Frank Hoffmann
Klaus Kriegel
Romain Grunert
Ludmila Scharf
André Schulz

Ws 2007/08

16. November 2007
Abgabe: 26. November 2007
vor der Vorlesung

5. Übung zur Vorlesung Höhere Algorithmik

1. Aufgabe (5 Punkte).
Gegeben sei ein quadratisches $n \times n$-Gitter $G = (V, E)$, also $V = \{(i, j) \mid 1 \leq i, j \leq n\}$ und $E = \{\{(i, j), (i', j')\} \mid |i - i'| + |j - j'| = 1\}$, mit Zahlen als Knotenbeschriftung. Nehmen Sie an, dass alle Zahlen verschieden sind. Ein lokales Minimum ist (vgl. 1. Aufgabe vom 4. Übungsblatt) ein Knoten, dessen Knotenbeschriftung kleiner ist als die seiner Nachbarn.

Entwickeln Sie einen Algorithmus, der ein lokales Minimum in Laufzeit $O(n)$ findet. Begründen Sie, warum Ihr Algorithmus korrekt ist und die gewünschte Laufzeit erreicht.

Hinweis: Überlegen Sie sich, wie man für eine Spalte (Zeile) entscheiden kann, ob sich ein lokales Minimum darauf bzw. links oder rechts davon (darunter oder darüber) befindet.

2. Aufgabe (5 Punkte).
Betrachten Sie ein reguläres Achteck mit Seitenlänge 1. Geben Sie eine optimale Triangulierung und ihre Länge (Summe der Kantenlängen, jede Kante nur einmal gezählt) an.

3. Aufgabe (5 Punkte).
In der Vorlesung haben Sie den Algorithmus von Floyd und Warshall kennengelernt.

(a) Zeigen Sie, dass der Algorithmus mit Speicherplatz der Größenordnung $O(n^2)$ auskommt, wobei n die Anzahl der Knoten des betrachteten Graphen ist.

(b) Ein negativer Zyklus ist ein Kreis als Untergraph des betrachteten Graphen, dessen Kantengewichte eine negative Summe haben. Wie kann man den Algorithmus dazu benutzen, negative Zyklen zu finden?

(c) Wie kann der Algorithmus dazu verwendet werden, die transitive Hülle eines Graphen zu berechnen?

4. Aufgabe (5 Punkte).
Entwickeln Sie einen Algorithmus, der den längsten Weg in einem gerichteten azyklischen Graphen findet. Begründen Sie die Korrektheit Ihres Algorithmus und analysieren Sie seine Laufzeit.

Hinweis: Topologisches Sortieren (vgl. 3. Aufgabe vom 1. Übungsblatt)

5. Aufgabe (5 Punkte).
Ein Graph $G = (V, E)$ heißt Pfad, wenn seine Knoten als $v_1, \ldots, v_n$ geschrieben werden

können, so dass genau dann eine Kante zwischen v_i und v_j existiert, wenn $|i-j|=1$ gilt. Jedem Knoten in v_i sei eine positive ganze Zahl w_i als Gewicht zugeordnet. Gesucht ist eine unabhängige Menge maximalen Gewichts.

(a) Zeigen Sie anhand von Gegenbeispielen, dass die folgenden zwei Algorithmen nicht immer eine unabhängige Menge maximalen Gewichts finden.

Algorithmus 1

```
\\ Der ``heaviest-first'' Greedy-Algorithmus
Setze S = ∅
While G ≠ ∅
Wähle Knoten v_i maximalen Gewichts
Füge v_i zu S hinzu
Lösche v_i und seine Nachbarn aus G
Endwhile
Return S
```

Algorithmus 2

```
Sei S_1 die Menge aller v_i mit i ungerade
Sei S_2 die Menge aller v_i mit i gerade
\\ Beachte, dass S_1 und S_2 unabhängige Mengen sind.
Bestimme, welche der Mengen S_1 und S_2 größeres Gewicht hat, und
gib diese aus.
```

(b) Geben Sie einen Algorithmus an, der als Eingabe Pfade mit Knotengewichten akzeptiert und eine unabhängige Menge maximalen Gewichts ausgibt. Die Laufzeit des Algorithmus sollte unabhängig von den Gewichten polynomiell in der Anzahl der Knoten sein. Begründen Sie die Korrektheit Ihres Algorithmus und warum die Laufzeit polynomiell ist.