INF3/4130 PRØVE-EKSAMEN
Gjennomgås 1/12-2005, 14.15 (lille aud.)

Oppgave 1 Uavgjørbarhet

L = {(M_1, M_2) | M_1 og M_2 er Turingmaskiner som er ekvivalente, dvs. gir samme output for samme input}

a) Bevis at L er uavgjørbart.

b) Kommenter betydning av øverste resultatet i følgende sammenheng: Det ville vare fint hvis vi kunne teste programmer ved å sammenligne dem automatisk (ved hjelp av et generelt testeprogram) med et kjent korrekt program for samme funksjon. Er dette mulig? Hvorfor?

Oppgave 2 NP-kompletthet etc.

Svar JA eller NEI og begrunn svaret ditt.

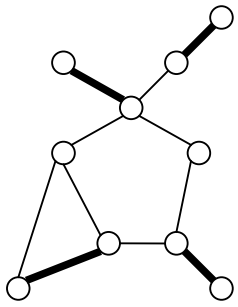
a) Bare problemer i NP kan løses effektivt med probabilistiske (Monte Carlo) algoritmer.

b) Noen NP-komplette problemer kan løses i polynomisk tid med dagens parallelle datamaskiner.

c) Noen NP-komplette problemer kan løses effektivt gjennomsnittlig (in average-case polynomial time).

Oppgave 3 Matching

Gitt grafen under, med den angitte matching. Vis hvordan algoritmen for maksimal matching i generelle grafer vil avgjøre om dette er en maksimal matching eller om den kan økes. Gi et eksempel på hvordan nodene kan være merket når algoritmen tar avgjørelsen. Merk blomster som oppstår med en passelig boks/ring rundt dem, og bruk stipling på kanter som hverken er med i noe tre eller i matchingen. Bruk ellers F(irkant)-noder, P(rikk)-noder og S(irkel)-noder, som i kompendiet. Angi hva som er avgjørende for at algoritmen tar den avgjørelsen den tar.



Oppgave 4 Dynamisk programmering

Vi skal løse følgende problem: Vi får som input to strenger: S som består bare av bokstaver, og T som også kan inneholde '*'-er. Spørsmålet er om strengene kan gjøres like, om vi får

lov til å erstatte hver '*' i T med null eller flere selvvalgte bokstaver. For S="abcbcf" og T="a*bc*f", kan dette gjøres slik: T'="a(bc)bc()f". For strengene "acdeb" og "a*c" er det derimot umulig. Beskriv en algoritme som avgjør om man kan oppnå likhet, og som bygger på dynamisk programmering. Skisser et program som implementerer algoritmen.

Oppgave 5 AVL-tre

Vi setter følgende sekvens av verdier inn i et tomt AVL-tre: 3, 5, 6, 2, 1, 4. Angi hvordan treet vil se ut etter hver innsetting.

Oppgave 6 Binomial-heaper

Anta at vi har to binomial-heaper H1 og H2 . H1 har $2^n - 1$ noder og H2 har $2^{n-1} - 1$ noder. Hvor mange sammenslåinger av (to) trær vil vi få når vi slår sammen ("merge") disse to heapene. Forklar hvorfor, og angi hvilke trær den resulterende heapen vil bestå av.

Oppgave 7 Amortisert kost

Når vi har en datastruktur som vi kan gjøre et antall forskjellige operasjoner på, så kan vi snakke om amortisert tids-analyse, og om amortisert kost (amortized time bound). Hvilke av utsagnene under er riktige for denne type analyse. Forklar kort.

a) Amortisert tids-anlyse bygger på gjennomsnittlig tid hver operasjon tar over M utførte operasjoner (oftest fra en tom datastruktur).

b) Om en operasjon tar tid O(1) worst case, så vil den også ta O(1) tid amortisert. Kan det her spille noen rolle om det er flere operasjoner som kan kalles?

c) Ved amortisert tids-analyse må man vite hvilken frekvensen de forskjellige operasjonene vil bli kalt med.

d) Ved amortisert tidsanalyse finner vi en øvre grense for hvor lang tid en vilkårlig sekvens av M utførte operasjoner (oftest fra en tom datastruktur) kan ta.

Oppgave 8 Nøyaktig søking i strenger

Som del av KPM-algoritmen beregnes en Next-tabell for P (det mønsteret vi skal lete etter) Beregn Next-tabellen for følgende tilfeller:

P = 0 0 0 1 0 0 0 og P = 1 1 2 1 1 0 1 2

Oppgave 9 Sterke komponenter

Anta at vi har en rettet graf G, og at vi lager en rettet graf G' ved å la de sterke komponentene i G være nodene i G', og at det går en kant mellom to noder i G' hvis og bare hvis det går (minst) en rettet kant mellom (noder i) de to tilsvarende sterke komponentene i G. Vis at G' er en løkkefri rettet graf (en DAG).