Linköpings Universitet
Institutionen för datavetenskap
Peter Jonsson

TDDD20/TDDA32 Konstruktion och analys av algoritmer

Tentamen onsdagen den 22:e oktober 2008, kl 8.00–13.00.

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

Poäng: Totalt kan 40 poäng erhållas. För godkänt krävs ca 16 poäng.

Jourhavande lärare: Peter Jonsson, tel 28 24 15 eller 070 773 43 89.

Allmänt: Skriv läsligt. Onödigt komplicerade lösningar kan leda till poängavdrag. Bristfälligt motiverade lösningar leder ofelbart till poängavdrag.

Uppgifterna i tentamen är inte ordnade efter svårighetsgrad.

Lycka till!

Peter

Uppgift 1. (8p)
Maximeringsproblemet MAX CUT definieras så här:

Indata: Oriktad graf $G = (V, E)$;
Lösning: Två mängder V_1, V_2 sådana att $V = V_1 \cup V_2$;
Mått: Antal bågar som har en ändpunkt i V_1 och en ändpunkt i V_2.

Konstruera en polynomisk algoritm som approximerar MAX CUT inom 2, d.v.s. hittar en lösning som har ett mått på minst 50% av den optimala lösningen. Algoritmen får vara randomiserad och då är kravet att den *förväntade* storleken på lösningen är 50% av optimum.

Uppgift 2. (8p)
Att lösa linjära ekvationssystem (d.v.s. givet en rationell $m \times n$-matris A och en rationell m-vektor b hitta en rationell n-vektor x sådan att $Ax = b$) är ett problem som är lösbart i polynomisk tid. Visa att problemet blir NP-hårt[1] om man tillåter sig att använda *kvadratiska* ekvationer; d.v.s. ekvationer där vissa av termerna kan vara kvadrerade. Exempel på sådana ekvationer är $x_i^2 + x_j + 2x_k^2 = 1$, $x_i^2 - 5x_j^2 = 0$ och $x_i + x_j + x_k = 1$.

Uppgift 3. (8p)
Vi säger att en sträng $s = s_1 s_2 \ldots s_m$ är *dold* i en sträng $t = t_1 t_2 \ldots t_n$ om det finns tal $1 \leq j_1 < \ldots < j_m \leq n$ sådana att $s = t_{j_1} t_{j_2} \ldots t_{j_m}$. Konstruera en algoritm som i $O(m \cdot n)$ tid räknar ut hur många olika sätt strängen s finns dold i strängen t. Exempelvis finns strängen `kost` dold på två sätt i strängen `konstruktion` medan `rim` endast finns dold på ett sätt i `algoritm`.

[1] Försök inte visa att problemet är i NP!

Uppgift 4. (8p)
Betrakta följande grafproblem:

Indata: Sammanhängande graf G med n noder och m bågar.
Utdata: Alla par av noder x, y sådana att om man tar bort x och y ur G så blir den resulterande grafen ej sammanhängande.

Konstruera en algoritm som löser detta problem i $O(n \cdot m)$ tid. Observera att man kan kontrollera om en graf är sammanhängande i $O(m)$ tid. Alltså kan problemet trivialt lösas i $O(n^2 \cdot m)$ tid.

Uppgift 5. (8p)
En instans av *mängdproblemet* består av en ändlig mängd trippler (V_1, C, V_2) där V_1, V_2 är variabler och $C \in \{\subseteq, \mathsf{disj}\}$. Låt $I = \{T_1, \ldots, T_k\}$ vara en godtycklig instans av detta problem över variablerna $X = \{x_1, \ldots, x_m\}$ och låt E vara mängden av alla icke-tomma, ändliga delmängder av de naturliga talen. Vi säger att I har en lösning om och endast om det finns en total funktion f från X till E sådan att

- $f(x_i) \subseteq f(x_j)$ för varje $(x_i, \subseteq, x_j)$ i I; och
- $f(x_i) \cap f(x_j) = \emptyset$ för varje $(x_i, \mathsf{disj}, x_j)$ i I.

Konstruera en polynomisk algoritm som undersöker om en godtycklig instans av mängdproblemet har en lösning.