12. Übungsblatt zu Grundzüge von Algorithmen und Datenstrukturen, WS 15/16

Prof. Markus Bläser
http://www-cc.cs.uni-saarland.de/course/50/

Abgabe: 4. Februar 2016, 12:00 Uhr

Aufgabe 12.1 Naive Matrizenmultiplikation multipliziert zwei $(n \times n)$-Matrizen A, B, indem für alle n^2 Einträge der Produktmatrix $A \cdot B$ ein Zeilenvektor von A mit einem Spaltenvektor von B multipliziert wird. Dieses Verfahren hat offensichtlich Laufzeit $\Theta(n^3)$. (In dieser Aufgabe sei die Laufzeit eines Algorithmus stets die Anzahl der Additionen und/oder Multiplikationen, die der Algorithmus ausführt.) Wir suchen ein schnelleres Verfahren und nutzen dazu Divide-and-Conquer. Zunächst nehmen wir an, dass n eine Zweierpotenz ist. Wir zerlegen die zwei zu multiplizierenden $(n \times n)$-Matrizen A, B in 8 $(\frac{n}{2} \times \frac{n}{2})$-Matrizen a, b, c, d, e, f, g, h, so dass folgendes Produkt zu berechnen ist:

$$\underbrace{\begin{pmatrix} a & b \\ c & d \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} e & f \\ g & h \end{pmatrix}}_{B} = \underbrace{\begin{pmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{pmatrix}}_{C}. \tag{1}$$

Nun kann man $A \cdot B$ berechnen, indem zunächst rekursiv die 8 Produkte ae, bg, af, bh, ce, dg, cf und dh berechnet und diese dann entsprechend (1) zu $A \cdot B$ zusammengesetzt werden.

(a) Stellen Sie eine Rekursiongleichung für die Anzahl der benötigten Rechenoperationen des oben skizzierten Divide-and-Conquer-Algorithmus auf und lösen Sie diese. Vergleichen Sie die Laufzeit dieses Verfahrens mit der Laufzeit der naiven Matrizenmultiplikation.

(b) Zeigen Sie, wie die Einträge der Matrix C durch geschickte Addition/Subtraktion folgender 7 Produkte berechnet werden können. Beachten Sie dabei, dass Matrizenmultiplikation nicht kommutativ ist.

$$\begin{array}{cccc} a(f-h) & (a+b)h & (c+d)e & d(g-e) \\ (a+d)(e+h) & (b-d)(g+h) & (a-c)(e+f) & \end{array}$$

(c) Welche Laufzeit hat ein rekursives Verfahren zur Matrizenmultiplikation, wenn entsprechend (b) vorgegangen wird? Leiten Sie dazu eine Rekursionsgleichung für die Laufzeit her.

(d) Erweitern Sie das Verfahren auf beliebige n. Geben Sie wieder eine Rekursionsgleichung für die Laufzeit an und bestimmen Sie das asymptotische Wachstum der Laufzeit.

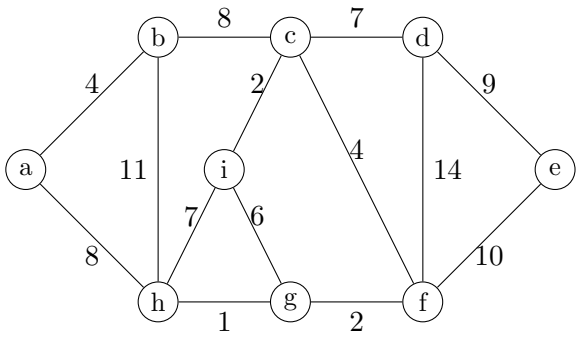

Abbildung 1:

Aufgabe 12.2 Führen Sie Kruskals Algorithmus und Prims Algorithmus jeweils auf dem Graphen in Abbildung 1 aus. Fangen Sie bei Prims Algorithmus am Knoten a an.

Aufgabe 12.3 a) Sei G ein gewichteter, zusammenhängender, ungerichteter Graph, so dass alle Kantengewichte paarweise verschieden sind. Zeigen Sie, dass es dann genau einen minimalen Spannbaum in G gibt.

b) Zeigen Sie: Für jeden minimalen Spannbaum T in einem Graphen gibt es eine Sortierung der Kanten, so dass Kruskals Algorithmus mit dieser Sortierung T berechnet. (Die Kantengewichte müssen nicht unbedingt verschieden sein!)