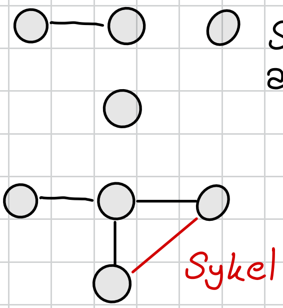
Forelesning 22
GRAFTEORI II

Innhold
- Spenntrær
- Vektede grafer
- Minimale spenntrær (MST)
- Algoritmer for å finne MST: Kruksal og Prim

Trær



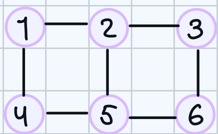
Skog som består av 3 trær

$$G = (\underbrace{V}_{\text{noder}}, \underbrace{E}_{\text{kanter}}) \qquad \underbrace{|V|}_{\text{antall noder}} = |E| + 1$$

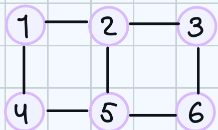
Spenntrær - eksempler

En graf (som ikke er et tre) kan ha mange spenntrær.

Vi kan først fjerne 2-5 og så 1-4.

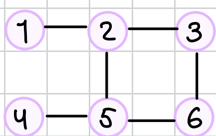


Eller vi kan først fjerne 1-4 og så 3-6.



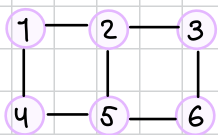
Spenntrær oppgave

Hvor mange forskjellige spenntrær har grafen?



4 forskjellige

Hvor mange forskjellige spenntrær har grafen?



I Fjern 2-5
Må fjerne en kant i tillegg
Kan velge mellom alle. 6 muligheter

II Ikke fjern 2-5
$3 \cdot 3$ muligheter $= 9$

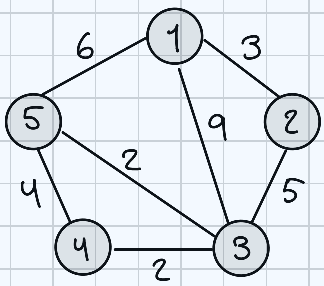
15 muligheter

Vektede grafer

En vektet graf er et par (G, w) der $G = (V, E)$ er en (enkel) graf og $w : E \to \mathbb{R}_{\geq 0}$ gir en vekt til hver kant. Mer uformelt: Hver kant har en vekt som er et eller annet ikke-negativt tall. Når vi tegner en merket graf, merker vi hver kant med vekten.

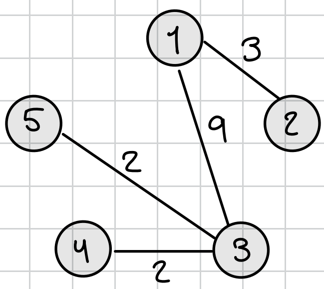
Eksempel

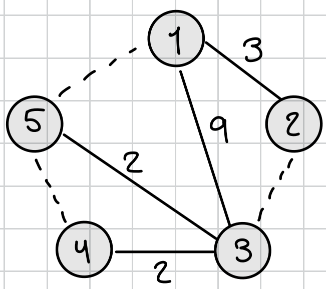
Dette er en vektet graf G. Her er alle vektene heltall.



Vektede trær

Dette er et vektet tre. Merk at treet T er et spenntre for grafen G i forrige eksempel.



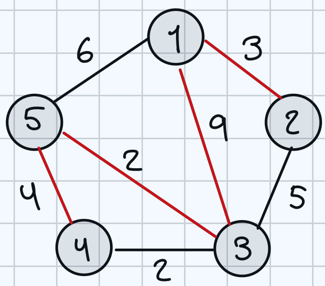


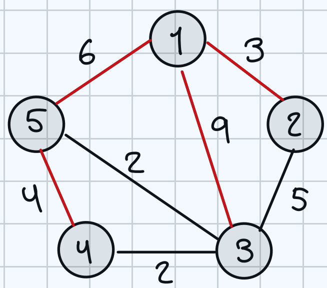
Totalvekten til et (spenn-)tre er summen av vektene.

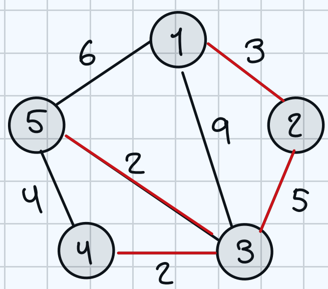
Her er totalvekten $2 + 2 + 9 + 3 = 16$

Minimale spenntrær - eksempel

Tre forskjellige spenntrær for samme graf.







Algoritmer for minimale spenntrær

Kruskal
Sorter først kentene etter vekt: start med en kent med lavest mulig vekt og legg til en og en, alltid med lavest mulig vekt, såfremt den ikke skaper sykel.

Prim
Start i vilkårlig node: utvid med kanten med lavest mulig vekt som er festet på denne noden: utvid med en kant (med lavest mulig vekt) slik at du har et tre: gjenta til du har et spenntre.

Begge gir MST for sammenhengende vektede grafer.
Forskjell: I Prim's algoritme bygger vi ut et større og større tre. I Kruskal's algoritme bygger vi en skog, som i siste steg, blir et tre (altså vi har ikke-sammenhengende grafer underveis i Kruskal's algoritme).

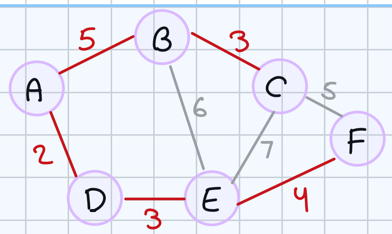
Prims algoritme

1. Start i en vilkårlig node. La dette være starttreet ditt. Du skal utvide dette til et spenntre ved å legge til nye kanter og noder.

2. Se på alle kanter som går fra en node i treet du har til en node utenfor. Av disse kantene, velg en med lavest mulig vekt. Legg den kanten og endenoden til treet.

3. Gjenta [steg 2] til du har et spenntre.

Prims algoritme eksempel



Start i A. Først: A-D (2). Neste billigste ut fra A,D er D-E (3), osv.

Totalvekt: 2+3+4+5+3=17

Kruskals algoritme

Vi kan også bruke Kruskals algoritme for å finne et minimalt spenntre til en vektet graf. Anta du har en vektet graf $G = (V, E)$ med $|V| = n$.

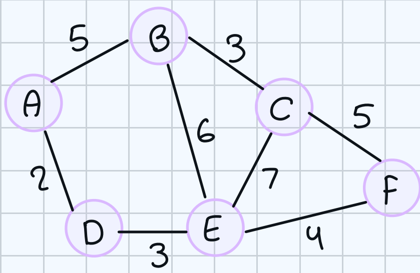
1 Start med en undergraf som består av alle nodene V og ingen kanter.

2 Lag en liste med alle kanter, sortert etter vekt (stigende).

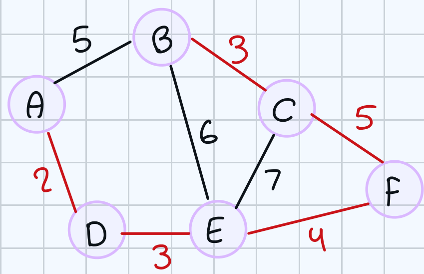
3 Gå gjennom kantene i denne rekkefølgen og legg til en kant hvis den ikke lager sykel med de allerede valgte kantene: ellers hopp over og gå videre på listen.

4 Når du har valgt n-1 kanter, har du et minimalt spenntre.

Kruskals algoritme - eksempel



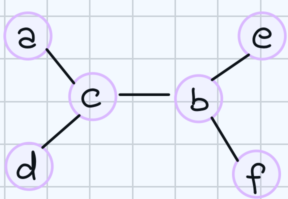
Kant	Vekt	Kant	Vekt
A-D	2	C-F	5
B-C	3	A-B	5
D-E	3	B-E	6
E-F	4	C-E	7



Det minimale spenntreet har totalvekt: $2 + 3 + 3 + 4 + 5 = 17$.

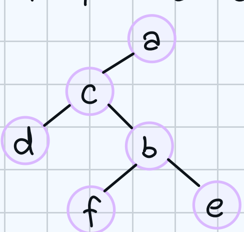
Rotfestede trær-eksempel

I datavitenskap er rotfestede trær en viktig struktur.

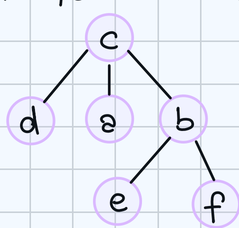


Grafen G, som er et tre.

Rotfestet tre med a som rot



Rotfestet tre med c som rot



Rotfestede trær

Et tre der en node er valgt som rot r.

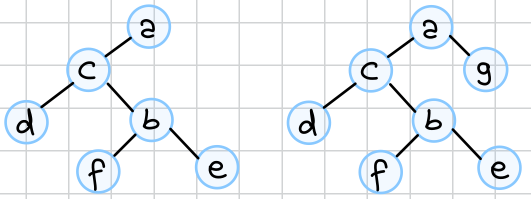
Gir hierarki-struktur: foreldre/barn struktur, søsken, undertrær.

Binærtrær

Binærtrær er rotfestede trær der hver node har høyst to barn (altså høyst to kanter nedover i hierarkiet).

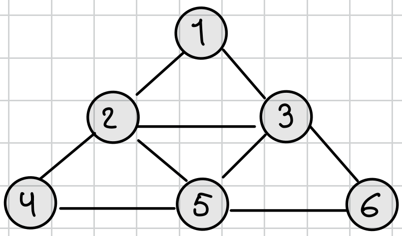
Et binærtre sies å være fullt, hvis hver node har enten 0 eller 2 barn.

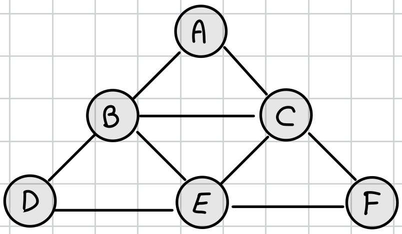
Eksempel (treet til høyre er et fullt binærtre):



Oppgave

Finn alle isomorfier mellom grafene.





Løsning: En grafisomorfi f må sende noder med grad t til noder av grad t. Så hvis f er en grafisomorfi, må $f(1) \in \{A, D, F\}$. Anta at $f(1) = A$. Siden 1 har kant til 2 og 3, og A har kant til B og C, må $f(2) \in \{B, C\}$. Hvis $f(2) = B$, må $f(3) = C$. Omvendt, hvis $f(2) = C$, må $f(3) = B$.

I begge tilfeller må $f(5) = E$, siden $f(5)$ må ha kant til både $f(2)$ og $f(3)$. Hvis $f(5) = B$ (og dermed $f(3) = C$ og $f(5) = E$), må $f(4) = D$, siden $f(4)$ skal ha kant til $f(2) = B$. Da er $f(6) = F$. Sjekk nå at $f(1) = A$, $f(2) = B$, $f(3) = C$, $f(4) = D$, $f(5) = E$, $f(6) = F$ er en isomorfi. Dvs. at det er en kant mellom x og y hvis og bare hvis det er kant mellom $f(x)$ og $f(y)$.

Tilsvarende hvis $f(2) = C$, må vi altså ha $f(3) = B$ og $f(5) = E$, og det følger at $f(4) = F$ og $f(6) = D$. Sjekk at $f(1) = A$, $f(2) = C$, $f(3) = B$, $f(4) = F$, $f(5) = E$, $f(6) = D$ er en isomorfi.

Dermed: Hvis $f(1) = A$ har to isomorfier. Tilsvarende (bruk symmetri), to isomorfier for $f(1) = D$, og to mulige isomorfier for $f(1) = F$.
Totalt: 6 isomorfier.