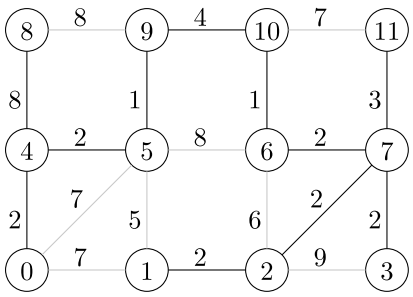
RWTH Aachen **SS 2016**
Lehrgebiet Theoretische Informatik Lösungsvorschlag 12
Rossmanith–Hensel–Kuinke–Sánchez 11. Juli 2016

Übung zur Vorlesung Datenstrukturen und Algorithmen

Aufgabe T36



Aufgabe T37

Zunächst sortieren wir die Aktivitäten nach der Endzeit, so daß die frühste Endzeit als erstes betrachtet wird. Dann gehen wir greedy vor:

1. Wähle Aktivität mit frühster Endzeit.
2. Lösche diese und alle die sich damit überschneiden.
3. falls Menge der Aktivitäten nicht leer, gehe zu 1.

Die Laufzeit beträgt $O(n \log n)$, da die Zeiten sortiert werden müssen.
Da jedes gewählte Intervall die frühste Endzeit von allen übrigen hat, werden in Schritt 2 nur Intervalle entfernt die sich alle gegenseitig schneiden, und deshalb nur maximal eins aus der Menge gewählt werden kann. Demnach ist die greedy Lösung auch optimal.

Aufgabe T38

Wir müssen folgende zwei Eigenschaften aus der Vorlesung zeigen:
Ein Matroid $M = (S, I)$ besteht aus einer Basis S und einer Familie $I \subseteq 2^S$ von unabhängigen Mengen mit:

1. Falls $A \subseteq B$ und $B \in I$, dann $A \in I$ (M ist hereditary).
2. Falls $A, B \in I$ und $|A| < |B|$ dann gibt es ein $x \in B$ so daß $A \cup \{x\} \in I$ (M hat die Austauscheigenschaft).

Die erste Eigenschaft ist einfach zu sehen, da das Entfernen einer Kante niemals Kreise hinzufügt oder Komponenten verbindet.
Für die zweite Eingeschaften beachten wir, daß ein Wald mit k Kanten aus $|V| - k$ Bäumen besteht. Sei nun $G_A = (V, A)$ und $G_B = (V, B)$ mit $A, B \in I$ und $|A| < |B|$.

Demnach hat G_B mehr Kanten und damit weniger Bäume als G_A. Da G_B mindestens aus drei Bäumen besteht muss G_A mindestens aus vier Bäumen bestehen. Nehmen wir an jede Kante in B schließt einen Kreis in $G[A]$. Diesen Kreis gibt es nicht in $G[B]$. Das heißt, daß dieser Kreis in $G[A]$ mit (u, v) eine Kante enthält die nicht in B ist für jede Kante aus B. Daraus folgt, daß A mindestens so groß wie B ist (Widerspruch). Es ist also möglich eine weitere Kante aus B in A einzufügen so daß $A \in \mathcal{F}$ erhalten bleibt.

Aufgabe H33

Wir betrachten die Menge $\{1, \ldots 16\}$. Da sich bei benutzung der Rangheuristik der Rang nur erhöht, wenn wir *union* auf zwei Bäumen mit dem gleichen Rang anwenden, werden wir keine Bäume mergen die unterschiedlichen Rang haben.

Desweiteren wollen wir nicht, daß *find* auf Elemente ausgeführt wird die nicht in der Wurzel stehen, damit die Bäume nicht durch Pfadkompression geglättet werden.

Der erste durchlauf ist also: *union*(1,2), *union*(3,4), *union*(5,6), *union*(7,8), ..., *union*(15,16). Um Bäume mit jeweils Rang eins zu erzeugen.

Und jetzt führen wir weitere merges benachbarter Bäume durch.

union(1,3), *union*(5,7), *union*(9,11), *union*(13,15).

union(1,5), *union*(9,13).

union(1,9).

Da sich bei jedem *union* der rang um eins erhöht is die Gesamthöhe danach vier.

Aufgabe H34

Wir können dieses Problem durch eine abgewandelte Version des Algorthmus von Kruskal lösen. Nach Aufgabe T38 wissen wir, daß ein Greedy-Algorithmus die optimale Lösung liefert. Wir wählen also minimale Kanten, die keinen Kreis erzeugen, und jedesmal wenn wir eine *union* Operation ausführen in der wir zwei Komponenten zusammenführen, erhöhen wir einen counter k um eins. Wir terminieren wenn $n - k = 3$.

Aufgabe H35

Die Knotenmenge des Flußnetzwerks ist $\{s, t\} \cup \{J_1, \ldots, J_n\} \cup \{M_1, \ldots, M_m\}$. Die Kanten des Netzwerks sind wie folgt:

1. Von der Quelle s läuft zu jedem Knoten J_i eine Kante mit Kapazität j_i, wenn j_i die für einen angenommenen Auftrag J_i erhaltene Zahlung ist.

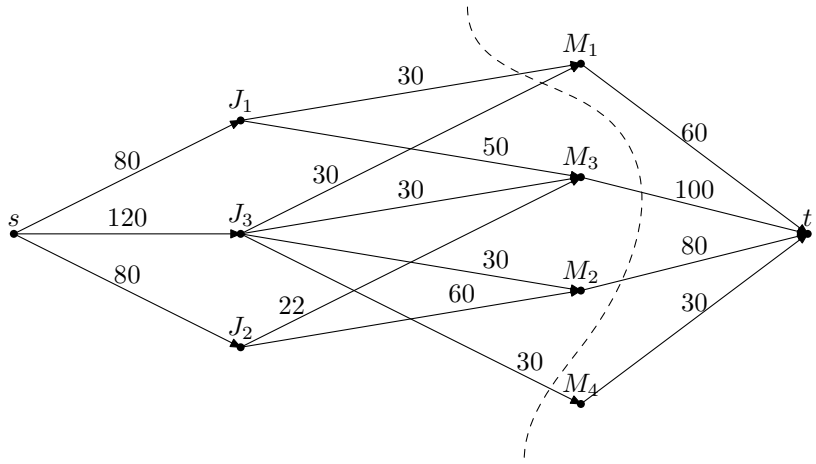
2. Von einem Auftrag J_i läuft eine Kante zur Maschine M_j, wenn M_j für den Auftrag J_i benötigt wird. Die Kapazität auf dieser Kante sind die Kosten, die für eine einmalige Anmietung der Maschine M_j für den Auftrag J_i entstehen (Tabelle 3).

3. Von jedem Knoten M_i läuft eine Kante mit Kapazität m_i in die Senke t, wenn m_i die Kosten sind, um die Maschine M_i zu kaufen.

Ein Min-Cut (S, T) für dieses Netzwerk ist natürlich endlich und ganzzahlig und kann effizient mit der Methode von Ford und Fulkerson berechnet werden. Wir wählen nun diejenigen Aufträge aus, die in S enthalten sind. Ebenso kaufen wir alle Maschinen in S.

Zur Begründung, warum dieses Verfahren korrekt ist: Die Kanten im Schnitt entsprechen anschaulich einem *Verzicht* auf einen Geldbetrag, welche der Kapazität dieser Kante entspricht. Wird etwa eine Kante $s \to J_i$ für einen Auftrag J_i vom Schnitt (S, T) geschnitten,

dann verzichtet man auf die Einnahmen in Höhe von j_i. Der Verzicht auf die Einnahmen kann günstiger sein (und zu einem kleineren Schnitt führen), als wenn man den Job annimmt und dafür viele andere Kanten (für Aufträge oder Maschinen) schneiden muß. Schneidet man analog eine Kante $M_i \to t$, dann muß man den entsprechenden Kaufpreis für eine Maschine bezahlen. Auch dieses kann günstiger sein, als die Maschine für jeden angenommen Job zu mieten. Liegt andererseits eine Maschine M_j in T, dann ist es offenbar günstiger, M_j für jeden angenommenen Auftrag J_i anzumieten; andernfalls könnte man im leicht einen kleineren Schnitt konstruieren, indem man M_j nach S verschiebt (und dem Fall entspricht, M_j zu kaufen), was im Widerspruch zur Minimalität des Schnittes steht.



Aufgabe H36