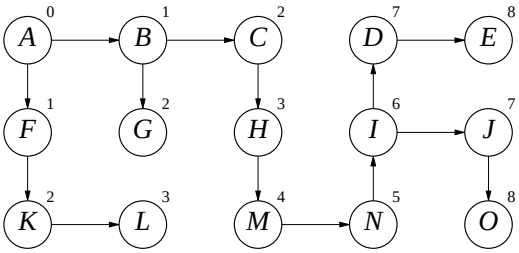
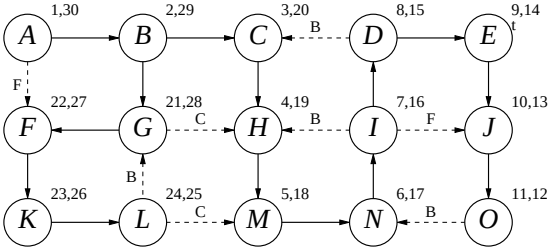
1a



Indsættelser i Q: $A, B, F, C, G, K, H, L, M, N, I, D, J, E, O$

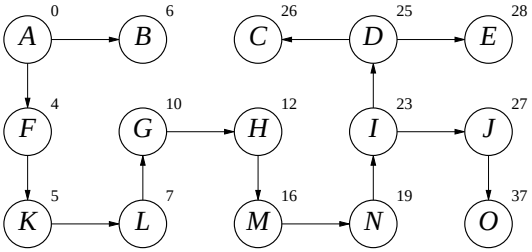
1b



1c

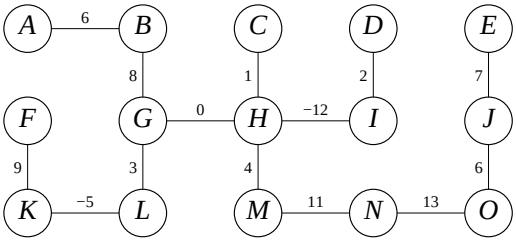
$\{A\}, \{B\}, \{F, G, K, L\}, \{C, D, E, H, I, J, M, N, O\}$

1d



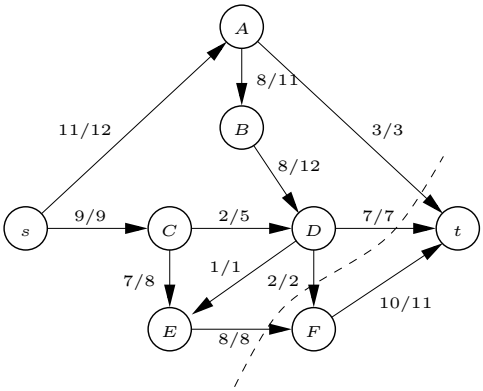
Fjernelser fra $A, F, K, B, L, G, H, M, N, I, D, C, J, E, O$

1e



Fjernelser fra Q: $A, B, G, H, I, C, D, L, K, M, F, N, O, J, E$

2a



Maximal strømning = 20.
Snit med kapacitet 20: $(\{s, A, B, C, D, E\},\{F, t\})$

2b

Forbedring	Sti
3	sAt
5	$sCDt$
2	$sABDt$
4	$sCEFt$
2	$sABDFt$
1	$sABDEFt$
3	$sABDCEFt$

3a

Kør BFS startende i p_i, hvor man kun kigger på kanter med længde $\leq R$, og rapporter BFS afstanden til p_j. Tid $O(n^2)$.

3b

Kør Prim's MST algoritme på den komplette graf i tid $O(n^2)$. R er den største kantvægt i MST-træet. Tid $O(n^2)$.

3c

Betragt grafen med alle kanter med længde højest R. Lav et flow-netværk, hvor $s = p_i$ og $t = p_j$, og alle *knuder* har kapacitet 1. Før Ford-Fulkerson algoritmen, indtil der er fundet to forbedrende stier fra s til t, eller at der ikke findes to forbedrende stier. Tid $O(n^2)$.

3d

Sorter alle kantvægtende. Lav binær søgning på kantvægtene efter en mindste kommunikationsradius, hvor der der for hver kantvægt anvendes 3c) til at checke om den aktuelle afstand er for lille eller tilstrækkelig. Tid $O(n^2 \log n)$.

4a

$i = 1$
for $j = 1$ to n
  if $i \leq m$ and $P[i] = T[j]$ then
    $i = i + 1$
if $i > m$ then print "P er en delsekvens af T"
else print "P er ikke en delsekvens af T"

Tid $O(n)$.

4b

$C(i,j)$	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	$+\infty$	$+\infty$	1	1	1	1	1	1	1
2	$+\infty$	$+\infty$	$+\infty$	$+\infty$	2	2	1	1	1
3	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	2

4c

for $i = 0$ to m
  for $j = 0$ to n
    if $i = 0$ then $C[i,j] = 0$
    else if $i > j$ then $C[i,j] = +\infty$
    else
      $d = 0$
      while $d < i$ and $P[i-d] = T[j-d]$ do
        $d = d + 1$
      $C[i,j] = C[i,j-1]$
      $D[i,j] = 0$ (for 4d)
      if $d > 0$ and $1 + C[i-d, j-d] < C[i,j]$ then
        $C[i,j] = 1 + C[i-d, j-d]$
        $D[i,j] = d$ (for 4d)
return $C[m,n]$

Tid $O(nm^2)$.

4d

code from 4c)
if $C[m,n] = +\infty$ then print "P is not a subsequence of T"
else report(m,n)

proc report(i,j)
  if $i > 0$ then
    $d = D[i,j]$
    if $d = 0$ then report$(i, j-1)$
    else
      report$(i-d, j-d)$
      print "positions $j - d + 1$ to j in T"

Tid $O(nm^2)$.

5a

Check v.h.a. KMP algoritmen om B forekommer i AA. B er et præfiks-suffiks swap af A hvis og kun hvis B forekommer i AA. Tid $O(n)$.

5b

Konstruer suffix-træet for $PP\#T\$$. Marker i et gennemløb af suffixtræet for hver knude om der i undertræet er et blad der svarer til et suffix startende i henholdsvis PP og T. Et præfiks-suffiks swap af P forekommer i T hvis og kun hvis der findes en knude hvor strengen α fra roden ned til knuden har længde mindst m, og i knudens undertræ findes suffikser der starter i både PP og T (og derfor har α som præfiks). Tid $O(n)$.