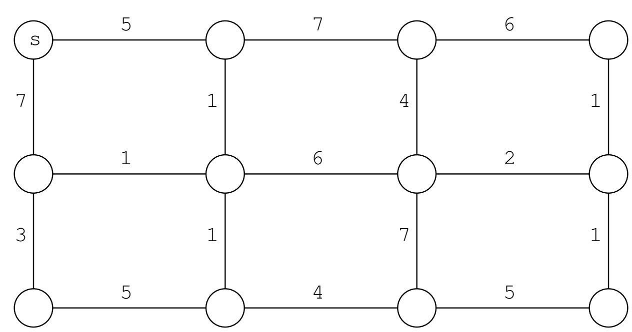
Universitetet i Bergen
Det matematisk-naturvitenskaplige fakultet
Eksamen i emnet Inf102 - Algoritmer, programmering og datastrukturer
Fredag 15. desember 2006, kl. 09-12
Bokmålstekst

Tillatte hjelpemidler: Kalkulator.

Du trenger ikke skrive Java-kode for noen av oppgavene. Det er tilstrekkelig med en skriftlig beskrivelse av algoritmene du lager, ev. kan du skrive pseudo-kode. Alle kjøretider skal gis med O-notasjon.

Oppgave 1

Gitt følgende vektete graf.



a) Marker kantene i det minste utspennende treet du får ved bruk av Prim-Jarnik algoritmen med start i noden merket s.

b) Hva er kjøretiden til Prim-Jarnik algoritmen dersom du bruker en haug (eng. *heap*) for å implementere prioritetskøen i algoritmen?

c) Hva er kjøretiden til Prim-Jarnik algoritmen dersom du bruker en usortert liste for å implementere prioritetskøen i algoritmen?

d) Diskuter når det kan være en fordel å bruke de ulike implementasjonene i b) og c).

Oppgave 2

a) Vis hvordan Quicksort sorterer følgende sekvens med tall når pivot-elementet alltid velges som siste element: 32, 67, 7, 92, 79, 71, 16, 41, 25, 21.

b) Utled beste kjøretid for Quicksort. Gi et eksempel med 7 tall som viser når denne forekommer.

c) Utled verste kjøretid for Quicksort. Gi et eksempel med 7 tall som viser når denne forekommer.

Oppgave 3
Gitt en tabell $A[n]$ med heltall (indekseringen av A starter fra 1 og går til og med n).

a) Beskriv og analyser kjøretiden til et effektivt program som fyller i verdiene i en heltallstabell $B[n]$ slik at hvert element $B[i] = \Sigma_{k=1}^{i} A[k]$, $1 \leq i \leq n$. Dvs. $B[i]$ inneholder summen av elementene fra og med $A[1]$ til og med $A[i]$.

b) Beskriv og analyser kjøretiden til et effektivt program som fyller i verdiene i en heltallstabell $C[n][n]$ slik at hvert element $C[i][j] = \Sigma_{k=i}^{j} A[k]$, $1 \leq i, j \leq n$. Dvs. $C[i][j]$ inneholder summen av elementene fra og med $A[i]$ til og med $A[j]$.

Oppgave 4
Marker hvilke av følgende utsagn som er sanne og hvilke som er usanne. Gi en kort begrunnelse for hvert svar.

a) Søking i en sortert liste med n elementer tar tid $O(\log n)$.

b) Sletting i et binært søketre tar tid $O(\log n)$.

c) Høyden til en heap kan bli større enn $\log n$.

I oppgavene d) og e) ser vi på en hash tabell med N bøtter og n elementer der vi bruker (usorterte) kjedete lister (eng. *separate chaining*).

d) get(x) operasjonen tar $O(\lceil n/N \rceil)$ tid.

e) put(x) operasjonen tar $O(\lceil n/N \rceil)$ tid.

f) Det tar $O(n)$ tid å bygge en heap med n elementer.

g) Radix-sortering av n 64-bits heltall tar $O(n)$ tid.

h) Dybde-først traversering av en graf er mer minne-effektivt enn Bredde-først traversering av den samme grafen.

i) Topologisk sortering av en rettet graf med n noder og m kanter tar $O(m \log n)$ tid.

j) Kruskals algoritme og Prim-Jarniks algoritme returnerer alltid det samme minste utspennende treet.

Fredrik Manne