

Institutt for datateknikk og informasjonsvitenskap

Eksamensoppgave i TDT4120 Algoritmer og datastrukturer

Faglig kontakt under eksamen

Magnus Lie Hetland

Telefon

918 51 949

Eksamensdato

17. desember, 2016

Eksamenstid (fra–til)

09:00–13:00

Hjelpemiddelkode/tillatte hjelpemidler

D

Annen informasjon

Oppgavearkene leveres inn, med svar i svarrute under hver oppgave

Målform/språk

Bokmål

Antall sider (uten forside)

8

Antall sider vedlegg

0

Informasjon om trykking av eksamensoppgave

Originalen er

1-sidig ☒ 2-sidig ☐

sort/hvit ☒ i farger ☐

Skal ha flervalgskjema ☐

Kvalitetssikret av

Ole Edsberg

Kontrollert av

Dato

Sign

Les dette nøye

- (i) Det at kjøretiden er oppgitt til å være $O(n)$ – tvinger det bøttestørrelsen til å være konstant? Er de større vil de jo bli færre. Hvor store kan de være, og likevel gi lineær tid, dersom man tar kvadratet av hver enkelt?
- (ii) Les hele eksamenssettet nøye før du begynner!
- (iii) Faglærer går normalt én runde gjennom lokalet. Ha evt. spørsmål klare!
- (iv) Skriv svarene dine i svarrutene og lever inn oppgavearket. Bruk gjerne blyant! Evt. kladd på eget ark først for å unngå overstrykninger, og for å få en egen kopi.
- (v) Ekstra ark kan legges ved om nødvendig, men det er meningen at svarene skal få plass i rutene på oppgavearkene. Lange svar teller ikke positivt.
- (vi) Eksamen har 20 oppgaver, totalt verdt 120 poeng. Av disse er 20 bonuspoeng, så en poengsum over 100 regnes som 100. Poengverdi er angitt ved hver oppgave.

Oppgaver

- (6 p) 1. Du skal sortere n heltall i verdiorrådet 0 til k , der $k = O(n)$. Hvilken sorteringsalgoritme i pensum vil det være mest naturlig å bruke?

- (6 p) 2. I læreboka er dybde-først-søk (*depth-first search*, DFS) implementert med rekursjon. Det er også mulig å implementere dybde-først-søk *uten* rekursjon. Hvordan da?

- (6 p) 3. Hvilken teknikk – inkrementell design (*incremental design*), splitt og hersk (*divide-and-conquer*), dynamisk programmering (*dynamic programming*), grådighet (*greedy algorithms*) eller amortisert analyse (*amortized analysis*) – er brukt i den vanlige løsningen på 0-1-ryggsekkproblemet (0-1 knapsack)?

- (6 p) 4. La tabellen $S[1..10]$ være $\langle 50, 70, 45, 15, 72, 41, 61, 22, 26, 64 \rangle$ og la $S.top = 5$. Utfør så følgende utsagn, i rekkefølge:

- 1 $x = \text{POP}(S)$
- 2 $y = \text{POP}(S)$
- 3 $\text{PUSH}(S, x)$
- 4 $\text{PUSH}(S, y)$

Hva er innholdet i tabellen S nå?

Merk: Det spørres her om innholdet i *hele* tabellen, ikke bare i stakken.

- (6p) 5. Etter at vi har utført en algoritme for å finne korteste vei fra én node s til alle andre i en graf (*the single-source shortest path problem*) har vi $v.\pi = \text{NIL}$ for en gitt node v . Hva er da verdien til $v.d$?

- (6p) 6. Hvis du har et (ikke nødvendigvis balansert) binært søketre med n noder og høyde h , hvor lang tid tar det å finne minste element (TREE-MINIMUM)? Uttrykk svaret med O-notasjon.

- (6p) 7. Hva er den amortiserte kjøretiden for innsetting i en dynamisk tabell (TABLE-INSERT)? Oppgi svaret i O-notasjon.

- (6p) 8. Om man bruker BUCKET-SORT på n uavhengig uniformt fordelte tall i området $[0, 1)$, så får man en kjøretid på $O(n)$. Underveis, som en del av BUCKET-SORT, kalles INSERTION-SORT flere ganger. Hva er den forventede kjøretiden til *hvert enkelt* av disse kallene til INSERTION-SORT? (Det er her altså snakk om en forventningsverdi.) Bruk O-notasjon og uttrykk svaret som funksjon av n .

Merk: Du skal *ikke* oppgi svaret som funksjon av input-størrelsen til INSERTION-SORT, men som funksjon av input-størrelsen til BUCKET-SORT.

- (6p) 9. $A[1..n]$ er en tabell med heltall. Du har nesten skrevet ferdig en algoritme basert på designmetoden *splitt og hersk* (*divide-and-conquer*) for å finne det minste elementet (eller ett av de minste elementene) i A . Det du har skrevet så langt ser slik ut:

MINIMUM(A, p, r)

```
1  if  $p == r$ 
2      return  $A[p]$ 
3  else  $m = \lfloor (p + r) / 2 \rfloor$ 
4       $x =$ 
5       $y =$ 
6      if  $x < y$ 
7          return  $x$ 
8      else return  $y$ 
```

For å finne minimum i A bruker du MINIMUM($A, 1, n$). Fyll ut det som mangler i de to rutene i pseudokoden over.

(6 p) 10. Løs følgende rekurrens, der $n \geq 0$ er et heltall:

$$T(n) = \begin{cases} 0 & \text{hvis } n = 1, \\ 10\,000 T(n/10) + n^4 + n^2 & \text{hvis } n > 1. \end{cases}$$

Oppgi svaret i Θ -notasjon.

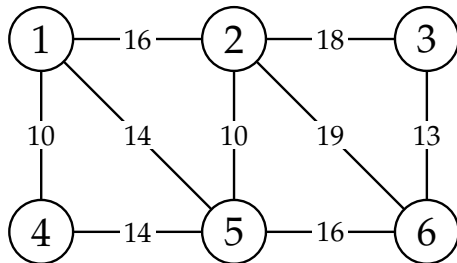
(6 p) 11. Din venn Lurvik mener følgende rekurrens har løsning $T(n) = n^3 - n^2$:

$$T(n) = \begin{cases} 0 & \text{if } n = 1, \\ 8T(n/2) + n^2 & \text{if } n > 1, \end{cases}$$

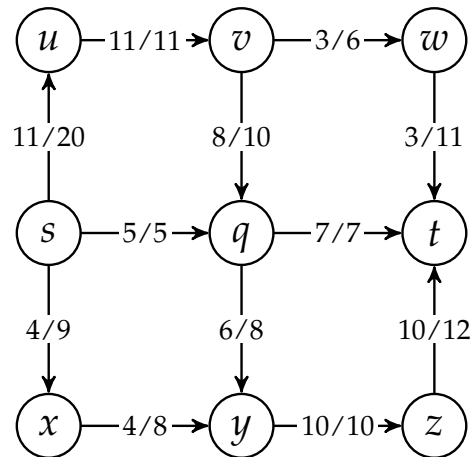
der $n = 2^k$, for et positivt heltall k . Vis at hun har rett.

(6 p) 12. Du har en urettet graf $G = (V, E)$, der hver kant $(u, v) \in E$ har en vekt $w(u, v) = -1$. Du har oppgitt to noder s og t , og ønsker å finne den lengste (dvs. tyngste) stien fra s til t , altså den med størst vekstsum. Hvordan ville du løse problemet?

(6 p) 13. I TRANSITIVE-CLOSURE brukes den binære variabelen $t_{ij}^{(k)}$ til å indikere om det går en sti fra i til j hvis alle noder på veien mellom dem *må* ligge i mengden $\{1, 2, \dots, k\}$. For eksempel er $t_{ij}^{(0)} = 1$ hvis og bare hvis $(i, j) \in E$. Hva er uttrykket for $t_{ij}^{(k)}$, når $k > 0$?



Figur 1: En vektet, urettet graf til bruk i oppgave 14



Figur 2: Flytnettverk brukt i oppgave 15

- (6 p) 14. Hvis du utfører MST-KRUSKAL på grafen i figur 1, hvilken kant vil velges som den femte i rekken? Det vil si, hvilken kant vil være den femte som legges til i løsningen?
Oppgi kanten på formen (i, j) , der $i < j$.

- (6 p) 15. Figur 2 viser flytnettverket G , med kilde s , sluk t og flyt f . Er flyten maksimal? Svar ja eller nei.
Hvis ja, oppgi også mengden av noder som kan nås (dvs., som det finnes stier til) fra s i G_f .
Hvis nei, oppgi også nodene i en flytforøkende sti (*augmenting path*), i rekkefølge.

		1	2	3	4
1	0	3	∞	7	
2	∞	0	∞	4	
3	1	2	0	6	
4	∞	∞	-1	0	

 $D^{(2)}$

		1	2	3	4
1	NIL	1	NIL	2	
2	NIL	NIL	NIL	2	
3	3	3	NIL	2	
4	NIL	NIL	4	NIL	

 $\Pi^{(2)}$

Figur 3: Forrige tilstand i utførelsen av FLOYD-WARSHALL, brukt i oppgave 16

- (6 p) 16. Under en kjøring av FLOYD-WARSHALL er $D^{(2)}$ og $\Pi^{(2)}$ som angitt i figur 3. Hva blir $D^{(3)}$ og $\Pi^{(3)}$? Fyll ut tabellene nedenfor.

Merk: Vi antar her en implementasjon som i læreboka, det vil si at vi i hver iterasjon k lager nye tabeller $D^{(k)}$ og $\Pi^{(k)}$, heller enn en mer plass-effektiv variant som overskriver tabellene.

		1	2	3	4
1					
2					
3					
4					

 $D^{(3)}$

		1	2	3	4
1					
2					
3					
4					

 $\Pi^{(3)}$

- (6 p) 17. Et flytnettverk er en rettet graf $G = (V, E)$ der hver kant $(u, v) \in E$ har en *kapasitet* $c(u, v) \geq 0$. Hvis $(u, v) \notin E$, lar vi $c(u, v) = 0$. En *flyt* i G er en reell funksjon $f : V \times V \rightarrow \mathbb{R}$ som tilfredsstiller to egenskaper. Hvilke egenskaper er dette?

Egenskapene uttrykkes fortrinnsvis med matematisk notasjon. En kort tekstlig beskrivelse kan gi opptil 3 poeng. Det holder *ikke* å oppgi navnene på dem. (Det er heller ikke *nødvendig* å oppgi navnene deres.)

- (6 p) 18. Er følgende utsagn riktig?

«For ethvert språk L , hvis L er i NP, så er L ikke i co-NP.»

Svar ja eller nei, og forklar svært kort.

- (6 p) 19. En *uavhengig mengde* (*independent set*) i en graf $G = (V, E)$ er en delmengde $U \subseteq V$ av nodene som er slik at hver kant i E er tilkoblet maksimalt én node i U (dvs., ingen av nodene i U er naboer). Et *fritt tre* (*free tree*) er en sammenhengende, asyklisk urettet graf (dvs., et tre uten noen angitt rot). Du skal løse følgende problem.

Input: Et fritt tre $T = (V, E)$ der hver node $v \in V$ har en vekt $w(v) \neq 0$.

Output: En uavhengig mengde U i T med maksimal vektsum $\sum_{v \in U} w(v)$.

Se figur 4 for et eksempel. Merk at nodevektene *kan være negative*.

For enkelhets skyld trenger du *ikke* beskrive hvordan du vil finne selve mengden U ; det holder at du finner korrekt vektsum. Beskriv konsist en algoritme som løser problemet effektivt. Oppgi kjøretiden så presist som mulig, i asymptotisk notasjon.

- (6 p) 20. En *dominerende mengde* (*dominating set*) i en urettet graf er en delmengde av nodene som tilsammen er naboer med alle de andre nodene. Det vil si, en dominerende mengde for en urettet graf $G = (V, E)$ er en mengde $U \subseteq V$ som er slik at for enhver node $v \in V - U$ så finnes det minst én node $u \in U$ der $(u, v) \in E$. (Se figur 5 for et eksempel.)

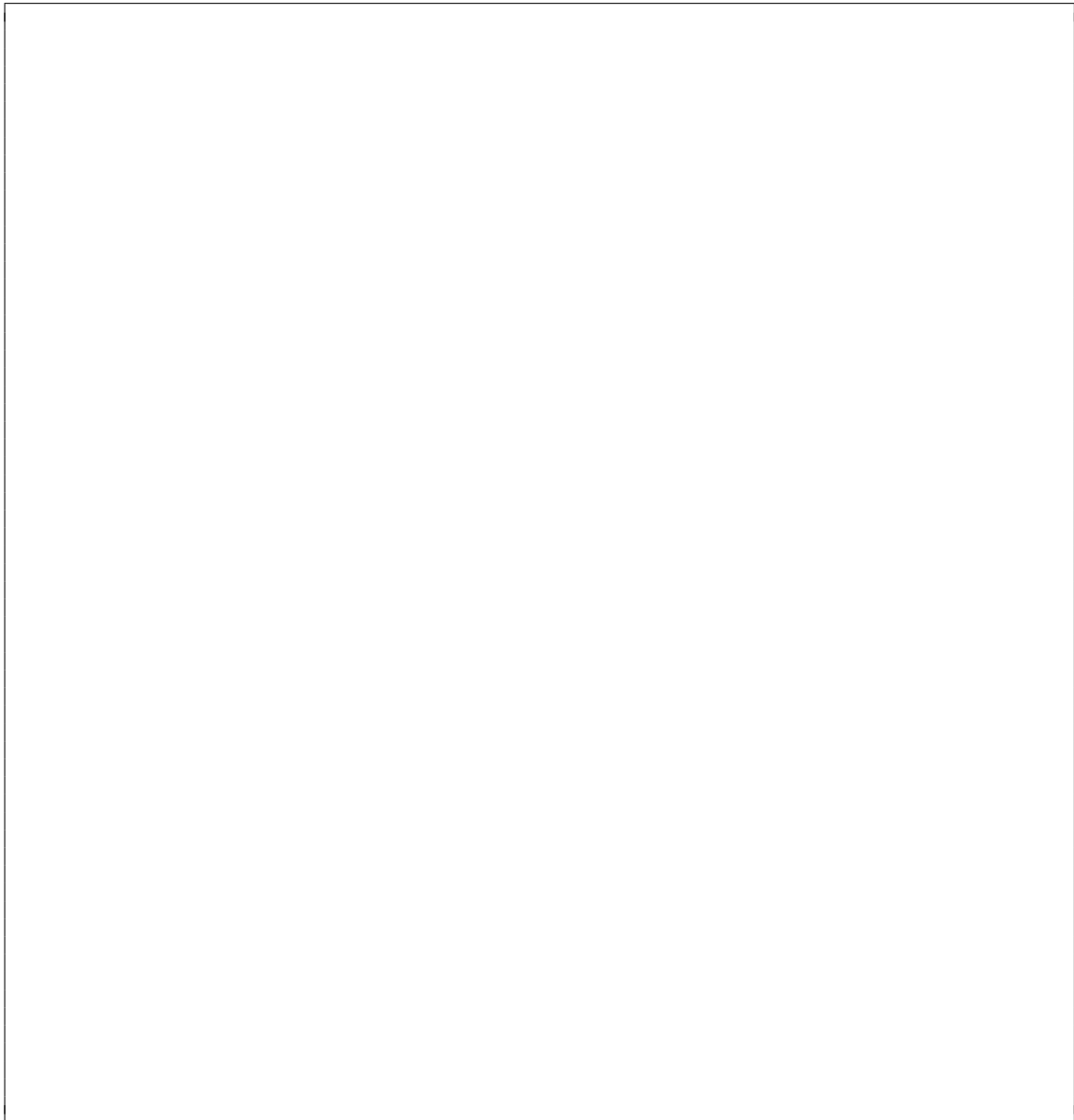
Det såkalte *dominating set problem* handler om å finne en dominerende mengde av minimal størrelse (dvs., med så få noder som mulig) i en gitt graf. Uttrykt som et beslutningsproblem, går det ut på å avgjøre om grafen har en dominerende mengde av en gitt størrelse k . Som et språk, definerer vi

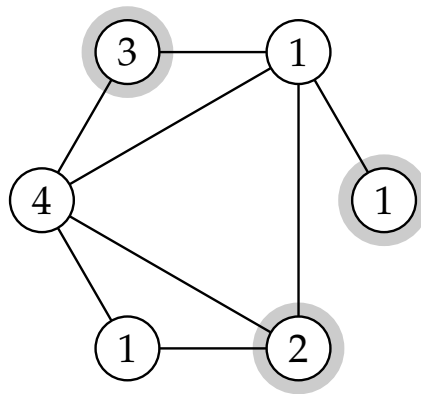
$\text{DOMINATING-SET} = \{ \langle G, k \rangle : \text{grafen } G \text{ har en dominerende mengde med } k \text{ noder} \}.$

Anta at du allerede vet at språket VERTEX-COVER er NP-komplett. Bruk denne kunnskapen til å vise at DOMINATING-SET er NP-komplett.

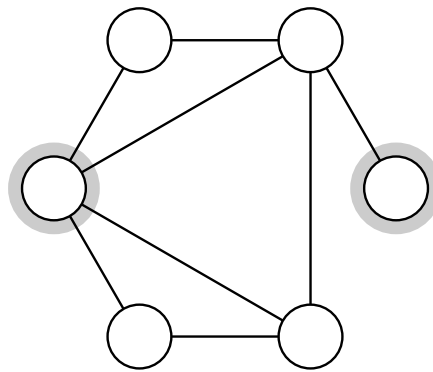
Merk: For full uttelling må alle bestanddelene i et NP-komplethetsbevis være med.

Hint: Anta at noden w kun har u og v som naboer, og at u og v er naboer med hverandre, som i figur 6. Dersom w er med i løsningen vår, kan vi alltid bytte den ut med enten u eller v , og fortsatt ha en løsning (en dominerende mengde) med samme antall noder.

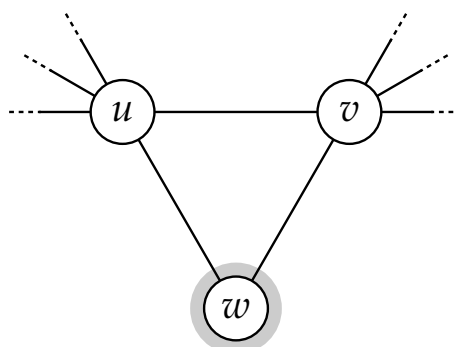




Figur 4: Eksempel på en maksimal, vektet uavhengig mengde (se oppgave 19). De tre uthevede nodene er med i den uavhengige mengden U , som har vektsum $\sum_{v \in U} w(v) = 1 + 2 + 3 = 6$



Figur 5: En dominerende mengde (se oppgave 20) med størrelse 2. De uthevede nodene er med i den dominerende mengden, og hver av de andre nodene har en kant til minst én av dem



Figur 6: Nodene u og v kan være koblet til andre noder i grafen, men w er kun koblet til u og v . Dersom w er med i en dominerende mengde av størrelse k , så kan vi bytte ut w med enten u eller v , og fortsatt ha en dominerende mengde av størrelse k