Universität Karlsruhe (TH)
Lehrstuhl Informatik für Ingenieure und Naturwissenschaftler
Dr. Thomas Worsch
Matthias Schulz

21. Mai 2008

Einführung in die Informatik

Lösungen zu Übungsblatt 4

– Greedy-Algorithmus, Färbungen –

Aufgabe 1:

a) Sei $A = Sg(A_G + A_G^2 + \ldots + A_G^m)$ und $A' = Sg(\sum_{i=1}^{m} c_i A_G^i)$.

Alle Einträge von A und A' sind entweder 0 oder 1. Falls $A_{i,j}$ 1 ist, gibt es ein $k \in \{1, \ldots, m\}$, für das $(A_G^k)_{i,j} > 0$ gilt. Dann gilt aber auch $A'_{i,j} = 1$, da auch $c_k(A_G^k)_{i,j} > 0$ gilt and alle Zahlen, die addiert werden, nicht negativ sind.

Ebenso gilt $A_{i,j} = 1$ wenn $A'_{i,j} = 1$ gilt, und damit folgt $A = A'$.

b) Angenommen, es gibt ein Paar i, j, so dass $Sg(A_G + A_G^2 + \ldots + A_G^m)$ für i, j eine 1 enthält während $Sg(A_G + A_G^2 + \ldots + A_G^n)$ für i, j eine 0 enthält.

Für $q > n$ enthält A_G^q für i, j genau dann einen Eintrag > 0, wenn es einen Weg W der Länge q von i nach j gibt (siehe Vorlesung). Dies bedeutet aber, dass auf diesem Weg mindestens ein Knoten k zweimal besucht wurde.

Damit gibt es einen kürzeren Weg von i nach j, indem man den Kreis von k nach k aus W löscht.

Dies kann man wiederholen, bis jeder Knoten höchstens einmal im Weg von i nach j vorkommt. Damit gibt es ein $p < n$, für das A_G^p für i, j ebenfalls 1 ist.

Damit muss auch $Sg(A_G + A_G^2 + \ldots + A_G^m)$ für i, j 1 sein, was der Voraussetzung widerspricht.

Da nur nicht negative Zahlen addiert werden, kann $Sg(A_G + A_G^2 + \ldots + A_G^m)$ an einer Stelle i, j nur 0 sein, wenn auch $Sg(A_G + A_G^2 + \ldots + A_G^n)$ für i, j 0 ist.

Damit sind die beiden Matrizen gleich.

c) Es gilt: $(I + A_G)^m = \sum_{i=0}^{m} \binom{m}{i} A_G^i$, da $A \cdot I^k = A$ für alle $k \in \mathbb{N}_0$ gilt.

Da $\binom{m}{i}$ immer positiv ist und $\sum_{i=0}^{m} \binom{m}{i} A_G^i - I = \sum_{i=1}^{m} \binom{m}{i} A_G^i$ gilt, folgt aus Aufgabe a) die Behauptung.

d) Die Matrix A_1 sei A. Für $i \geq 1$ wird die Matrix $A_{i+1} = A_i \cdot A_i$ jeweils mit einer Matrixmultiplikation berechnet.

Nach k Matrixmultilpikationen hat man $A_k = A^{(2^k)}$.

e) Sei $k = \lceil \log_2 n \rceil$. Man berechnet in $O(n^2)$ die Matrix $(A_G + I)$, führt k Matrixmultiplikationen durch, die jeweils eine Zeitkomplexität in $O(n^{\log_2 7})$ benötigen, zieht in $O(n)$ Schritten I von der erhaltenen Matrix A ab und berechnet in $O(n^2)$ Schritten $Sg(M)$.

Das Ergebnis ist nach Aufgabenteilen b) und c) die Wegematrix.

Der Algorithmus ist schneller als der Warshall-Algorithmus, da $\lceil \log_2 n \rceil \cdot n^{log_2 7} \in O(n^3)$, aber $n^3 \notin O(\lceil \log_2 n \rceil \cdot n^{\log_2 7})$ gilt.

Aufgabe 2:

a) Vollständige Induktion:

Induktionsanfang: Der Graph bestehend aus zwei verbundenen Knoten ist zweifärbbar.

Induktionsschritt: Entfernt man ein Blatt x von einem Baum mit $k+1$ Knoten, so erhält man einen Baum mit k Knoten, der nach Induktionsvoraussetzung bipartit ist. x ist mit genau einem Knoten y dieses kleineren Baumes verbunden und kann in derjenigen Farbe angemalt werden, die y nicht hat. Somit ist auch der größere Baum zweifärbbar.

b) $K_{3,3}$ ist bekannterweise nicht planar. Andererseits ist $K_{3,3}$ bipartit und damit sogar zweifärbbar, also insbesondere auch 4-färbbar.

c) K_4 kann weder K_5 noch $K_{3,3}$ als Teilgraph enthalten, da K_4 jeweils weniger Knoten besitzt. Somit ist K_4 planar.

Angenommen, man könnte den K_4 mit drei Farben färben. Dann müssten zwei Knoten u, v die gleiche Farbe haben. Da im K_4 jeder Knoten jedoch mit jedem anderen verbunden ist, sind auch u und v durch eine Kante verbunden, so dass die Färbung nicht konfliktfrei ist.

Somit braucht man 4 Farben, um K_4 konfliktfrei einzufärben.

d) Wir versuchen eine 3-Färbung von U.

Knoten 1 habe Farbe 1, Knoten 2 habe Farbe 2.

Da Knoten 5 sowohl mit 1 als auch mit 2 verbunden ist, muss Knoten 5 die Farbe 3 haben.

Da Knoten 6 sowohl mit 5 als auch mit 2 verbunden ist, muss Knoten 6 die Farbe 1 haben.

Da Knoten 3 mit 2 und 6 verbunden ist, muss Knoten 3 die Farbe 3 haben.

Da Knoten 7 mit 3 und 6 verbunden ist, muss Knoten 7 die Farbe 2 haben.

Da Knoten 4 mit 1, 3 und 7 verbunden ist, kann 4 nicht konfliktfrei mit einer der drei Farben gefärbt werden.

Färbt man Knoten 4 jedoch in einer vierten Farbe, so ergibt sich eine konfliktfreie Färbung, wie man leicht überprüfen kann.

Somit ist die minimale Anzahl an Farben, die nötig sind, um U konfliktfrei einzufärben, 4.

Aufgabe 3: Ein Matroid (bei den Bezeichnungen wie auf dem Übungsblatt) erfüllt folgende Eigenschaften:

- $\emptyset \in U$.
- $A \in U, B \subseteq A \Rightarrow B \in U$.
- $A, B \in U, |A| < |B| \Rightarrow \exists x \in B \setminus A : A \cup \{x\} \in U$.

a) Wir prüfen nacheinander die Bedingungen:

$\emptyset \in U$ ist erfüllt.
Jede Teilmenge von $\emptyset$ liegt in U. (Einzige Teilmenge ist $\emptyset$)
Da alle Elemente von U 0 Elemente enthalten, gilt die dritte Bedingung.

b) Da $\emptyset \notin U$, liegt hier kein Matroid vor.

c)
- $\emptyset \in U$.
- Alle ein- und zweielementigen Teilmengen von E sind in U, alle Elemente von U enthalten höchstens drei Elemente $\Rightarrow$ Wenn $A \in U$ gilt, ist auch jede Teilmenge von A in U enthalten.
- Zu jeder 0- und 1-elementigen Teilmenge lässt sich ein beliebiges Element hinzufügen, so dass die erhaltene Menge wieder in U liegt.
  Jede dreielementige Menge enthält die maximale Anzahl von Elementen, so dass man nur überprüfen muss, ob man aus jeder 3-elementigen Menge zu jeder 2-elementigen Menge ein Element hinzufügen kann, so dass die erhaltene 3-elementige Menge wieder in U liegt.

  Jede 3-elementige Menge enthält e_3 oder e_4, so dass $\{e_1, e_2\}$ immer zu $\{e_1, e_2, e_3\} \in U$ oder $\{e_1, e_2, e_4\} \in U$, $\{e_1, e_5\}$ immer zu $\{e_1, e_3, e_5\} \in U$ oder $\{e_1, e_4, e_5\} \in U$ und $\{e_2, e_5\}$ immer zu $\{e_2, e_3, e_5\} \in U$ oder $\{e_2, e_4, e_5\} \in U$ ergänzt werden kann.

  Jede 3-elementige Menge enthält e_2 oder e_5, so dass $\{e_1, e_3\}$ immer zu $\{e_1, e_2, e_3\} \in U$ oder $\{e_1, e_3, e_5\} \in U$, $\{e_1, e_4\}$ immer zu $\{e_1, e_2, e_4\} \in U$ oder $\{e_1, e_4, e_5\} \in U$ und $\{e_3, e_4\}$ immer zu $\{e_2, e_3, e_4\} \in U$ oder $\{e_3, e_4, e_5\} \in U$ ergänzt werden kann.

  Jede 3-elementige Menge enthält e_1, e_4 oder e_5, so dass $\{e_2, e_3\}$ immer zu $\{e_1, e_2, e_3\} \in U$ oder $\{e_2, e_3, e_4\} \in U$ oder $\{e_2, e_3, e_5\} \in U$ ergänzt werden kann.
  Jede 3-elementige Menge enthält e_1, e_3 oder e_5, so dass $\{e_2, e_4\}$ immer zu $\{e_1, e_2, e_4\} \in U$ oder $\{e_2, e_3, e_4\} \in U$ oder $\{e_2, e_4, e_5\} \in U$ ergänzt werden kann.

Jede 3-elementige Menge enthält e_1, e_2 oder e_4, so dass $\{e_3, e_5\}$ immer zu $\{e_1, e_3, e_5\} \in U$ oder $\{e_2, e_3, e_5\} \in U$ oder $\{e_3, e_4, e_5\} \in U$ ergänzt werden kann.
Jede 3-elementige Menge enthält e_1, e_2 oder e_3, so dass $\{e_4, e_5\}$ immer zu $\{e_1, e_4, e_5\} \in U$ oder $\{e_2, e_4, e_5\} \in U$ oder $\{e_3, e_4, e_5\} \in U$ ergänzt werden kann.
Damit liegt bei U ein Matroid vor.

Aufgabe 4:

a) Wir verwenden den Greedy-Algorithmus. Die Tatsache, dass der Greedy-Algorithmus immer das beste Ergebnis liefert, wird in Teil c) bewiesen.

Greedy-Algorithmus bedeutet hier: Wir fügen zur Wagenladung möglichst viele möglichst wertvolle Waren hinzu, wertvoll im Sinne von Wert : Gewicht.

Zuerst berechnen wir also den Wert pro Gewichtseinheit der einzelnen Waren:

Objekt	a	b	c	d	e	f
Gewicht	3	3	5	5	9	6
Wert	6	5	9	8	12	9
relativer Wert	2	$1,66\ldots$	$1,8$	$1,6$	$1,33\ldots$	$1,5$

Der Greedy-Algorithmus liefert nun folgende Werte:

$C = 17$:

	Objekt	*Gewicht*	*Wert*	*Gesamtgewicht*
1	a	3	6	3
2	c	5	9	8
3	b	3	5	11
4	d	5	8	16
5	f	1	1,5	17

$\rightarrow$ Gesamtwert ist 29,5.

$C = 26$:

	Objekt	*Gewicht*	*Wert*	*Gesamtgewicht*
1	a	3	6	3
2	c	5	9	8
3	b	3	5	11
4	d	5	8	16
5	f	6	9	22
6	e	4	$5,33\ldots$	26

$\rightarrow$ Gesamtwert ist 42,33....

b) Ein Programm sieht wie folgt aus:

1: Sortiere alle Objekte gemäß ihrem relativen Wert (w_i)pro Gewichtseinheit (g_i) und numeriere die Objekte neu, so daß $\frac{w_1}{g_1} \geq \frac{w_2}{g_2} \geq \ldots \frac{w_n}{g_n}$ gilt **;**
2: $s \leftarrow 0$
3: **für** $k = 1$ **bis** n **:**
3.1: **wenn** $s + g_k \leq C$ **dann** $a_k = 1$ **, sonst**
$a_k = (C - s)/g_k$
3.2: $s \leftarrow s + a_k g_k$

c) Angenommen, der Greedy-Algorithmus liefert nicht die optimale Lösung.

Gegeben seien die Objekte $o_1, \ldots o_n$ mit Gewichten $g_1, \ldots g_n$ und Werten $w_1, \ldots w_n$, wobei für $1 \leq k \leq n-1$ gilt $\frac{w_k}{g_k} \geq \frac{w_{k+1}}{g_{k+1}}$, sowie eine Kapazität C, so dass der Greedy-Algorithmus (GA) nicht die optimale Lösung findet.

Der GA weist den Objekten $o_1, \ldots o_n$ die Anteile $a_i = \min\{1, \max\{0, \frac{(C - \sum_{k=1}^{i-1} g_k)}{g_i}\}\}$ zu. (Wähle Anteil 1, solange Grenze nicht überschritten ist; wenn Anteil 1 über Grenze hinausgehen würde, wähle maximalen Anteil, alle weiteren Anteile sind 0.)

Die optimale Lösung weise den Objekten $o_1, \ldots o_n$ die Anteile a'_i mit $0 \leq a'_i \leq 1$ zu.

Es gilt: $\sum_{i=1}^{n} a'_i g_i = C, \sum_{i=1}^{n} a_i g_i = C$ und
$\sum_{i=1}^{n} a'_i w_i > \sum_{i=1}^{n} a_i w_i$.

Sei i die kleinste Zahl, für die $a'_i > a_i$ gilt. Damit muss $a_i \neq 1$ gelten und für $j > i$ gilt $a_j = 0$. Damit gilt für alle $j \geq i : a'_j \geq a_j$.

Es gilt nun $\sum_{j=i}^{n} (a'_j - a_j) g_j = \sum_{j=1}^{i-1} (a_j - a'_j) g_j$, wobei alle Summanden nicht negativ sind.

Somit gilt auch (unter Verwendung der Ordnung nach fallendem Wert pro Gewichtseinheit):

$\sum_{j=i}^{n} (a'_j - a_j) g_j \frac{w_j}{g_j} \leq$
$\sum_{j=i}^{n} (a'_j - a_j) g_j \frac{w_{i-1}}{g_{i-1}} =$
$\sum_{j=1}^{i-1} (a_j - a'_j) g_j \frac{w_{i-1}}{g_{i-1}} \leq$
$\sum_{j=1}^{i-1} (a_j - a'_j) g_j \frac{w_j}{g_j}$.

Daraus ergibt sich $\sum_{i=1}^{n} a'_j w_j \leq \sum_{i=1}^{n} a_j w_j$, im Widerspruch zur Vorausstzung.

Damit ist die Annahme, dass der GA eine nicht-optimale Lösung liefern kann, widerlegt $\Rightarrow$ GA liefert immert immer die optimale Lösung.