Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Avsluttende eksamen i
TDT4125 Algoritmekonstruksjon

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.

Poeng er oppgitt for hver deloppgave. Disse summerer til 120, men maksimal score er 100 poeng. Med andre ord: **En poengsum på over 100 teller som 100.**

Ikke kast bort tid på å skrive for lange forklaringer!

Det viktigste er å vise at du har forstått de sentrale punktene. Det holder stort sett med en setning eller to.

Oppgave 1

a) Hvis vi ikke klarer å finne optimum for et problem, hvordan er det mulig å garantere at vi er en gitt faktor unna? Forklar kort. (6 p)
Løsning: Vi kan gi garantier i forhold til en *optimistisk grense* for optimum.
(Relevant pensum: F.eks. Cormen et al., 3. utg., kap. 35)

b) Hvis du uttrykker et problem som et lineært program med heltallsrestriksjoner, og så lager et nytt lineært program som kun skiller seg fra det første ved at du har fjernet heltallsrestriksjonene, hva vet du om forholdet mellom disse programmene? (6 p)

Løsning: Det andre programmet vil ha en minst like bra objektivverdi. (Det kan vi garantere fordi det er en *relaxation*.)
(Relevant pensum: Williamson & Shmoys, s. 18)

c) Hva er forskjellen på et *polynomial-time approximation scheme* (PTAS) og et *fully polynomial-time approximation scheme* (FPTAS)? Forklar kort. (6 p)
Løsning: Begge lar deg velge ε og gir deg en $(1+\varepsilon)$-approksimasjon. PTAS er polynomisk i n og FPTAS er polynomisk i både n og $1/\varepsilon$.
(Relevant pensum: Cormen et al., s. 1107)

d) Hvis (x, k) er en instans av et parametrisert problem, hvordan definerer vi *størrelsen* til denne instansen? (6 p)
Løsning: $|x| + k$, der $|x|$ er lengden til strengen x.
(Relevant pensum: Cygan et al., Dfn. 1.2, s. 13)

e) Hvis du har en instans (G, k) av problemet Vertex Cover, hvorfor er det trygt å redusere denne til $(G - v, k - 1)$ dersom v har minst $k + 1$ naboer? (6 p)
Løsning: Noden v må være med i ethvert vertex cover av størrelse maks k. Ellers måtte vi ha tatt med alle de $k + 1$ naboene.
(Relevant pensum: Cygan et al., § 2.2.1)

f) Hva er de to reduksjonene (FAST.1 og FAST.2) for problemet Feedback Arc Set in Tournaments? (6 p)
Løsning:
FAST.1: Hvis en kant e er del av minst $k + 1$ triangler, reversér e og redusér k med 1.
FAST.2: Hvis en node v ikke finnes i noen triangler, slett v.
(Relevant pensum: Cygan et al., § 2.2.2)

g) Forklart kort hvorfor T_P for en grådig scheduler maksimalt er en faktor 2 unna tiden for en optimal scheduler. (6 p)
Løsning: Følger direkte av at $T_P \leq T_1/P + T_\infty$ holder for grådig scheduling, mens $T_1/P \leq T_P$ og $T_\infty \leq T_P$ holder uansett.
(Relevant pensum: Cormen et al., 3. utg., (27.2), (27.3), Thm. 27.1)

h) Forklar kort hvordan man kan simulere CRCW ved hjelp av EREW. Beskriv hvordan både skriving og lesing håndteres. (6 p)
Løsning: Skriving: Hver prosessor har sitt eget minneområde, og man bruker parallell sortering i logaritmisk tid for å avgjøre hvilke prosessorer (den første av dem med samme verdi) som skal få skrive til globalt minne. Lesing: Sortér basert på adresse igjen, la den

første lese, og spre så verdien med pekerdobling e.l.
(Relevant pensum: Cormen et al., 1. utg., s. 706–708; § 30.1)

i) Anta at du har en vektet, rettet graf $G = (V, E)$ med en angitt startnode s og en slutt-node t. Du vil finne en sti p fra s til t som er slik at den største kantvekten i p er minst mulig. Uttrykk dette problemet som et lineært program uten heltallsrestriksjoner. (6 p)
Løsning:

La A være E sortert etter vekt, og la $i(u, v)$ være indeksen til (u, v).

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + 2^{i(u,v)} & \quad \text{for each } (u,v) \in E, \\ & d_s & = & 0. & \end{array}$$

Optimum på binær form angir da hvilke elementer av A som er med i den optimale stien.

Dette er programmet for korteste vei, der vektene $w(u, v)$ er byttet ut med unike toer-potenser på formen $2^{i(u,v)}$. Enhver slik toerpotens er større en summen av alle lavere potenser, så stien med lavest sum av potenser vil også ha lavest maksimums-potens og maksimums-indeks, og dermed en minimal maksimums-vekt. Binærrepresentasjonen angir hvilke toerpotenser som er med i summen.

Som et alternativ kunne man ha brukt disse vektene i programmet for *min-cost-flow*-problemet, med kapasiteter og flyt satt til 1. Det optimale hjørnet for løsnings-polytopen har heltallsverdier, og flyten angir eksplisitt hvilken sti som er valgt.

Oppgaven har vist seg å være noe mer utfordrende enn den var ment å være, så poeng-givningen har vært generøs.
(Relevant pensum: Cormen et al., 3. utg., (29.44–29.46))

j) Gitt en qubit $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$, hva er sannsynlighetene for utfallene $|0\rangle$ og $|1\rangle$? (6 p)
Løsning: $|\alpha|^2$ og $|\beta|^2$.
(Relevant pensum: Forelesningsfoiler om temaet, s. 24)

k) Hva slags matriser bruker vi til å representere kvanteporter (*quantum gates*)? (6 p)
Løsning: Unitære matriser (*unitary matrices*). Det viktige her er at de bevarer lengden på komplekse vektorer, så summen av sannsynlighetene holdes konstant lik 1. Om man forklarer dette uten å ha med navnet, gir det også full score.
(Relevant pensum: Forelesningsfoiler om temaet, s. 43)

Oppgave 2

Din venn Lurvik har skrevet en semesteroppgave om "fraværet av klikker" i en graf. Teksten skifter mellom to tolkninger av denne beskrivelsen: Enten at det finnes en fraværende klikk (altså en uavhengig mengde, *independent set*) av størrelse k eller mer i grafen, eller at det ikke finnes noen klikk (*clique*) av størrelse k eller mer i grafen. Betrakt disse to tolkningene som beslutningsproblemer.

a) Du påpeker at disse problemene ikke er identiske, men Lurvik mener de er ekvivalente, dvs. at de kan reduseres til hverandre i polynomisk tid. Er du enig? Diskutér kort. (7 p)

Løsning: INDEPENDENT SET er et NP-komplett problem (triviell reduksjon fra CLIQUE, ved å ta kant-komplementet av grafen). Det andre problemet er *komplementet* av CLIQUE. Siden CLIQUE er NP-komplett er altså det andre problemet coNP-komplett. Dersom vi kunne redusere mellom dem i polynomisk tid, ville det bety at NP = coNP, så problemene vi kan konkludere med at det er svært lite trolig.

(Relevant pensum: Arara & Barak, § 2.6.1)

Lurvik prøver nå å finne et *maksimalt snitt* i en urettet graf, altså en partisjonering av nodene i to mengder A og B slik at det er så mange kanter mellom A og B som mulig.

Hun har tatt et fag om heuristisk optimering, og hun har nå laget en algoritme basert på såkalt *lokalt søk* eller *hill-climbing*, der en løsning gradvis forbedres til man ikke kan komme lenger. Den grunnleggende operasjonen er å *flytte* en node, altså la den skifte fra A til B eller omvendt. Vi sier at en node *bør flyttes* dersom det å flytte den gir en bedre løsning. Algoritmen hennes virker som følger:

```
HEURISTIC-MAX-CUT(G)
lag en tilfeldig partisjonering A, B av G.V
while det finnes en v ∈ G.V slik at v bør flyttes
    flytt v
return A, B
```

Algoritmer basert på lokalt søk har generelt flere ulemper. Det er for eksempel ikke sikkert de terminerer i polynomisk tid, og de kan sette seg fast i lokale optima, og dermed ikke løse problemet optimalt. Likevel synes du det er noe besnærende ved Lurviks algoritme.

b) Lurvik mener algoritmen helt sikkert vil finne en *ganske god* løsning innen rimelig tid, for eksempel at den garantert vil oppnå en viss prosent av optimum og at den vil terminere i polynomisk tid. Er du enig? Diskutér kort. (7 p)

Løsning: Den er en approksimasjonsalgoritme med $\alpha = 0.5$. Vi ser at algoritmen terminerer i polynomisk tid, siden det tar polynomisk tid å sjekke om noen noder bør flyttes, og evt. flytte én, og snittet kan maksimalt øke $|G.E|$ ganger. En node bør (og vil dermed) flyttes når den har flere naboer i egen partisjon enn i den andre. Når algoritmen terminerer er det altså ingen noder som har det. For hver kant i A eller B finnes det altså to kanter over snittet, der nodene i den andre enden kanskje har en kant mellom seg. Med andre ord er minst halvparten av kantene med i snittet, og dermed også minst halvparten av en optimal løsning.
(Relevant pensum: Stoff om approksimering generelt)

c) Kan du si noe lignende om algoritmen dersom du sletter linje 2 og 3? Oppgi eventuelle antagelser. Diskutér kort. (6 p)
Løsning: Hvis vi antar at hver node har sannsynlighet 0.5 for å havne i A (altså uniform sannsynlighet over partisjonene), så vil forventningsverdien for indikatorvariabelen for en kant over snittet også være 0.5, som gir en forventet α på 0.5.
(Relevant pensum: Williamson & Shmoys, § 1.7)

d) Hvis du skulle ta utgangspunkt i algoritmen i deloppgave c for å konstruere en deterministisk algoritme, hvordan ville du ha gått frem? Forklar kort. (6 p)
Løsning: Her vil det være naturlig å bruke derandomisering med *the method of conditional expectations*, der man setter én og én variabel slik at man maksimerer forventningen over de gjenværende. Ingen grundig forklaring kreves. Om man gir en tydelig forklaring men ikke husker navnet på metoden, så gir det også full uttelling.
(Relevant pensum: Williamson & Shmoys, § 5.2)

Oppgave 3

Du skal plassere utrykningskjøretøy for å minimere utrykningstid. Fra et sett L med n lokasjoner skal du velge ut et sett $S \subseteq L$ med m lokasjoner som skal være *stasjoner*, og som hver får ett kjøretøy. De andre lokasjonene er *destinasjoner*. For ethvert par av lokasjoner $i, j \in L$ har du oppgitt utrykningstiden t_{ij} fra i til j i minutter, som et heltall. Du kan anta at $t_{ij} = t_{ji}$, og at alle mulige snarveier benyttes, slik at *trekantulikheten* gjelder:

$$t_{ik} \leq t_{ij} + t_{jk} \qquad \forall i, j, k \in L\,.$$

Under en utrykning regner vi det som (uniformt) tilfeldig hvilken stasjon som responderer. Det betyr at enhver destinasjon $j \notin S$ har en *forventet utrykningstid* som er gjennomsnittet av utrykningstidene fra de m stasjonene, altså

$$e_j = \frac{1}{m} \sum_{i \in S} t_{ij}\,.$$

Du har fått i oppgave å gi garantier om forventet utrykningstid, og ønsker dermed å velge S slik at du minimerer den maksimale forventede utrykningstiden $\max_{j \in L \setminus S} e_j$.

Heller enn bare å løse denne ene instansen, ønsker du å konstruere en algoritme som løser problemet generelt. Der det er naturlig å parametrisere problemet, skal m brukes som parameter.

a) Vis at problemet er i XP (*slice-wise polynomial*). (7 p)

Løsning: Bare prøv alle subsett, med endring fra ett subsett til det neste i konstant tid. Kjøretiden blir $O(n^m)$, altså $O(f(m) \cdot n^{g(m)})$, med $f(m) = 1$ og $g(m) = m$.

(Relevant pensum: Cygan et al., s. 7)

b) Vis at problemet er W[2]-hardt. (7 p)

Løsning: F.eks. reduksjon fra DOMINATING SET. Oversett grafen til et sett med utrykningstider, f.eks. 1 eller 2, for "kant" eller "ikke kant." Svaret vil da bli mindre enn 2 hvis og bare hvis grafen har et dominating set av størrelse m. For å vise W[2]-kompletthet må man også vise/poengtere at parameteren m er bevart.

(Relevant pensum: Cygan et al., § 13.1, 13.3)

c) Vis at problemet ikke har en PTAS, med mindre P = NP. (7 p)

Reduksjonen gir et NP-hardhets-bevis for $\alpha < 2/(2 - 1/m) = 1 + 1/(2m - 1)$.

(Relevant pensum: Williamson & Shmoys, § 16.1)

d) Beskriv en approksimeringsalgoritme for problemet, med $\alpha = 2$. Hva blir kjøretiden? (7 p)

Hint: Vis $t_{jk} \leq e_j + e_k$. For delvis uttelling, bare anta dette.

Løsning:

For hver i har vi $t_{jk} \leq t_{ij} + t_{ik}$. Tar vi gjennomsnittet over i, får vi $t_{jk} \leq e_j + e_k$. Om man bare antar dette, kan man oppnå maksimalt 5 poeng.

Alle andre enn de m utvalgte vil være innen en avstand av $2 \cdot \mathrm{OPT}$ av hverandre, pga. resultatet over. Hver av dem vil også være nærmere minst ett av de utvalgte enn OPT, siden gjennomsnittet er nærmere enn dette. Du har altså $n - m$ stk. som alle er nærmere $n - m - 1$ stk. enn $2 \cdot \mathrm{OPT}$. Som første lokasjon, velg én som har en "radius" som er mindre enn minst m andre har, og bruk så de $m - 1$ som er lengst unna (og som kanskje faller utenfor radien) som de gjenværende lokasjonene. Kjøretid $O(mn)$, gitt at man bruker lineær-tids-seleksjon. (Ellers får man en $\log m$-faktor i tillegg – evt. $\log n$, om man gjør det mindre effektivt.)

(Relevant pensum: Stoff om approksimering generelt)