4. Aufgabenblatt zur Vorlesung

Entwurf und Analyse von Algorithmen

(Günter Rote, Astrid Kaffanke, Christian Knauer)

Abgabe: 16. November 2000

1. **Minimieren der maximalen Kantenlänge** (3 Punkte).

   Für einen Pfad P in einem ungerichteten, gewichteten Graphen $G = (V, E)$ mit Gewichtsfunktion w sei $\tilde{\mu}(P) = \max\{w(e) \mid e \in P\}$ die Länge der längsten Kante auf diesem Pfad. Für zwei Knoten $u, v \in V$ sei $\mu(u, v) = \min\{\tilde{\mu}(P) \mid P \text{ ist Pfad von } u \text{ nach } v\}$ die minimale Länge der längsten Kante auf Pfaden von u nach v. Gesucht ist für alle Knoten v der Wert $\mu(s, v)$.

   Welcher Halbring liegt dieser Problemformulierung zu Grunde? Zeige, dass das wirklich ein Halbring ist.

2. **Geld verdienen** (5 Punkte).

   Arbitrage ist ein „Ausnutzen der Kursunterschiede an verschiedenen Börsen“ [Quelle: Duden, Die Rechtschreibung, Jubiläumsausgabe von 1980]. Damit kann auf geschickte Weise aus einer Währungseinheit mehr als eine Einheit der selben Währung gemacht werden.

   Beispiel: Für einen US–Dollar bekommt man 0.7 britische Pfund, für ein britisches Pfund bekommt man 9.5 französische Franc und für einen französischen Franc bekommt man 0.16 US–Dollar. Dann kann man auf diese Weise aus einem Dollar $0.7 \times 9.5 \times 0.16 = 1.064$ Dollar machen.

   Gegeben seien n Währungen $c_1, c_2, \ldots, c_n$ und eine $n \times n$ Tabelle R mit den Wechselkursen, d.h. für eine Einheit von Währung c_i kann man $R[i, j]$ Einheiten von Währung c_j kaufen.

   Gib einen effizienten Algorithmus an, der herausfindet, ob es eine Folge von Währungen $c_{i_1}, \ldots, c_{i_k}$ gibt, so dass

   $$R[i_1, i_2] \cdot R[i_2, i_3] \cdot \ldots \cdot R[i_{k-1}, i_k] \cdot R[i_k, i_1] > 1$$

   ist. Analysiere die Laufzeit des Algorithmus.

   Zusatzaufgabe (*1000 Zusatzpunkte*).

   Gib einen Algorithmus an, der in Zeit $O(n^k)$ für ein festes k den Kreis findet, der am meisten Gewinn bringt.

3. **Halbringe** (3 Punkte)

   Es sei die Menge $M = \{(a, b) \mid a \leq b \leq \infty\}$ mit den Operationen

   $(a, b) \oplus (c, d) = \min_2(a, b, c, d)$ und $(a, b) \otimes (c, d) = \min_2(a + c, a + d, b + c, b + d)$.

   gegeben. Dabei sei $\min_2(A)$ das Paar, das aus den beiden kleinsten Elementen von A besteht.

Zeige: $(M, \oplus, \otimes)$ ist ein Halbring mit Nullelement (∞, ∞). Was ist in diesem Halbring das Einselement?

Wofür ist dieser Halbring gut, d.h. welches Problem kann damit modelliert werden?

4. **Inkrementelles dynamisches Programmieren** (5 Punkte)

   Der Algorithmus von *Dantzig* zum Finden kürzester Wege in einem gewichteten Graphen mit n Knoten geht wie folgt:

   Für $k = 1 \dots n$

   Berechne kürzesten Wege zwischen allen Knotenpaaren im Graphen mit den Knoten $\{v_1, \dots, v_k\}$.

   Erkläre ausführlich, wie dieser Algorithmus arbeitet, zeige seine Korrektheit und schätze die Laufzeit ab.

5. **Implementierung von Graphalgorithmen**[1] (10 Punkte).

   Praktische Aufgabe! Abgabe bis zum 23.11.2000

   Implementiere abstrakte Datentypen `Graph` und `DiGraph`, die gerichtete bzw. ungerichtete schleifenlose Graphen auf n Knoten realisieren.

   Die interne Darstellung der Graphen (Adjazenzlisten, Adjazenzmatrix) sollte vom Benutzer frei wählbar sein; insbesondere sollten Methoden zur Umwandlung der Darstellungen untereinander zur Verfügung stehen.

   Überlege Dir, welche *elementaren* Methoden für diese Datentypen noch benötigt werden und implementiere diese.

   Schreibe eine Methode `Random_Graph` mit vier Parametern n, $0 \leq p \leq 1$, $a \leq b$, die einen zufälligen gerichteten Graphen auf n Knoten erzeugt. Dabei soll gelten: Ein Punktepaar ist eine Kante mit Wahrscheinlichkeit p und eine Nichtkante mit Wahrscheinlichkeit $1 - p$, mit zufälligem Kantengewicht aus dem Intervall $[a, b]$ (gleichverteilt).

   Teste Deine Implementierung, indem Du für einige Werte von n und p zufällige Graphen erzeugst, und dann die Anzahl der Kanten dieser Graphen tabellierst; die Kantenzahl sollte nicht zu sehr vom Erwartungswert abweichen!

   Schreibe außerdem eine Methode die testet, ob alle Knoten von einem vorgegebenen Knoten s aus erreichbar sind. Finde durch testen heraus, wie groß die Wahrscheinlichkeit p für die Kanten sein muss, damit „fast immer" *alle* Knoten erreichbar sind.

6. **Zuverlässige Pfade in Netzwerken** (freiwillig).

   Gegeben sei ein gerichteter Graph $G = (V, E)$, in dem jede Kante $(u, v) \in E$ einen Wert $0 \leq r(u, v) \leq 1$ hat; diese Zahl stellt die Zuverlässigkeit einer Verbindung vom Knoten u zum Knoten v dar. Im folgenden wird $r(u, v)$ als die Wahrscheinlichkeit, dass die Verbindung von u nach v nicht ausfällt, interpretiert; diese Wahrscheinlichkeiten seien unabhängig. Geben Sie einen effizienten Algorithmus an, der den zuverlässigsten Pfad zwischen zwei gegebenen Knoten findet.

   Welcher Halbring liegt dieser Problemformulierung zu Grunde?

   Hinweis. Für zwei unabhängige Ereignisse A und B gilt: $Prob(A \text{ und } B) = Prob(A) \cdot Prob(B)$.

[1] Hinweise zur Abgabe von Implementierungen im Rahmen der praktischen Übungsaufgaben siehe zweites Übungsblatt.