Übungsblatt 9 für "Diskrete und geometrische Algorithmen"

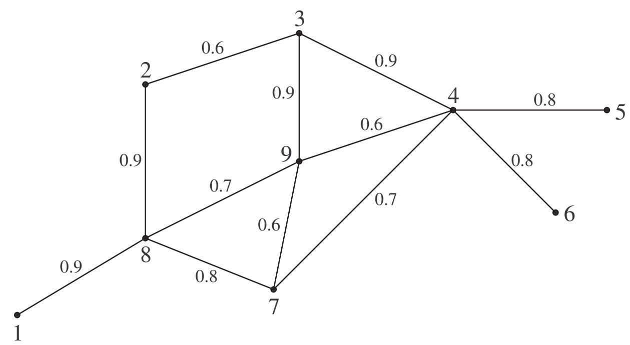
60.) Geben Sie eine (nicht ineffiziente) Variante des Bellman-Ford-Algorithmus an, die folgendes leistet:

- kann der Knoten v von s aus durch einen Pfad (also eine Folge gerichteter Kanten) erreicht werden, der einen Zyklus mit negativen Gewicht enthält, dann sei $v.d = -\infty$,
- ansonsten sei $v.d$ die minimale Distanz zu s.

61.) Gegeben sei ein gerichteter oder ungerichteter Graph $G = (V, E)$, in dem jeder Kante $(u, v) \in E$ ein reeller Wert $r(u, v)$ mit $0 \leq r(u, v) \leq 1$ zugeordnet ist, der die Zuverlässigkeit einer Datenübertragung vom Knoten u zum Knoten v darstellt. Wir interpretieren $r(u, v)$ als die Wahrscheinlichkeit, daß der Kanal von u nach v nicht versagt, und setzen voraus, daß diese Wahrscheinlichkeiten paarweise unabhängig sind.

(a) Geben Sie einen effizienten Algorithmus an, der den zuverlässigsten Kanal zwischen zwei gegebenen Knoten bestimmt.

(b) Wenden Sie diesen Algorithmus auf folgenden Graphen an um einen möglichst zuverlässigen Kommunikationskanal zwischen den Knoten 1 und 5 herzustellen, wobei die Wahrscheinlichkeiten $r(u, v)$ bei den entsprechenden Kanten angegeben sind.



62.) Eigenschaften kürzester Pfade.

(a) Sei $G = (V, E)$ ein gewichteter gerichteter Graph mit Startknoten s und sei G durch die Prozedur INITIALIZE-SINGLE-SOURCE(G, s)) (siehe VO bzw. Cormen et al.) initialisiert worden. Beweisen Sie, daß G einen Zyklus mit negativem Gewicht enthält, wenn eine Folge von Relaxationsschritten das Attribut $s.\pi$ auf einen von NIL verschiedenen Wert setzt.

(b) Sei $G = (V, E)$ ein gewichteter gerichteter Graph, der keine Zyklen mit negativem Gewicht enthält. Sei $s \in V$ der Startknoten und sei G durch die Prozedur INITIALIZE-SINGLE-SOURCE(G, s) initialisiert worden. Beweisen Sie, daß es eine Folge von $|V| - 1$ Relaxationsschritten gibt, die für alle Knoten $v \in V$ zu $v.d = \delta(s, v)$ führt.

63.) Fragen zum in der Vorlesung besprochenen "Aktivitäten-Auswahl-Problem".

(a) Nehmen Sie an, daß wir statt der zuerst endenden Aktivität stets die zuletzt beginnende Aktivität auswählen würden, die zu den bereits ausgewählten Aktivitäten kompativel ist. Erklären Sie, warum dies ein Greedy Algorithmus ist, und beweisen Sie, daß dieser Ansatz zu einer optimalen Lösung führt.

(b) Nicht jeder Greedy-Ansatz für das Aktivitäten-Auswahl-Problem erzeugt eine maximal große Menge paarweise zueinander kompatibler Aktivitäten. Geben Sie ein Beispiel dafür an, daß die Methode, die Aktivität mit der kleinsten Dauer unter den Aktivitäten auszuwählen, die kompatibel zu den vorher ausgewählten Aktivitäten ist, nicht immer zu einer optimalen Lösung führt. Geben Sie ebenso ein Beispiel dafür an, daß der Ansatz, immer eine verbliebene Aktivität mit der frühesten Startzeit, die kompatibel zu den vorher ausgewählten Aktivitäten ist, auszuwählen, nicht immer zu einer optimalen Lösung führt.

64.) Das "kontinuierliche Rucksack-Problem". Sie haben gewonnen! Und zwar dürfen Sie sich von n Gegenständen (welche Sie beliebig portionieren können; denken Sie z.B. an "Goldstaub", etc.), wobei der i-te Gegenstand v_i Euro wert ist und w_i Kilo wiegt, so viel nehmen und in Ihren Rucksack hineinpacken, so viel Sie tragen können; wir nehmen an, daß Sie höchstens W Kilo tragen können. Wir nehmen dabei weiters immer an, daß v_i, w_i und W positive ganze Zahlen sind. Die Aufgabe beim "kontinuierlichen Rucksack-Problem" besteht nun darin, anzugeben, wie viel Sie unter Beachtung der Gewichtsbeschränkung (es hilft Ihnen niemand beim Tragen!) von jedem Gegenstand mitnehmen sollen, sodaß der Gesamtwert der eingepackten Gegenstände möglichst groß ist.

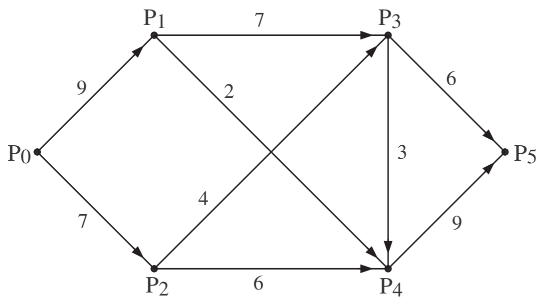
Zeigen Sie, daß die folgende Greedy-Strategie zu einer optimalen Lösung des Problems führt: Berechne zuerst den Wert pro Kilo v_i/w_i jedes Gegenstandes und sortiere die Gegenstände absteigend entsprechend diesem "Relativwert". Nehmen Sie nun in jedem Schritt so viel wie möglich von dem noch verfügbaren Gegenstand mit dem größten "Relativwert" bis Sie entweder alle Gegenstände aufgebraucht haben oder Sie nicht mehr tragen können, d.h. Ihr Gewichtslimit W erreicht haben.

65.) Betrachten Sie das Problem, Wechselgeld für n Cent so zusammenzustellen, daß so wenige Münzen wie möglich verwendet werden.

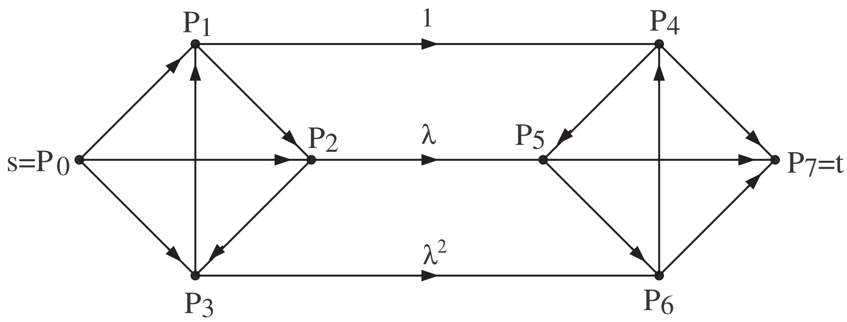
(a) Nehmen wir zunächst an, daß die verfügbaren Münzen Nennwerte haben, die Potenzen von einer natürlichen Zahl $q > 1$ sind, genauer: die Nennwerte der verfügbaren Münzen seien gegeben durch $1 = q^0, q^1, q^2, \ldots, q^k$, für ein $k \geq 1$. Zeigen Sie, daß ein naheliegender Greedy-Algorithmus ("in jedem Schritt nehme man die Münze mit größtmöglichem Nennwert hinzu") zu einer optimalen Lösung führt.

(b) Man gebe eine Menge von Münzennennwerten an, für die ein Greedy-Algorithmus nicht immer eine optimale Lösung liefert. Ihre Menge sollte die 1 Cent Münze enthalten, sodaß es für jeden Wert von n eine Lösung gibt.

66.) Man erkläre die Ford-Fulkerson-Methode zum Lösen des "Maximalen Fluss-Problems" an Hand des Auffinden eines maximalen Flusses vom Knoten P_0 zum Knoten P_5 im nachfolgend angegebenen Flussnetzwerk. Weiters gebe man auch einen minimalen Schnitt in diesem Netzwerk an.



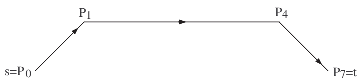
67.) Sei λ die positive Lösung der Gleichung $\lambda^2 + \lambda - 1 = 0$. Betrachten Sie nun das folgende Flussnetzwerk.



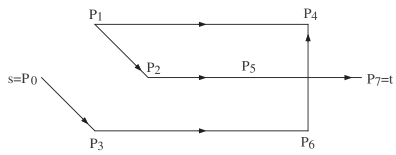
Alle unmarkierten Kanten haben die Kapazität $\lambda + 2$. Zeigen Sie mittels "max flow-min cut"-Theorems, daß es einen maximalen Fluß mit Wert 2 gibt.

Man betrachte nun die angegebene Folge Q_i von Pfaden für $i \geq 0$:

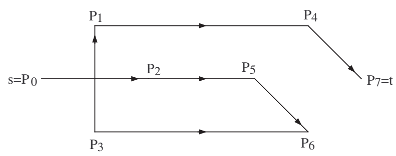
$Q_0 =$



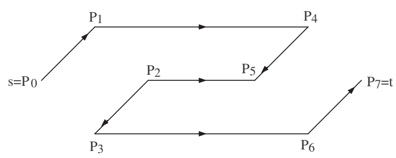
$Q_i =$



für $i = 3k + 1$,



für $i = 3k + 2$, mit $k \geq 0$.



für $i = 3k + 3$,

Man zeige, daß diese Pfade tatsächlich eine Folge von Erweiterungspfaden in den entsprechenden Restnetzwerken darstellen, sodaß die durch Erhöhung entlang Q_i gewonnenen Flüsse zwar gegen den Wert 2 konvergieren, diesen Wert jedoch nicht annehmen. Sie können dabei für den Nachweis die Beziehungen verwenden:

$$\lambda + 2 = \frac{1}{1-\lambda} = \sum_{i=0}^{\infty} \lambda^i \quad \text{und} \quad \lambda^{i+2} = \lambda^i - \lambda^{i+1} \text{ für alle } i \geq 0.$$