Linköpings Universitet
Institutionen för datavetenskap
Peter Jonsson

TDDA32 Konstruktion och analys av algoritmer

Tentamen onsdagen den 20:e oktober 1999, kl 14.00–18.00.

Hjälpmedel: Kursboken "Introduction to Algorithms" (Cormen, Leiserson & Rivest).

Poäng: Totalt kan 36 poäng erhållas. För godkänt krävs ca 14 poäng.

Jourhavande lärare: Peter Jonsson, tel 28 24 15.

Allmänt: Skriv läsligt. Onödigt komplicerade lösningar kan leda till poängavdrag.

Uppgifterna i tentamen är inte ordnade efter svårighetsgrad.

Lycka till!

Peter

Uppgift 1. (5p)
En *hamiltonväg* i en oriktad graf $G = \langle V, E \rangle$ är en väg som passerar varje nod *exakt* en gång. Att avgöra om en godtycklig graf innehåller en hamiltonväg är (inte helt överraskande) ett NP-fullständigt problem.

Vi säger att en graf G är av *grad 2* om och endast om varje nod har högst två grannar. Konstruera en polynomisk algoritm som avgör om det finns en hamiltonväg i en given graf av grad 2. Ange din algoritms tidskomplexitet så noggrant som möjligt.

Uppgift 2. (5p)
Visa att följande problem är NP-fullständigt:

Instans: En samling $C = \{c_1, \ldots, c_n\}$ där varje c_i är en ändlig mängd samt ett heltal K sådant att $K \leq n$.

Fråga: Finns det K eller fler parvis disjunkta mängder i C?

Som ett exempel, betrakta $C = \{\{a, b\}, \{b, c\}, \{c, d\}, \{e, f\}\}$. Här ser vi att mängderna $\{a, b\}$, $\{c, d\}$ och $\{e, f\}$ är parvis disjunkta (tag två mängder vilka som helst och notera att deras snitt är tomt). Dock är inte alla mängder i C parvis disjunkta ($\{a, b\} \cap \{b, c\} = \{b\}$). Alltså har instansen C, K en lösning för alla $K \leq 3$ men den saknar lösning för $K = 4$.

Uppgift 3. (5p)
Antag att A är en algoritm som korrekt löser följande problem i $O(1)$ tid[1]:
Givet en mängd heltal S och ett heltal k så svarar algoritmen "ja" om och endast om det finns en delmängd $S' \subseteq S$ sådan att summan av talen i S' är lika med k.

Använd A för att konstruera en algoritm B som inte bara svarar "ja" eller "nej" på föregående problem, utan dessutom returnerar delmängden S' om den existerar. Algoritmen B får anropa A högst $O(n)$ gånger, där n är antalet tal i S. Vidare får den totala tidskomplexiteten för B högst vara $O(n)$.

[1] En sådan algoritm existerar naturligtvis inte. Den här typen av hypotetiska algoritmer brukar man kalla "orakel" och de spelar en av huvudrollerna i komplexitetsteori.

Uppgift 4. (5p)
En *triangel* i en graf är en cykel med längden 3. Konstruera en algoritm som givet en grafs närhetsmatris kontrollerar om grafen innehåller en triangel. Algoritmen ska gå *strikt* snabbare än $O(n^3)$ där n är antalet noder i n. (M.a.o. algoritmen skall gå i $O(n^{3-\epsilon})$ tid för något fixt $\epsilon > 0$).

Uppgift 5. (8p)
Låt $L = l_1, l_2, \ldots, l_n$ vara en sekvens innehållande n distinkta heltal (d.v.s. alla talen i L är olika). Konstruera en algoritm som hittar den *längsta ökande delsekvensen* i L. En sådan sekvens är helt enkelt ett urval av element $l_{i_1}, l_{i_2}, \ldots, l_{i_k}$ ur L med följande två egenskaper:

- $i_1 < i_2 < \ldots < i_k$
- $l_{i_1} < l_{i_2} \ldots l_{i_k}$.

Betrakta exempelvis sekvensen 11, 17, 5, 8, 6, 4, 7, 12, 3. Den längsta ökande delsekvensen är då 5, 6, 7, 12. Algoritmen får högst ha tidskomplexitet $O(n^2)$.

Uppgift 6. (8p)
Om en nod med tillhörande bågar plockas bort från ett träd så får man en samling av ett, två eller flera träd (d.v.s en skog). Konstruera en algoritm som tar ett godtyckligt träd T (innehållande n noder) som indata och hittar en nod v med följande egenskap:

Om v tas bort från T så kommer varje återstående delträd att innehålla maximalt $n/2$ noder.

För full poäng krävs att algoritmens tidskomplexitet är högst $O(n)$.