Prof. G. Zachmann
David Mainzer

TU Clausthal
Institut für Informatik
15. Juni 2010

Sommersemester 2010

Übungen zu Informatik II - Blatt 10

Abgabe in der Übung am 22. 06 / 23. 06. 2010

Bitte beachten Sie, dass die Programmieraufgaben von Ihnen in der Übung vorgeführt und erklärt werden müssen. Zusätzlich senden Sie die Lösung, unter Angabe ihres Namens, an **dm@tu-clausthal.de**.

Aufgabe 1 (Greedy-Algorithmen, *2+3+2 Punkte*)

Mit Ihrem neuen Erdgas-Auto planen Sie einen Ausflug über die Alpen zum Schiefen Turm von Pisa. Einerseits wollen Sie auf der Fahrt so selten wie möglich zum Tanken anhalten, andererseits müssen Sie beachten, dass bisher nur ausgewählte Tankstellen Erdgas anbieten. Mit einem vollem Tank können Sie maximal D Kilometer fahren. Sie starten mit vollem Tank in Clausthal-Zellerfeld (E_0) und fahren bis nach Pisa (E_n). Unterwegs kommen Sie der Reihe nach an den Erdgas-Tankstellen $E_1, E_2, \ldots, E_{n-1}$ vorbei. Der Abstand zwischen E_{i-1} und E_i beträgt d_i Kilometer (mit $i = 1, \ldots, n$), wobei jeweils $d_i \leq D$ gilt.

a) Geben Sie eine Greedy-Strategie für dieses Problem an.

b) Implementieren Sie ihre Strategie in Python, so dass Ihr Programm die minimale Anzahl an Tank-Stops ermittelt und eine Liste der anzufahrenden Tankstellen zurück gibt (die Position der Tankstelle im Array ist hier ausreichend). Implementieren Sie ihren Ansatz so, dass beim Aufruf `array_Tankstellen = IHRE_STRATEGIE ( D,d )` ein Array mit den anzufahrenden Tankstellen ausgegeben wird.

```
# Beispiel:
#
# Reichweite mit einer Tankfüllung
D = 400
# Bsp. Array für Abstände der Tankstellen, beachten Sie, dass
# für jeden Wert im Array d < D gelten muss
d = [100,20,304,50,43,34,2,34,124,45,232,45,90]
meine_Tankstops = berechnet ( D,d )
# Lsg. --> meine_Tankstops = [1,4,9]
```

c) Warum hat Ihre Greedy-Strategie die Greedy-Choice-Eigenschaft, d.h., warum liefert sie eine optimale Lösung?

Aufgabe 2 (Greedy Pfadplanung, *5 Punkte*)

Beschreiben Sie einen Algorithmus als Pseudo-Code, um folgendes Problem zu lösen. Gegeben sind eine Menge achsenparallele Rechtecke in der Ebene als Hindernisse, die sich nicht gegenseitig überlappen und nicht berühren. Außerdem ist ein Startpunkt (grün) und ein Endpunkt (rot) gegeben, die sich nicht in einem der Rechtecke befinden.

Die Aufgabe ist, einen möglichst kurzen Weg zu finden, der nicht durch die Hindernisse geht. (Der Weg darf auf dem Rand der Vierecke verlaufen, quasi “dicht an der Wand” entlang.)

Sie können davon ausgehen, dass eine Routine zur Berechnung des nähesten Schnittpunktes zwischen einem Geradensegment und einer Menge von achsenparallelen Rechtecken vorhanden ist (diese Routine liefert ggf. auch die Antwort “kein Schnitt”). Außerdem liefert diese Routine die Seite des Vierecks, auf dem dieser Schnittpunkt liegt. Sie können auch davon ausgehen, dass Sie zu einer Viereckseite die beiden Eckpunkte bekommen können, und alle Informationen, die Sie sonst noch über Vierecke benötigen.