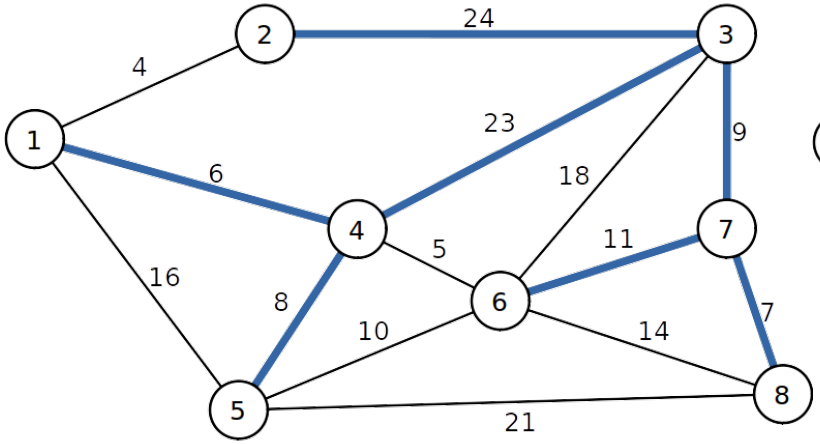
Mindste udspændende træ

- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
- Prims (Jarníks) algoritme
- Kruskals algoritme

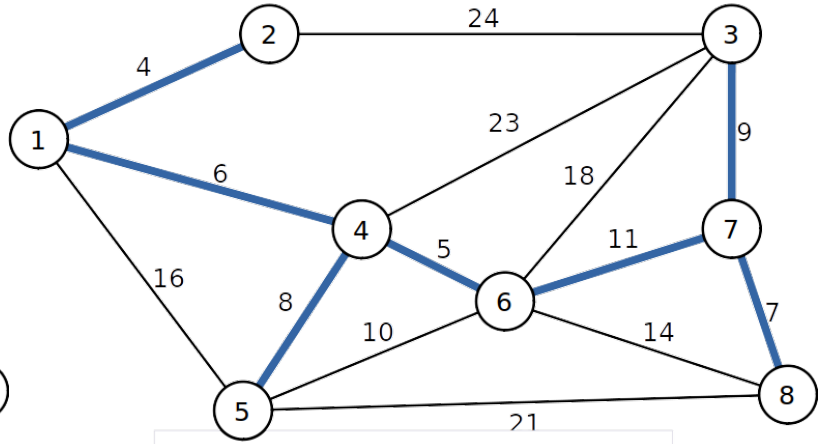
Mindste udspændende træ

Input: Vægtet sammenhængende graf.

Hver kant e har en vægt $v(e)$. Alle knuder forbundne.



Vægt: 6 + 8 + 23 + 24 + 9 + 11 + 7 = 88



Vægt: 4 + 6 + 5 + 8 + 11 + 9 + 7 = 50

Udspændende træ: Forbinder alle knuderne. Indeholder ikke kredse.

Et udspændende træ er en speciel delgraf (delmængde af grafens kanter).

Vægt af delgraf: samlet vægt af kanter i delgrafen.

Mindste udspændende træ: udspændende træ med mindst mulig vægt.

Algoritmisk problem: Find et mindste udspændende træ for den vægtede graf.

Mindste udspændende træ

Anvendelser:

- Netværksdesign
- Approksimationsalgoritmer
- Mange andre (se f.eks: `http://www.ics.uci.edu/~eppstein/gina/mst`)

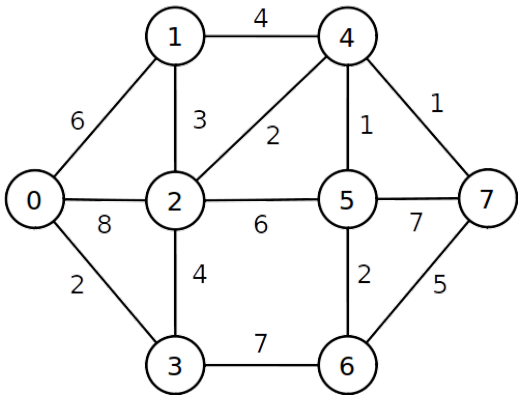
Mindste udspændende træ

- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
- Prims (Jarníks) algoritme
- Kruskals algoritme

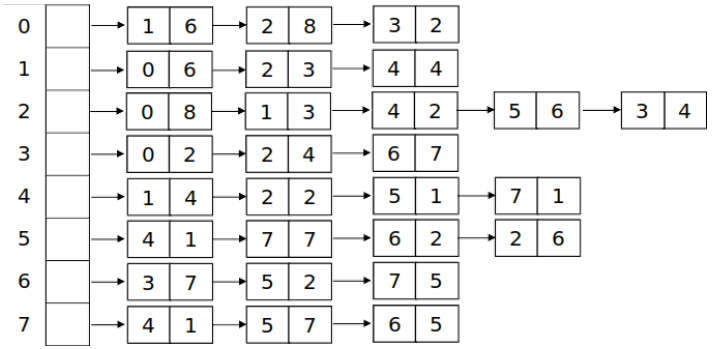
Mindste udspændende træ – repræsentation

Repræsentation:

Incidensmatrix og incidensliste.



	0	1	2	3	4	5	6	7
0	0	6	8	2	0	0	0	0
1	6	0	3	0	4	0	0	0
2	8	3	0	4	2	6	0	0
3	2	0	4	0	0	0	7	0
4	0	4	2	0	0	1	0	1
5	0	0	6	0	1	0	2	7
6	0	0	0	7	0	2	0	5
7	0	0	0	0	1	7	5	0



Mindste udspændende træ

- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
  - Et lille, nyttigt trick
  - Snitegenskaben
  - Kredsegenskaben
- Prims (Jarníks) algoritme
- Kruskals algoritme

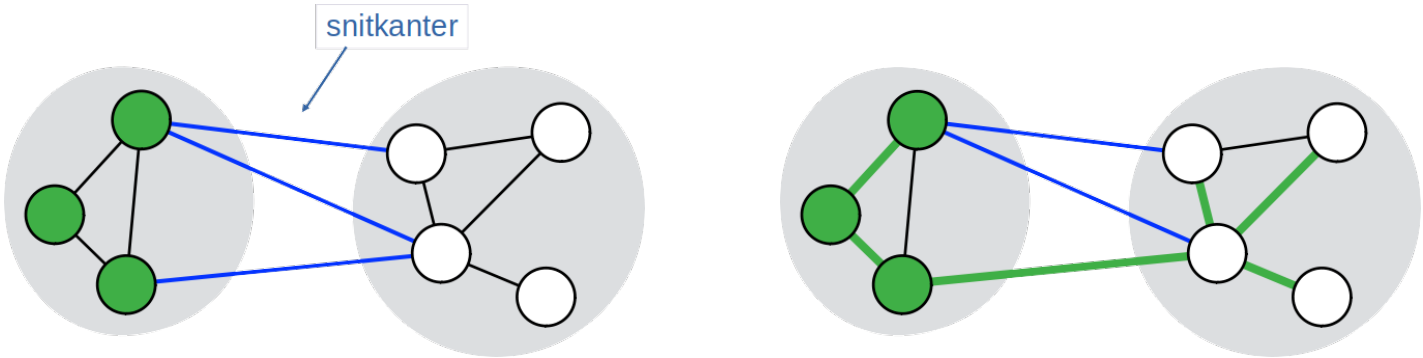
Mindste udspændende træ – Egenskaber – et nyttigt trick

Vi ønsker at finde et udspændende træ af mindste vægt.
Hvad nu hvis flere kanter vejer det samme? Kan der findes mange forskellige træer, som alle har den mindste vægt? Ja! (Vi vil bare have én af dem.)
Nyttigt trick: Afgøre uafgjort (tiebreaking). Når to kanter vejer det samme, indfør mekanisme til at foretrække en af dem.
Mekanisme: Foretræk laveste knudenummer.
Eksempel: Kant e mellem 6 og 7, kant f mellem 24 og 2. Kant f er mindst, fordi 2 er mindst.
Eksempel: Kant e mellem 6 og 7, kant f mellem 6 og 52. Kant e er mindst, fordi 7 er mindst.
Konsekvens: Vi kan antage, kanterne kan sorteres på en entydig måde. (Gør beviser simplere.) Se også opg.6.

Mindste udspændende træ – Egenskaber – Snitegenskaben

Snit: Opdeling af knuder i to ikke-tomme mængder.

Snitkanter: Kanter med et endepunkt i hver lejr.



Snitegenskaben: Letteste snitkant er med i det mindste udspændende træ.

Intuition: De to lejre skal forbindes; bedst at gøre det billigst.

Bevis: (kontraposition) Lad e være den letteste kant.

Betragt et udspændende træ T hvor e ikke er med.

Så har $T \cup \{e\}$ en kreds, som indeholder endnu en snitkant f.

Nyt udspændende træ: $T \cup \{e\} \setminus \{f\}$, dvs. vi har fjernet f og tilføjet e.

Dette træ er mindre, da e var lettere end f. Ergo: T var ikke mindste.

Mindste udspændende træ – Egenskaber – Kredsegenskaben

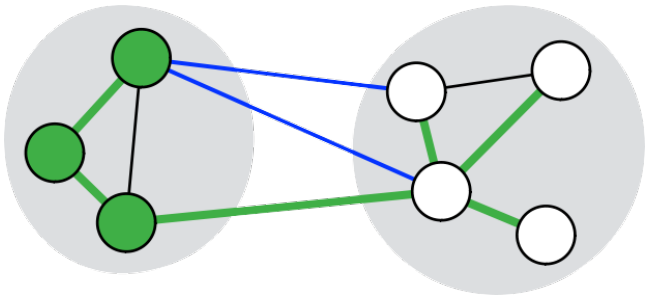
Kredsegenskaben: Den tungeste kant på en kreds er aldrig med i det mindste udspændende træ.

Intuition: Det er billigere at forbinde dens endepunkter på en anden måde.

Bevis: (kontraposition) Lad f være den dyreste kant på en kreds K.

Betragt et udspændende træ T hvor f er med.

Hvis vi sletter f fra T deler vi grafen op i to lejre.



Kredsen K må indeholde en anden kant $e \neq f$ dvs e lettere end f.

Nyt udspændende træ: $T \cup \{e\} \setminus \{f\}$, dvs. vi har fjernet f og tilføjet e.

Dette træ er mindre, da e var lettere end f. Ergo: T var ikke mindste.

Mindste udspændende træ – Egenskaber

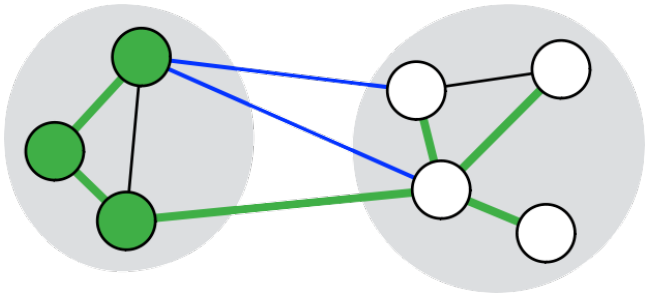
Vi kan antage, kanterne har en entydig rangordning.

Snitegenskaben: Letteste snitkant er med i det mindste udspændende træ.

- Vi viste det ved at betragte en kreds. $(T \cup \{e\})$

Kredsegenskaben: Den tungeste kant på en kreds er aldrig med i det mindste udspændende træ.

- Vi viste det ved at betragte et snit. $(T \setminus \{f\})$



Mindste udspændende træ

- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
- Prims (Jarníks) algoritme
- Kruskals algoritme

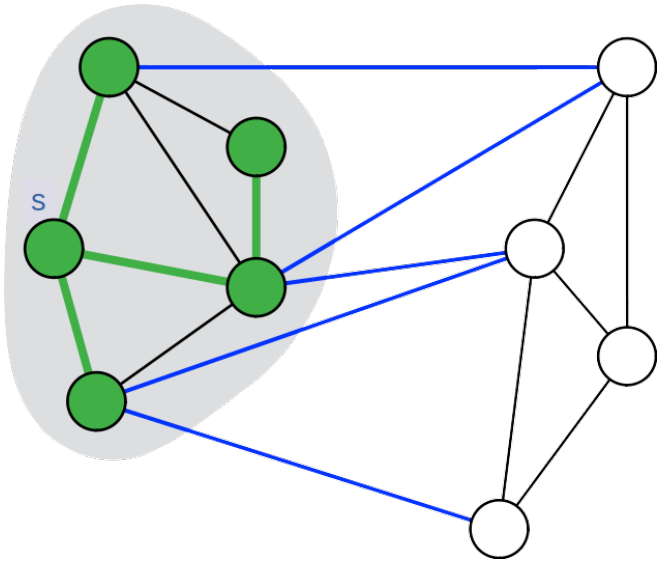
Prims (Jarníks) algoritme

Start: Tilføj knude s til træet T.

Skridt: Kig på kanter mellem T og resten af grafen.

Tilføj letteste snitkant til T.

Efter $n-1$ skridt: T udspænder hele grafen.



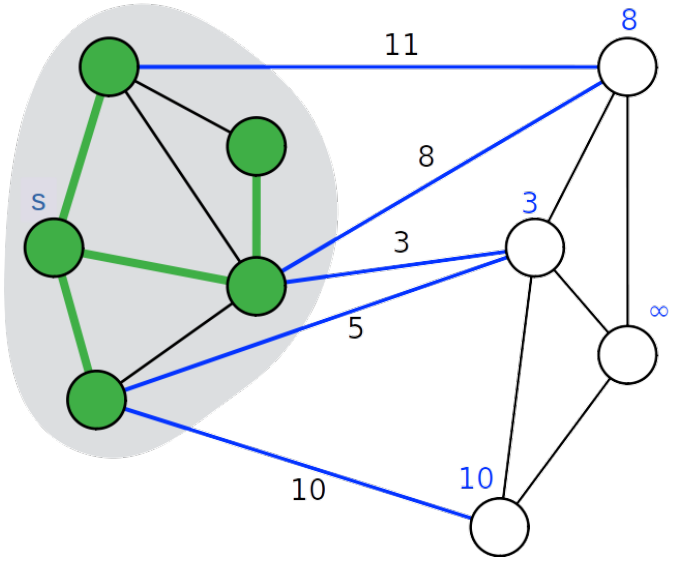
Korrekthed: Følger af snitegenskaben. Tid: Udfordring!

Hvordan finder vi hele tiden den letteste kant i snittet?

Prims (Jarníks) algoritme

Start: Tilføj knude s til træet T.

Skridt: Find letteste snitkant mellem T og resten af grafen, tilføj til T.



Implementation: Prioritetskø med knuderne som ikke er i T endnu.

Key(v) vægt af letteste kant mellem T og v.

Letteste snitkant Find letteste snitkant = Extract-min.

Tilføj til T Ny knude i T. Nye snitkanter.

Opdater den nye knudes naboer = Decrease-key.

Prims (Jarníks) algoritme

```
PRIM(G, s)
    for alle knuder v∈V
        v.key = ∞
        v.π = null
        INSERT(P,v)
    DECREASE-KEY(P,s,0)
    while (not(P.IsEmpty))
        u = EXTRACT-MIN(P)
        for alle naboer v af u
            if (v ∈ P and w(u,v)<key[v])
                DECREASE-KEY(P,v,w(u,v))
                v.π = u
```

Tid.

- Extract-min: $\times(n-1)$.
- Insert: $\times(n-1)$.
- Decrease-key: $\times m$.

Samlet: $O(n+n+m) = O(m)$ hob-operationer. $O(m \log n)$ tid.

Hurtigere implementation: Fibbonaccihob.

Prims (Jarníks) algoritme

Tid: Afhænger af prioritetskø.

Prioritetskø	INSERT	EXTRACT-MIN	DECREASE-KEY	Total
Hob	O(log n)	O(log n)	O(log n)	O(m log n)
Fibonaccihob	O(1)†	O(log n)†	O(1)†	O(m + n log n)

Mindste udspændende træ

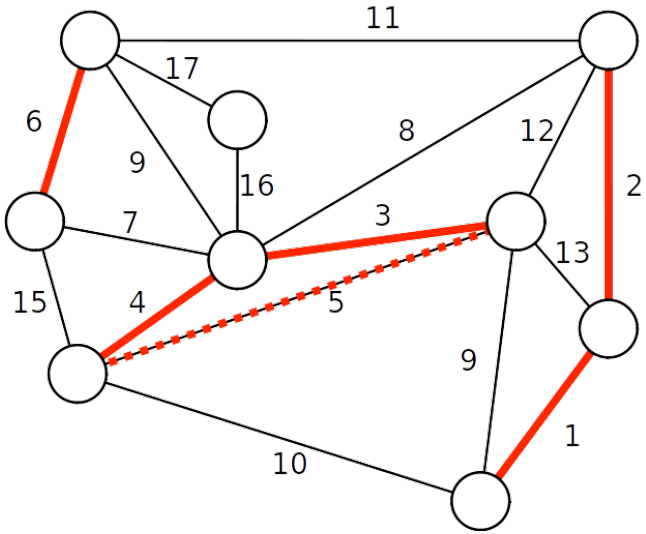
- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
- Prims (Jarníks) algoritme
- Kruskals algoritme

Kruskals algoritme

Kig på kanter fra letteste til tungeste.

Skridt: tilføj kant til T hvis den ikke medfører en kreds i T.

Efter $n-1$ kanter har vi et udspændende træ.



Analyse:

- Korrekthed?
- Tid?

Kruskals algoritme – korrekthed

Kig på kanter fra letteste til tungeste.

Skridt: tilføj kant til T hvis den ikke medfører en kreds i T.

Korrekthed:

Tilfælde 1: Kanten e danner en kreds og bliver ikke tilføjet.

- e er tungeste kant på en kreds.
- Kredsegenskaben: e ligger ikke i mindste udspændende træ.

Tilfælde 2: Kanten e (mellem u og v) danner ikke en kreds og skal tilføjes.

- Kig på u's sammenhængskomponent S i træet T.
- Knuden v ligger ude i 'resten af grafen'. $(G \setminus S)$
- Kanten e må være mindste kant i det snit. (S vs. $G \setminus S$)
- Snitegenskaben: e ligger i mindste udspændende træ.

Kruskals algoritme – tid

```
KRUSKAL(G)
    Sortér kanter
    INIT(n)
    for alle kanter (u,v) i sorteret orden
        if (not CONNECTED(u,v))
            INSERT(u,v)
    return indsatte kanter
```

Tid:

- Sortering af m kanter: $O(m \log m)$ tid.
- m spørgsmål: "Danner kreds?" Forbundne (2× Find)
- n gange: indsæt kant. Forbind (Union)

Udfordring: Datastruktur til at afgøre, om nuværende kants endepunkter allerede er forbundne? Hint: Vi har lige lært om den.

Forbind-Forbundne Foren-og-Find

Samlet tid: $O(m \log m)$

Mindste udspændende træ – tid

Algoritmiske resultater oversigt:

År	Tid	Opdaget af
1926	O(m log n)	Borůvka, Jarník, Kruskal, Prim, Dijkstra, ?
1975	O(m log log n)	Yao
1986	O(m log* n)	Fredman, Tarjan
1995	O(m)‡	Karger, Klein, Tarjan
2000	O(nα(m,n))	Chazelle
2002	optimal	Pettie, Ramachandran

Ubesvaret: Kan man løse problemet i lineær tid?

Mindste udspændende træ

- Mindste udspændende træ
- Repræsentation af vægtede grafer
- Egenskaber for mindste udspændende træ
- Prims (Jarníks) algoritme
- Kruskals algoritme