beispiel Graph:



Führen Sie jetzt folgende Analyse durch:

(a) Probieren Sie den Algorithmus zunächst an dem gegebenen Beispiel aus. Betrachten Sie die Knoten in alphabetischer Reihenfolge.

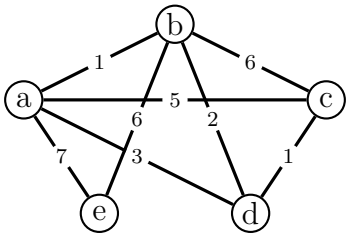
(b) Geben Sie die Worst-Case Laufzeit in Θ-Notation in Abhängigkeit von $n = |V|$ und $m = |E|$ an.

(c) Zeigen Sie, dass GREEDY-MAX-CUT kein 4/5-Approximationsalgorithmus ist. Geben Sie dazu einen geeigneten Graphen, die optimale Lösung und die Lösung von GREEDY-MAX-CUT an. Halten Sie den Graphen dabei möglichst klein.

(d) Zeigen Sie, dass GREEDY-MAX-CUT ein 1/2-Approximationsalgorithmus ist.

Freiwillige Zusatzaufgabe: Zeigen Sie, dass 1/2 die beste Gütegarantie ist, die für diesen Algorithmus gilt. D.h. geben Sie für jede bessere Gütegarantie $1/2 + \varepsilon$ einen Graphen an, bei dem diese Gütegarantie nicht erfüllt ist.

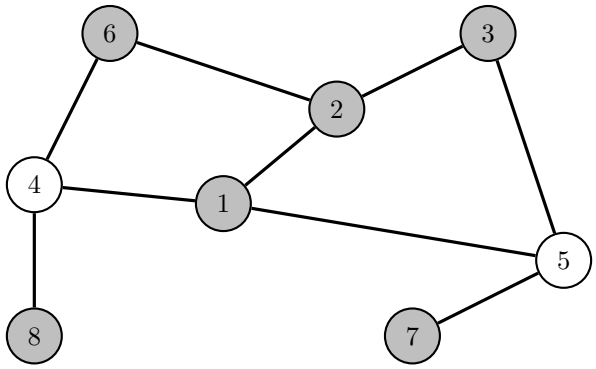
Aufgabe 4. Betrachten Sie das MAX-CUT Problem aus Aufgabe 3. Wir wenden nun lokale Suche für das MAX-CUT Problem an. Dazu verwenden wir die aus der Vorlesung bekannte Flip-Nachbarschaftsstruktur N für MAX-CUT. Es ist folgende Instanz von MAX-CUT gegeben:



(a) Bestimmen Sie $N((A, B))$ für den Schnitt $(A, B) = (\{a, c\}, \{b, d, e\})$.

(b) Wenden Sie lokale Suche mit der Flip-Nachbarschaftsstruktur N und Ausgangslösung $(A, B) = (\{a, c\}, \{b, d, e\})$ auf die obige Instanz an, bis ein lokales Optimum erreicht wird. Durchmustern Sie die aktuelle Nachbarschaft so, dass jedenfalls immer eine bessere Nachbarlösung gefunden wird, wenn eine solche in der Nachbarschaft existiert (Sie können frei zwischen First Improvement oder Best Improvement als Schrittfunktion wählen). Geben Sie alle Zwischenschritte an, also jeden Schnitt, welcher im Laufe der lokalen Suche ausgewählt wird.

Aufgabe 5. Wir wenden lokale Suche für das VERTEX COVER Problem an. Dazu verwenden wir die zwei aus der Vorlesung bekannten Nachbarschaftstrukturen N (simples löschen) und N' (alternative Nachbarschaft) für Vertex Cover. Gegeben ist nachfolgend ein Graph $G = (V, E)$ und eine Ausgangslösung $S = \{1, 2, 3, 6, 7, 8\}$ (Knoten mit grauem Hintergrund).



(a) Geben Sie $N(S)$ und $N'(S)$ an.

(b) Kann eine lokale Suche welche N verwendet, angewandt auf G, mit Ausgangslösung S ein minimales Vertex Cover finden? Falls ja, geben Sie einen Durchlauf der lokalen Suche an, der solch ein minimales Vertex Cover findet. Falls nein, zeigen Sie, dass die lokale Suche in keinem globalen Optimum enden kann.

(c) Beantworten Sie die selbe Frage wie bei Aufgabe 5(b), aber mit N' statt N.

(d) Überlegen Sie sich, welchen Einfluss die Wahl der Schrittfunktion (Best Improvement, First Improvement, Random Neighbor) bei der lokalen Suche für Vertex Cover mit Nachbarschafstruktur N bzw. N' hat. Vergleichen Sie dies mit dem Einfluss der Schrittfunktion in der lokalen Suche für das MAX-CUT Problem.
