LØSNINGSSKISSE / EKSAMEN I FAG
45011 ALGORITMER OG DATASTRUKTURER

Fredag 20. desember 1996, kl 0900 - 1300

Oppgave 1

(a) M=7, A=6*4 = 24

(b) $\Theta(n)$

(c)
```
function P((x,y:vektor; i,j:indeks) : real;
var k : indeks;  sp : real;
begin sp := 0.0;  for k := i to j do sp := sp + x[i] * y[i] ;
        P := sp;
end;
```

Oppgave 2

Godkjenner tre-rettinger som gir E I E K L M K S P T O T (minst først),
eller T S T P O M K K I L E E (størst først)
Dersom en ved likhet ikke lar laveste indeks rykke opp, fås:
E E K I L M K S P T O T, og tilsv.

Oppgave 3

(a) Følgende prosedyre-endringer / tillegg gjør jobben:

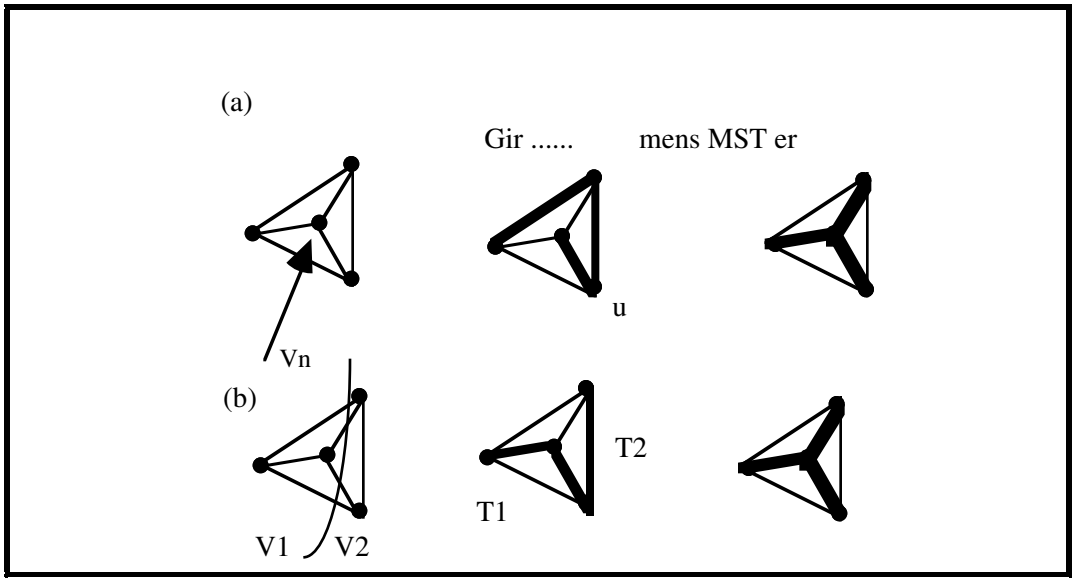
Initialize-S-S :
Føy til:
AntKant(v) := ∞ ; AntKant(s) := 0 ;
Relax:
Endre til:
if d(v) > d(u) + w(u,v)
OR (d(v) = d(u) + w(u,v) AND AntKant(v) > AntKant(u) +1)
THEN d(v) := d(u) + w(u,v)
π(v) := u
AntKant(v) := AntKant(u) +1

nb: Ingen endring i Extract-Min er nødvendig, men det blir ikke galt om en ved likhet trekker ut en node med minst kantantall fra s. (Noe bør trekkes for å gjøre noe unødvendig.)

(b) Problemet er at vektene w i Dijkstras algoritme er antatt å kunne være av typen Real. Dette betyr at differansen mellom lengste og nest lengste vei kan være vilkårlig liten og ukjent inntil problemet er løst. Det er derfor generelt ikke mulig å endre datasettet for å finne en korteste vei med det minste antall kanter fra s.
I spesielle, men likevel vanlige, tilfeller ville en løsning som følger kunne fungere: Desimaldelen i w-vektene er .0 : Siden differansen mellom korteste og nest korteste vei da blir minst 1.0, kunne en få valgt vei ved å addere Δw = 1/antall kanter(E) til alle vektene. Summen av Δw-bidragene kan da ikke overstige 1.0 fordi høyst alle kanter inngår i veien.

Oppgave 4

Svaret er NEI på både (a) og (b) (ingen av algoritmene gir generelt et MST). Dette vises mest rasjonelt med så enkle moteksempler som mulig:



Oppgave 5

(a) $r\Theta(s) + n(\Theta(r) + \Theta(s)) = ... = \Theta(n(r+s)) = \Theta(n(r+n/r))$ (forenkling kreves)

(b) Optimal r:s deling gir at metoden blir av orden $\Theta(n\sqrt{n})$

(c) Algoritme S er bedre enn Quicksort i Worst-case tilfellene, idet en da sammenligner en n^2-metode med en $n\sqrt{n}$-metode. I gjennomsnittlige tilfeller er Quicksort (n logn) bedre enn en $n\sqrt{n}$-metode, unntatt i (det åpne) intervallet n= 4..16. Quicksort anbefales erstattet at Sortering v/ins(m)etting dersom n > 10, og en står da igjen med et lite (åpent) intervall 11..16, der Alg. S er bedre en Quicksort, forutsatt at konstantene da ikke snur bildet. Uansett kan Alg. S kun i marginale tilfeller anbefales framfor Quicksort. (Elementer av denne diskusjonen må være med for å få poeng her.)

Oppgave 6

La T være summen av alle t(i)-verdiene. Problemet vårt er da et (0,1)-ryggsekk-problem (ZOK), der det her gjelder å pakke sekken nærmest opp til verdien T/2. Problemet er NP-komplett. (Vi har, på ett eller annet enhetsnivå (sekund, millisekund...) med "heltall" å gjøre.) Merk at Dynamisk Programmering krever heltall og en matrise med T/2 rader, og at en (f.eks) 10-dobling av T også 10-dobler matrisestørrelsen.