Avsluttende eksamen i TDT4120 Algoritmer og datastrukturer

Eksamensdato	30. november 2009
Eksamenstid	0900–1300
Sensurdato	21. desember
Språk/målform	Bokmål
Kontakt under eksamen	Magnus Lie Hetland (tlf. 91851949)
Tillatte hjelpemidler	Ingen trykte/håndskrevne; bestemt, enkel kalkulator

Vennligst les hele oppgavesettet før du begynner, disponer tiden og forbered evt. spørsmål til faglærer kommer til eksamenslokalet. Gjør antagelser der det er nødvendig. Skriv kort og konsist på angitt sted. Lange forklaringer og utledninger som ikke direkte besvarer oppgaven tillegges liten eller ingen vekt.

Algoritmer kan beskrives med tekst, pseudokode eller programkode, etter eget ønske, så lenge det klart fremgår hvordan den beskrevne algoritmen fungerer. Korte, abstrakte forklaringer kan være vel så gode som utførlig pseudokode, så lenge de er presise nok. Kjøretider oppgis med asymptotisk notasjon, så presist som mulig.

Oppgave 1 (56%)

a) Hva er forskjellen på et *problem* og en *probleminstans*?

Svar (7%):

b) Hvilket krav stiller vi normalt til input til bøttesortering (*bucket sort*)?

Svar (7%):

c) Hvis du i et flytnettverk har en kant fra u til v med kapasitet 7 og flyt 3, hvor mye kan du øke flyten med fra v til u? Hva vil flyten fra v til u være da?

Svar (7%):

d) Å finne korteste vei (*simple*, *weighted path*) i en vektet graf med negative sykler er NP-hard. Hvordan håndterer BELLMAN–FORD dette?

Svar (7%):

e) Om du står overfor et ukjent problem A i NP, hvordan vil du vise at det er NP-komplett?

Svar (7%):

f) Er det mulig å bevise at *noen* NP-komplette problemer kun kan løses av algoritmer med eksponentiell kjøretid uten at man avgjør hvorvidt de andre kan løses i polynomisk tid? Begrunn svaret kort.

Svar (7%):

g) Gi en nedre og øvre grense for T_P ut fra *work, span* og P på en ideell parallell maskin med grådig arbeidsfordeling (*scheduling*).

Svar (7%):

Et k-regulært tre er et tre der alle interne noder har k barn. Det trenger ikke være balansert. Du kan anta at treet har minst én intern node.

h) Hvor mange løvnoder har et k-regulært tre med n interne noder? Sett opp en rekurrensligning og løs den. Oppgi svaret eksakt (det vil si, uten asymptotisk notasjon eller ukjente konstanter), som funksjon av k og n. Vis utregningen.

Svar (7%):

Oppgave 2 (20%)

Anta at du har en mengde **U** med bilder og en funksjon $s(\cdot, \cdot)$ som beskriver hvor like to bilder er, i form av et reelt tall. Du kan anta at $s(x, y) = s(y, x)$. Du ønsker nå å utføre en enkel «clustering»: Du vil dele **U** i to delmengder **X** og **Y**, slik at summen av likhetsverdier fra alle objekter i **X** til alle i **Y** blir så liten som mulig. Verken **X** eller **Y** skal være tom.

a) Vis at problemet kan løses polynomisk, *eller* vis at problemet er NP-hard.

Svar (6%):

Du ønsker nå å vise frem bildene i et lysbildeshow, slik at to etterfølgende bilder ligner så mye på hverandre som mulig – og for dramatisk effekt vil du ende opp på det første bildet igjen til slutt. Du ønsker altså å maksimere summen av likhetsverdier mellom par med etterfølgende bilder for hele lysbildeshowet, og du vil vise hvert bilde nøyaktig én gang, bortsett fra første/siste bilde, som altså vises to ganger.

b) Beskriv kort en algoritme som løser problemet effektivt og oppgi kjøretiden, *eller* vis at problemet er NP-hard.

Svar (7%):

Du ønsker å lage en nettside med bildene dine, der man kan navigere til «nabobilder». Du ønsker å gjøre dette slik at hvis du kan navigere direkte fra bilde x til y kan du også navigere direkte fra bilde y til x. Det skal være mulig å komme seg fra hvilket som helst bilde til et hvilket som helst annet ved å gå fra nabo til nabo, men det skal ikke være mulig å gå i ring. Du vil legge opp navigeringen slik at summen av likhet over alle par med nabobilder blir størst mulig.

c) Beskriv kort en algoritme som løser problemet effektivt og oppgi kjøretiden, *eller* vis at problemet er NP-hard.

Svar (7%):

Oppgave 3 (24%)

Anta følgende: For en problemstørrelse på n kan vi behandle tall med $O(\log n)$ siffer i konstant tid, og vi kan bruke dem som indekser i tabeller, med konstant oppslagstid. (Dette er den normale antagelsen i læreboka.)

Anta også at du har en urettet graf med n noder og m kanter, representert ved hjelp av nabolister lagret i en tabell indeksert med nodenummer (standard nabolisterepresentasjon). Du ønsker nå å sortere nabolistene slik at nabolisten for hver node inneholder naboene i stigende rekkefølge.

a) Forklar hvordan du kan sortere alle nabolistene som beskrevet med total kjøretid $O(m+n)$.

Svar (8%):

Anta at du har en algoritme A som finner en sirkulasjon med minimal kostnad i en rettet graf $G = (V, E, w, b, u)$ med noder V, kanter E, kant-vekter w og nedre og øvre kapasiteter hhv b *og* u, eller som sier at ingen sirkulasjon eksisterer.

b) Vis kort hvordan du kan bruke A for å avgjøre om det finnes en sti fra en node u til en node v i en urettet graf $G' = (V', E')$.

Svar (8%):

La $T = (V, E)$ være et urettet tre, og la $h(u) = \max \{ d(u, v) \mid v \in V \}$, der $d(u, v)$ er lengden på stien fra u til v, målt i antall kanter. Definer et midtpunkt i T til å være en node $u \in V$ slik at $h(u) = \min \{ h(v) \mid v \in V \}$.

c) Forklar (uten detaljert kode) hvordan du kan finne alle midtpunktene i T ved hjelp av topologisk sortering.

Svar (8%):