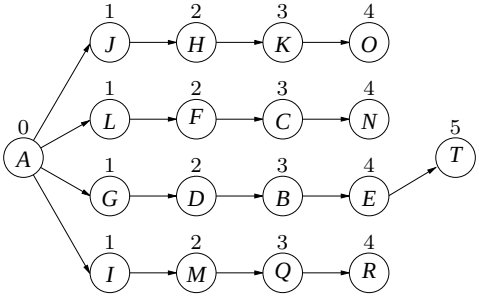
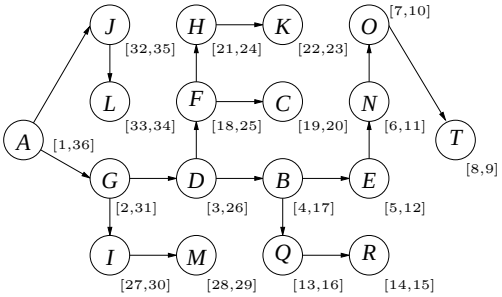
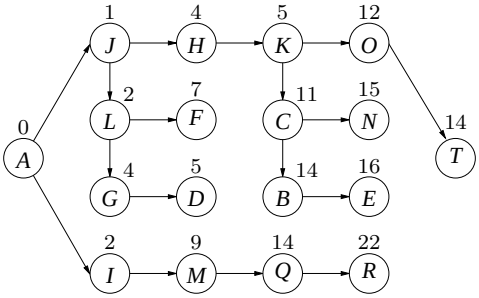
1a



1b



1c



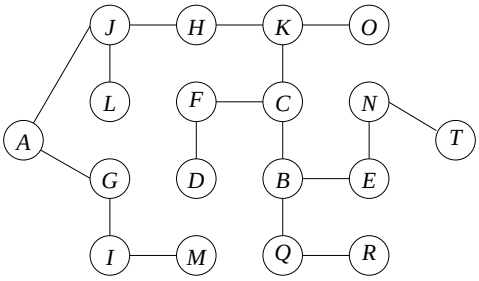
1d

$\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{G\}, \{H\}, \{I\}, \{J\}, \{K\}, \{L\}, \{M\}, \{N\}, \{O\}, \{Q\}, \{R\}, \{T\}$

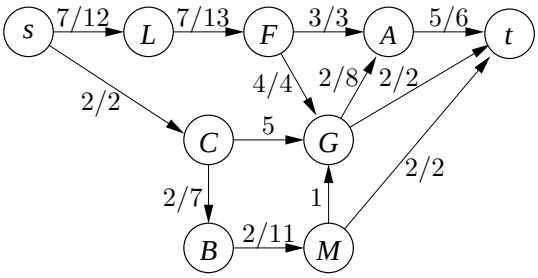
1e

$AJLGIMDFHKCBQRENOT$

1f



2a



Maximal strømning = 9.
Snit med kapacitet 9: $(\{s, L, F\},\{A, B, C, G, M, t\})$

2b

Sti	Forbedring
$sCGt$	2
$sLFAt$	3
$sLFGAt$	3
$sLFGCBMt$	1

3a

Da hver knude højst har to udgående kanter, er antal kanter $\leq 2n$, hvor n er antal knuder i grafen. For at finde den korteste vej fra s_1 til t, relaxer vi først alle kanterne i den yderste ring fra s_1 og imod uret. Derefter relaxer vi alle kanter fra den yderste til den næstyderste ring. Derefter tager vi ringene udefra og indefter: For hver ring relaxer rundt langs ringen to gange, og relaxer derefter alle kanter til den næste ring. Da alle kanter højst relaxeres to gange bliver tiden $O(n)$.

3b

Vend alle kanterne i modsat retning, og find den korteste vej fra t til alle s_i som i spørgsmål (a) ved at tage cyklerne indefra og udefter. Tid $O(n)$.

3c

Lav en ny graf hvor alle negative kanter i den oprindelige graf fjernes, og alle andre kanter får vægt 1. Lav en ny knude s der er forbundet til alle s_i med kanter med vægt 1. Kør For-Fulkerson på den resulterende graf. Returner værdien af det fundne flow. Da flowet højst er k, findes højst k forbedrende stier, hver i tid $O(n)$. Total tid $O(nk)$.

4a

Korteste hundesnor $29 = d(p_2, q_5)$. Mulig tur:

p_1	p_2	p_2	p_2	p_2	p_2	p_3	p_4	p_5
q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_7	q_8

4b

Hvis `D[i][j]` angiver $d(p_i, q_j)$ har vi løsningen:

```
for i=1 to n
  for j=1 to m
    if i=1 and j=1 then L[1,1]=D[1,1]
    if i=1 and j>1 then L[i,j]=max(D[1,j],L[1,j-1])
    if i>1 and j=1 then L[i,j]=max(D[i,1],L[i-1,1])
    if j>1 and i>1 then L[i,j]=max(D[i,j],min(L[i-1,j-1],L[i,j-1],L[i-1,j]))
return L[i,j]
```

Tid $O(nm)$.

4c

```
code from 4b)
report(n,m)

proc report(i,j)
  if i=1 and j>1 then report(1,j-1)
  if i>1 and j=1 then report(i-1,1)
  if i>1 and j>1 then
    if      L[i,j]>=L[i,j-1] then report(i,j-1)
    else if L[i,j]>=L[i-1,j] then report(i-1,j)
    else report(i-1,j-1)
  print p_i,q_j
```

Tid $O(nm)$.

5a

Positionerne $\{6, 7, 15\}$ dækkes ikke: $T = \underline{\texttt{a c a}}\,\underline{\texttt{b a}}\,\texttt{b c}\,\underline{\texttt{a c a}}\,\underline{\texttt{b}}\,\underline{\texttt{a b b}}\,\texttt{a}\,\underline{\texttt{a c a}}\,\underline{\texttt{b b}}$

5b

Løsning 1: For hver streng S_i, kør KMP med S_i og T, og marker alle positioner i T der dækkes af S_i. Tid $O(n + |S_i|)$ for hver S_i, dvs. total tid $O(kn + N)$.

Løsning 2: Konstruer suffixtræet for T i tid $O(n)$. For hver S_i find kanten i suffix træet hvor strengen S_i ender, og marker kanten med $|S_i|$, i tid $O(N)$. Lav et DFS gennemløb af suffix-træet og marker hvert blad med den maksimale længde $|S_i|$, der står på en kant fra roden ned til bladet, i tid $O(n)$. Marker hver position j i T med markeringen af bladet svarende til suffixet $T[j..n]$ i tid $O(n)$, dvs. $T[j]$ er markeret med længden af det største S_i der har et match startende i $T[j]$. Løb T igennem fra venstre mod højre, hvor man hele tiden husker det højreste indeks, der kan dækkes af et S_i der har et match startende i eller til venstre for den aktuelle position, i tid $O(n)$. Ikke dækkede positioner udskrives undervejs i dette sidste gennemløb. Total tid $O(n + N)$.