Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Eksamenforfattere:
Magnus Lie Hetland
Ole Edsberg
Lars Greger Hagen

Kvalitetskontroll:
Pål Sætrom

Kontakter under eksamen:
Magnus Lie Hetland (918 51 949)
Ole Edsberg (952 81 586)

Slutteksamen i
TDT4125 Algoritmekonstruksjon, videregående kurs

Onsdag 22. mai, 2013
Tid: 09:00 – 13:00

Hjelpemidler: (B) Alle trykte/håndskrevne; spesifikk, enkel kalkulator

Språk: Norsk (bokmål)

Les hele eksamen før du begynner, disponer tiden, og forbered spørsmål til fagstabens hjelperunde. Gjør antakelser der det er nødvendig. Svar kort og konsist. Lange forklaringer som ikke direkte besvarer spørsmålet gis ingen vekt.

Det er 20 underoppgaver. Hver underoppgave teller 5 % av den totale poengsummen.

Oppgave 1

a) Beskriv hvordan Minimum Set Covering-problemet kan løses med metoden branch-and-bound. Beskriv bare prosedyrene for *branching* og *bounding* (inkludert avkutting av subtrær).

Oppgave 2

a) Definer en nabolag-funksjon som gjør det mulig å anvende lokalt søk (basic local search) på Minimum Set Covering-problemet.

b) Makespan-skeduleringsproblemet (MS) er et minimeringsproblem hvor vi må finne en *skedulering* for n jobber med prosesseringstider $p_1, p_2, \ldots, p_2$ på $m \geq 2$ identiske prosessorer på en slik måte at alle prosessorene får minst en jobb, og tiden frem til alle jobbene er ferdig utført blir minimert. En skedulering er en plassering av hver jobb på en prosessor.

For eksempel kan en innputt-instans av MS ha $n = 2$ jobber $\{1, 2, 3\}$ med prosesseringstider $p_1 = 3, p_2 = 2, p_3 = 1$ og $m = 2$ prosessorer. En optimal løsning for denne innputt-instansen er skeduleringen som plasserer jobben 1 på den første prosessoren og jobbene 2 og 3 på den andre prosessoren. Vi kunne skrive denne skeduleringen som et sett av sett $\{\{1\}, \{2, 3\}\}$. Kostnaden (tid til ferdig utførelse) for den optimale løsningen er 3.

Vi kan definere et mulig nabolag for MS ved å la naboene til en skedulering x være settet av alle skeduleringer som kan lages av x ved å velge en jobb fra en prosessor med mer enn en jobb, og så plassere den valgte jobben på en hvilken som helst annen prosessor. Naboene til skeduleringen $\{\{1\}, \{2, 3\}\}$ ville være settet av tre skeduleringer $\{\{\{1\}, \{2, 3\}\}, \{\{1, 2\}, \{3\}\}, \{\{1, 3\}, \{2\}\}\}$.

Finnes det noen innputt-instans for MS hvor lokalt søk med dette nabolaget kan bli sittende fast i et lokalt, ikke-globalt optimum? Hvis du svarer nei, bevis det. Hvis du svarer ja, gi et eksempel på en innputt-instans hvor lokalt søk ville bli sittende fast, og skriv løsningen i et lokalt, ikke-globalt optimum for ditt eksempel.

c) En venn av deg påstår at han har funnet et eksakt polynomisk-tid-søkbart nabolag for et NP-hardt problem X. Vennen din påstår at dette beviser at P=NP fordi alle problemer i NP er polynomisk-tid-reduserbare til X, og X kan løses i polynomisk tid med vanlig lokalt søk med det nyoppdagede nabolaget. Hva er galt med din venns resonnement?

d) Gi et eksempel på et optimeringsproblem som det finnes et eksakt polynomisk-tid-søkbart nabolag for. Det er OK hvis problemet er et leketøyseksempel som du finner opp for denne oppgaven.

Oppgave 3

a) I maksimum-klikk-problemet (MAX-CL) får du en graf $G = (V, E)$ og må finne et subsett $S \subseteq V$ slik at hvert par av noder i S har en kant mellom seg og størrelsen av S maksimeres. Reduser MAX-CL til IP eller 0/1-LP og vis hvordan du kan relaksere det til (reelt-tallig) LP.

b) Anta at du har en instans x av et maksimeringsproblem. Du reduserer den til en instans $I(x)$ av IP, relakserer $I(x)$ til en instans $I_{rel}(x)$ av LP, og finner dualen $Dual(I_{rel}(x))$. La $Opt(x)$ være kostnaden til en optimal løsning for x. Uttrykk så presist som mulig hvilke slutninger du kan gjøre om forholdet mellom $Opt(x)$ og verdien α i hvert av de følgende tilfellene. Bruk operatorene $=$, $\neq$, $>$, $\geq$, $<$ eller $\leq$ der det passer.

1. En mulig (feasible) løsning av $I(x)$ har kostnad α.
2. En optimal løsning på $I(x)$ har kostnad α.
3. En mulig løsning på $I_{rel}(x)$ har kostnad α.
4. En optimal løsning på $I_{rel}(x)$ har kostnad α.
5. En mulig løsning på $Dual(I_{rel}(x))$ har kostnad α.
6. En optimal løsning på $Dual(I_{rel}(x))$ har kostnad α.
7. En mulig løsning på $I_{rel}(x)$ har kostnad α, og en mulig løsning på $Dual(I_{rel}(x))$ har også kostnad α.
8. En optimal løsning på $I_{rel}(x)$ har kostnad α, og en optimal løsning på $Dual(I_{rel}(x))$ har også kostnad α.
9. En mulig løsning på $I_{rel}(x)$ har kostnad α, en mulig løsning på $Dual(I_{rel}(x))$ har også kostnad α, og en mulig løsning på $I(x)$ har også α.
10. En optimal løsning på $I_{rel}(x)$ har kostnad α, og en optimal løsning på $Dual(I_{rel}(x))$ har også kostnad α, og en mulig løsning på $I(x)$ har også kostnad α.

c) Anta at du har et optimeringsproblem som du reduserer til IP og deretter relakserer til LP. Hvis du har et bevis for at de optimale løsningene på det relakserte problemet alltid har heltallige løsninger, hva impliserer dette for kompleksitetsklassen til det opprinnelige problemet? Begrunn svaret.

Oppgave 4 Ta for deg problemet å velge hvilke personer du skal invitere til en fest. La *Peop* være settet av personer du kan invitere, og la $f(x, y)$ være en funksjon som for hvert par av personer x og y gir et ikke-negativt heltall som måler hvor godt x og y kjenner hverandre. Det er alltid slik at $f(x, y) = f(y, x)$. Du ønsker å invitere et subsett $Inv \subseteq Peop$ av personer slik at:

1. Hvert par av forskjellige personer a og b du inviterer har i minste fall litegrann kjennskap til hverandre. Med andre ord,

$$\forall(a, b \in Inv, a \neq b) f(a, b) > 0.$$

2. Summen av kjennskap over alle par av personer du inviterer er maksimert. Med andre ord,

$$\sum_{a,b \in Inv, a \neq b} f(a, b)$$

er maksimert.

a) Er dette problemet sterkt NP-hardt? Begrunn svaret.

Oppgave 5

a) La oss late som om du må løse Minimum Vertex Cover-problemet med en genetisk algoritme (GA). Spesifiser de følgende komponentene av GA for dette problemet: (i) representasjon av løsninger som individer, (ii) crossover-operatoren, (iii) mutasjons-operatoren, og (iv) fitness-funksjonen. Ikke forklar GAer generelt. Hold svaret enkelt.

b) Skjemateoremet er basert på en antakelse om en crossover-rate på 100 %. Hvordan ville skjemateoremet forandret seg hvis crossover-raten var 0 %? Begrunn svaret.

Oppgave 6

a) Approksimeringsalgoritmer gir garantier for hvor unøyaktige de er, på tross av at optimum er ukjent. Hvordan er dette mulig? Forklar veldig kort, med dine egne ord.

En gåte: Jeg kan ikke nødvendigvis gi deg det eksakte svaret på problemet ditt, men jeg kan komme så nært som du ber meg om, i polynomisk tid (selv om den tiden kan vokse eksponentielt som en funksjon av min nøyaktighet). Jeg kan til og med gjøre dette for et hvilket som helst problem innen en gitt distanse fra ditt, selv om jo lenger unna det er, jo lenger vil det ta.

b) Hvem er jeg?

For en gitt familie av kant-vektede funksjoner i $\text{TSP}_\triangle$ har du funnet en metode for å konstruere et minimalt/minimum spenntre (MST) slik at alle de indre nodene har liketallig grad (even degree), og alle løvnodene kan matches med en kostnad $\frac{1}{\sqrt{5}} \cdot \text{OPT}$, hvor OPT er kostnaden ved optimum.

c) Hvordan ville du brukt MST-en og matchingen din for å lage en approksimeringsalgoritme for $\text{TSP}_{\triangle}$, og hva ville ϵ bli?

Et *pangram* eller en *holoalfabetisk setning* for et gitt alfabet er en setning hvor hvert tegn i alfabetet forekommer minst en gang. Eksempler på engelsk er "The quick brown fox jumps over the lazy dog," og "Quick hijinx swiftly revamped gazebo." Ta for deg det følgende generaliserte, forenklede problemet.

Forenklet Pangram Problem (SPP)

Innputt: Et alfabet Σ og et språk (i.e., sett av ord) $L \subset \Sigma^*$, hvor $|L| = n$ og $|\Sigma| = m$.

Avgrensninger (constraints): $\mathcal{M}(\Sigma, L) = \{S \subseteq L \mid \Sigma = \bigcup_{w \in S} w\}$.

Kostnader: For hvert $S \in \mathcal{M}(\Sigma, L)$, $cost(S, \Sigma, L) = |S|$.

Mål: *minimum*.

d) Beskriv en effektiv approksimeringsalgoritme for SPP. Hva er approksimeringsratioen (δ) for denne algoritmen?

Oppgave 7 La G være en sammenkoblet (connected) graf, med $|V(G)| \geq 2$. Ta for deg følgende approksimeringsalgoritme for minimum vertex cover-problemet (MIN-VCP). Finn et dybde-først tre T over G, og utputt settet S av alle indre (ikke-løv-) noder i T.

a) Vis at S er et vertex cover for G, men at det ikke nødvendigvis hadde vært det, hvis vi hadde brukt bredde-først søk i stedet.

b) Vis at det finnes en matching $M \subseteq E(G)$ slik at $|M| \geq |S|/2$.

c) Hvis OPT er minimum-størrelsen for et vertex cover for G, vis at $|S| \leq 2 \cdot \text{OPT}$.

Oppgave 8 Det såkalte *begrenset korteste vei*-problemet (RSP) er definert som følger: Du får en graf G med n noder og m kanter. Hver kant $\{i, j\} \in E(G)$ har en positiv heltallig kostnad $c(i, j)$, og en positiv heltallig forsinkelse $d(i, j)$. Kostnaden (forsinkelsen) for en vei er summen av kostnader (forsinkelser) langs alle dens kanter. Vi ønsker å finne den minste mulige kostnaden over s-t-veier (for to gitte noder $s, t \in V(G)$) hvis totale forsinkelse ikke overstiger en gitt terskel D. RSP er NP-hardt, men det finnes en FPTAS for problemet, med kjøretd $\mathcal{O}(mn/\epsilon)$.

Ta nå for deg problemet RSP_k, hvor maksimum-kostnaden over alle kanter er mindre enn n^k, for en eller annen konstant k.

a) Vis hvordan FPTASen for RSP kan brukes for å løse RSP_k i polynomisk tid. Hva er kjøretiden? (Forklar resonnementet ditt.)

Oppgave 9 Et *hitting set* er definert som følger: La U være universet av objekter og la $\mathcal{S}$ være settet av sett $\{S_1, S_2, \ldots, S_n\}$ hvor $S_i \subseteq U$ for alle $S_i \in \mathcal{S}$. Et hitting set for $\mathcal{S}$ er et sett $H \subseteq U$ som har ikke-tomt snitt (intersection) med alle sett i $\mathcal{S}$. Med andre ord, for alle $S_i \in \mathcal{S}$, har vi at $H \cap S_i \neq \emptyset$. Vi kaller $|H|$ størrelsen av hitting settet.

For eksempel, hvis vi har $U = \{a, b, c, d\}$, og $\mathcal{S} = \{\{a, b\}, \{b, c, d\}, \{c, d\}\}$, da er $H = \{b, c\}$ et hitting set for $\mathcal{S}$, og størrelsen dets er 2.

Hitting set-problemet (HSP) er et decision-problem definert som følger: Du får et univers av objekter U, et sett av sett $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ hvor $S_i \subseteq U$ for alle $S_i \in \mathcal{S}$, og et positiv heltall k. Oppgaven er å svare ja hvis det finnes et hitting set for $\mathcal{S}$ av størrelse k, og nei hvis ikke.

For eksempel, ta for deg problem-instansen $U = \{a, b, c, d\}$, $\mathcal{S} = \{\{a, b, c\}, \{a, c\}, \{d\}\}$, $k = 2$. Svaret for denne instansen er ja. De mulige hitting settene er $\{a, d\}$ og $\{c, d\}$. Hvis vi endrer k til 1, blir svaret nei.

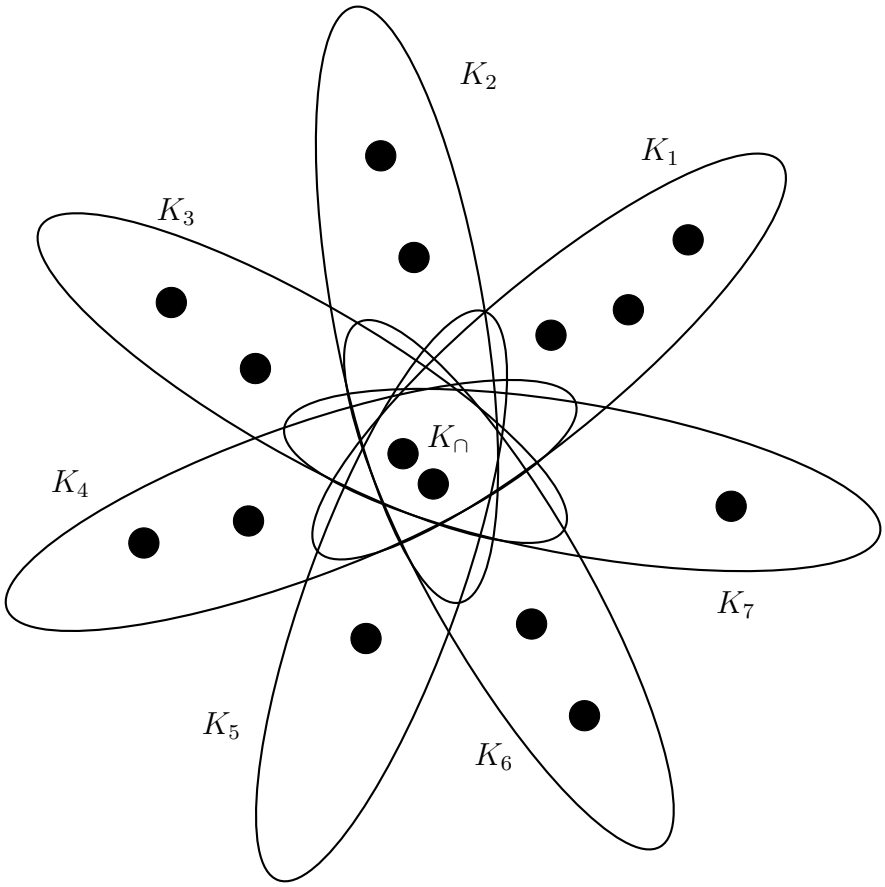
a) Ta for deg parametriseringen

$$Pat(U, \mathcal{S}, k) = \max\{k, d(\mathcal{S})\}$$

av HSP, hvor $d(\mathcal{S})$ er størrelsen av det største settet i $\mathcal{S}$. Gi en Pat-parametrisert polynomisk-tid algoritme for HSP.

Hint: Se Appendikset

Appendiks



Solsikke: Settene $\mathcal{K} = \{K_1, K_2, \ldots, K_k\}$ utgjør en *solsikke* hvis det finnes et sett $K_\cap$ slik at for alle $K_i, K_j \in \mathcal{K}$, $i \neq j$, er snittet (intersection) $K_i \cap K_j$ identisk med $K_\cap$.

Solsikkelemmaet: Hvis en samling av sett $\mathcal{K} = \{K_1, K_2, \ldots, K_m\}$ består av $p^{d+1} \cdot d!$ sett, og alle sett i $\mathcal{K}$ har størrelse mindre enn eller lik d, da inneholder $\mathcal{K}$ en solsikke av størrelse $p+1$. Videre kan denne solsikken finnes i polynomisk tid.

Observasjon: Anta at en instans $(\mathcal{S}, k)$ av hitting set-problemet inneholder en solsikke $\mathcal{K}$ av størrelse $k+1$ med snitt-sett $K_\cap$. Hvis $K_\cap = \emptyset$, er det $k+1$ disjunkte sett, så det er ingen løsning. Hvis $K_\cap \neq \emptyset$, er problemet ekvivalent med å fjerne solsikkesettene fra $\mathcal{S}$, og legge til $S_\cap$ i stedet.