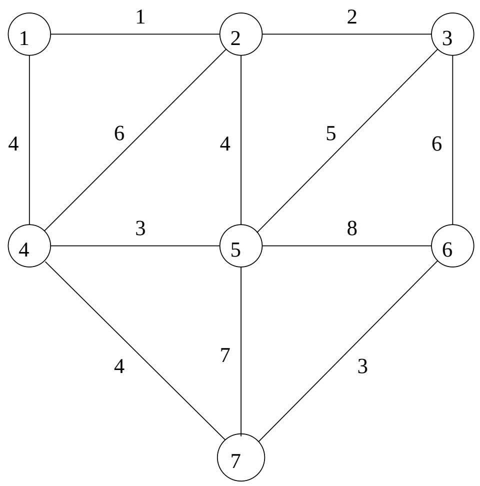
Entwurf und Analyse von Algorithmen

WS 1999/2000
Prof. Dr. K. Madlener
Übungsblatt 6

Aufgabe 6.1. Wir betrachten noch einmal das Problem der Konstruktion eines minimalen aufspannenden Baumes. Eine Menge von Kanten eines Graphen, die keinen Kreis enthält, heiße zulässig. Weiter nennen wir eine zulässige Kantenmenge vielversprechend, falls sie so erweitert werden kann, dass eine optimale Lösung (d. h. die Kantenmenge eines minimalen aufspannenden Baumes) durch Hinzunahme weiterer Kanten nach wie vor möglich ist. Es ist klar, dass die leere Kantenmenge vielversprechend ist. Auch die Menge der Kanten eines minimalen aufspannenden Baumes selbst ist vielversprechend: Man nimmt einfach keine Kanten mehr hinzu (triviale Erweiterung). Noch eine Bezeichnung: Eine Kante verlässt eine Knotenmenge, wenn genau ein Endknoten der Kante in der Knotenmenge liegt.

Sei nun $G = (V, E)$ ein zusammenhängender ungerichteter Graph mit (positiv) gewichteten Kanten. Sei $B \subset V$ eine echte Knotenteilmenge und $F \subseteq E$ eine vielversprechende Kantenmenge, so dass keine Kante in F die Knotenmenge B verlässt. Sei e die kürzeste Kante in E, die B verlässt (oder eine der kürzesten, wenn es mehrere solche gibt). Zeigen Sie, dass dann auch $F \cup \{e\}$ vielversprechend ist.

Geben Sie einen Greedy-Algorithmus zur Berechnung (der Kantenmenge) eines minimalen aufspannenden Baumes an, der diese Tatsache wesentlich verwendet. (Dies ist der Algorithmus von Prim.) Begründen Sie die Korrektheit Ihres Algorithmus und demonstrieren Sie den Ablauf an folgendem Graphen:



Die Knotennummerierung dient nur der Bezeichnung.

Aufgabe 6.2. Gegeben seien n sortierte Listen $L_1, \ldots, L_n$. Wir wollen nun durch wiederholtes paarweises Mischen letztendlich eine einzige Liste gewinnen. Dabei kommt es für die Gesamtanzahl der nötigen Vergleiche auf die Reihenfolge an: Überzeugen Sie sich davon anhand des Beispiels, dass L_1 aus 30, L_2 aus 20 und L_3 aus 10 Elementen bestehe.

Geben Sie einen Greedy-Algorithmus an, der die optimale Mischreihenfolge für n gegebene sortierte Listen (nur die Längen der Listen sind relevante Eingabegrößen) findet und begründen Sie, warum Ihr Algorithmus korrekt ist. Prüfen Sie Ihren Algorithmus an folgendem Beispiel: Gegeben seien sechs Listen der Längen 2, 3, 5, 7, 9 und 13.

Der Algorithmus, den Sie angegeben haben, sollte im schlechtesten Fall eine Laufzeit in $O(n^2)$ haben; zeigen Sie dies. Finden Sie dann eine Variante des Algorithmus, der im schlechtesten Fall eine Laufzeit in $O(n \log n)$ hat.

Aufgabe 6.3. Sei $(E,\mathcal{U})$ ein Matroid. Wir nennen ein Element $x \notin A$ eine Erweiterung von $A \in \mathcal{U}$, falls $A \cup \{x\} \in \mathcal{U}$. Zeigen Sie: Ist $x \in E$ keine Erweiterung von $\emptyset$, dann ist x auch keine Erweiterung irgendeiner Menge $A \in \mathcal{U}$.

Sei nunmehr x das erste Element von E, das der kanonische Greedy-Algorithmus für das gewichtete Matroid $(E,\mathcal{U})$ auswähle. Das nun verbleibende Problem, eine maximale Teilmenge aus $\mathcal{U}$ optimalen Gewichtes, die x enthält, zu finden, reduziert sich nun auf das Problem, eine maximale Teilmenge aus $\mathcal{U}'$ optimalen Gewichtes für das Matroid $(E',\mathcal{U}')$ zu finden, wobei

$$\begin{aligned} E' &= \{y \in E \,:\, \{x,y\} \in \mathcal{U}\} \\ \mathcal{U}' &= \{B \subseteq E \setminus \{x\} \,:\, B \cup \{x\} \in \mathcal{U}\}, \end{aligned}$$

und die Gewichtsfunktion für $(E',\mathcal{U}')$ die auf E' eingeschränkte Gewichtsfunktion für $(E,\mathcal{U})$ sei. Zeigen Sie dies.

Aufgabe 6.4. Betrachten Sie folgenden Algorithmus:

function COVER(V, E)
begin
  $U := \emptyset$; $E' := E$
  repeat
    Sei $v \in V$ ein Knoten maximalen Grades
    $U := U \cup \{v\}$; $V := V \setminus \{v\}$
    $E' := E' \setminus \{(u, w) \,:\, u = v \text{ oder } w = v\}$
  until $E' = \emptyset$
  return U
end

Berechnet dieser Algorithmus für einen ungerichteten Graphen $G = (V, E)$ stets eine Knotenüberdeckung (zur Definition siehe Blatt 2)? Ist sie minimal (d. h. enthält sie so wenig Knoten wie möglich)? Beweisen Sie jeweils Ihre Behauptung.

Informationen zur Vorlesung:
http://www-madlener.informatik.uni-kl.de/ag-madlener/lectures/eaa/ws1999-2000/eaa.html