

**Les dette nøye**

- (i) Les hele eksamenssettet nøye før du begynner!
- (ii) Faglærer går normalt én runde gjennom lokalet. Ha evt. spørsmål klare!
- (iii) 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.
- (iv) 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.

B21

**Oppgaver**

- (5%) 1. (a) Hvilken traverseringsalgoritme i pensum kan brukes til å finne korteste vei i uvektede grafer?

BFS

✓

- (5%) (b) Én nøkkelingrediens i dynamisk programmering er optimal substruktur. Hva er den andre?

Overlapping Subproblems

✓

- (5%) (c) Hva er kjøretiden til RANDOMIZED-SELECT i verste tilfelle (*worst-case*)?

$O(n^2)$   $\Theta(n^2)$

✓

- (5%) (d) RADIX-SORT sorterer etter ett og ett siffer. I hvilken rekkefølge?

Least significant to most significant

✓

- (5%) (e) Hva har grafen hvis BELLMAN-FORD feiler (dvs., returnerer FALSE)?

A negative cycle

✓

- (5%) (f) Hva representerer attributten  $v.\pi$  i (for eksempel) PRIM og DIJKSTRA?

The previous vertex (in the shortest path / minimal spanning tree)

✓

- (5%) (g) Hva er input til kretsen i NPC-beviset for CIRCUIT-SAT (dvs.  $y$ , der  $C(y) = A(x, y)$ )?

Certificate for verifying the decision problem.

✓

2. I de følgende deloppgavene, sørg for å ikke «kaste bort» informasjon under forenklingen. Gi tette grenser og bruk den mest presise asymptotiske notasjonen du kan (av  $O$ ,  $\Omega$  eller  $\Theta$ ).

- (5%) (a) Forenkle uttrykket  $\Theta(n) + O(n^2) + \Omega(n^3)$ .

$\Omega(n^3)$  ✓

- (5%) (b) Forenkle uttrykket  $\Theta(n + \sqrt{n}) + O(n^2 + n) + \Omega(n \cdot (n + \lg n))$ .

~~XXXXXXXXXX~~  $\Omega(n^2)$  ✓

- (5%) (c) Løs rekurensen  $T(n) = T(n/3) + \lg n$ . Skriv svaret med  $\Theta$ -notasjon.

$\Theta(\lg n)$  ✓

3. Dine venner Smartnes og Lurvik har laget hver sin algoritme som tar inn en binær maks-haug (*binary max-heap*) som input. Haugen er representert som en tabell  $A[1..n]$  på vanlig måte. Alle elementene er forskjellige og haugen er *komplett*.<sup>1</sup>

- (5%) (a) Lurvik sin algoritme baserer seg på antagelsen at de  $\lceil n/2 \rceil$  minste elementene alltid vil ligge sist (dvs., i løvnodene). Stemmer det? Begrunn svaret kort.

No, for instance

```

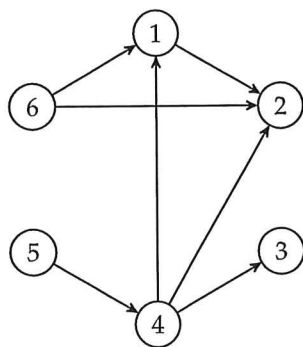
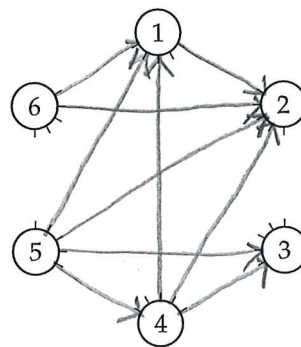
      10
     /  \
    8    5
   / \  / \
  7 6 3 2
  
```

is a counterexample since  $5 < \text{some of the leaves}$ . ✓

- (5%) (b) Smartnes mener hun har en algoritme som utnytter haug-strukturen og bygger et balansert binært søketre for elementene til  $A$  i lineær tid. Tror du hun kan ha rett? Begrunn svaret kort.

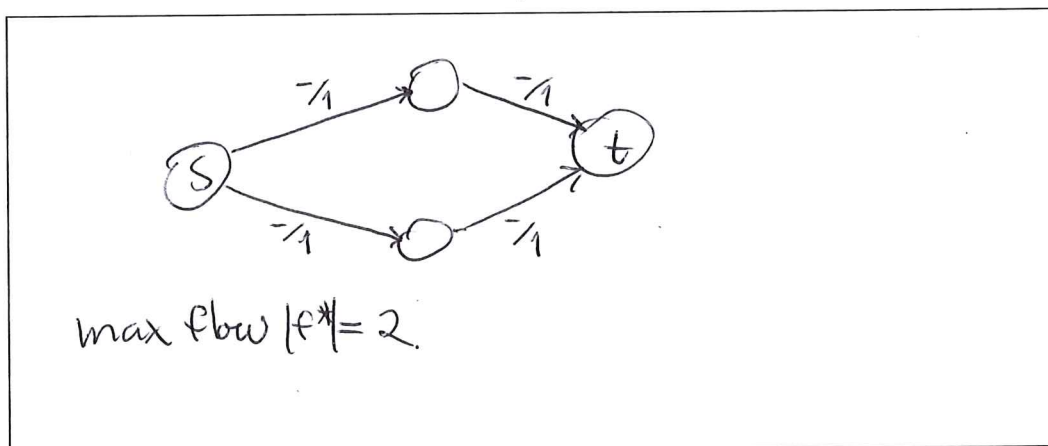
This sounds reasonable, since if we sort nodes on the same level, the heap will turn into a binary search tree.  
It's not however obvious that this sorting will take linear time, but since we sort atmost  $\lceil \frac{n}{2} \rceil$  elements at the same time, it may. ✗

<sup>1</sup>Alle interne noder har to barn og alle løvnoder er på samme nivå, dvs.,  $n = 2^k - 1$  for et heltall  $k \geq 1$ .

Figur 1:  $T^{(3)}$  – brukt i oppgave 4Figur 2:  $T^{(4)}$  – svar på oppgave 4

- (5%) 4. Du skal simulere én iterasjon av TRANSITIVE-CLOSURE. I figur 1 ser du den rettede grafen som tilsvarende  $T^{(3)}$ . Du skal beregne  $T^{(4)}$  og tegne den tilhørende rettede grafen. Fyll inn kantene som mangler i figur 2. Angi retning tydelig.  
(Merk: Hver  $T^{(k)}$  er en helt ny tabell. Dvs., vi gjenbraker ikke den samme i hver iterasjon.)

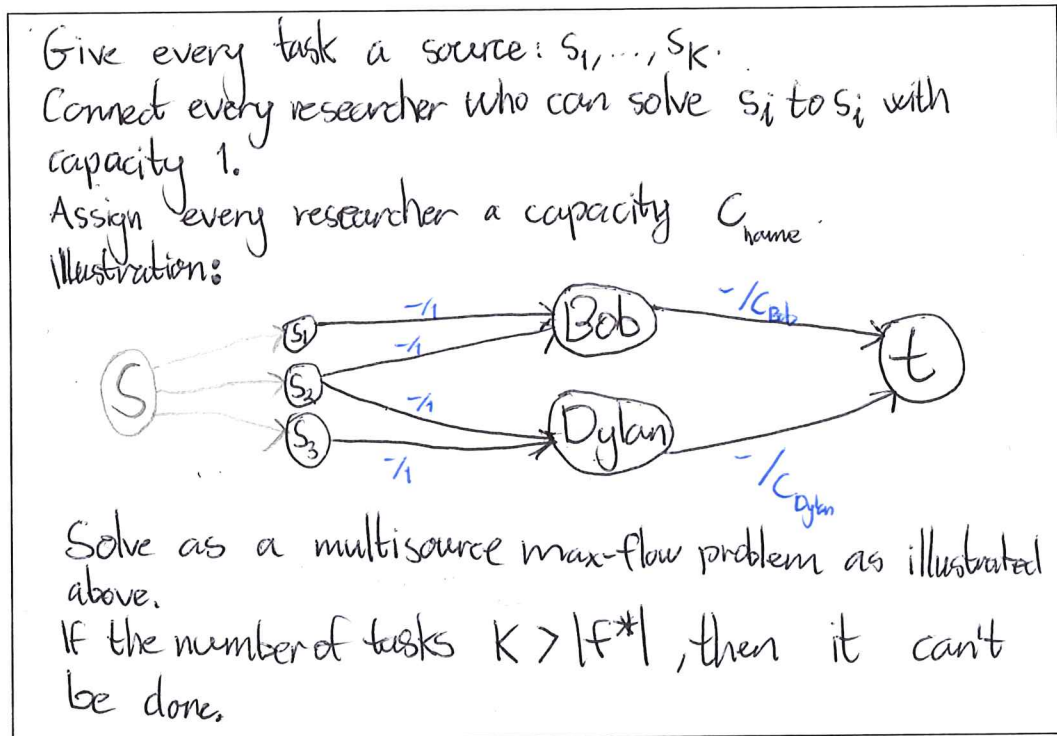
- (5%) 5. (a) Tegn et flytnettverk der kapasitetene er oddetall men der maksimal flytverdi er et partall.



- (5%) (b) I et flytnettverk der kapasitetene er partall vil maksimal flytverdi være et partall. Hvorfor?

Because the minimum cut must be even.  
Also, using the Ford-Fulkerson method, we could always augment by 2, so the max flow would be a multiple of 2.

- (5%) (c) Et team med forskere skal gjennomføre et sett med prosjekter.
- Hvert prosjekt består av flere oppgaver, og et visst antall av prosjektets oppgaver (f.eks. 7 av 12) må gjøres. Ulike prosjekt kan ha ulike antall.
  - Hver oppgave skal utføres av én forsker.
  - Hver forsker har en viss kapasitet, dvs., et antall oppgaver hun rekker å gjøre.
  - Hver forsker er kompetent til å gjøre noen av oppgavene, men ikke nødvendigvis alle.
- Du skal avgjøre hvem som skal gjøre hvilke oppgaver, eller finne ut at det ikke går. Beskriv hvordan du kan løse dette som et maks-flyt-problem. Tegn gjerne et flytnettverk.

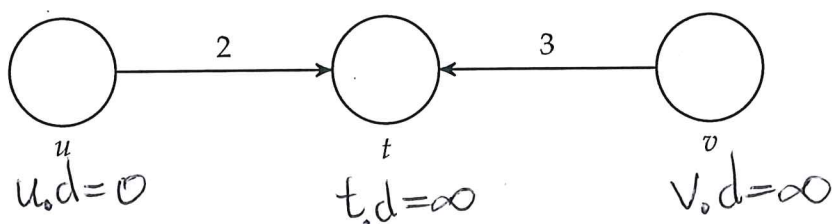


6. Din venn Klokland vil finne korteste vei til en node  $t$  (single-destination shortest-path) i en rettet graf  $G = (V, E)$  med vektfunksjon  $w : E \rightarrow \mathbb{R}$ . Han har endret på RELAX så den oppdaterer  $u$  basert på  $w(u, v)$  og  $v.d$ , i stedet for omvendt.

Han bruker så denne nye RELAX' i en modifisert versjon av DFS: Først initialiserer han grafen som vanlig, og når den rekursive traverseringen returnerer til  $u$  etter å ha besøkt  $v$  (og farget den svart), bruker han sin RELAX' på kanten  $(u, v)$ . (Om ønskelig, se pseudokode på s. 6 for presisering.)

Klokland påstår at algoritmen fungerer for rettede, asykliske grafer. Du er skeptisk, og vil lage et moteksempel. Resultatet er grafen som er vist nedenfor.

- (3%) (a) Hva er resultatet av algoritmen om  $u$  besøkes først? Fyll inn  $u.d$ ,  $t.d$  og  $v.d$  nedenfor.





- (7%) (b) Beskriv kort hvordan algoritmen kan endres så den blir korrekt for (vektede, rettede) asykliske grafer. (Her holder det med en svært kort forklaring.)

3 modifications:

- 1) Change  $w(u,v)$  to  $w(v,u)$
- 2) Relax on edge  $(v,u)$  instead of  $(u,v)$
- 3) Relax before DFS-VISIT.

7. Du får oppgitt et spillbrett som vist i figur 3 (på side 6), med  $n \geq 3$  ruter, der du starter helt til venstre og skal flytte mot høyre. I rutene står det hvor langt du maksimalt kan flytte i ett trekk når du står på ruten. Målet er å komme frem til den siste ruten med så få trekk som mulig.

(Du kan anta input er slik at at det alltid er mulig å komme frem til den siste ruten.)

- (7%) (a) Beskriv en algoritme som effektivt finner ut hvor mange trekk du må bruke. (Du trenger ikke finne ut hvilke trekk.) Beskriv den gjerne med egne ord, eller bruk formler, figurer, kode eller pseudokode etter eget ønske.

MINIMUM-JUMPS(A):

Create cost array  $C = [0, \infty, \dots, \infty]$

for  $i = 1$  to  $A.length$ :

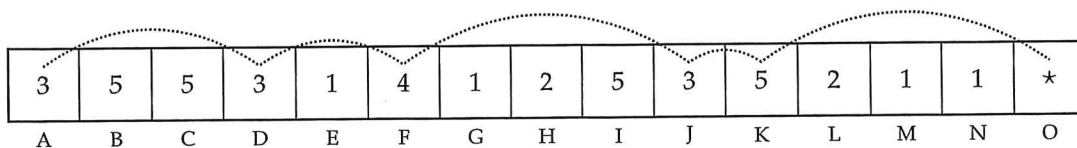
max\_jump =  $\min\{A[i], A.length - i\}$

for  $j = 1$  to max\_jump:

$C[i+j] = \min\{C[i+j], C[i] + 1\}$

return  $C[A.length]$  // last element of C

Explanation:  $C[i]$  is the minimum required jumps to reach position  $i$ ,



Figur 3: Eksempel til oppgave 7. I første trekk kan du flytte fra rute A til B, C eller D. Flytter du til D kan du så flytte til E, F eller G i ett trekk. (Merk: Eksempel-løsningen er ikke optimal.)

- (3%) (b) Hva blir kjøretiden i verste tilfelle? Oppgi svaret i  $\Theta$ -notasjon. Forklar kort.

$$\underline{\underline{\Theta(n^2)}}$$

The outer loop will run  $n$  times. The inner will run atmost  $n-i$  times.

$$\Rightarrow \text{Total} \sim n \cdot \sum_{i=1}^n (n-i) = \Theta(n^2)$$

### Pseudokode til oppgave 6

Det følgende er en mer detaljert beskrivelse av algoritmen som omtales i oppgave 6. INITIALIZE-SINGLE-SOURCE er som beskrevet i pensum. Pseudokoden er ment som et supplement, og det er mulig å besvare oppgaven uten den. Merk at det *ikke* er nødvendig å skrive pseudokode, eller referere til denne koden, i besvarelsen.

RELAX'(u, v, w)

```

1  if  $u.d > v.d + w(u, v)$ 
2       $u.d = v.d + w(u, v)$ 
3       $u.\pi = v$ 

```

DFS'(G, t)

```

1  INITIALIZE-SINGLE-SOURCE(G, t)
2  for each vertex  $u \in G.V$ 
3       $u.color = \text{WHITE}$ 
4  for each vertex  $u \in G.V$ 
5      if  $u.color == \text{WHITE}$ 
6          DFS-VISIT'(G, u)

```

DFS-VISIT'(G, u)

```

1   $u.color = \text{GRAY}$ 
2  for each  $v \in G.Adj[u]$ 
3      if  $v.color == \text{WHITE}$ 
4          DFS-VISIT'(G, v)
5      RELAX'(u, v, w)
6   $u.color = \text{BLACK}$ 

```

Department of Computer and Information Science

## Examination paper for TDT4120 Algorithms and Data Structures

**Academic contact during examination** Magnus Lie Hetland  
**Phone** 918 51 949

**Examination date** December 4, 2017  
**Examination time (from-to)** 09:00–13:00  
**Support material code** D

**Other information** The problem sheets are handed in, with answers in answer boxes under the problems

**Language** English  
**Number of pages (front page excluded)** 6  
**Number of pages enclosed** 0

**Informasjon om trykking av eksamensoppgave**

**Originalen er**

**1-sidig** ☒ **2-sidig** ☐

**sort/hvit** ☒ **i farger** ☐


**Skal ha flervalgskjema** ☐

**Quality assured by**

Pål Sætrom

**Checked by**

27/11-17  
Date

  
Signature