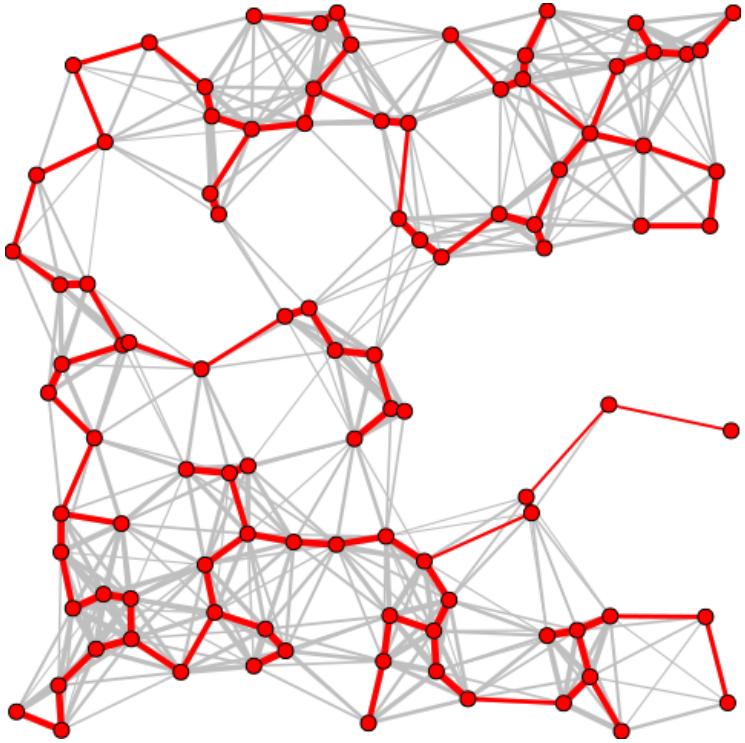


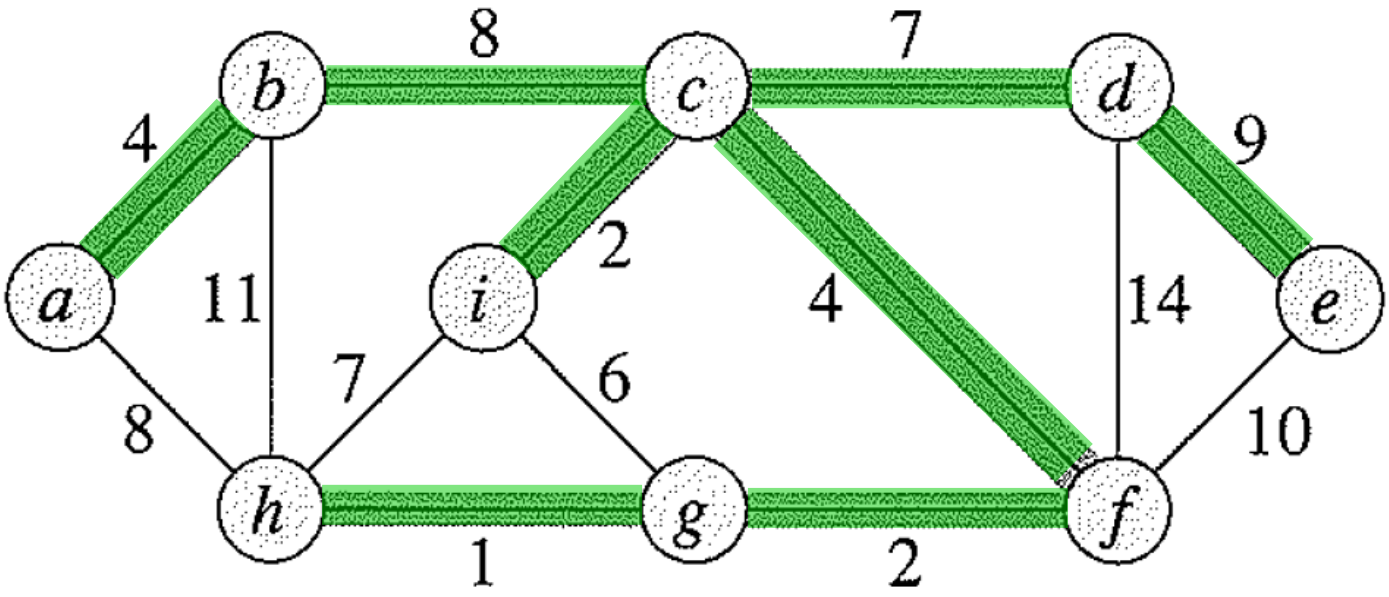
Algoritmer og Datastrukturer 2

Gerth Stølting Brodal

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

AARHUS UNIVERSITET

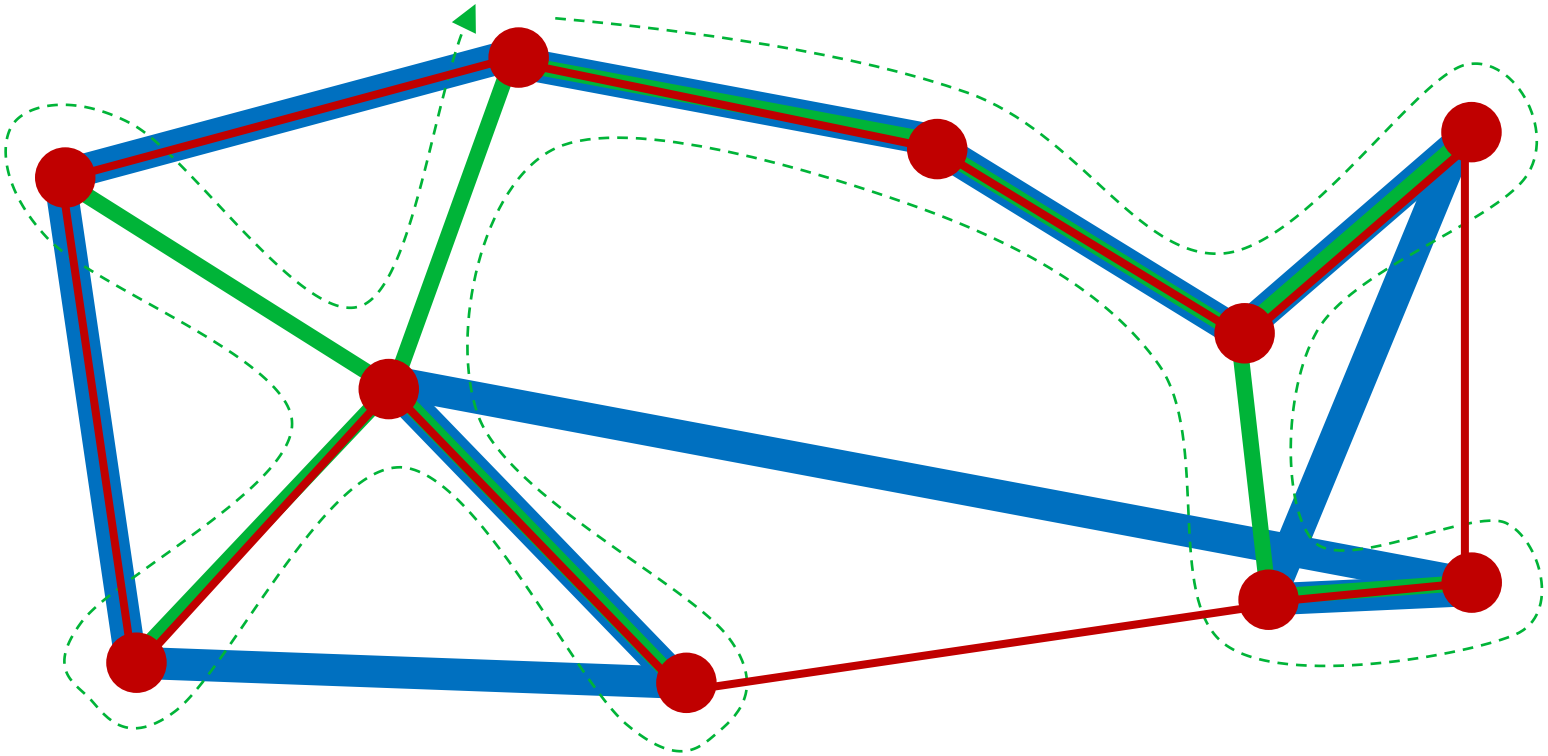
Minimum Udspændende Træ (MST)



Problem

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

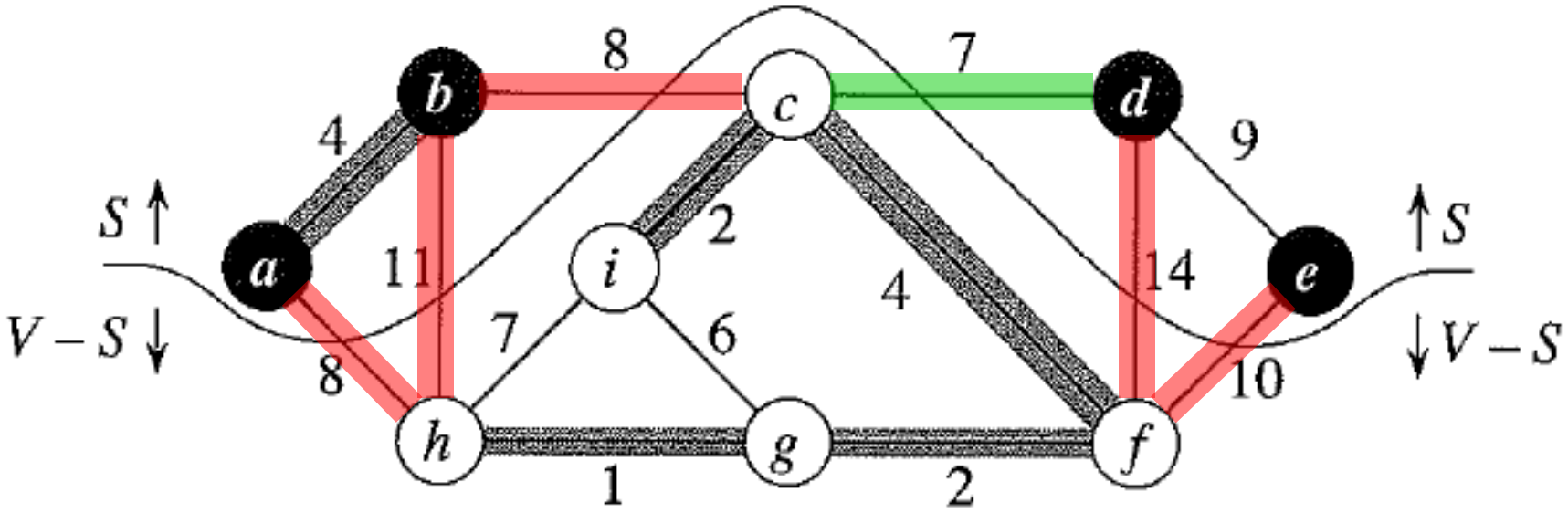
En anvendelse af MST :
(Euklidisk) Korteste Hamiltonske Cykel



Sætning : Touren fundet vha. et **MST**
er en 2-approximation til en **korteste cykel**

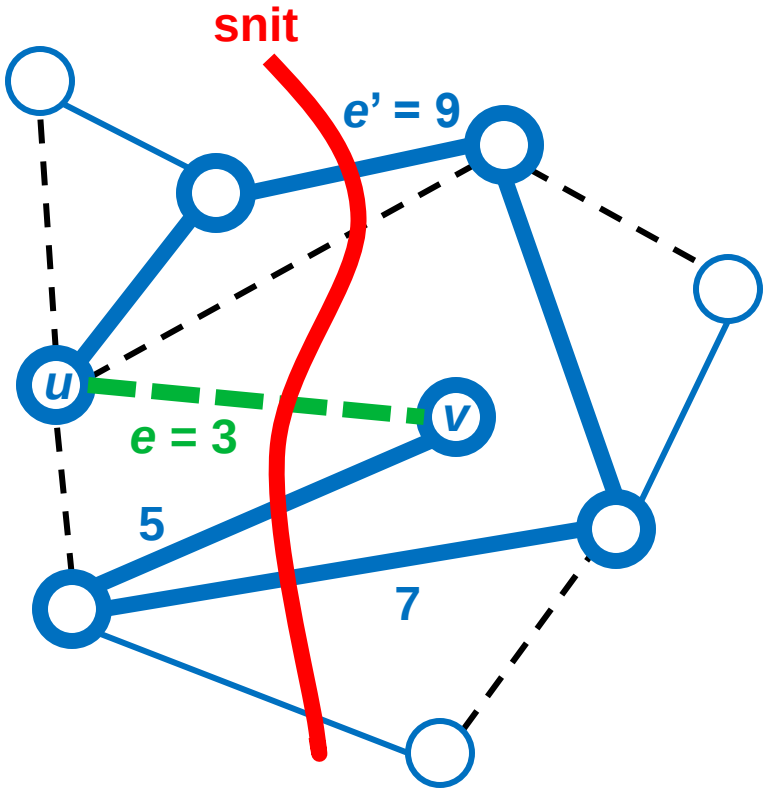
(|**Touren**| ≤ 2·|**MST**| og |**MST**| ≤ |**korteste cykel**|)

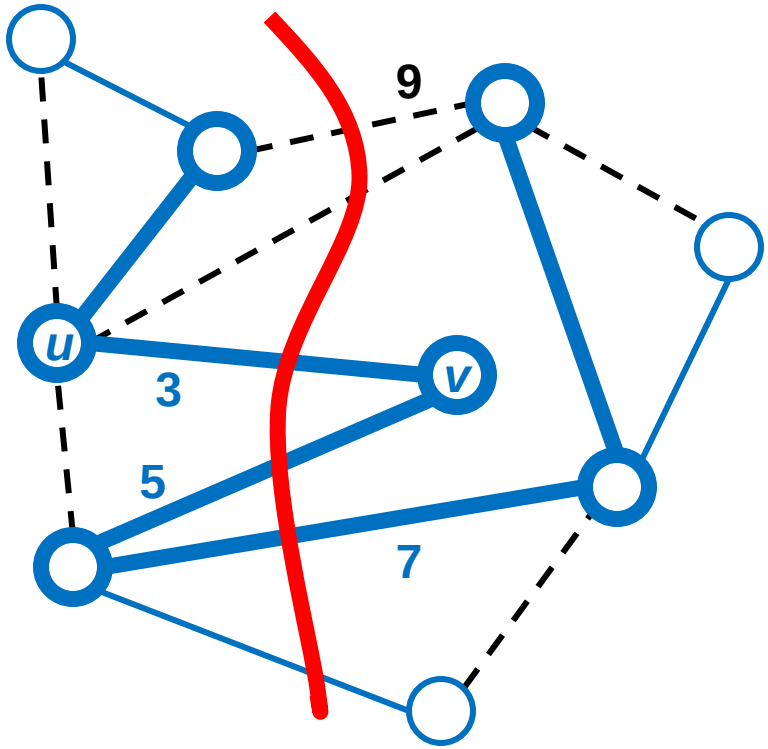
Minimum Udspændende Træer: Snit



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

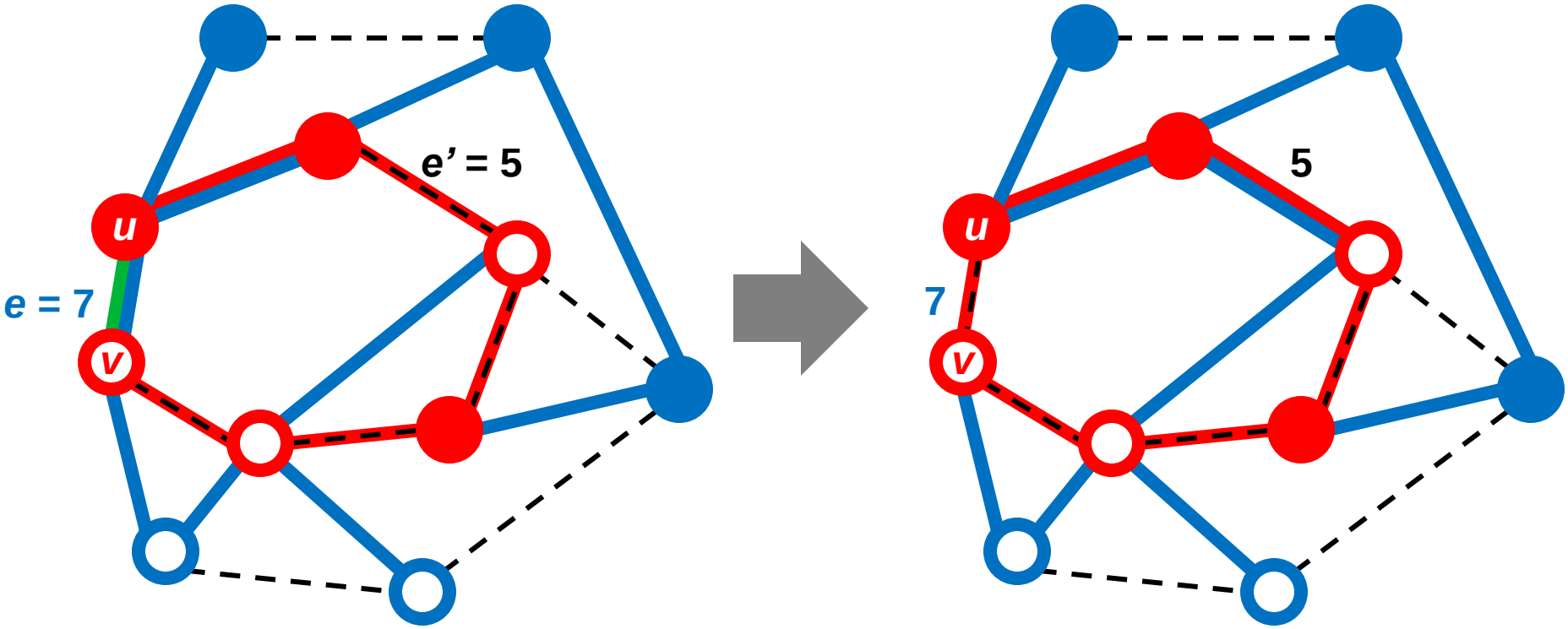
Nyt udspændende træ med mindre vægt

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      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

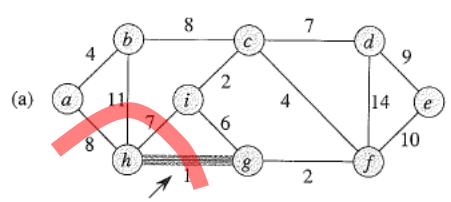
```
MST-KRUSKAL(G, w)
1  A = ∅
2  for each vertex v ∈ G.V
3       MAKE-SET(v)
4  sort the edges of G.E into nondecreasing order by weight w
5  for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
6       if FIND-SET(u) ≠ FIND-SET(v)
7            A = A ∪ {(u, v)}
8            UNION(u, v)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

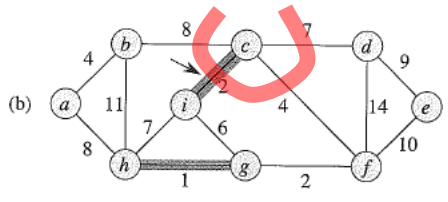
flaskehals i algoritmen

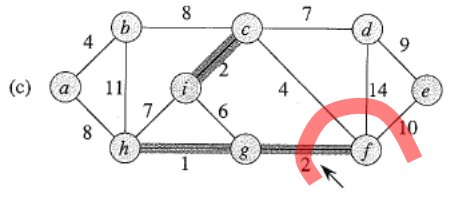
Union-Find datastruktur
$O(m \cdot \alpha(n))$
[CLRS, Sætning 21.14]

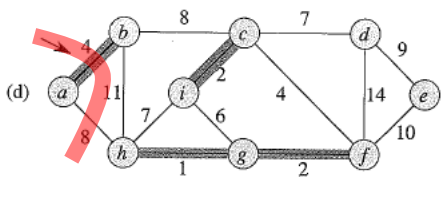
Tid $O(m \cdot \log n)$

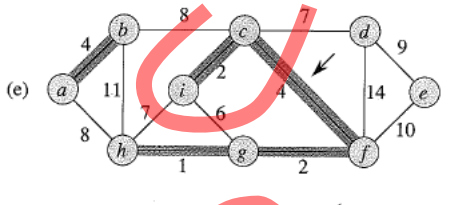
Kruskall's Algoritme: Eksempel

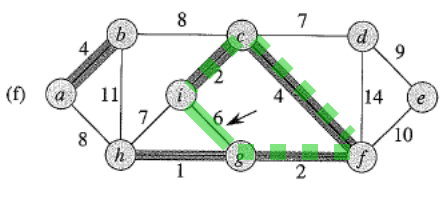


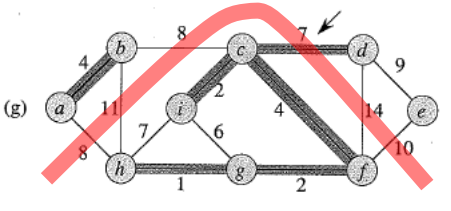


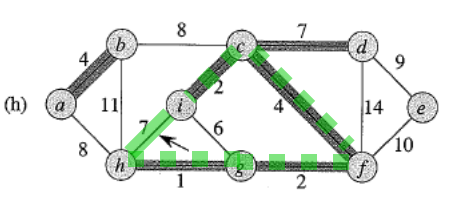


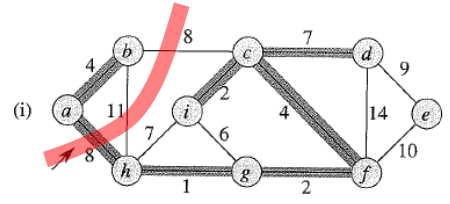


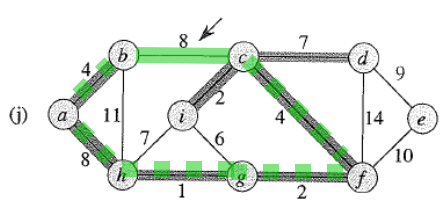


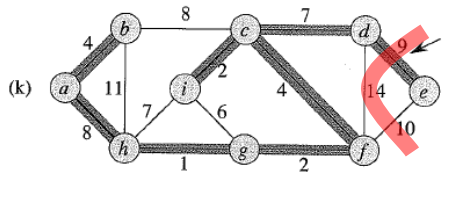


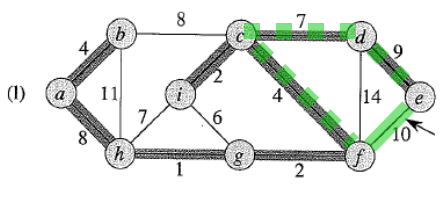


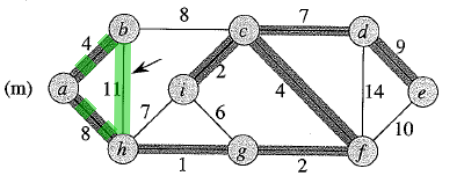


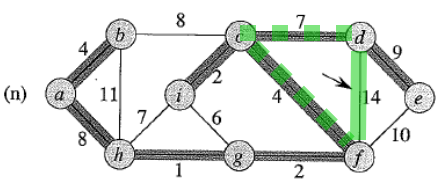










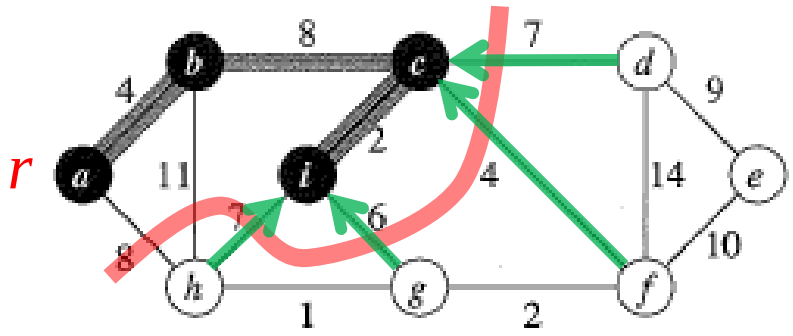


Kantene betrages efter stigende vægte (angivet med ↗)

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

Prim's Algoritme

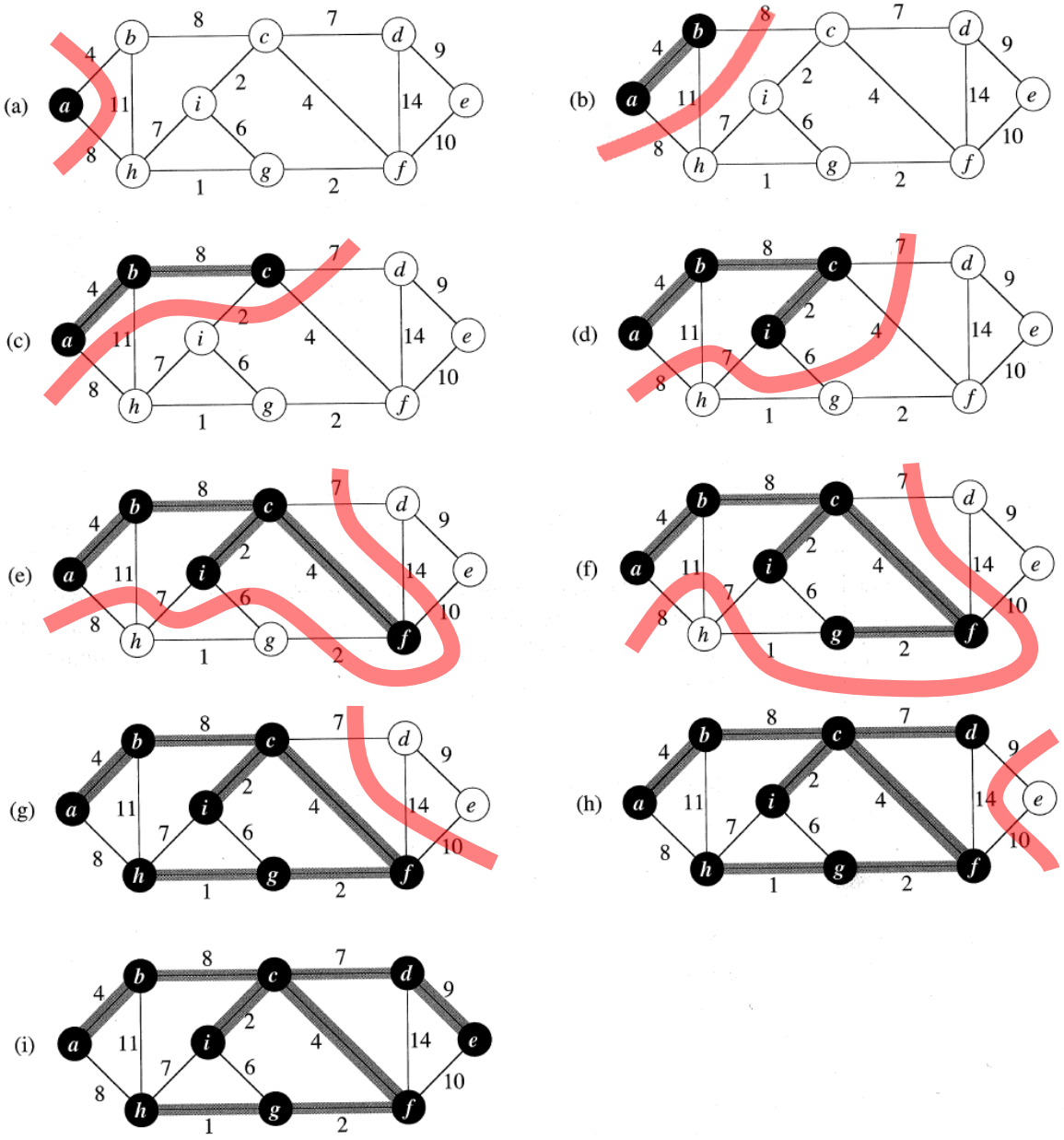
```
MST-PRIM(G, w, r)
 1  for each u ∈ G.V
 2      u.key = ∞
 3      u.π = NIL
 4  r.key = 0
 5  Q = G.V
 6  while Q ≠ ∅
 7      u = EXTRACT-MIN(Q)
 8      for each v ∈ G.Adj[u]
 9          if v ∈ Q and w(u, v) < v.key
10              v.π = u
11              v.key = w(u, v)
```



flaskehals i algoritmen
- prioritetskø

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

Prim's Algoritme: Eksempel



Eksamensopgave 3(c) august 2005

Spørgsmål c: Beskriv en algoritme, der givet en vægtet graf G og en kant e i grafen, afgør om e er indeholdt i et minimum udspændende træ for G. Det antages at alle vægtene er forskellige. Udførselstiden for algoritmen skal være $O(m)$, hvor m er antal kanter i grafen. □

Minimum Udspænding Træer

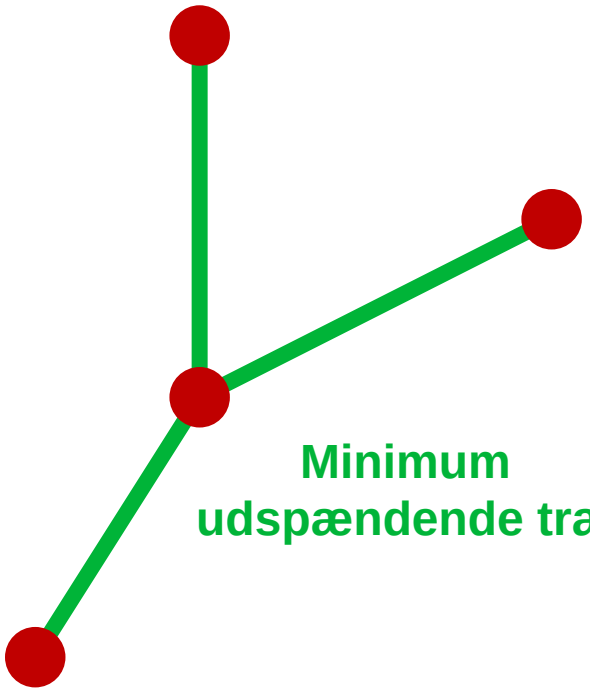
Kruskall (1956) (mange træer; sorterer kanterne)	$O(m \cdot \log n)$
Prim (1930) (et træ; prioritetskø over naboknuder)	$O(m \cdot \log n)$
	$O(m+n \cdot \log n)$ (Fibonnaci heaps [CLRS, kapitel 19] (1984))
Borůvka (1926) (mange træer samtidigt; kontraktion)	$O(m \cdot \log n)$
Fredman, Tarjan (1984) (Borůvka (1926) + Fibonnaci heaps)	$O(m \cdot \log^* n)$
Chazelle (1997)	$O(m \cdot \alpha(m,n))$
Pettie, Ramachandran (2000)	? (men optimal deterministisk sammenligningsbaseret)
Karger, Klein, Tarjan (1995) (Randomiseret) Fredmand, Willard (1994) (RAM)	$O(m)$

IKKE PENSUM

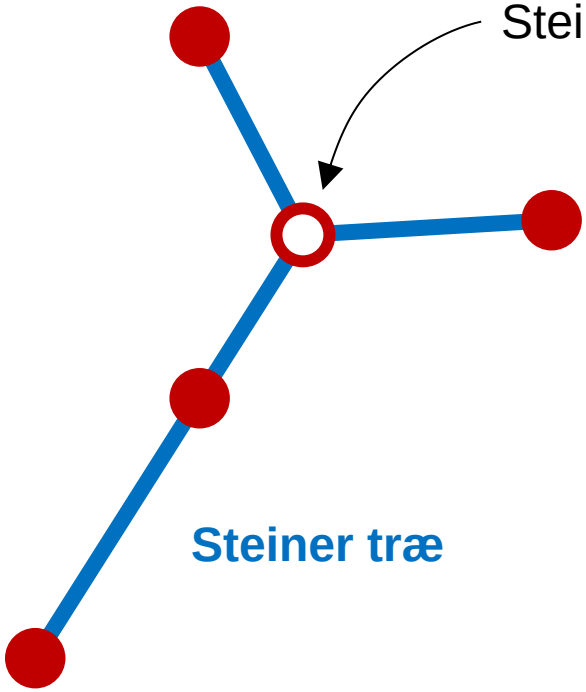
Jakob Steiner [1796-1863]

(Euklidiske) Steiner Træer

Givet en mængde punkter, find et træ med minimum vægt som forbinder alle knuder, muligvis ved at tilføje **nye punkter**



Minimum udspændende træ



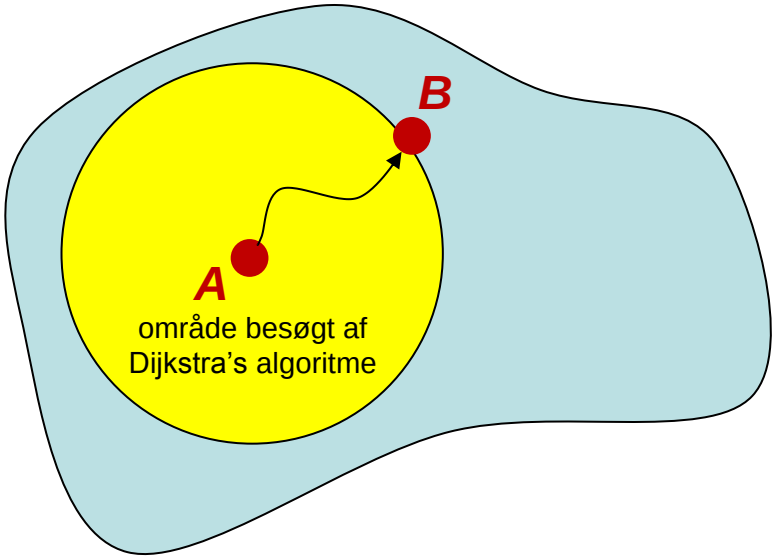
Steiner træ

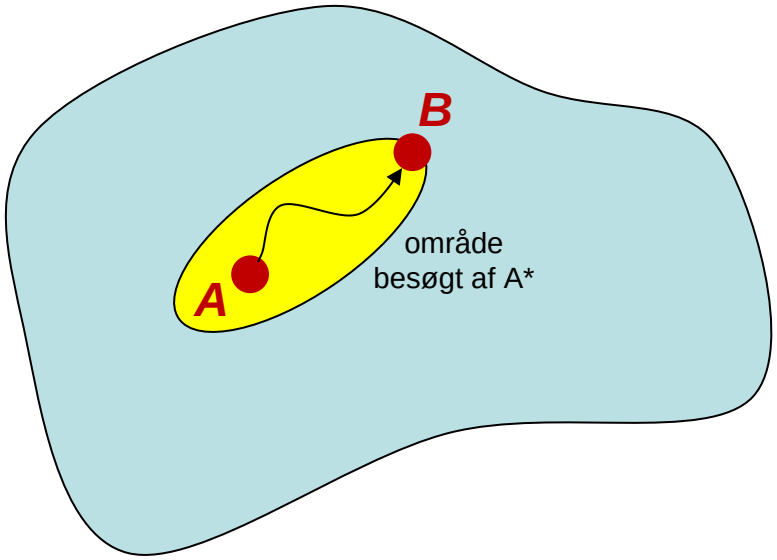
Conjecture (Steiner Ratio): $|\text{MST}| \leq \frac{2}{\sqrt{3}} \cdot |\text{Steiner Tree}|$

NP hårdt; polynomial-time approximation scheme (PTAS)

IKKE PENSUM

Korteste Veje : A* Heuristik





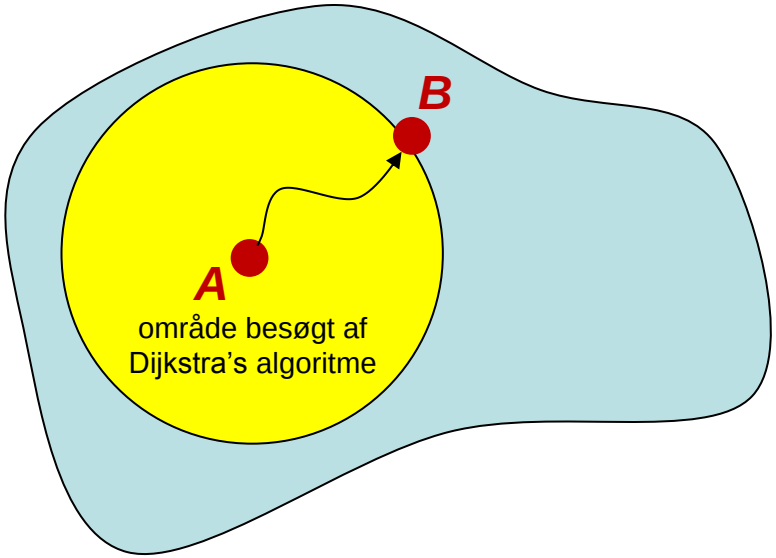
Intuitivt: Tildel hver punkt ***x*** en højde ***h*(*x*)**, og erstat vægten *w*(*u*,*v*) med ***w'*(*u*,*v*) = w(*u*,*v*) + *h*(*v*) – *h*(*u*)**

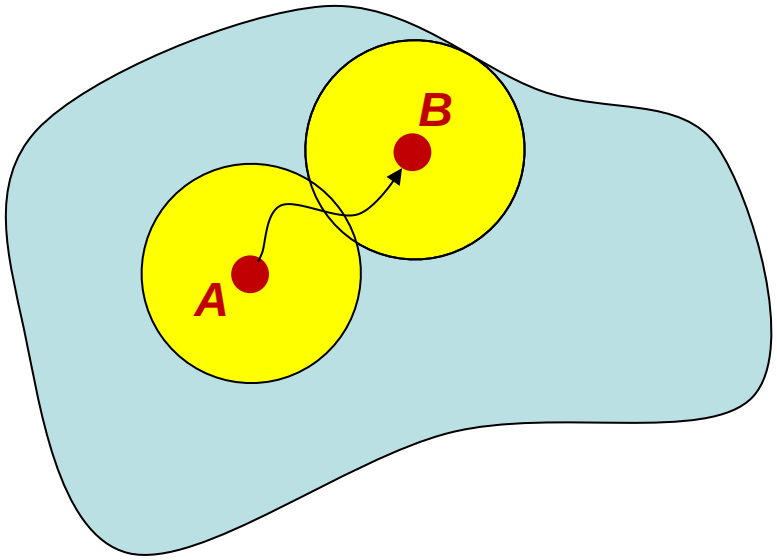
Korteste stier forbliver korteste stier (dog *h*(*B*)-*h*(*A*) længere)

Kan anvende Dijkstra's algoritme hvis *w'*(*u*,*v*) ≥ 0
Euklidisk: Anvend f.eks. *h*(*x*) = |*B* - *x*|

IKKE PENSUM

Korteste veje : Dobbeltrettet





Intuitivt: Start Dijkstra's algoritme både i ***A*** og ***B***

IKKE PENSUM

Korteste veje : Preprocessering

	Europe			USA		
method	preprocessing time [h:m]	space [GB]	query [ns]	preprocessing time [h:m]	space [GB]	query [ns]
CH [5]	0:13	0.4	93 995	0:14	0.5	67 885
CHASE [5]	0:52	0.6	9 034	1:59	0.7	9 922
HPML [9]	≈12:00	3.0	9 817	≈12:00	5.1	10 078
TNR [5]	0:58	3.7	1 775	0:47	5.4	1 566
TNR+AF [5]	2:00	5.7	992	1:22	6.3	888
HL prefix	2:31 + 0:45	5.7	527	2:17 + 0:40	6.4	542
HL local	2:31 + 0:08	20.1	572	2:17 + 0:07	22.7	627
HL global	2:31 + 0:14	21.3	276	2:17 + 0:18	25.4	266
Table Lookup	> 11:03	1 208 358.7	56	> 22:44	2 293 902.1	56

A Hub-Based Labeling Algorithm for Shortest Paths in Road Networks,
Ittai Abraham, Daniel Delling, Andrew V. Goldberg, Renato F. Werneck (SEA'2011)