

Algoritmer og Datastrukturer 2

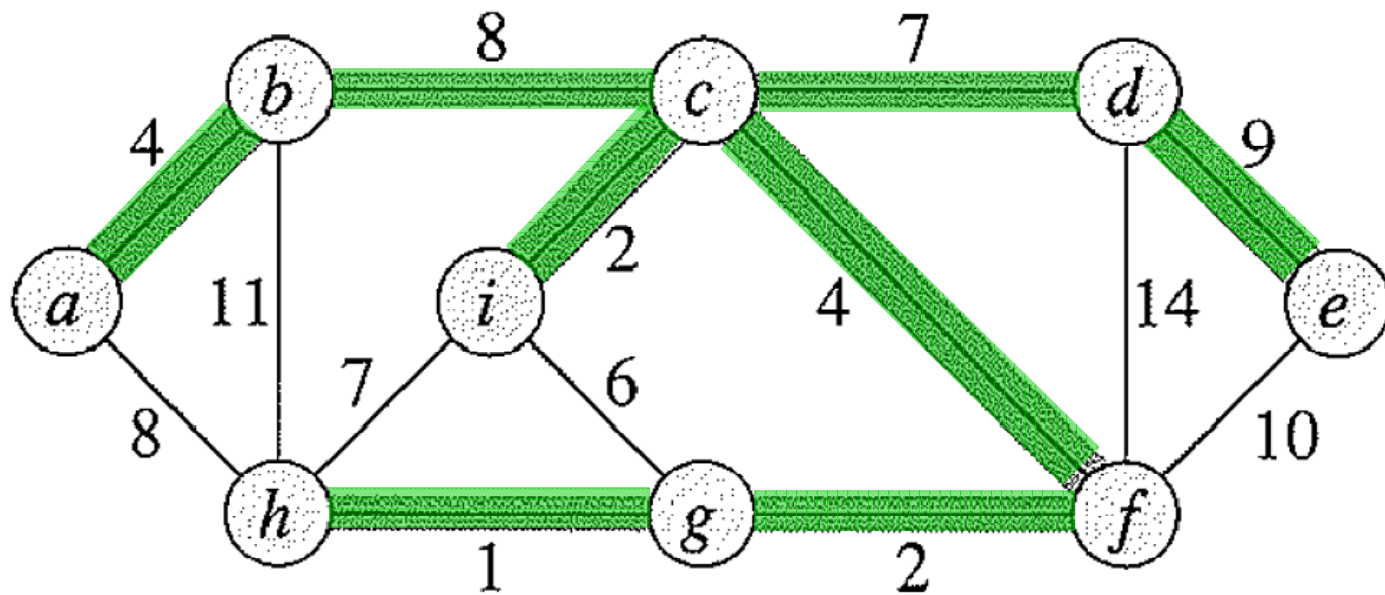
Gerth Stølting Brodal

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



AARHUS UNIVERSITET

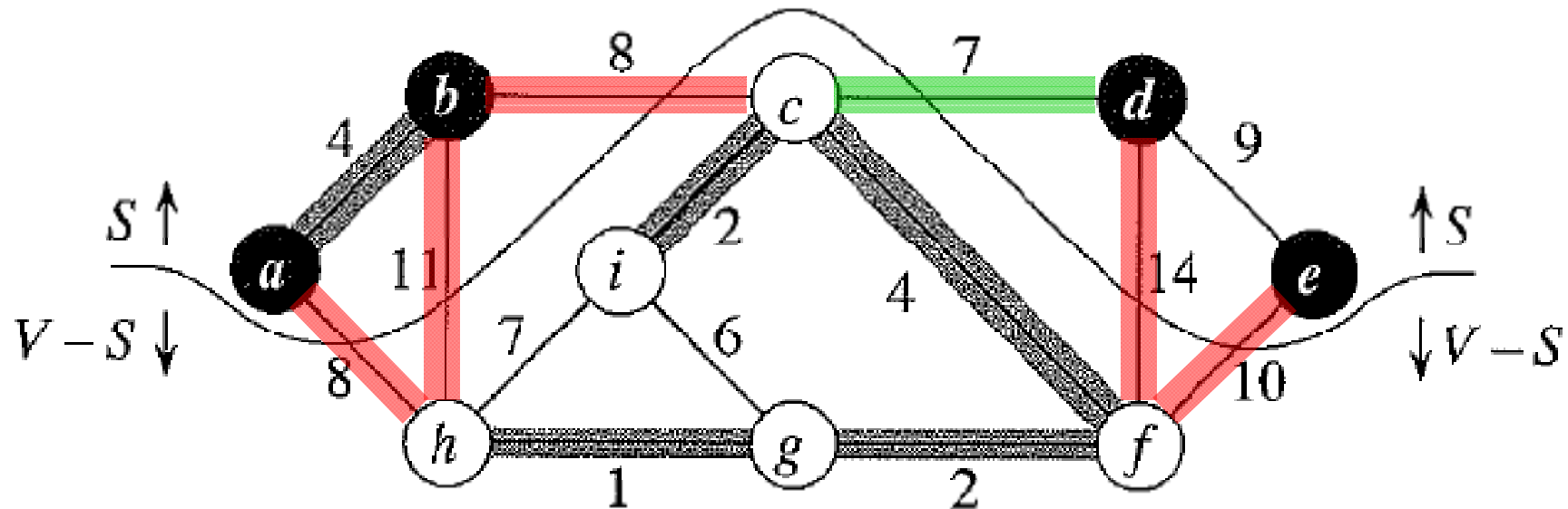
Minimum Udspændende Træ (MST)



Problem

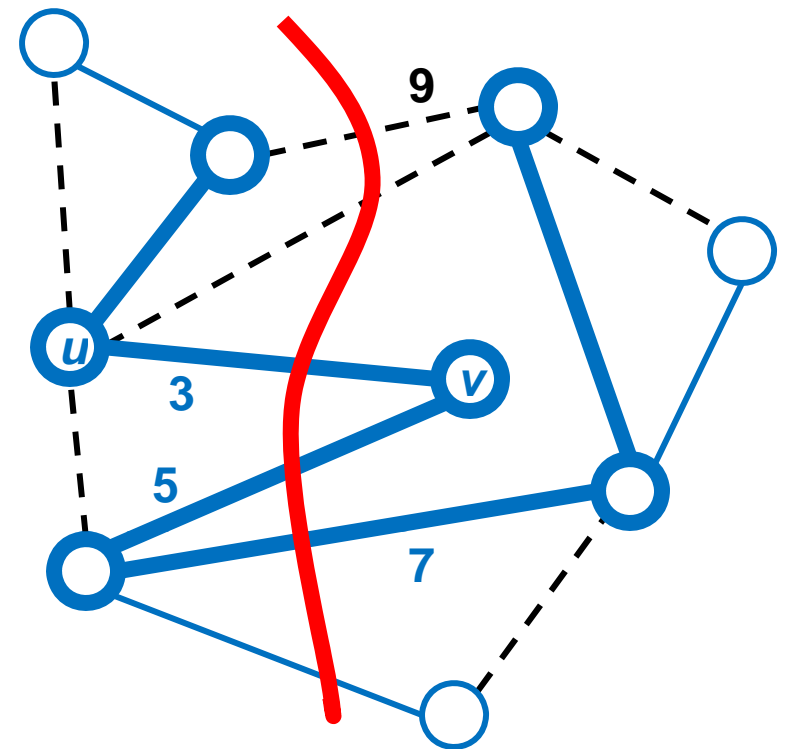
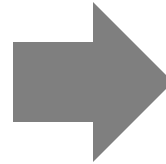
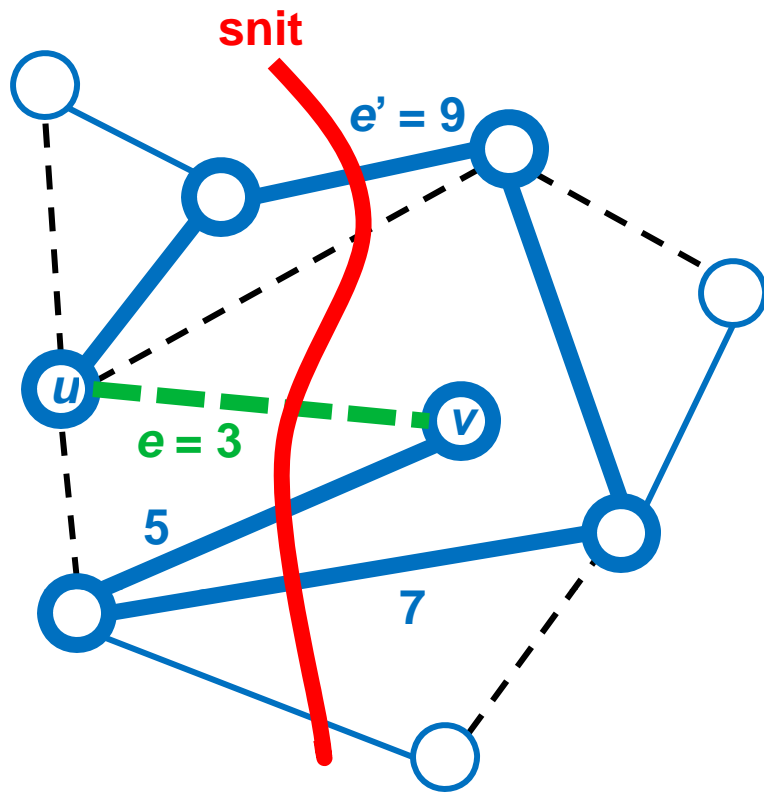
Find et **udspændende træ** for en **sammenhængende uorienteret vægtet** graf således at **summen af kanterne er mindst mulig**

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

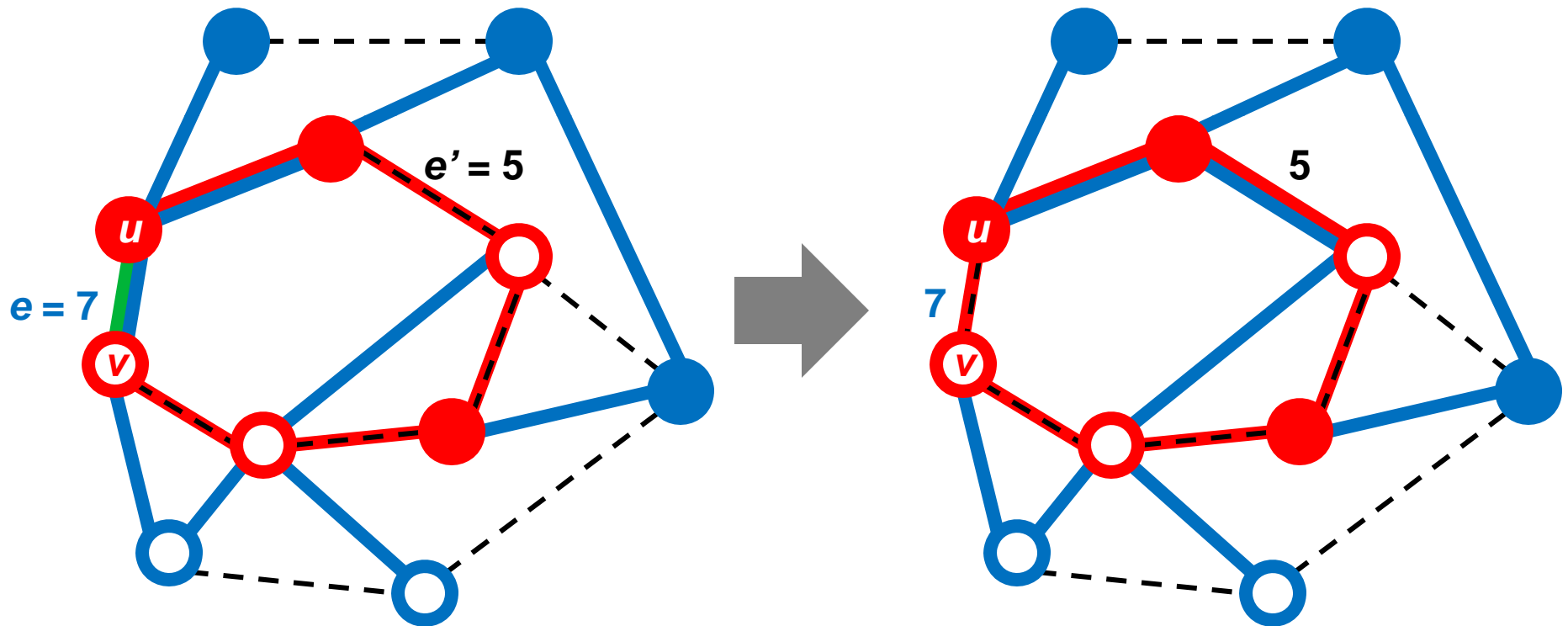
Nyt udspændende træ
 med mindre vægt ⚡

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 $A = \emptyset$
- 2 **while** A does not form a spanning tree
- 3 find an edge (u, v) that is safe for A
- 4 $A = A \cup \{(u, v)\}$
- 5 **return** A



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

Kruskall's Algoritme

MST-KRUSKAL(G, w)

1 $A = \emptyset$

2 **for** each vertex $v \in G.V$

3 MAKE-SET(v)

flaskehals i algoritmen

4 sort the edges of $G.E$ into nondecreasing order by weight w

5 **for** each edge $(u, v) \in G.E$, taken in nondecreasing order by weight

6 **if** FIND-SET(u) \neq FIND-SET(v)

7 $A = A \cup \{(u, v)\}$

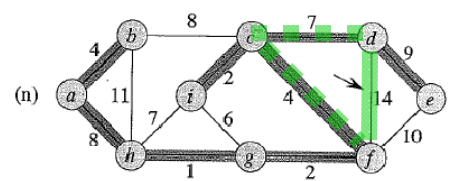
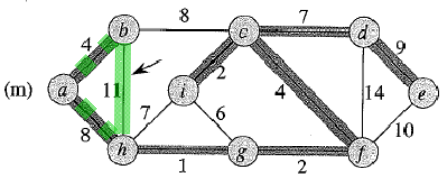
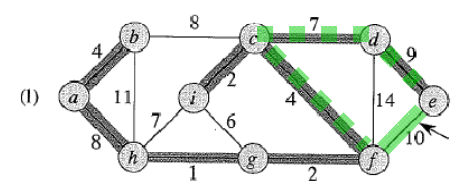
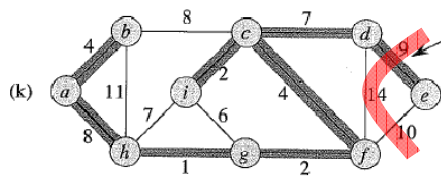
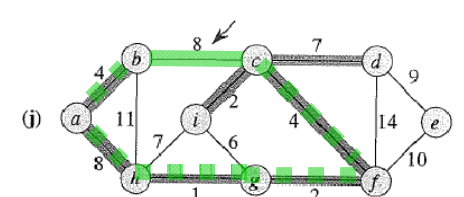
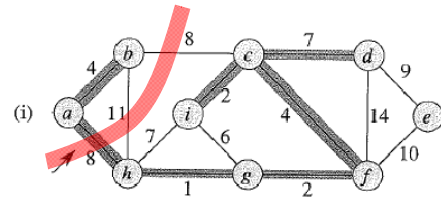
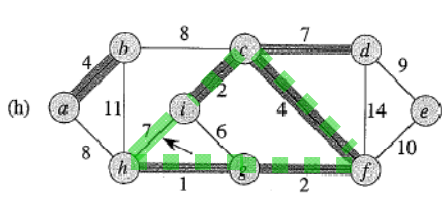
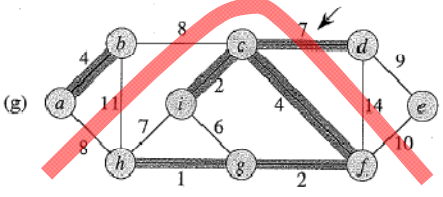
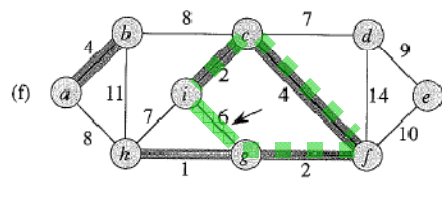
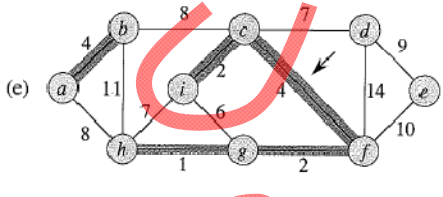
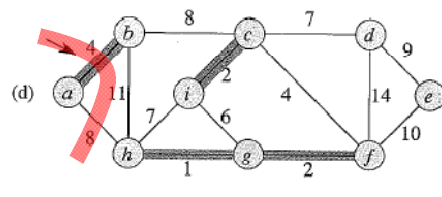
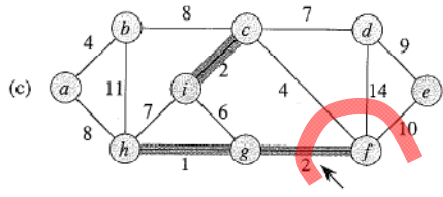
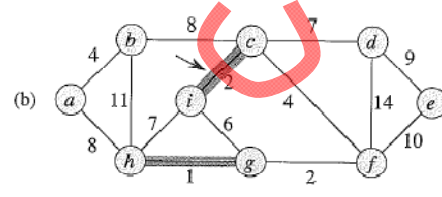
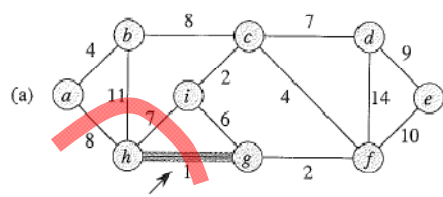
8 UNION(u, v)

Union-Find
datastruktur

9 **return** A

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

Kruskall's Algorithm: Eksempel



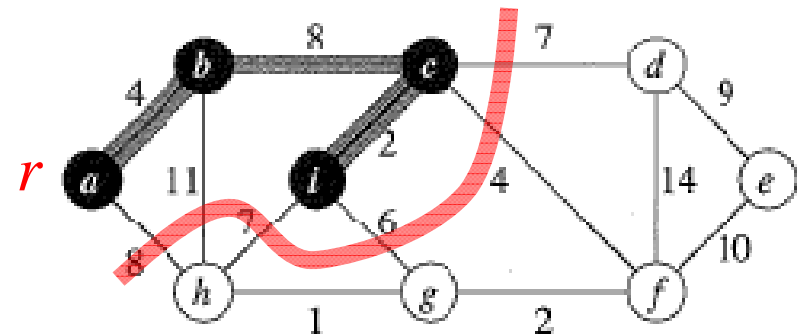
Kantene betrages efter stigende vægte (angivet med ↗)

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

Prim's Algorithm

MST-PRIM(G, w, r)

```
1  for each  $u \in G.V$ 
2       $u.key = \infty$ 
3       $u.\pi = \text{NIL}$ 
4   $r.key = 0$ 
5   $Q = G.V$ 
6  while  $Q \neq \emptyset$ 
7       $u = \text{EXTRACT-MIN}(Q)$ 
8      for each  $v \in G.Adj[u]$ 
9          if  $v \in Q$  and  $w(u, v) < v.key$ 
10              $v.\pi = u$ 
11              $v.key = w(u, v)$ 
```



flaskehals i algoritmen
- prioritetskø

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

Prim's Algorithm: Eksempel

