TU Wien
Institut für Logic and Computation
Forschungsbereich Algorithms and Complexity

186.866 Algorithmen und Datenstrukturen VU

Übungsblatt 7

PDF erstellt am: 23. Mai 2022

Deadline für dieses Übungsblatt ist **Montag, 20.06.2022, 20:00 Uhr**. Um Aufgaben für diese Übung anerkannt zu bekommen, 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 7*
     Button *Abgabe hinzufügen*
     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 7*
     Button *Abgabe bearbeiten*
     Bearbeitete Aufgaben anhaken und *Änderungen speichern*.

Bitte beachten Sie:

- Bis zur Deadline können Sie sowohl Ihr hochgeladenes PDF, als auch Ihre angekreuzten Aufgaben beliebig oft verändern. Nach der Deadline ist keine Veränderung mehr möglich. Es werden ausnahmslos keine Nachabgabeversuche (z.B. per E-Mail) akzeptiert.
- Sie können Ihre Lösungen entweder direkt in einem Textverarbeitungsprogramm erstellen, 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 finden Sie in den Folien der Vorbesprechung).

Aufgabe 1. Gegeben ist folgende Instanz des Rucksackproblems:

Kapazität $G = 11$

#	Wert	Gewicht
1	2	1
2	3	2
3	7	4
4	9	5
5	13	8

(a) Wenden Sie den aus der Vorlesung bekannten Greedy-Algorithmus für das Rucksackproblem auf die obige Instanz an. Welche Gegenstände werden ausgewählt? Welchen Gesamtwert haben diese?

(b) Lösen Sie die obige Instanz jetzt durch dynamische Programmierung. Geben Sie dazu, wie aus der Vorlesung bekannt, die vollständige Belegung der 2-dimensionalen Lösungstabelle an.

Geben Sie die Menge der für die optimale Lösung ausgewählten Gegenstände an und markieren Sie in der Tabelle all jene Felder, die der Algorithmus Find-Solution(M) aus der Vorlesung bei der Berechnung der Lösungsmenge A ausliest und verwendet. Welchen Gesamtwert hat die optimale Lösung?

	0	1	2	3	4	5	6	7	8	9	10	11
$\emptyset$												
$\{1\}$												
$\{1,2\}$												
$\{1,2,3\}$												
$\{1,2,3,4\}$												
$\{1,2,3,4,5\}$												

Aufgabe 2. Gegeben sei ein Graph $G = (V, E)$. Der folgende Greedy-Algorithmus findet ein Independent Set in G, also eine Knoten-Menge $I \subseteq V$, in der keine zwei Knoten adjazent zueinander sind.

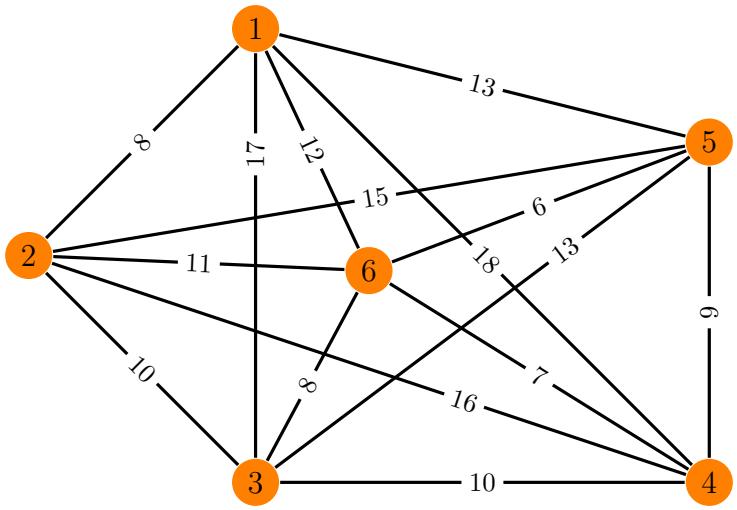
```
1 I ← {}
2 while V(G) has vertices
3     choose v ∈ V(G) with deg(v) minimal
4     I ← I ∪ {v}
5     remove N(v) ∪ {v} and all incident edges from G
6 return I
```

(a) Warum werden in dem Algorithmus in jeder Iteration auch die Nachbarn von v aus dem Graphen entfernt?

(b) Angenommen die Graphen, auf die der Algorithmus angewandt wird, haben maximalen Knotengrad Δ, d.h. kein Knoten in G hat mehr als Δ Nachbarn. Zeigen Sie, dass der gegebene Algorithmus dann eine Gütegarantie von $1/(\Delta + 1)$ für das Maximum Independent Set Problem besitzt. Beim Maximum Independent Set Problem geht es darum, ein Independent Set mit der maximalen Anzahl an Knoten zu finden.

Aufgabe 3. Betrachten Sie nochmal den Algorithmus aus dem vorhergehenden Beispiel zum Auffinden eines Independent Sets, diesmal für beliebige schlichte Graphen. Zeigen Sie, dass der Algorithmus keine Gütegarantie größer 0 hat.
Hinweis: Zeigen Sie für ein gegebenes $n \in \mathbb{N}$, dass die Gütegarantie kleiner als $1/n$ ist. Dies kann man beispielsweise erreichen, indem man eine Klasse von Graphen findet, für welchen der Algorithmus eine Lösung konstanter Größe liefert, aber die Optimallösung beliebig groß werden kann.

Aufgabe 4. Gegeben sei folgende symmetrische Traveling Salesperson Problem (TSP) Instanz in Form eines vollständigen Graphen G mit Knoten $V = \{1, 2, 3, 4, 5, 6\}$ und Kanten E mit Gewichten wie eingezeichnet:



(a) Wenden Sie *zweimal* die in der Vorlesung gelernte Spanning-Tree-Heuristik auf diese Instanz an und beschreiben Sie die einzelnen Schritte im Detail, mit jeweils unterschiedlichen Eulerkreisen, sodass zwei unterschiedliche Touren erzeugt werden.

(b) Finden Sie eine optimale Tour durch „Hinsehen“. Wie viele Touren müssten Sie mindestens allgemein für das symmetrische TSP und hier konkret in einem Enumerations-Verfahren durchprobieren, um die bewiesen optimalen Lösungen zu finden?

Hinweis: $n!$ kann verbessert werden.

(c) Besteht eine Gütegarantie auf der gegebenen Instanz und wenn ja, welche? Welche konkreten Approximationsgüten haben Ihre beiden in Unteraufgabe (a) gefundenen Touren?

(d) Welche asymptotische Laufzeit in Θ-Notation hat die Spanning-Tree-Heuristik?

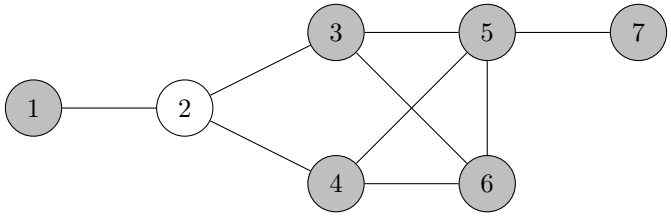
Aufgabe 5. Betrachten Sie das Problem *Lastverteilung* aus der Vorlesung zu Approximationsalgorithmen und den zugehörigen Algorithmus `List-Scheduling`, der, wie in der Vorlesung gezeigt, im Allgemeinen eine Approximationsgüte von 2 garantiert.

Wir schränken die zulässigen Probleminstanzen nun wie folgt ein:

1. Es gibt mindestens 120 Jobs.
2. Es gibt höchstens 4 Maschinen.
3. Jeder Job j hat eine Bearbeitungszeit $1 \leq t_j \leq 10$.

Liefert der Algorithmus `List-Scheduling` auf solchen Instanzen eine bessere Approximationsgüte als 2? Falls ja, geben Sie eine möglichst kleine Schranke an, die für alle oben beschriebenen Instanzen gilt und begründen Sie Ihre Antwort. Falls nein, geben Sie ein Gegenbeispiel an, für das die Approximationsgüte weiterhin bei 2 liegt.

Aufgabe 6. Gegeben ist ein Graph $G = (V, E)$ mit Vertex Cover $S = V \setminus \{2\}$.



(a) Finden Sie durch „Hinsehen“ ein minimales Vertex Cover für G und argumentieren Sie warum es kein kleineres Vertex Cover geben kann.

(b) Sei N die aus der Vorlesung bekannte naive Nachbarschaftsstruktur für Vertex Cover, also $C' \in N(C)$ wenn C' aus C durch Löschen eines einzigen Knotens erzeugt werden kann und noch immer ein Vertex Cover ist.

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) Sei N' die aus der Vorlesung bekannte *verbesserte* Nachbarschaftsstruktur für Vertex Cover, also $C' \in N'(C)$ wenn $C' \in N(C)$ oder wenn C' durch Hinzufügen eines Knotens von $V \setminus C$ und Entfernen von zwei Knoten aus C gebildet werden kann und noch immer ein Vertex Cover ist.

Beantworten Sie die selbe Frage wie bei Aufgabe 6(b), aber mit N' statt N.

(d) Liefert eine lokale Suche nach einem Vertex Cover mit N' auf jedem Graphen ein globales Optimum? Falls ja, begründen Sie Ihre Antwort. Falls nein, geben Sie ein Gegenbeispiel an.
