Technische Universität München
Institut für Informatik I-16
Dr. Tobias Lasser
Jakob Vogel

WiSe 2012/13
Übungsblatt 9
20. Dezember 2012

Algorithmen und Datenstrukturen

Aufgabe 1 (P) Traveling Salesman

In der Vorlesung wurde das *Traveling Salesman Problem* (TSP) vorgestellt, und eine Lösung über das *Backtracking*-Entwurfsschema präsentiert. Dabei wurde eine Teillösung immer um einen Schritt erweitert, und dann die Gesamtmenge aller gültigen Rundreisen nach der kürzesten durchsucht.

In dieser Aufgabe soll nun die Komplexität des Verfahrens untersucht werden.

a) Implementieren Sie eine Methode, die eine Adjazenzmatrix für einen gewichteten, ungerichteten, vollständigen Graphen mit gegebener Knotenzahl zufällig erzeugt.

b) Setzen Sie den in der Vorlesung gegebenen Code in C++ um, und berechnen Sie die Menge aller Lösungen.

c) Wie groß darf der Graph (bzw. die Adjazenzmatrix) werden, so dass die Lösung in unter einer Minute berechenbar ist?

Aufgabe 2 (P) Min-Heap

Eine weitere Datenstruktur, die in der Vorlesung besprochen wurde, ist der *Min-Heap*. Es handelt sich dabei um einen fast vollständigen Binärbaum, bei dem die Knoten jeweils einen speziellen Wert zugeordnet bekommen. Die geforderte *Min-Heap-Eigenschaft* ist dabei, dass der Wert des Vaterknotens stets kleiner (oder gleich) im Vergleich zu den Werten seiner Kinder zu sein hat.

Implementieren Sie nun einen Min-Heap in C++ unter Verwendung eines Arrays bzw. eines `std::vector<...>`.

Aufgabe 3 (P) Min-Priority-Queue

Mittels des in der vorherigen Aufgabe besprochenen Min-Heaps lässt sich eine Prioritäts-Warteschlange realisieren. Es handelt sich dabei um eine Queue, die nicht nach FI-FO-Prinzip arbeitet, sondern, wo eine den Elementen zugeordnete Priorität die Reihenfolge des Auslesens bestimmt.

In einer späteren Aufgabe benötigen wir eine *Min-Priority-Queue*, bei der – wir werden kürzere Teilpfade bevorzugen – Elemente kleineren Werts vorgezogen werden sollen. Verwenden Sie also oben implementierten Min-Heap zur Erstellung einer derartigen Warteschlange!

Aufgabe 4 (P) KMP-Suche in Zeichenketten

Oft stellt sich das Problem, eine kürzere Zeichenkette innerhalb einer längeren Zeichenkette zu finden. Eine mögliche Lösung dafür ist der KMP-Algorithmus, wie er in der Vorlesung besprochen wurde.

Implementieren Sie die KMP-Stringsuche und vergegenwärtigen Sie sich die Abläufe!