RWTH Aachen **SoSe 2007**
Lehrgebiet Theoretische Informatik Blatt 11
Kneis—Mölle—Richter—Rossmanith—Thiemann 25.6.2007

Übungen zur Vorlesung Datenstrukturen und Algorithmen

T30

Die internationale Raumstation ISS steht auch Weltraumtouristen offen. Sie sollen nun entscheiden, welche Touristen Sie mitnehmen wollen, um möglichst viel Geld zu verdienen. Gegeben sind Kandidaten $K_1, \ldots, K_n$, welche jeweils bereit sind, $k_1, \ldots, k_n$ US-Dollar zu zahlen. Allerdings sind sie anspruchsvoll und erwarten auf der ISS auch ein Unterhaltungsprogramm (der Erstbesucher Cameron wollte zum Beispiel einen Weltraumspaziergang machen). Zu diesem Zweck stehen eine Menge „Spielzeuge“ $Z_1, \ldots, Z_m$ zur Verfügung. Bei der Mitnahme eines Spielzeugs zur ISS entstehen allerdings jeweils Kosten $z_1, \ldots, z_m$. Der Kandidat K_i ist nur bereit zu zahlen, wenn die Spielzeuge $R_i \subseteq \{Z_1, \ldots, Z_m\}$ mitgenommen werden.
Entwerfen Sie einen effizienten Algorithmus, der eine Menge von Kandidaten auswählt, um die Einnahmen (also die gezahlten Gebühren der Touristen minus die Kosten für die Spielzeuge) zu maximieren. Jedes Spielzeug muß nur einmal mitgenommen werden, selbst wenn mehrere es benutzen wollen.

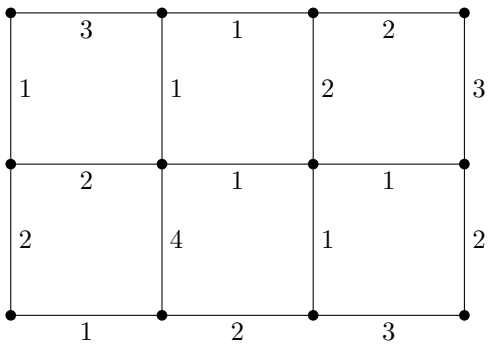
Lösungsvorschlag:
Wir konstruieren auf der Knotenmenge $\{s, t\} \cup \{K_1, \ldots, K_n\} \cup \{Z_1, \ldots, Z_m\}$ das folgende Flußproblem:

1. Von der Quelle s läuft zu jedem Knoten K_i eine Kante mit Kapazität k_i.
2. Für jedes $Z_j \in R_i$ läuft eine Kante mit unendlicher Kapazität von K_i nach Z_j.
3. Von jedem Knoten Z_i läuft eine Kante mit Kapazität z_i in die Senke t.

Ein Min-Cut für dieses Netzwerk ist natürlich endlich, da es endliche Schnitte gibt (beispielsweise die Menge aller Kanten, die in s beginnen). Die gestrichenen Kanten entsprechen jeweils verzichtetem Gewinn (durch Verzicht auf einen Touristen oder Verzicht auf die Ersparnis durch Nichtmitnahme eines Spielzeugs). Somit entspricht jeder Min-Cut einer legalen Konfiguration (und andersherum), und so weiter...

T31

Führen Sie die Algorithmen von Prim und Kruskal auf folgendem Graphen aus.



Lösungsvorschlag:
Bei korrekter Ausführung ergibt sich jeweils ein Spannbaum minimalen Gewichts, vermutlich 15.

H25 (10 Punkte)
Es sei $G = (V, E)$ ein Flußnetzwerk und f ein Fluß. Weiter gelte $S \cup M \cup T = V$, und S, M, T seien paarweise disjunkt. Schließlich gelte noch $s \in S$ und $t \in T$.
Zeigen oder widerlegen Sie: $f(S, M) = f(M, T)$.

Lösungsvorschlag:

$$f(S, M) = f(V, M) - f(M \cup T, M) = f(M, M \cup T) = f(M, M) + f(M, T) = f(M, T)$$

H26 (10 Punkte)
In Manchukuo gibt es Briefmarken mit 7, 10, 13 und 15 Fen.
Beweisen oder widerlegen Sie: Es gibt ein Porto, das durch Anwendung des Greedy-Algorithmus für den Münzwechsel (immer die größte Münze oder Briefmarke zu nehmen) zu einem Ergebnis führt, das aber nicht optimal ist.
In der Vorlesung sahen wir bereits, daß 20 Fen zu gar keinem Ergebnis führt.

Lösungsvorschlag:
Es sei $n \in \mathbf{N}$ eine Zahl und $n = 7a + 10b + 13c + 15d$ mit $a, b, c, d \in \mathbf{N}$, so daß die Summe $a + b + c + d$ (also die Zahl der Briefmarken) minimal ist.
Weil der Greedy-Algorithmus mindestens d-mal die 15 wählen würde, ist es vollkommen ausreichend, die Fälle mit $d = 0$ zu betrachten.
Wir haben vorausgesetzt, daß die Wechselstrategie $n = 7a + 10b + 13c$ optimal ist. Dann muß $a < 4$ gelten, weil man $4 \cdot 7 = 28$ via $13 + 15 = 28$ billiger erzeugen kann. Weiterhin muß $b < 3$ gelten, da $3 \cdot 10 = 30$ auch durch $2 \cdot 15 = 30$ ersetzbar ist. Ohne Beschränkung der Allgemeinheit darf schließlich noch $c < 4$ angenommen werden, weil $4 \cdot 13 = 52$ nicht billiger als $3 \cdot 15 + 7 = 52$ ist (jeweils vier Briefmarken).
Folglich sind nur $4 \cdot 3 \cdot 4 = 48$ Fälle übrig. Diese lassen sich beispielsweise durch ein kleines Programm überprüfen. Zur Abwechslung mal in C:

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  int a,b,c,n,ga,gb,gc;
  for (a=0; a<4; a++)
    for (b=0; b<3; b++)
      for (c=0; c<4; c++) {
        n = 7*a + 10*b + 13*c;
        gc = n/13; n -= 13*gc;
        gb = n/10; n -= 10*gb;
```

```
        ga = n/7; n -= 7*ga;
        if (n ≡ 0 ∧ga+gb+gc>a+b+c) {
          printf ("Greedy-Loesung existiert, ist aber suboptimal:\n"
                  "7*%d+10*%d+13*%d ist teurer als 7*%d+10*%d+13*%d.\n",
                  ga, gb, gc, a, b, c);
        }
      }
  return 0;
}
```

Wenn man das Programm laufen läßt, macht es überhaupt keine Ausgabe. Damit ist die zu untersuchende Aussage widerlegt.