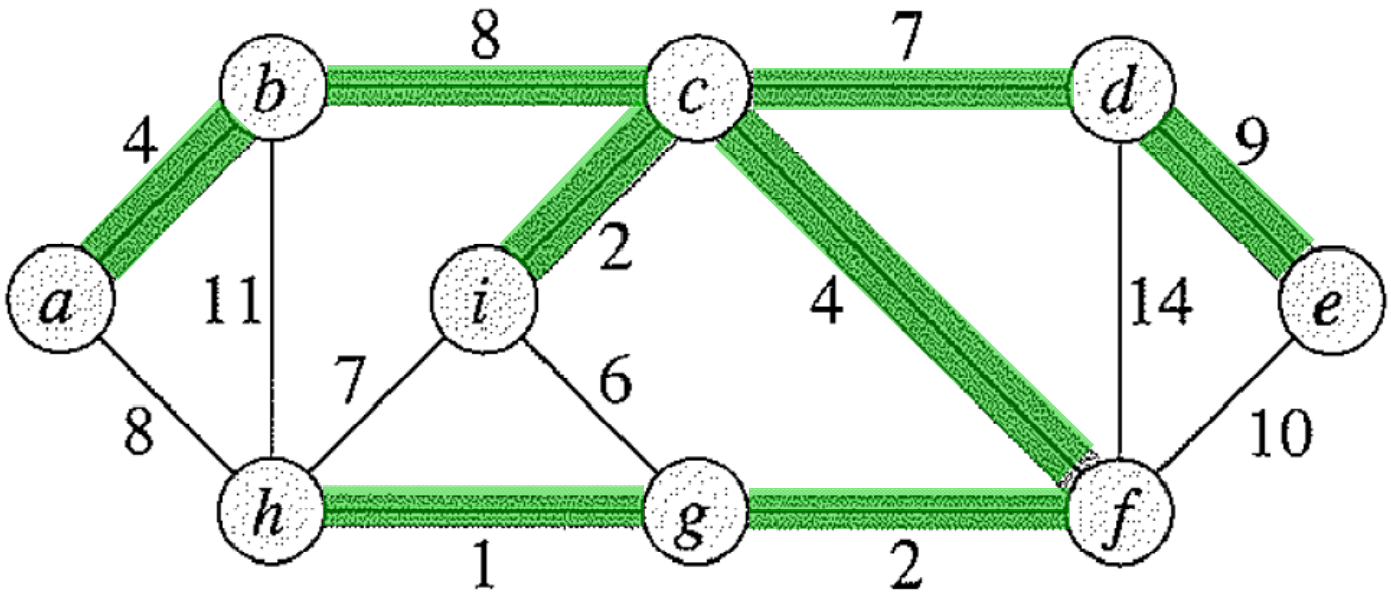
Algoritmer og Datastrukturer 2

Gerth Stølting Brodal

Minimum Udspændende Træer (MST)
[CLRS, kapitel 23]

AARHUS UNIVERSITET

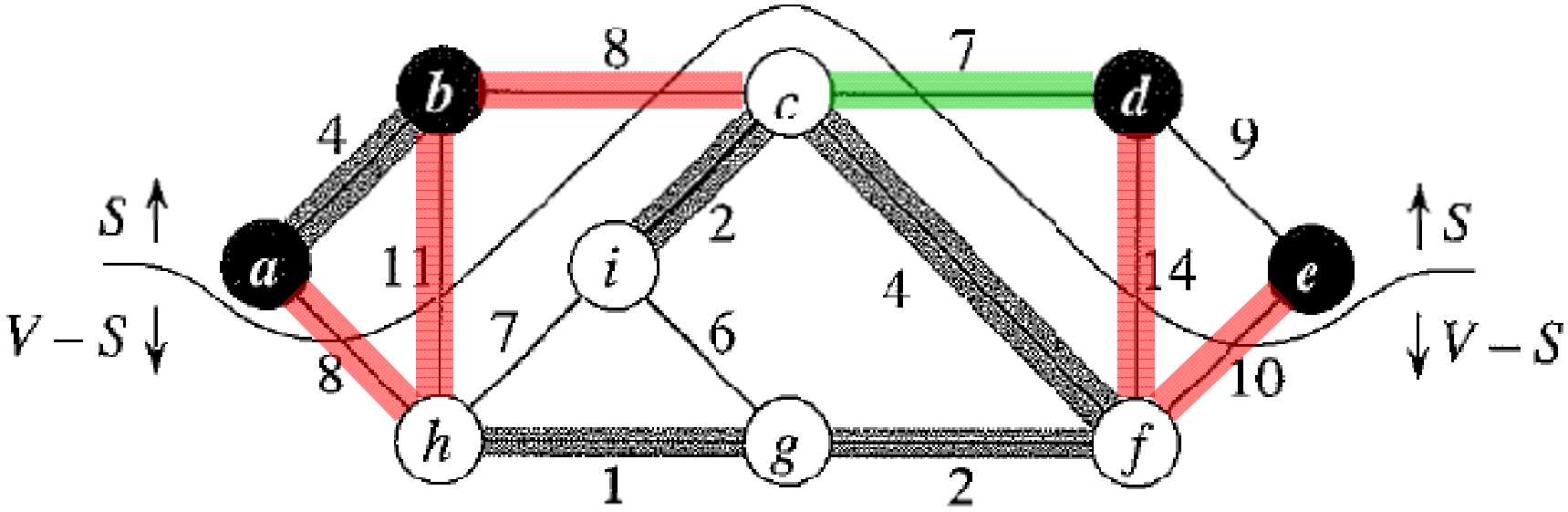
Minimum Udspændende Træ (MST)



Problem

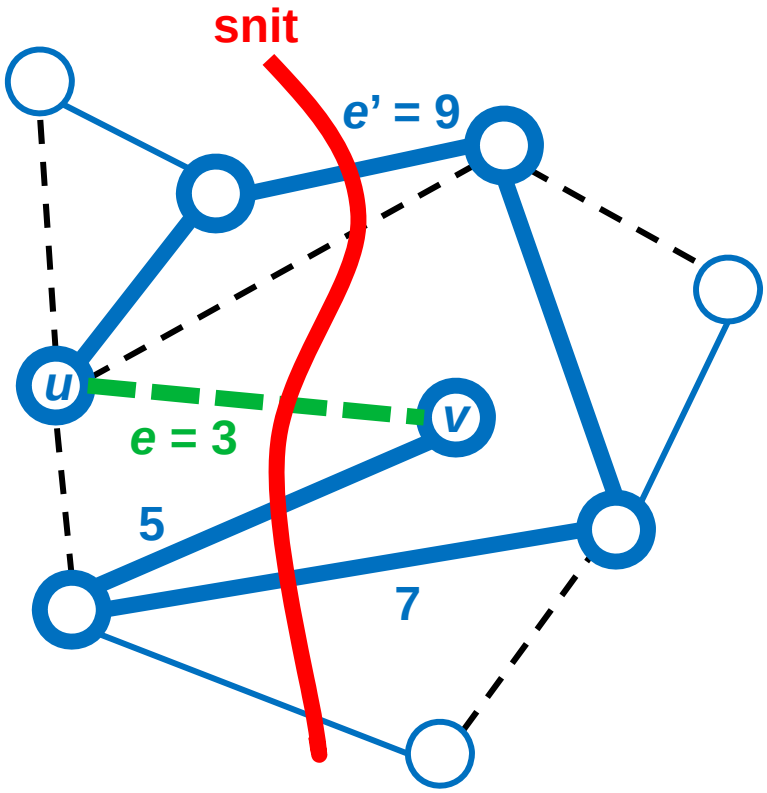
Find et udspændende træ for en sammenhængende uorienteret vægtet graf således at summen af kanterne er mindst mulig

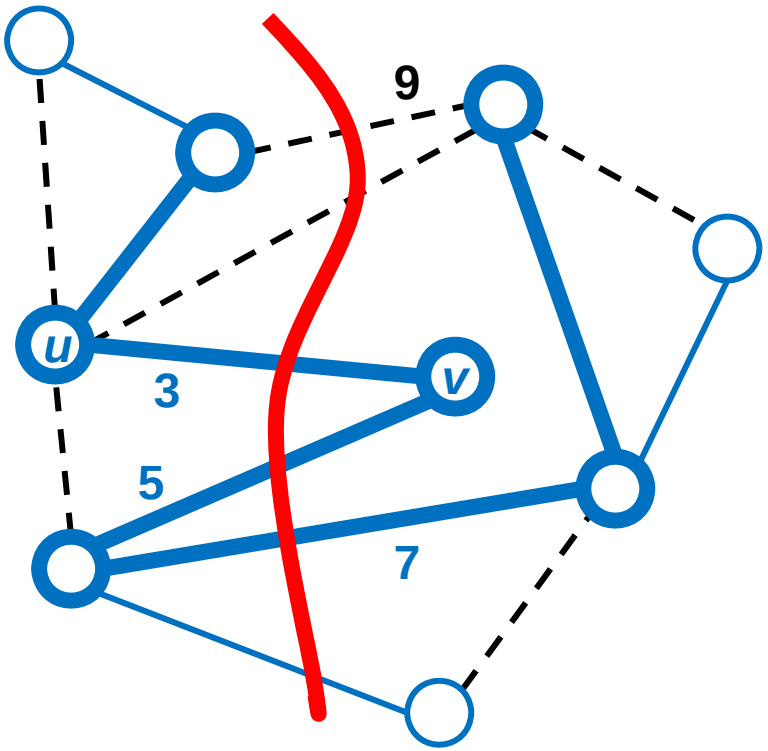
Minimum Udspændende Træer: Snit



Sætning

Hvis alle vægte er forskellige, så gælder der for *ethvert* **snit (*S*,*V*-*S*)** at den **letteste kant** der krydser snittet er med i et minimum udspændende træ





Antag modsætningsvis et MST med et snit hvor mindste kant *e* ikke er med i MST

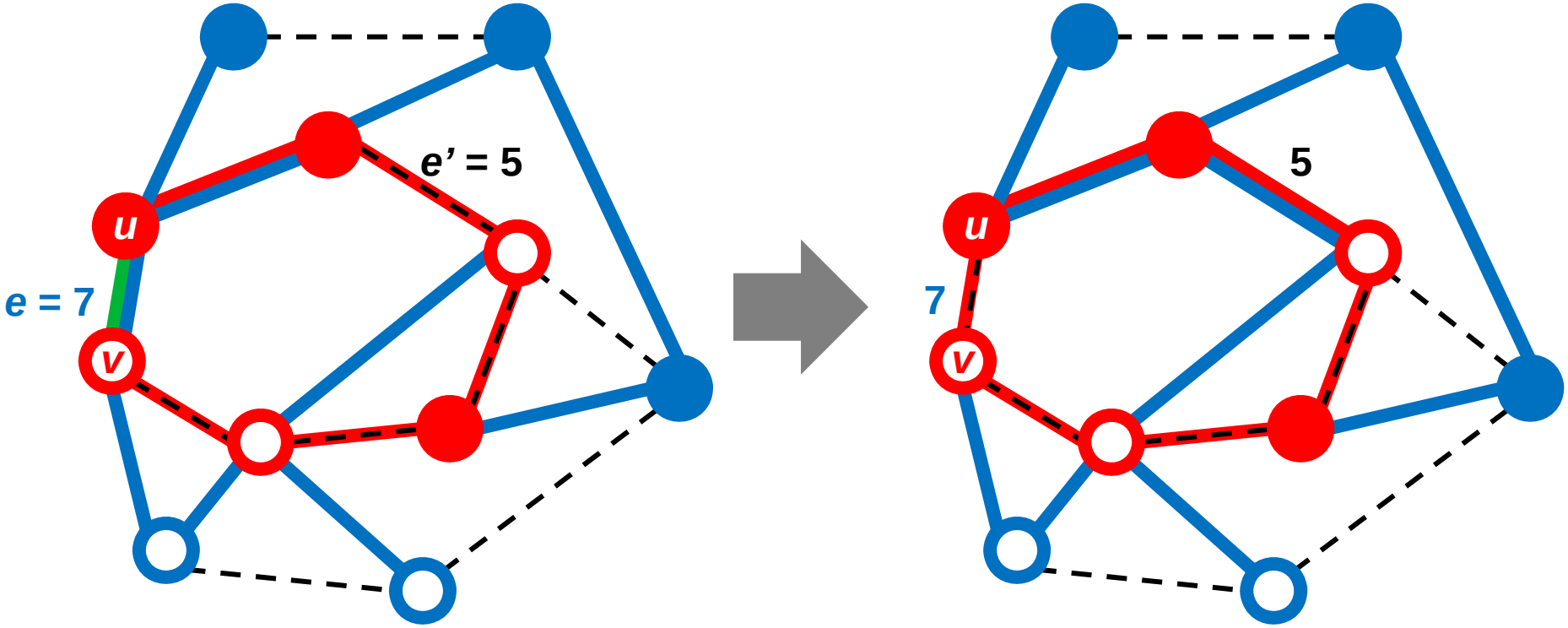
Nyt udspændende træ med mindre vægt

Sætning

Hvis alle vægte er forskellige, så gælder der for *ethvert* **snit (*S*,*V*-*S*)** at den **letteste kant** der krydser snittet er med i et minimum udspændende træ

Sætning

Hvis alle vægte er forskellige, så gælder der for *enhver* **cykel** at den **tungeste kant i cyklen** ikke er med i et minimum udspændende træ



Antag modsætningsvis et MST og cykel hvor tungeste kant *e* er med i MST

Nyt udspændende træ med mindre vægt

Minimum Udspændende Træer: Grådig generel algoritme

```
GENERIC-MST(G, w)
1  A ← ∅
2  while A does not form a spanning tree
3       do find an edge (u, v) that is safe for A
4            A ← A ∪ {(u, v)}
5  return A
```

En letteste kant over et snit (som ikke allerede indeholder kanter fra A)

Kruskall's Algoritme

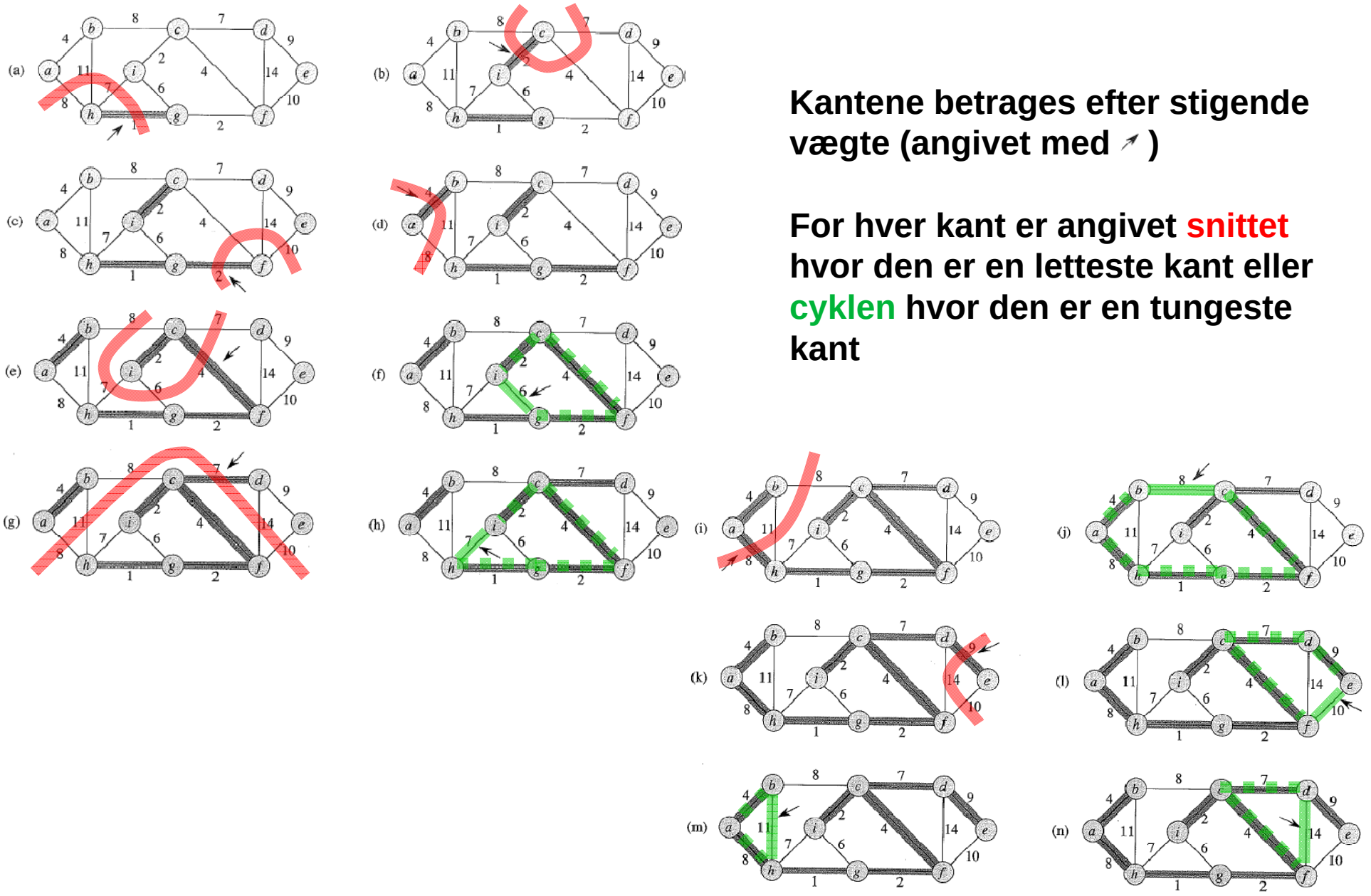
```
MST-KRUSKAL(G, w)
1  A ← ∅
2  for each vertex v ∈ V[G]
3       do MAKE-SET(v)
4  sort the edges of E into nondecreasing order by weight w
5  for each edge (u, v) ∈ E, taken in nondecreasing order by weight
6       do if FIND-SET(u) ≠ FIND-SET(v)
7            then A ← A ∪ {(u, v)}
8                 UNION(u, v)
9  return A
```

flaskehals i algoritmen

Union-Find datastruktur

Tid $O(m \cdot \log n)$

Kruskall's Algoritme: Eksempel

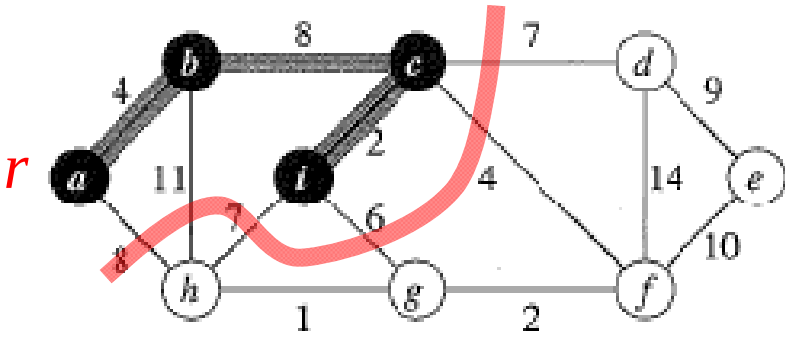


Kantene betrages efter stigende vægte (angivet med ↗)

For hver kant er angivet snittet hvor den er en letteste kant eller cyklen hvor den er en tungeste kant

Prim's Algoritme

```
MST-PRIM(G, w, r)
 1  for each u ∈ V[G]
 2        do key[u] ← ∞
 3           π[u] ← NIL
 4  key[r] ← 0
 5  Q ← V[G]
 6  while Q ≠ ∅
 7        do u ← EXTRACT-MIN(Q)
 8           for each v ∈ Adj[u]
 9                 do if v ∈ Q and w(u, v) < key[v]
10                       then π[v] ← u
11                            key[v] ← w(u, v)
```



flaskehals i algoritmen
- prioritetskø

Tid $O(m \cdot \log n)$

Prim's Algoritme: Eksempel