MAT1030 – Diskret matematikk

Forelesning 23: Grafteori

Dag Normann

Matematisk Institutt, Universitetet i Oslo

16. april 2008

Oppsummering

- En graf består av *noder* og *kanter*

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
  - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
  - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.
  - Håndhilselemmaet: Det er alltid et partall antall noder av odde grad i en graf.

Oppsummering

- En graf består av *noder* og *kanter*
- Kanter ligger *inntil* noder, og noder kan være *naboer*.
- Vi bør kjenne til begrepene om *sammenhengende* grafer, *tomme* grafer, *løkker*, *parallelle* kanter, *enkle* grafer og *komplette* grafer.
- Hver node har en *grad*.
  - Summen av gradene til alle nodene i en graf er lik 2 ganger antallet kanter.
  - Håndhilselemmaet: Det er alltid et partall antall noder av odde grad i en graf.
- Vi skal kjenne til *komplementet* av en graf og *matriserepresentasjoner*.

Oppsummering

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.

Oppsummering

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.
- Mange algoritmer/egenskaper kan forstås bedre ved å bruke grafer.

Oppsummering

- Grafer kan brukes til å representere omtrent alt som fins av relasjoner.
- Mange algoritmer/egenskaper kan forstås bedre ved å bruke grafer.
- Ofte er løsningen å kunne identifisere et problem som et *grafteorisk* problem.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.
- Vi definerte stier og

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.
- Vi definerte stier og
- kretser

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.
- Vi definerte stier og
- kretser
- En sti er en følge av noder og kanter slik at vi går fra node til node via kantene mellom dem.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.
- Vi definerte stier og
- kretser
- En sti er en følge av noder og kanter slik at vi går fra node til node via kantene mellom dem.

  Den presise definisjonen ble gitt mandag.

Oppsummering

- Vi har sett på isomorfibegrepet for grafer.
- To grafer er isomorfe hvis alle de viktige egenskapene er de samme.
- Mer presist
  - Det finnes en bijeksjon mellom nodene og mellom kantene slik at bildet av en kant går mellom bildet av to noder hvis og bare hvis kanten går mellom nodene.
- Vi definerte stier og
- kretser
- En sti er en følge av noder og kanter slik at vi går fra node til node via kantene mellom dem.

  Den presise definisjonen ble gitt mandag.
- En krets er en sti som begynner og slutter samme sted.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:
  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.
- Hvis vi representerer landene som noder og grensene som kanter, er dette egentlig et grafteoretisk problem.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.
- Hvis vi representerer landene som noder og grensene som kanter, er dette egentlig et grafteoretisk problem.
- Grafteori, som en matematisk tung disiplin, har mye å hente fra forsøkene på å løse dette problemet.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.
- Hvis vi representerer landene som noder og grensene som kanter, er dette egentlig et grafteoretisk problem.
- Grafteori, som en matematisk tung disiplin, har mye å hente fra forsøkene på å løse dette problemet.
- Måten problemet ble løst på har interesse i seg selv.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.
- Hvis vi representerer landene som noder og grensene som kanter, er dette egentlig et grafteoretisk problem.
- Grafteori, som en matematisk tung disiplin, har mye å hente fra forsøkene på å løse dette problemet.
- Måten problemet ble løst på har interesse i seg selv.
- De som løste det, reduserte problemet til et stort antall enkelttilfeller, som deretter ble sjekket av en datamaskin.

Digresjon: Firefarveproblemet

- I mange, mange år var følgende et åpent matematisk problem:

  Anta at vi har et plant kart over landområder (land, fylker, stater o.l.).

  Er det alltid mulig å trykke kartet ved hjelp av bare fire farver slik at to landområder som grenser opp mot hverandre alltid har forskjellig farve.
- Hvis vi representerer landene som noder og grensene som kanter, er dette egentlig et grafteoretisk problem.
- Grafteori, som en matematisk tung disiplin, har mye å hente fra forsøkene på å løse dette problemet.
- Måten problemet ble løst på har interesse i seg selv.
- De som løste det, reduserte problemet til et stort antall enkelttilfeller, som deretter ble sjekket av en datamaskin.
- Var det mennesker eller datamaskinen som løste problemet?

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.
- Mandag så vi på en algoritme som fant en Eulerkrets når det var mulig.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.
- Mandag så vi på en algoritme som fant en Eulerkrets når det var mulig.
- Den kan også brukes til å finne en Eulersti, det vil si en sti som er innom alle kantene nøyaktig en gang, men som kan begynne og slutte på forskjellige steder.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.
- Mandag så vi på en algoritme som fant en Eulerkrets når det var mulig.
- Den kan også brukes til å finne en Eulersti, det vil si en sti som er innom alle kantene nøyaktig en gang, men som kan begynne og slutte på forskjellige steder.
- Disse stedene må da være de to nodene med odde grad.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.
- Mandag så vi på en algoritme som fant en Eulerkrets når det var mulig.
- Den kan også brukes til å finne en Eulersti, det vil si en sti som er innom alle kantene nøyaktig en gang, men som kan begynne og slutte på forskjellige steder.
- Disse stedene må da være de to nodene med odde grad.
- Utvider vi grafen med en kant mellom disse to nodene, kan vi lage en Eulerkrets.

Eulerstier

- En Eulerkrets er en krets i en graf som inneholder hver kant nøyaktig en gang.
- Euler fant ut at en graf har en Eulerkrets nøyaktig når den er sammenhengende og hver node har et partall som grad.
- Mandag så vi på en algoritme som fant en Eulerkrets når det var mulig.
- Den kan også brukes til å finne en Eulersti, det vil si en sti som er innom alle kantene nøyaktig en gang, men som kan begynne og slutte på forskjellige steder.
- Disse stedene må da være de to nodene med odde grad.
- Utvider vi grafen med en kant mellom disse to nodene, kan vi lage en Eulerkrets.
- Tar vi bort den nye kanten, får vi en Eulersti.

Eulerstier

Eksempel

Eulerstier

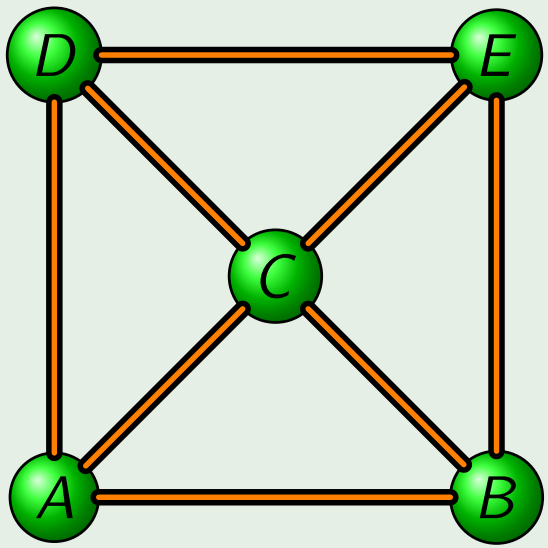
Eksempel

La oss se på et eksempel:

Eulerstier

Eksempel

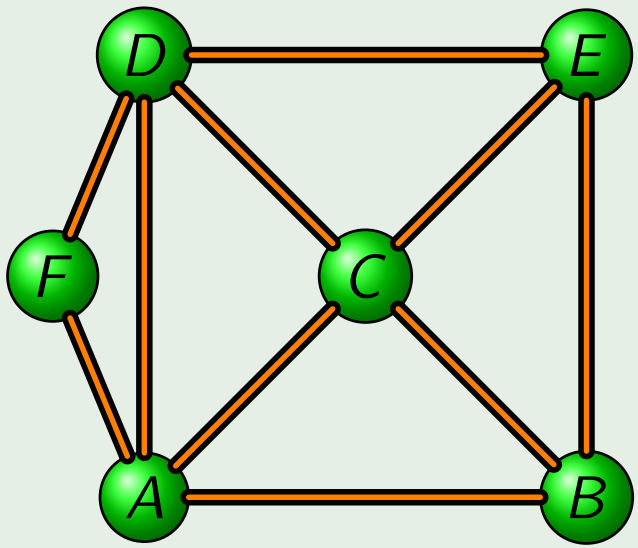
La oss se på et eksempel:



Eulerstier

Eksempel

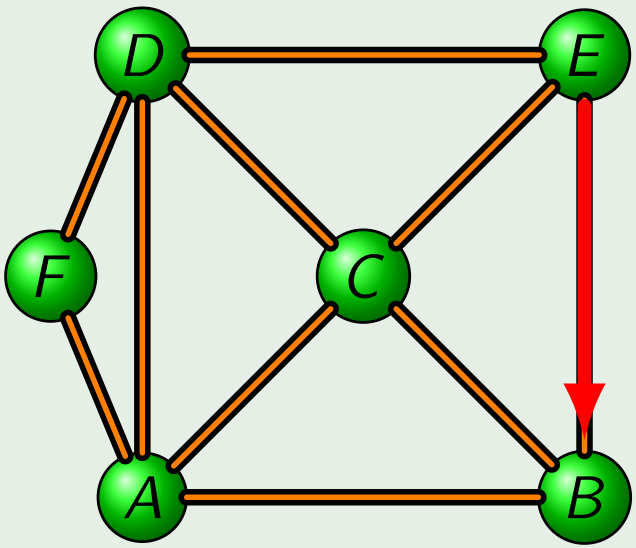
La oss se på et eksempel:



Eulerstier

Eksempel

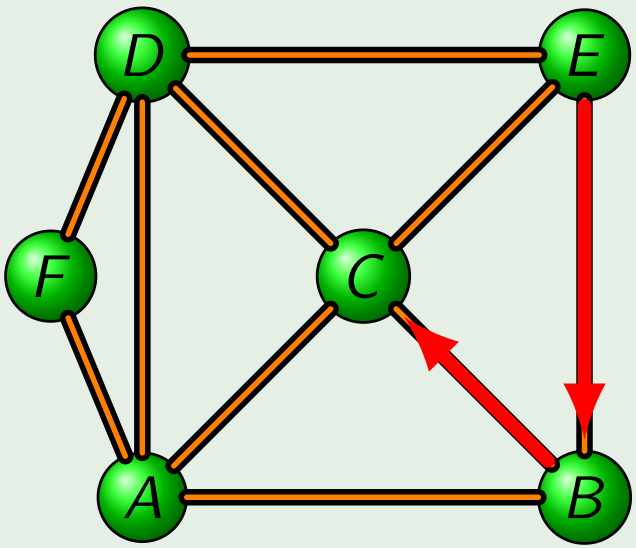
La oss se på et eksempel:



Eulerstier

Eksempel

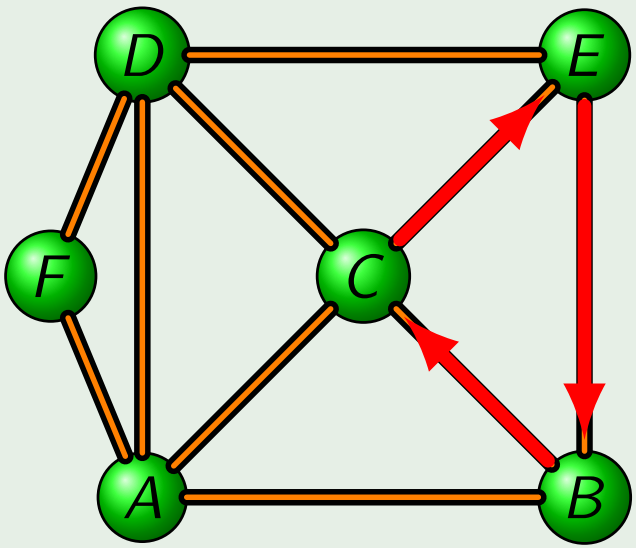
La oss se på et eksempel:



Eulerstier

Eksempel

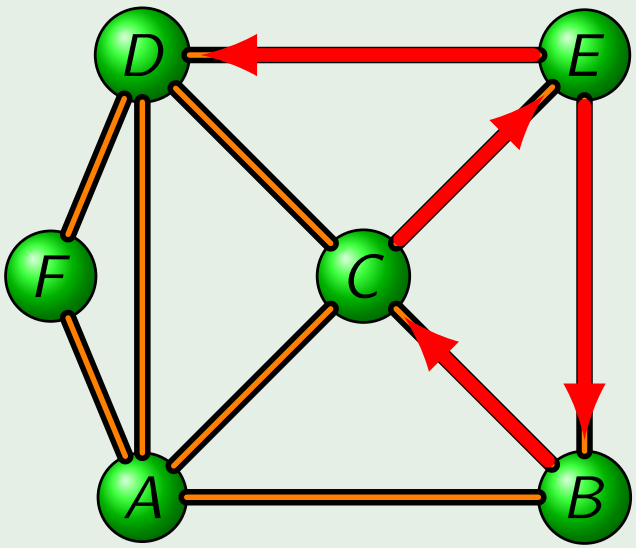
La oss se på et eksempel:



Eulerstier

Eksempel

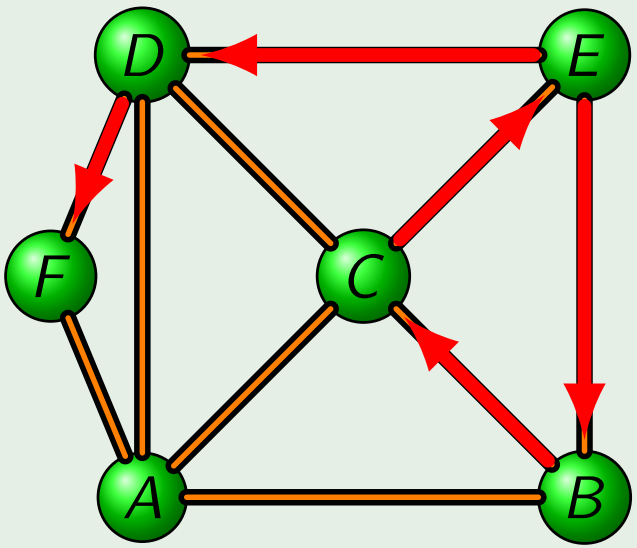
La oss se på et eksempel:



Eulerstier

Eksempel

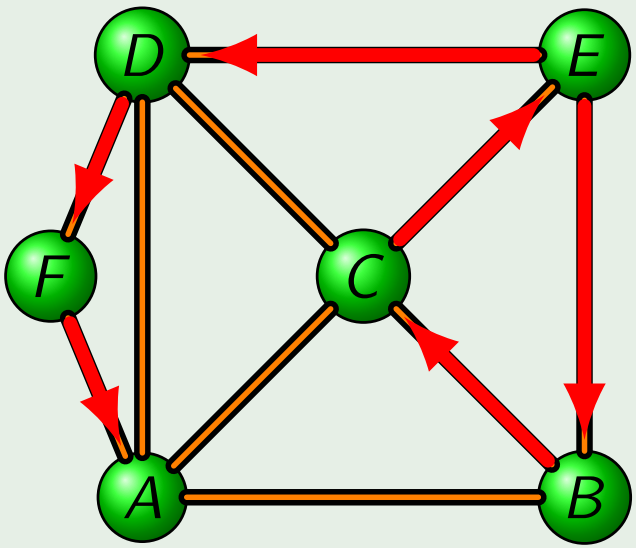
La oss se på et eksempel:



Eulerstier

Eksempel

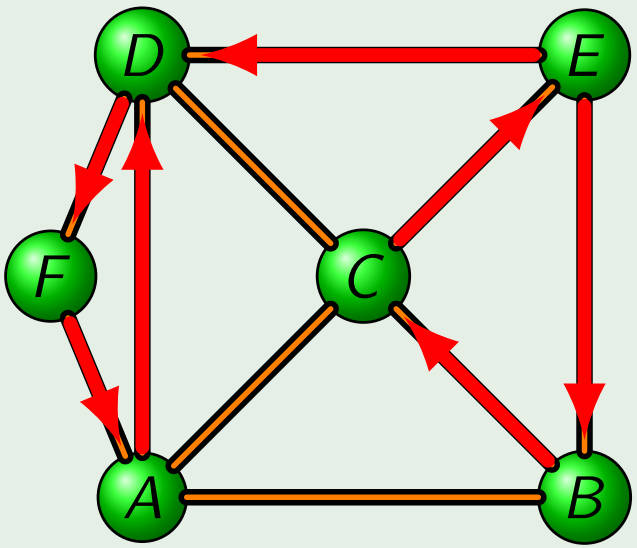
La oss se på et eksempel:



Eulerstier

Eksempel

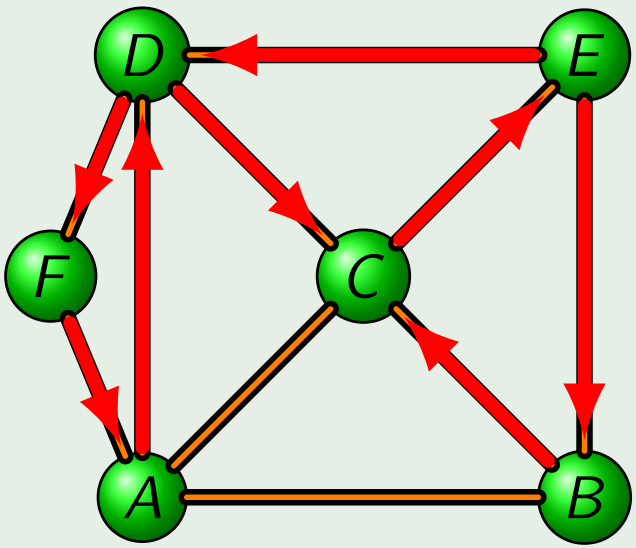
La oss se på et eksempel:



Eulerstier

Eksempel

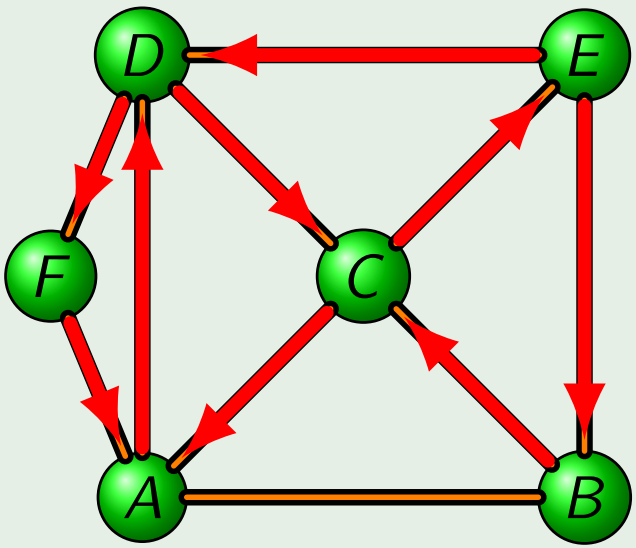
La oss se på et eksempel:



Eulerstier

Eksempel

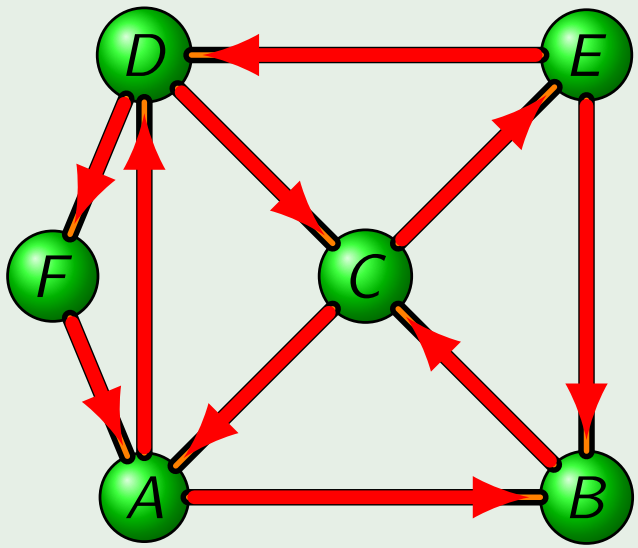
La oss se på et eksempel:



Eulerstier

Eksempel

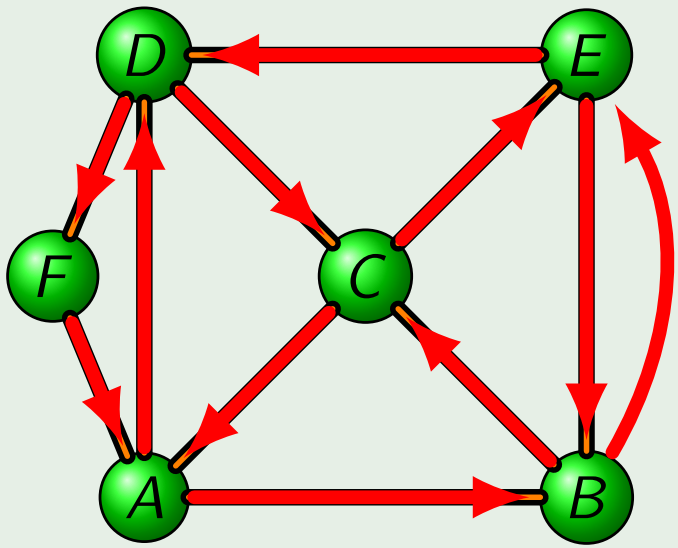
La oss se på et eksempel:



Eulerstier

Eksempel

La oss se på et eksempel:



Eulerstier

- Det er to strategier for å bevise en setning som Eulers.

Eulerstier

- Det er to strategier for å bevise en setning som Eulers.
- Vi kan bruke et induksjonsbevis.

Eulerstier

- Det er to strategier for å bevise en setning som Eulers.
- Vi kan bruke et induksjonsbevis.
- Induksjonsbevis gir ofte opphav til algoritmer, og den algoritmen boka presenterer kan sees på som utledet av et induksjonsbevis.

Eulerstier

- Det er to strategier for å bevise en setning som Eulers.
- Vi kan bruke et induksjonsbevis.
- Induksjonsbevis gir ofte opphav til algoritmer, og den algoritmen boka presenterer kan sees på som utledet av et induksjonsbevis.
- Vi kan også bevise teoremet *direkte*, uten å argumentere via algoritmen.

Eulerstier

- Det er to strategier for å bevise en setning som Eulers.
- Vi kan bruke et induksjonsbevis.
- Induksjonsbevis gir ofte opphav til algoritmer, og den algoritmen boka presenterer kan sees på som utledet av et induksjonsbevis.
- Vi kan også bevise teoremet *direkte*, uten å argumentere via algoritmen.
- Her følger et bevis for at det er alltid finnes en Eulerkrets hvis hver node har grad lik et partall.

Eulerstier

- Det er to strategier for å bevise en setning som Eulers.
- Vi kan bruke et induksjonsbevis.
- Induksjonsbevis gir ofte opphav til algoritmer, og den algoritmen boka presenterer kan sees på som utledet av et induksjonsbevis.
- Vi kan også bevise teoremet *direkte*, uten å argumentere via algoritmen.
- Her følger et bevis for at det er alltid finnes en Eulerkrets hvis hver node har grad lik et partall.
- Det er også mulig å trekke en algoritme ut av dette beviset, og algoritmen blir veldig lik den vi så på på mandag.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets. Vi skal gjøre dette ved å vise følgende tre påstander:

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets. Vi skal gjøre dette ved å vise følgende tre påstander:

(a) S er en krets.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets. Vi skal gjøre dette ved å vise følgende tre påstander:

(a) S er en krets.

(b) S inneholder alle nodene i grafen.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets. Vi skal gjøre dette ved å vise følgende tre påstander:

(a) S er en krets.

(b) S inneholder alle nodene i grafen.

(c) S inneholder alle kantene i grafen.

Eulerstier

Bevis (1)

La G være en sammenhengende graf hvor gradene til alle nodene er partall. La S være den *lengste* stien

$$v_0 e_1 v_1 e_2 v_2 \ldots e_n v_n$$

slik at ingen kant forekommer to ganger. Vi skal vise at S er en Eulerkrets. Vi skal gjøre dette ved å vise følgende tre påstander:

(a) S er en krets.

(b) S inneholder alle nodene i grafen.

(c) S inneholder alle kantene i grafen.

Siden ingen kant forekommer to ganger, må S være en Eulerkrets.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S, og da hadde ikke S vært maksimal.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S, og da hadde ikke S vært maksimal. Siden graden til v_n er et partall

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S, og da hadde ikke S vært maksimal. Siden graden til v_n er et partall, så må vi tidligere i stien ha gått ut av v_n.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S, og da hadde ikke S vært maksimal. Siden graden til v_n er et partall, så må vi tidligere i stien ha gått ut av v_n. Den eneste muligheten er at v_n er lik v_0.

Bevis (Fortsatt)

(a) Noden v_0 må være lik v_n. Når vi går ut av den første noden, v_0, via kanten e_1, så bruker vi opp én av kantene som ligger inntil v_0. For hver node vi går inn i og ut av, så bruker vi opp to kanter. Når vi er fremme ved den siste noden i stien, v_n, så fins det ingen ubrukt kant som ligger inntil v_n. Hadde det vært en slik kant, så ville vi hatt en sti som var lenger enn S, og da hadde ikke S vært maksimal. Siden graden til v_n er et partall, så må vi tidligere i stien ha gått ut av v_n. Den eneste muligheten er at v_n er lik v_0. Dermed er S en *krets*.

Bevis

(b) S må bestå av alle nodene i grafen.

Bevis

(b) S må bestå av alle nodene i grafen. Det er fordi grafen er sammenhengende og S er maksimal.

Bevis

(b) S må bestå av alle nodene i grafen. Det er fordi grafen er sammenhengende og S er maksimal. Hvis en node v ikke hadde vært med

Bevis

(b) S må bestå av alle nodene i grafen. Det er fordi grafen er sammenhengende og S er maksimal. Hvis en node v ikke hadde vært med, så kunne vi ha laget en krets som var lenger enn S.

Bevis

(c) S inneholder alle kantene fra grafen.

Bevis

(c) S inneholder alle kantene fra grafen. Anta for motsigelse at det fins en kant e, som forbinder nodene u og v, som ikke er med i S.

Bevis

(c) S inneholder alle kantene fra grafen. Anta for motsigelse at det fins en kant e, som forbinder nodene u og v, som ikke er med i S. Siden S inneholder alle nodene fra grafen, så må v være lik v_k for en passende k.

Bevis

(c) S inneholder alle kantene fra grafen. Anta for motsigelse at det fins en kant e, som forbinder nodene u og v, som ikke er med i S. Siden S inneholder alle nodene fra grafen, så må v være lik v_k for en passende k. Da kan vi lage en sti som er lenger enn S ved å begynne med ue og fortsette med S:

Bevis

(c) S inneholder alle kantene fra grafen. Anta for motsigelse at det fins en kant e, som forbinder nodene u og v, som ikke er med i S. Siden S inneholder alle nodene fra grafen, så må v være lik v_k for en passende k. Da kan vi lage en sti som er lenger enn S ved å begynne med ue og fortsette med S:

$$ue \underbrace{v_k e_{k+1} v_{k+1} \ldots e_n v_n e_1 v_1 e_2 v_2 \ldots e_{k-1} v_{k-1} e_k v_k}_{S}$$

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

Definisjon

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

Definisjon

La G være en sammenhengende graf.

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

Definisjon

La G være en sammenhengende graf. En *Hamiltonsti* er en sti som inneholder hver node fra G nøyaktig én gang.

- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

Definisjon

La G være en sammenhengende graf. En *Hamiltonsti* er en sti som inneholder hver node fra G nøyaktig én gang. En *Hamiltonkrets* er en Hamiltonsti hvor den første og den siste noden sammenfaller.

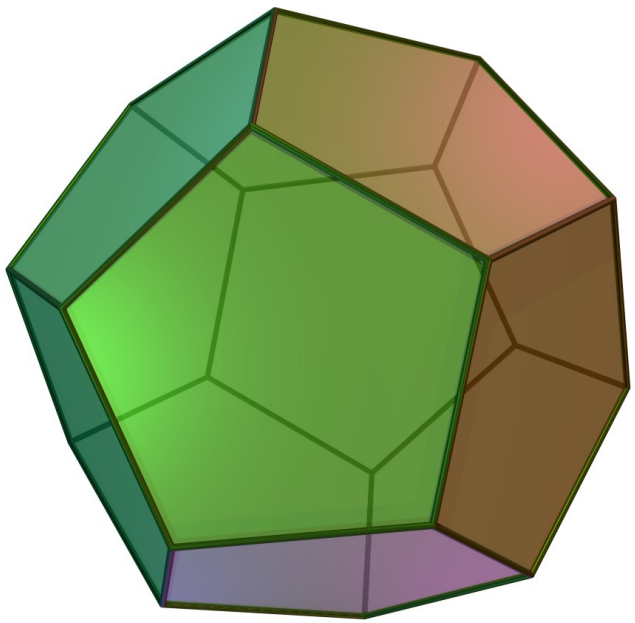
- Vi må også si litt om stier som inneholder alle *nodene* i en graf, uavhengig hvorvidt alle kantene er med eller en kant er med flere ganger.
- "Den handelsreisendes problem" er et slikt problem, hvor man er ute etter den *korteste* stien som går gjennom alle byene i en mengde.

Definisjon

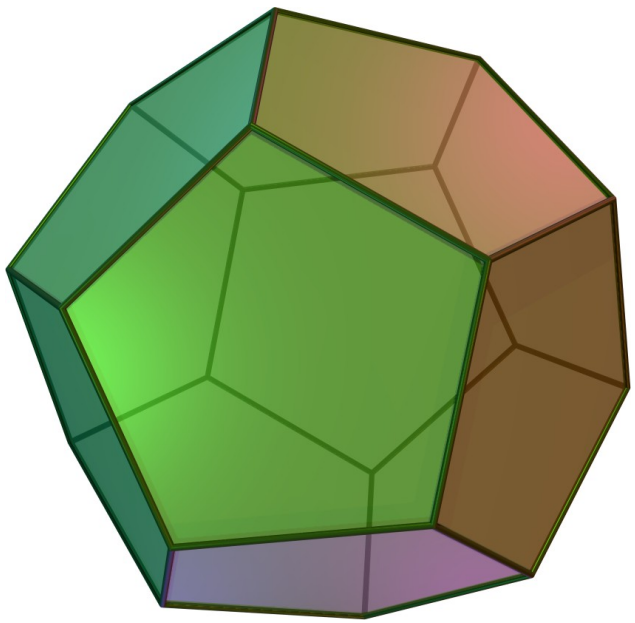
La G være en sammenhengende graf. En *Hamiltonsti* er en sti som inneholder hver node fra G nøyaktig én gang. En *Hamiltonkrets* er en Hamiltonsti hvor den første og den siste noden sammenfaller. En sammenhengende graf som har en Hamiltonkrets kalles *Hamiltonsk*.

- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene)

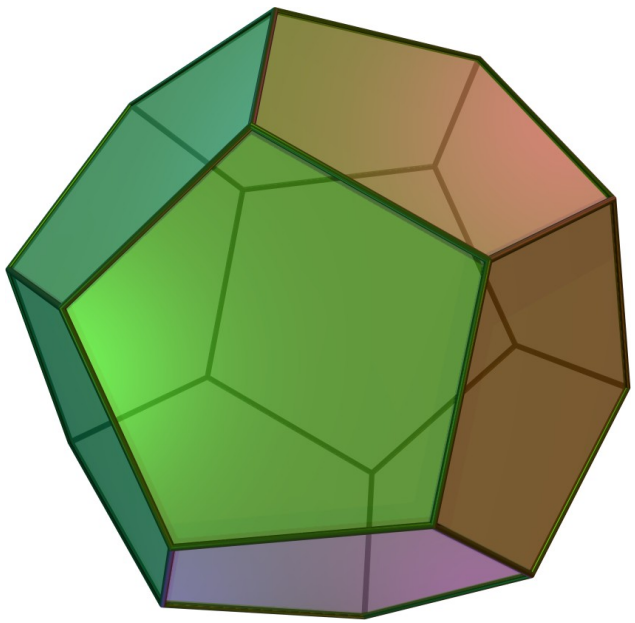
- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene)



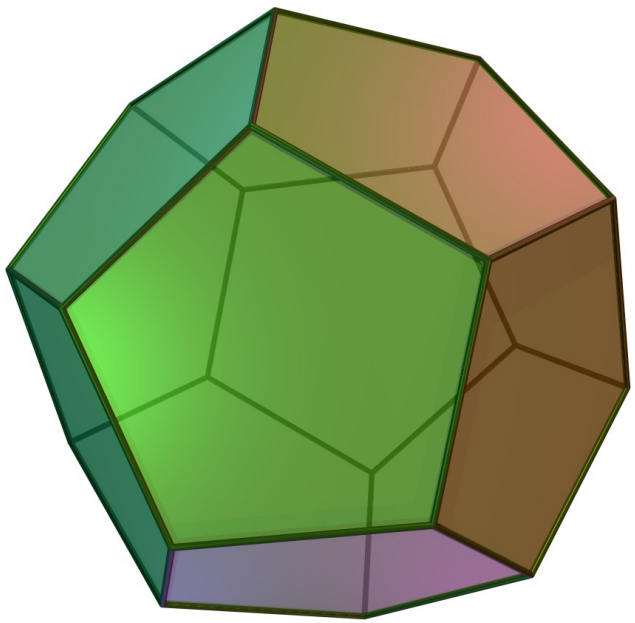
- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by.



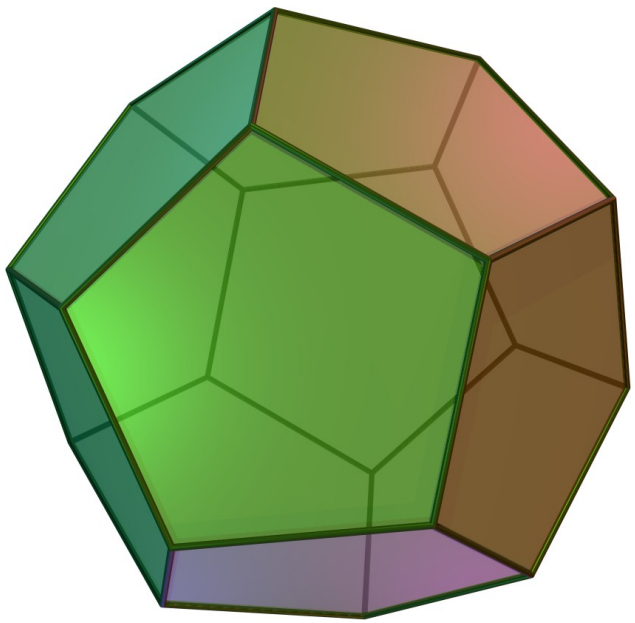
- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by. Spørsmålet han stilte var om det var mulig å reise gjennom alle byene nøyaktig én gang.

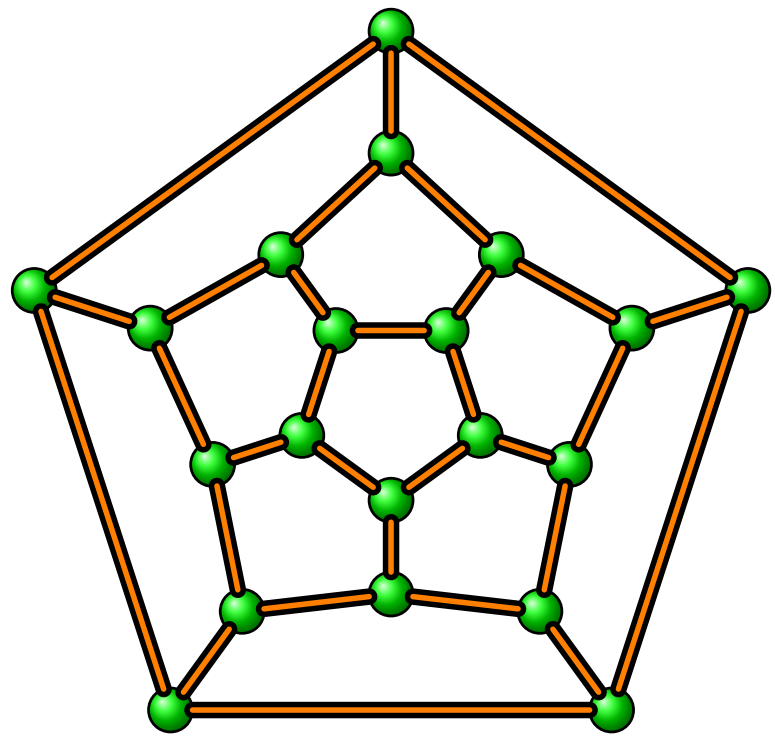


- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by. Spørsmålet han stilte var om det var mulig å reise gjennom alle byene nøyaktig én gang. Vi ser at dette spørsmålet er det samme som om den tilhørende grafen har en Hamiltonsti.

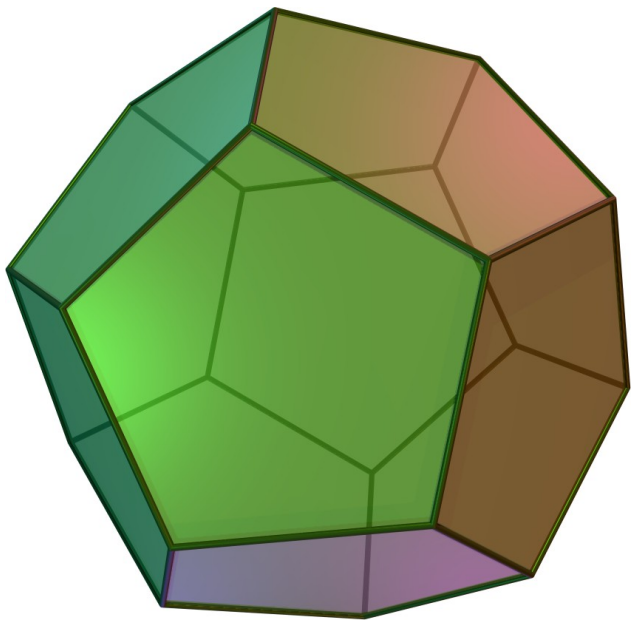


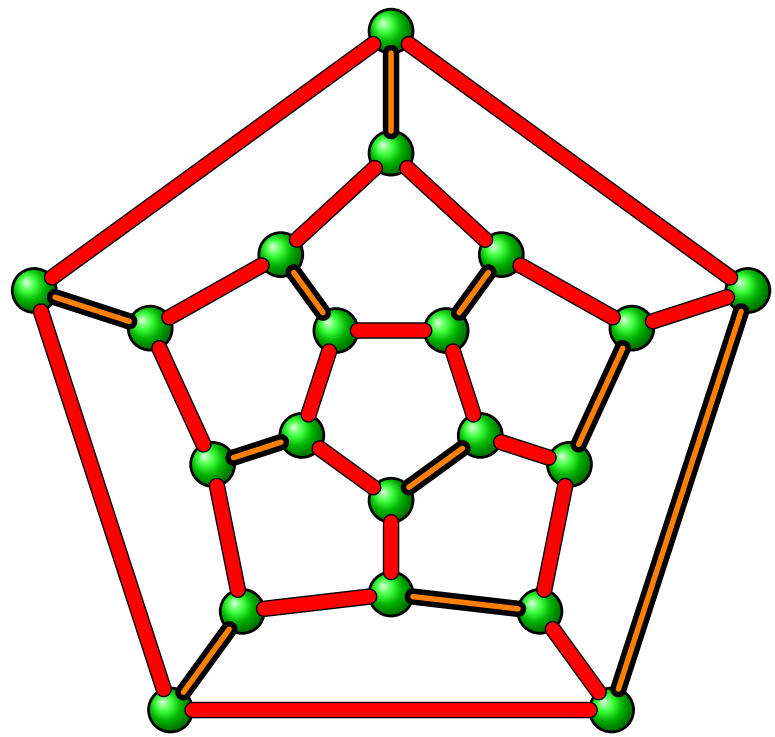
- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by. Spørsmålet han stilte var om det var mulig å reise gjennom alle byene nøyaktig én gang. Vi ser at dette spørsmålet er det samme som om den tilhørende grafen har en Hamiltonsti.



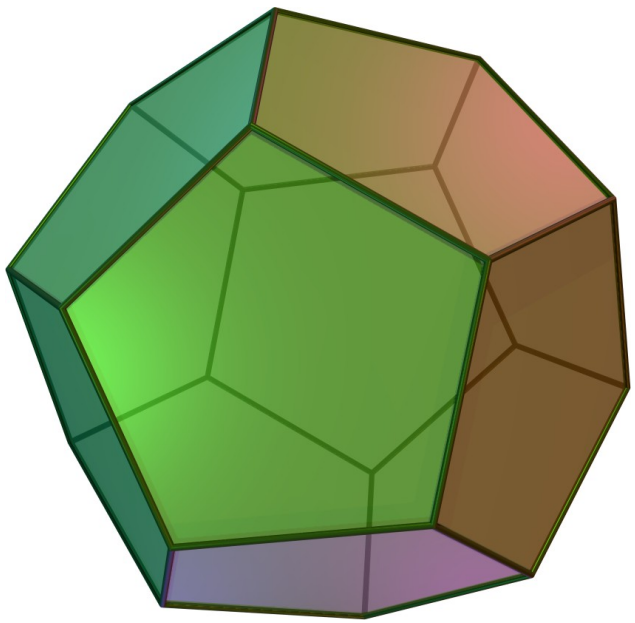


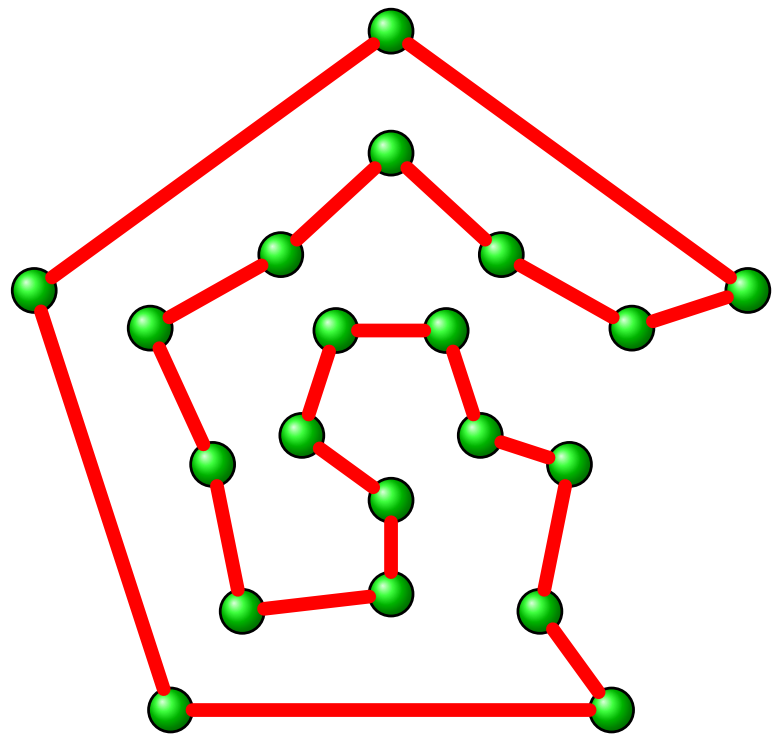
- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by. Spørsmålet han stilte var om det var mulig å reise gjennom alle byene nøyaktig én gang. Vi ser at dette spørsmålet er det samme som om den tilhørende grafen har en Hamiltonsti.





- Hamiltons puzzle tar utgangspunkt i et dodekaeder (et av de fem Platonske legemene) hvor hvert hjørne er merket med navnet på en by. Spørsmålet han stilte var om det var mulig å reise gjennom alle byene nøyaktig én gang. Vi ser at dette spørsmålet er det samme som om den tilhørende grafen har en Hamiltonsti.





- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.
- Dette er "like vanskelig" som å bestemme om et utsagnslogisk utsagn er en tautologi.

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.
- Dette er "like vanskelig" som å bestemme om et utsagnslogisk utsagn er en tautologi.

  [Det tilhører klassen av **NP**-komplette problemer.]

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.
- Dette er "like vanskelig" som å bestemme om et utsagnslogisk utsagn er en tautologi.

  [Det tilhører klassen av **NP**-komplette problemer.]
- I praksis er det sjeldent at man virkelig trenger å finne en Hamiltonkrets.

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.
- Dette er "like vanskelig" som å bestemme om et utsagnslogisk utsagn er en tautologi.

  [Det tilhører klassen av **NP**-komplette problemer.]
- I praksis er det sjeldent at man virkelig trenger å finne en Hamiltonkrets.
- Ofte er det tilstrekkelig å finne en Eulerkrets, eller greit å gå over noder flere ganger.

- Euler studerte også et tilsvarende problem: når det er mulig for en springer å gå over *alle* rutene på sjakkbrett av ulike størrelser.
- Det er heller ingen som har klart å lage en *effektiv* algoritme for å finne ut om det fins en Hamiltonkrets i en graf.
- Dette er "like vanskelig" som å bestemme om et utsagnslogisk utsagn er en tautologi.

  [Det tilhører klassen av **NP**-komplette problemer.]
- I praksis er det sjeldent at man virkelig trenger å finne en Hamiltonkrets.
- Ofte er det tilstrekkelig å finne en Eulerkrets, eller greit å gå over noder flere ganger.
- Det fins mange spesialtilfeller og heuristikker man kan benytte seg av.

Trær

- Et tre er en spesiell type graf.

Trær

- Et tre er en spesiell type graf.
- Intuitivt er et tre noe som vokser fra en rot

Trær

- Et tre er en spesiell type graf.
- Intuitivt er et tre noe som vokser fra en rot

  og så forgrener seg uten noe sted å vokse sammen igjen.

Trær

- Et tre er en spesiell type graf.
- Intuitivt er et tre noe som vokser fra en rot

  og så forgrener seg uten noe sted å vokse sammen igjen.
- Vi kan se på et biologisk tre som en graf, ved å la hvert forgreningspunkt være nodene, og delene av en stamme, gren eller kvist mellom to forgreningspunkter være kantene.

Trær

- Et tre er en spesiell type graf.
- Intuitivt er et tre noe som vokser fra en rot

  og så forgrener seg uten noe sted å vokse sammen igjen.
- Vi kan se på et biologisk tre som en graf, ved å la hvert forgreningspunkt være nodene, og delene av en stamme, gren eller kvist mellom to forgreningspunkter være kantene.
- Vi skal gi en presis definisjon av når en graf kan betraktes som et tre.

Trær

- Hvorfor skal vi lære om trær?

Trær

- Hvorfor skal vi lære om trær?
- Hvis en graf representerer et nettverk, vil et tre svare til et nettverk hvor det bare finnes en sti fra en node til en annen.

Trær

- Hvorfor skal vi lære om trær?
- Hvis en graf representerer et nettverk, vil et tre svare til et nettverk hvor det bare finnes en sti fra en node til en annen.
- Hvis nettverket består av kabler eller andre medier som formidler informasjon, kan det være hensiktsmessig at signaler bare går langs en vei, slik at systemet ikke forstyrres av at samme informasjon kommer med små tidsforskjeller.

Trær

- Hvorfor skal vi lære om trær?
- Hvis en graf representerer et nettverk, vil et tre svare til et nettverk hvor det bare finnes en sti fra en node til en annen.
- Hvis nettverket består av kabler eller andre medier som formidler informasjon, kan det være hensiktsmessig at signaler bare går langs en vei, slik at systemet ikke forstyrres av at samme informasjon kommer med små tidsforskjeller.
- Dataobjekter som sammensatte algebraiske uttrykk, utsagnslogiske formler eller program har ofte en trestruktur som beskrive hvordan komplekse objekter er bygget opp fra enklere objekter.

Trær

- Hvorfor skal vi lære om trær?
- Hvis en graf representerer et nettverk, vil et tre svare til et nettverk hvor det bare finnes en sti fra en node til en annen.
- Hvis nettverket består av kabler eller andre medier som formidler informasjon, kan det være hensiktsmessig at signaler bare går langs en vei, slik at systemet ikke forstyrres av at samme informasjon kommer med små tidsforskjeller.
- Dataobjekter som sammensatte algebraiske uttrykk, utsagnslogiske formler eller program har ofte en trestruktur som beskrive hvordan komplekse objekter er bygget opp fra enklere objekter.
- For å undersøke om et utsagn formalisert i matematikken kan bevises eller ikke, kan man prøve å bygge opp et tre av utsagn hvor forgreningen stopper når vi har nådd aksiomene.

Trær

- Hvorfor skal vi lære om trær?
- Hvis en graf representerer et nettverk, vil et tre svare til et nettverk hvor det bare finnes en sti fra en node til en annen.
- Hvis nettverket består av kabler eller andre medier som formidler informasjon, kan det være hensiktsmessig at signaler bare går langs en vei, slik at systemet ikke forstyrres av at samme informasjon kommer med små tidsforskjeller.
- Dataobjekter som sammensatte algebraiske uttrykk, utsagnslogiske formler eller program har ofte en trestruktur som beskrive hvordan komplekse objekter er bygget opp fra enklere objekter.
- For å undersøke om et utsagn formalisert i matematikken kan bevises eller ikke, kan man prøve å bygge opp et tre av utsagn hvor forgreningen stopper når vi har nådd aksiomene.

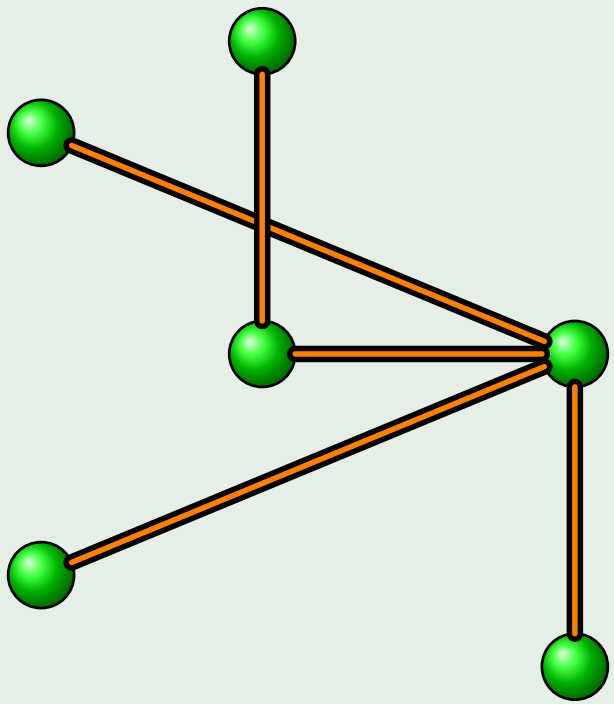
  Denne naive ideen danner grunnlag for enkelte automatiske bevis-søkere.

Trær

Trær

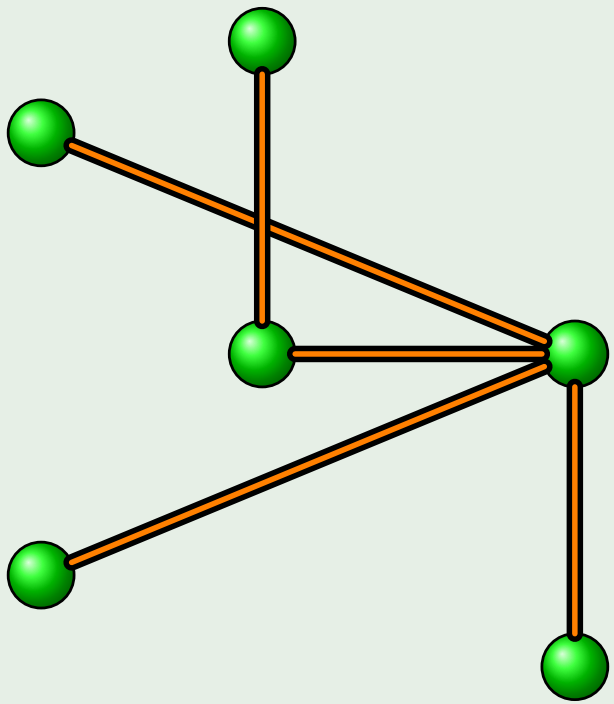
Trær

Eksempel



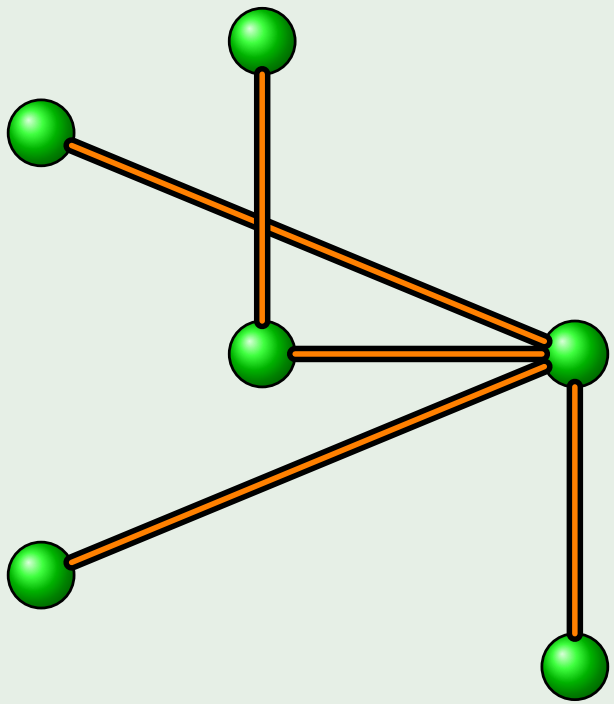
Trær

Eksempel

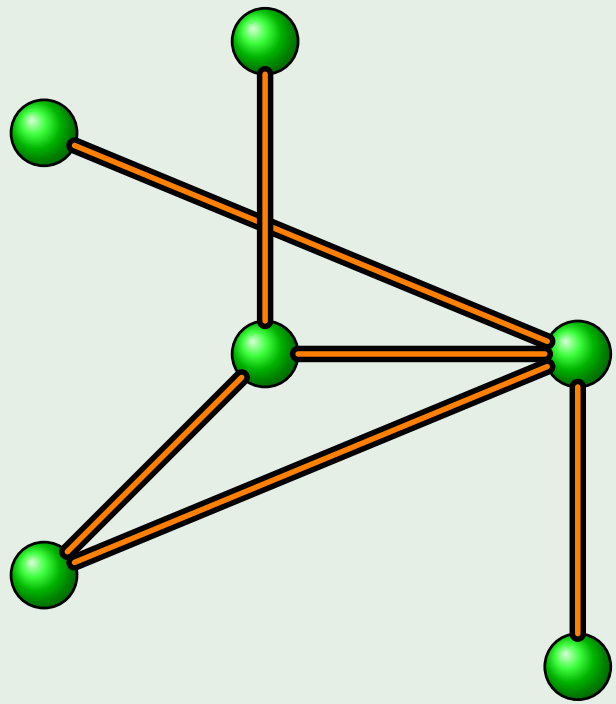


Trær

Eksempel



Eksempel



Trær

Definisjon

Trær

Definisjon

a) En sykel(Eng. cycle) i en graf er en sti med følgende egenskaper:

Trær

Definisjon

a) En sykel(Eng. cycle) i en graf er en sti med følgende egenskaper:
   - Stien inneholder minst en kant.

Trær

Definisjon

a) En sykel(Eng. cycle) i en graf er en sti med følgende egenskaper:
   - Stien inneholder minst en kant.
   - Ingen kant forekommer mer enn en gang.

Trær

Definisjon

a) En sykel(Eng. cycle) i en graf er en sti med følgende egenskaper:
   - Stien inneholder minst en kant.
   - Ingen kant forekommer mer enn en gang.
   - Stien er en krets, dvs. den begynner og slutter i samme node.

Trær

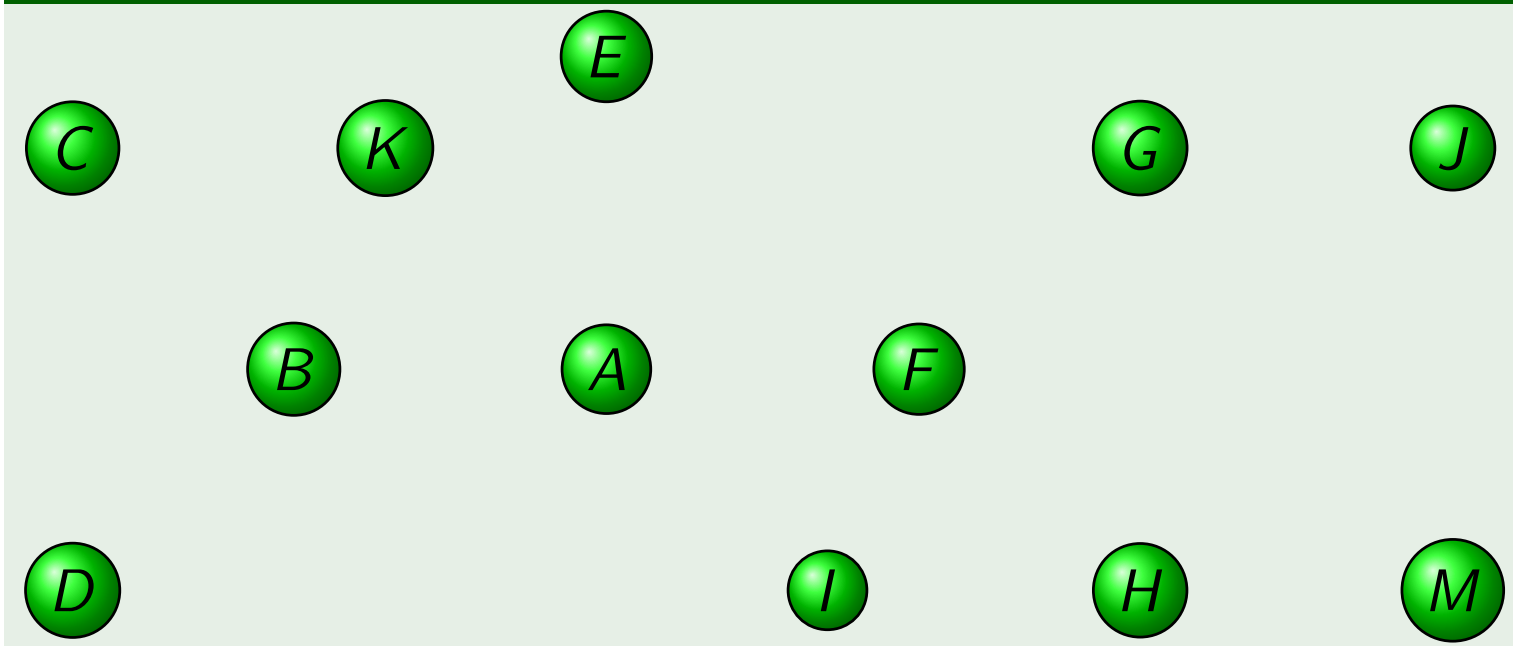
Definisjon

a) En sykel(Eng. cycle) i en graf er en sti med følgende egenskaper:
   - Stien inneholder minst en kant.
   - Ingen kant forekommer mer enn en gang.
   - Stien er en krets, dvs. den begynner og slutter i samme node.

b) En graf er et tre hvis grafen er sammenhengende og grafen ikke inneholder noen sykler.

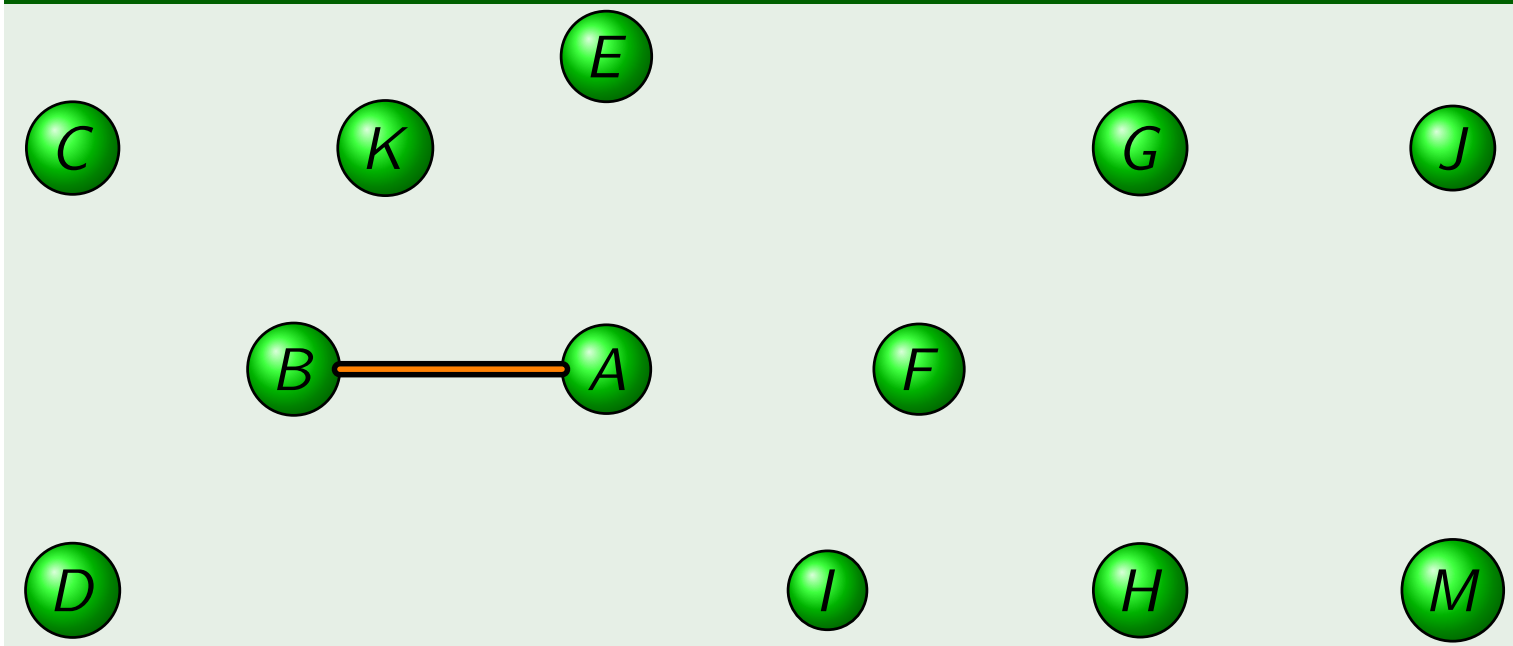
Trær

Eksempel



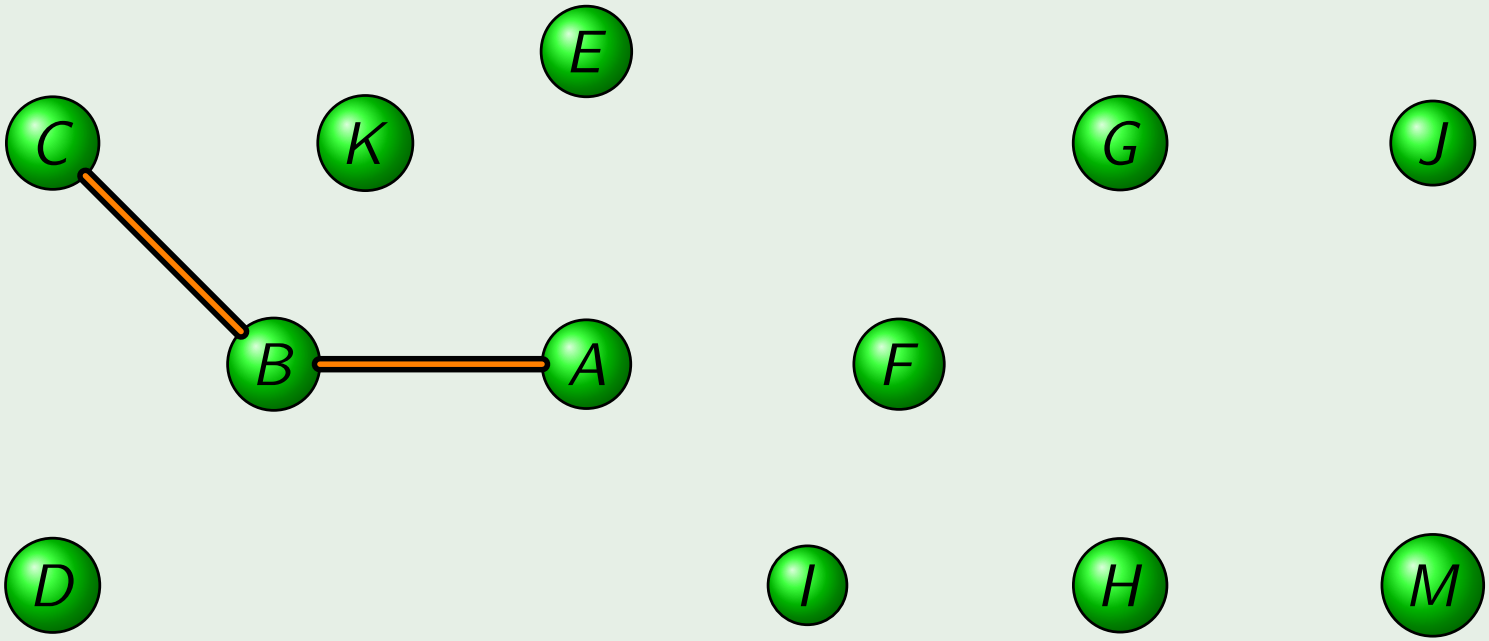
Trær

Eksempel



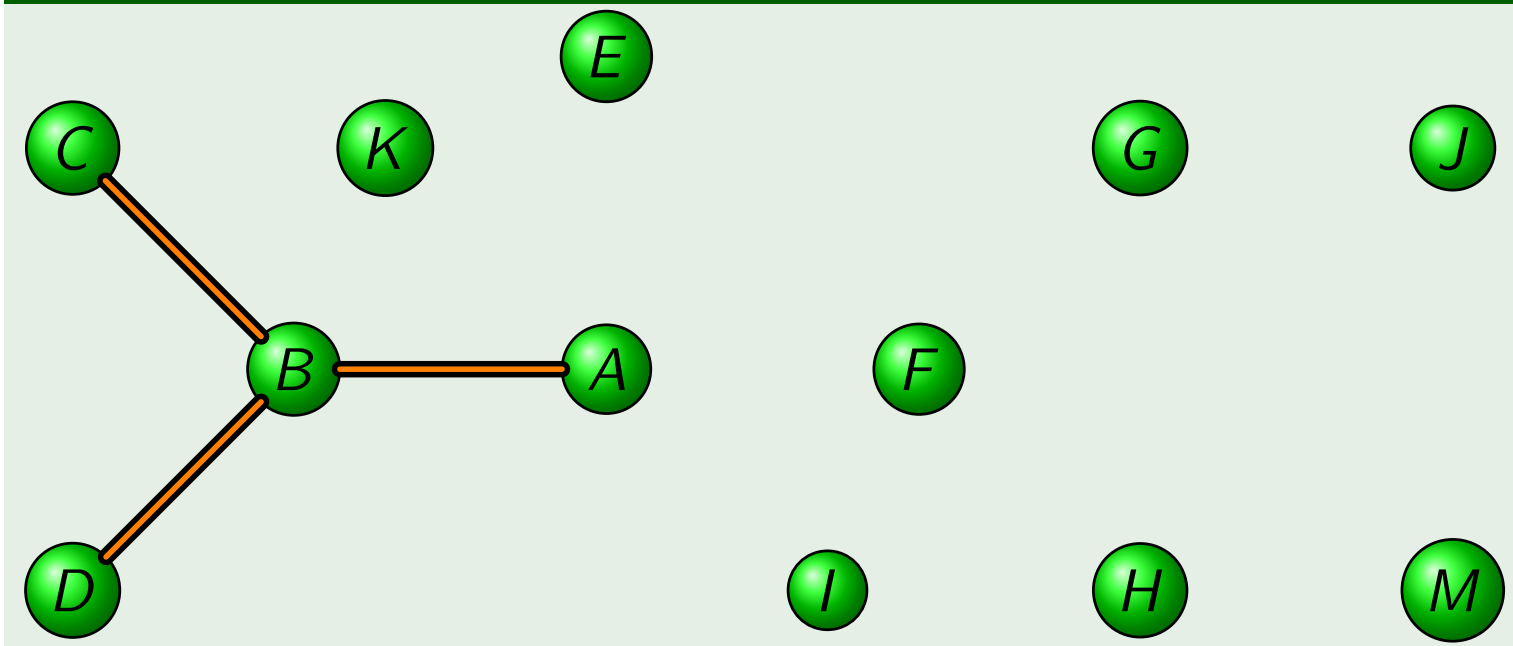
Trær

Eksempel



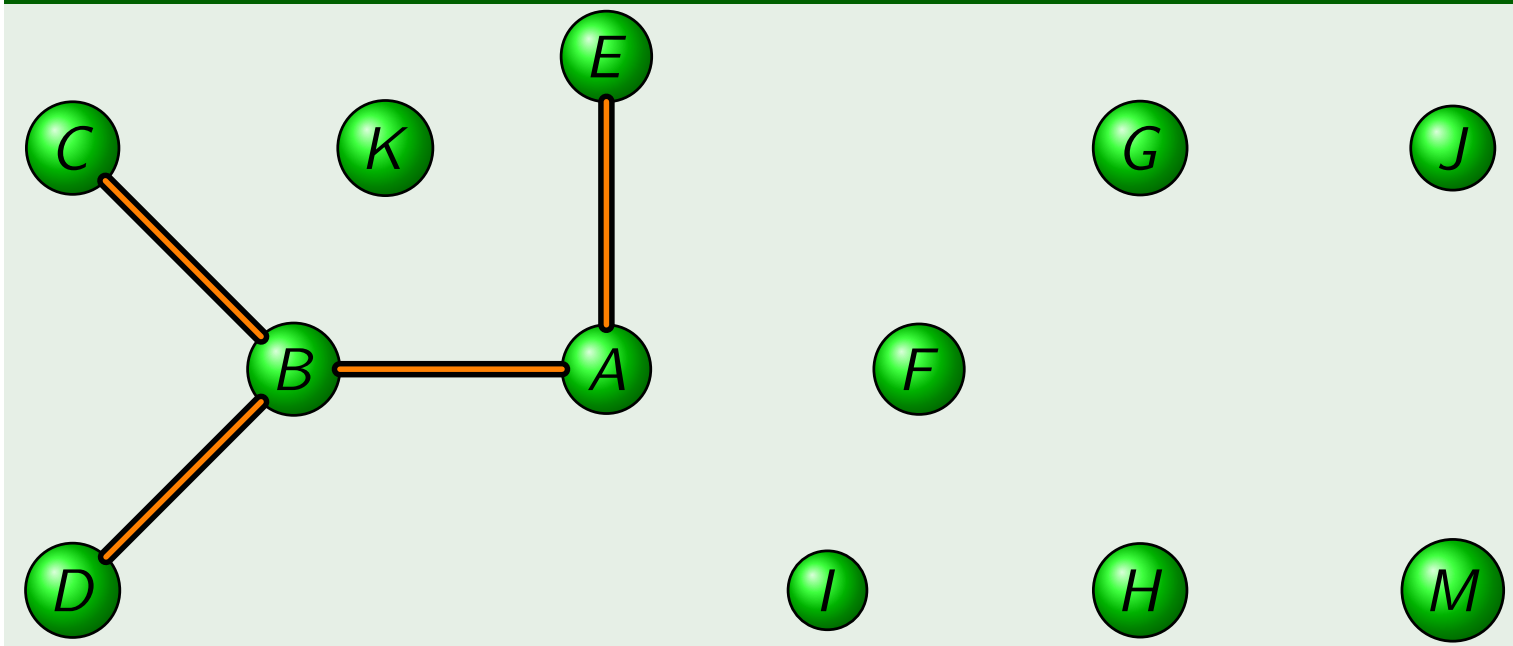
Trær

Eksempel



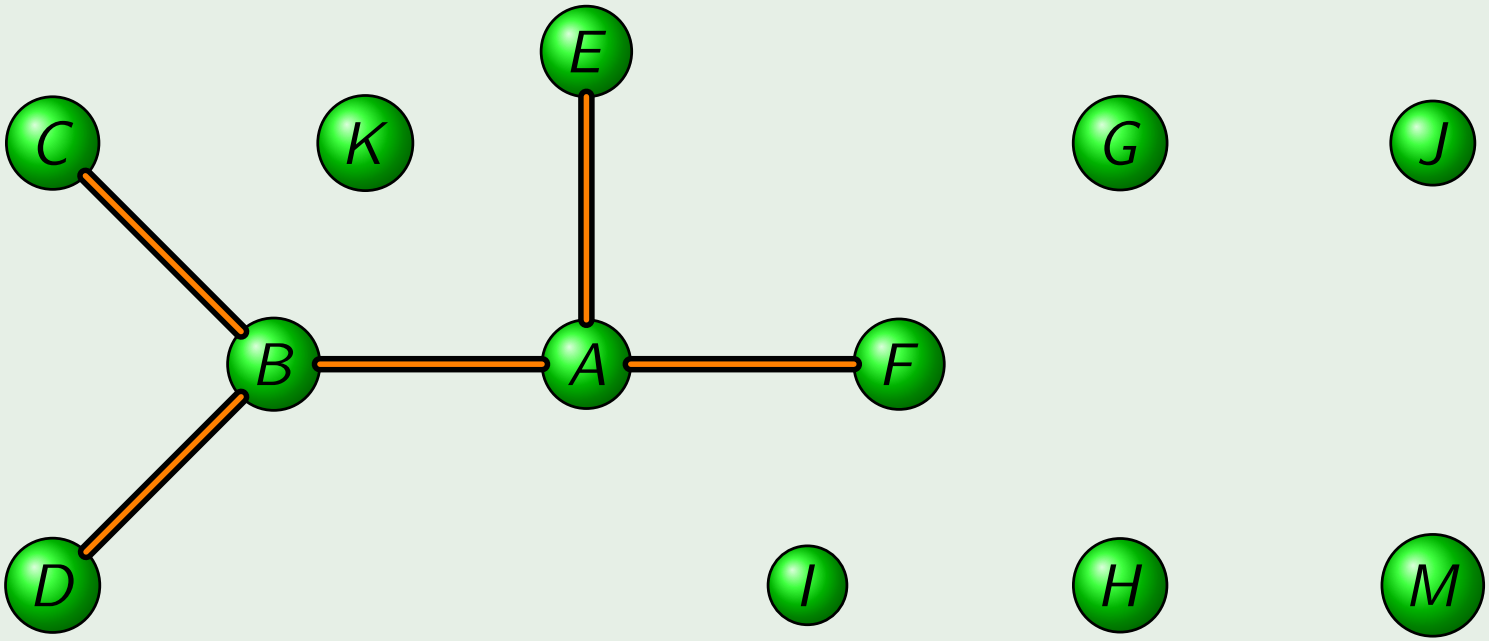
Trær

Eksempel



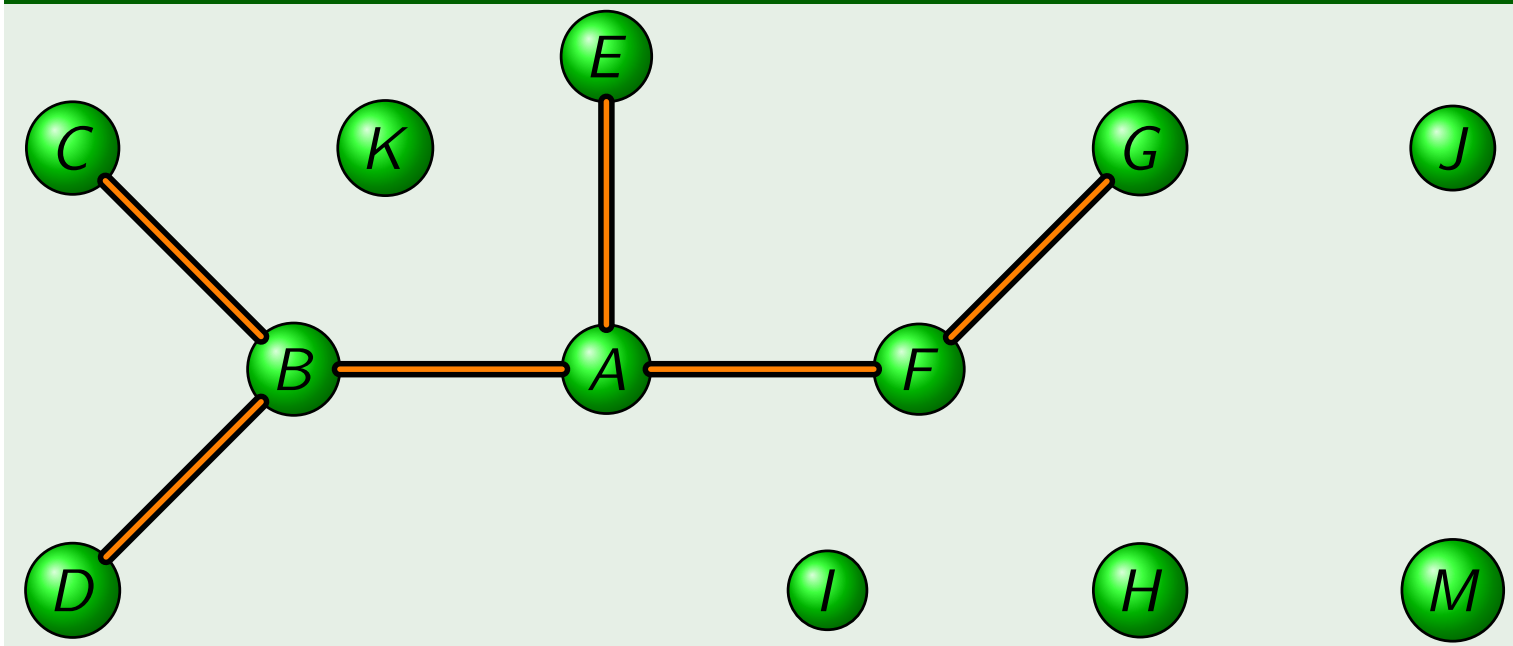
Trær

Eksempel



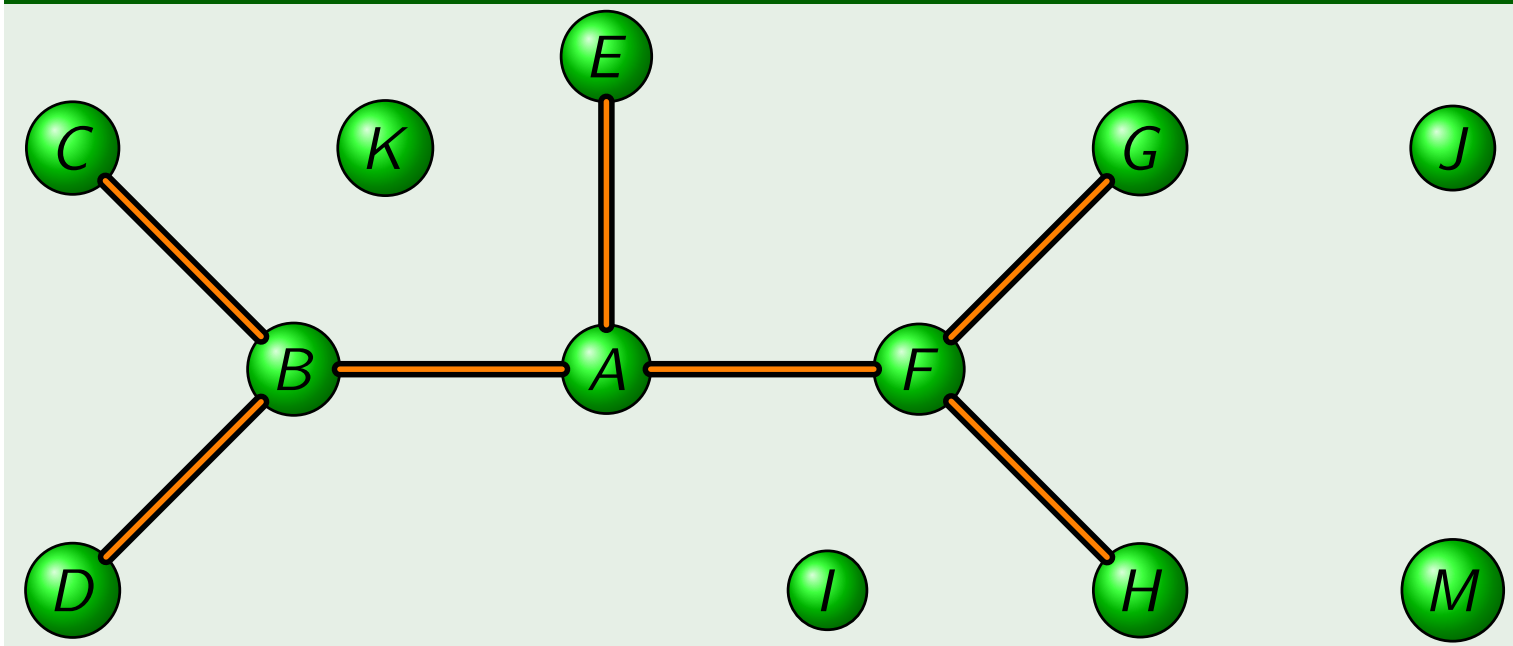
Trær

Eksempel



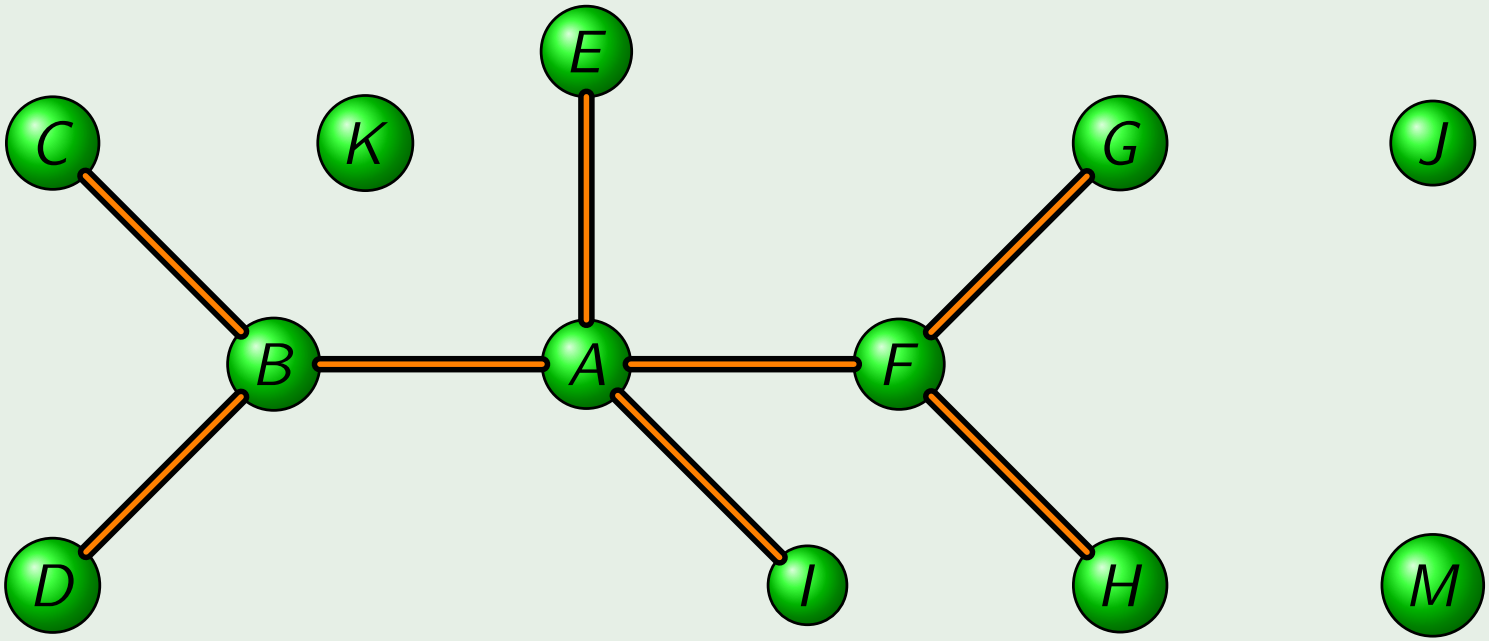
Trær

Eksempel



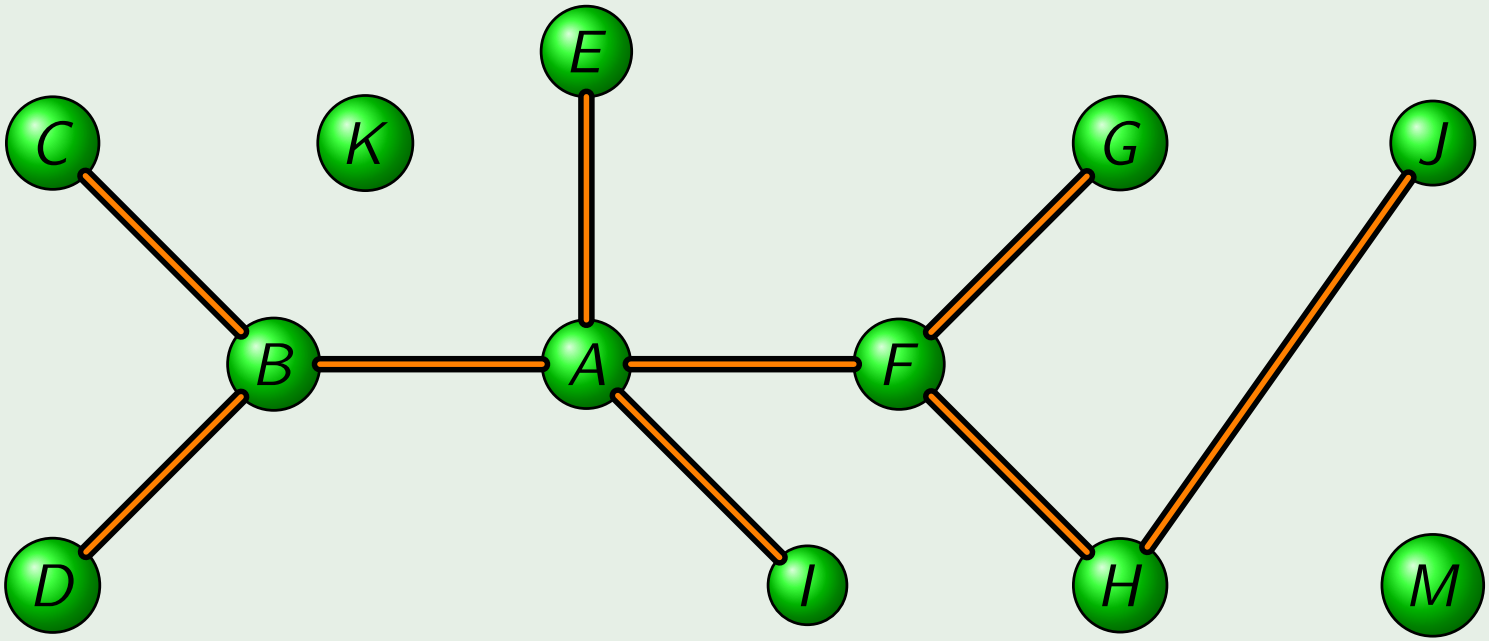
Trær

Eksempel



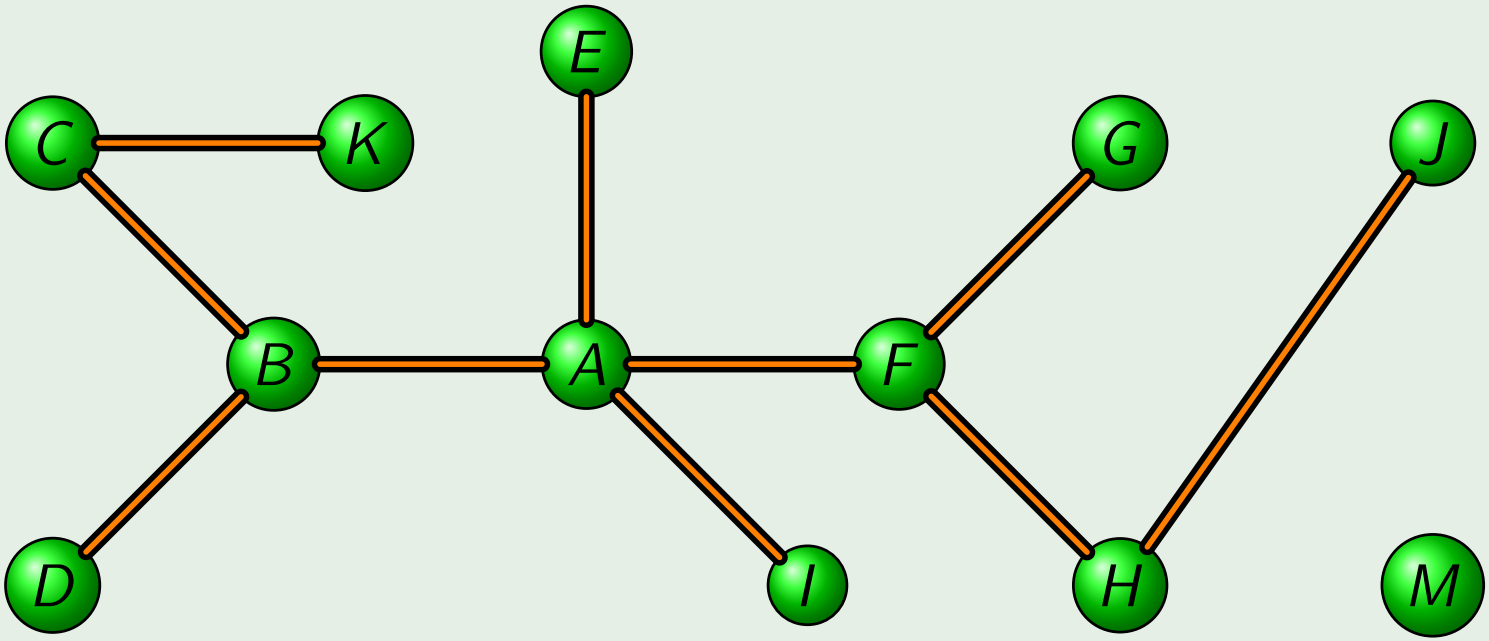
Trær

Eksempel



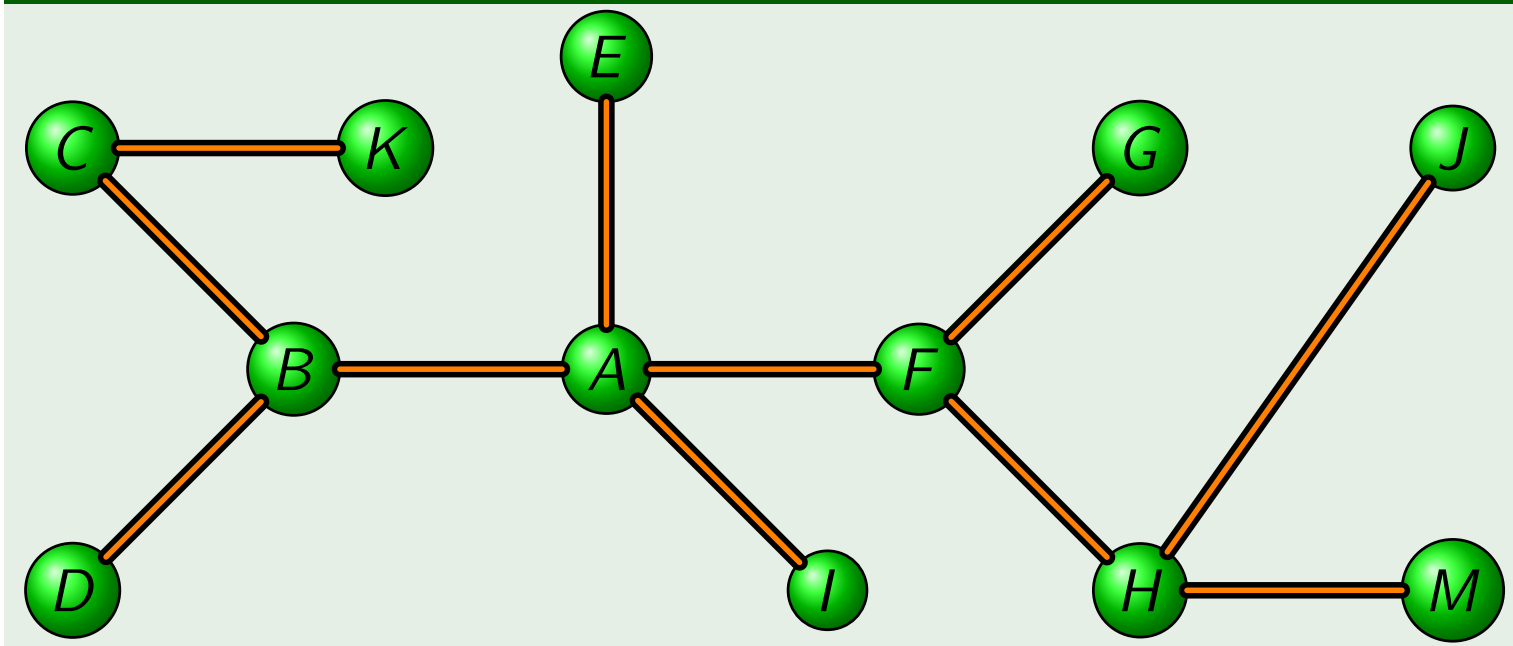
Trær

Eksempel



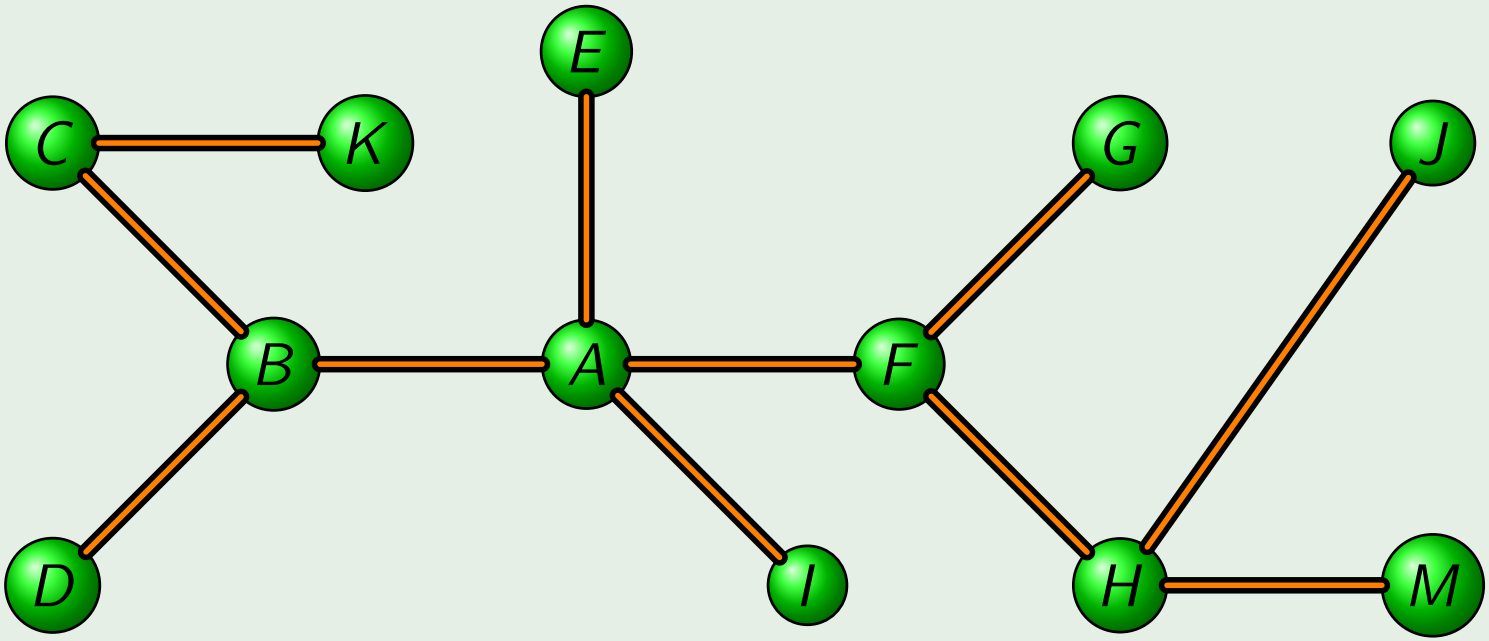
Trær

Eksempel



Trær

Eksempel



Vi ser at denne grafen har 12 noder og 11 kanter.

Trær

Eksempel

Trær

Eksempel

Et tre trenger ikke å ha noen forgreningspunkter:

Trær

Eksempel

Et tre trenger ikke å ha noen forgreningspunkter:

Trær

Eksempel

Et tre trenger ikke å ha noen forgreningspunkter:

Her har vi fem noder og fire kanter.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.
- Husk at en graf alltid har minst en node.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.
- Husk at en graf alltid har minst en node.

  Grafen med en node og ingen kanter er et tre. Alle andre trær vil ha endenoder.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.
- Husk at en graf alltid har minst en node.

  Grafen med en node og ingen kanter er et tre. Alle andre trær vil ha endenoder.
- I de eksemplene vi har sett har alle trærne en node mer enn de har kanter.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.
- Husk at en graf alltid har minst en node.

  Grafen med en node og ingen kanter er et tre. Alle andre trær vil ha endenoder.
- I de eksemplene vi har sett har alle trærne en node mer enn de har kanter.

  Dette er en egenskap som alle endelige trær har.

Trær

- I de eksemplene vi har sett på har vi alltid endenoder i et tre, det vil si noder av grad 1.
- Husk at en graf alltid har minst en node.

  Grafen med en node og ingen kanter er et tre. Alle andre trær vil ha endenoder.
- I de eksemplene vi har sett har alle trærne en node mer enn de har kanter.

  Dette er en egenskap som alle endelige trær har.

  Det er ingenting i definisjonen av grafer og trær som sier at de skal være endelige, men vi kommer til å begrense oss til endelige grafer og trær hvis vi ikke sier noe annet. Boka forutsetter også at vi bare arbeider med endelige grafer og trær i dette kurset

Trær

Teorem

Trær

Teorem

a) Hvis et tre har minst en kant, har treet en node med grad 1 (En slik node kaller vi en endenode eller bladnode).

Trær

Teorem

a) Hvis et tre har minst en kant, har treet en node med grad 1 (En slik node kaller vi en endenode eller bladnode).

b) I ethvert tre finnes det nøyaktig en node mer enn det finnes kanter.

Trær

Bevis

Trær

Bevis

a) La

$$v_0 e_1 \cdots e_n v_n$$

være en sti med maksimal lengde.

Trær

Bevis

a) La

$$v_0 e_1 \cdots e_n v_n$$

være en sti med maksimal lengde.

Siden grafen er et tre, kan ikke stien være innom samme node to ganger.

Trær

Bevis

a) La

$$v_0 e_1 \cdots e_n v_n$$

være en sti med maksimal lengde.

Siden grafen er et tre, kan ikke stien være innom samme node to ganger.

Endenodene v_0 og v_n må være bladnoder, siden vi ellers ville kunnet gjøre stien lengere.

Bevis (Fortsatt)

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Hvis det bare finnes en node, har vi ingen kanter, og påstanden stemmer.

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Hvis det bare finnes en node, har vi ingen kanter, og påstanden stemmer.

Hvis det finnes mer enn en node, kan vi anta at påstanden holder for alle mindre trær.

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Hvis det bare finnes en node, har vi ingen kanter, og påstanden stemmer.

Hvis det finnes mer enn en node, kan vi anta at påstanden holder for alle mindre trær.

Tar vi bort en bladnode og den ene kanten som ligger inntil denne noden, får vi et mindre tre.

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Hvis det bare finnes en node, har vi ingen kanter, og påstanden stemmer.

Hvis det finnes mer enn en node, kan vi anta at påstanden holder for alle mindre trær.

Tar vi bort en bladnode og den ene kanten som ligger inntil denne noden, får vi et mindre tre.

Siden vi har tatt bort en node og en kant, og da har en node mer enn vi har kanter, må dette være tilfellet i det gitte treet også.

Trær

Bevis (Fortsatt)

b) Vi bruker induksjon på antall noder i treet.

Hvis det bare finnes en node, har vi ingen kanter, og påstanden stemmer.

Hvis det finnes mer enn en node, kan vi anta at påstanden holder for alle mindre trær.

Tar vi bort en bladnode og den ene kanten som ligger inntil denne noden, får vi et mindre tre.

Siden vi har tatt bort en node og en kant, og da har en node mer enn vi har kanter, må dette være tilfellet i det gitte treet også.

Resonementet illustreres på tavla.

Vektede grafer

- Hvis en graf representerer et veinett er det av interesse å vite hvor lange de enkelte veistrekningene er.

Vektede grafer

- Hvis en graf representerer et veinett er det av interesse å vite hvor lange de enkelte veistrekningene er.
- Hvis en graf representerer et ledningsnett, kan anleggskostnader og driftskostnader ved de enkelte strekningene være av interesse.

Vektede grafer

- Hvis en graf representerer et veinett er det av interesse å vite hvor lange de enkelte veistrekningene er.
- Hvis en graf representerer et ledningsnett, kan anleggskostnader og driftskostnader ved de enkelte strekningene være av interesse.
- Hvis nodene i en graf står for land og kantene for grenseoverganger mellom dem, kan tollsatsene eller korrupsjonskoeffisienten ved de forskjellige grenseovergangene bety noe.

Vektede grafer

- Hvis en graf representerer et veinett er det av interesse å vite hvor lange de enkelte veistrekningene er.
- Hvis en graf representerer et ledningsnett, kan anleggskostnader og driftskostnader ved de enkelte strekningene være av interesse.
- Hvis nodene i en graf står for land og kantene for grenseoverganger mellom dem, kan tollsatsene eller korrupsjonskoeffisienten ved de forskjellige grenseovergangene bety noe.
- Siden vi har mange eksempler på grafer hvor det er viktige tallstørrelser knyttet til de enkelte kantene, studerer vi vektede grafer som et eget begrep.

Trær

Definisjon

Trær

Definisjon

En vektet graf er en graf hvor hver kant har fått en vekt, et positivt reelt tall.

Trær

Definisjon

En vektet graf er en graf hvor hver kant har fått en vekt, et positivt reelt tall.

Merk

Trær

Definisjon

En vektet graf er en graf hvor hver kant har fått en vekt, et positivt reelt tall.

Merk

Formelt sett kan vi definere en vektet graf som et par (G, f) hvor G er en graf og f er en funksjon fra mengden av kanter i G til de positive reelle tallene.

Trær

Definisjon

En vektet graf er en graf hvor hver kant har fått en vekt, et positivt reelt tall.

Merk

Formelt sett kan vi definere en vektet graf som et par (G, f) hvor G er en graf og f er en funksjon fra mengden av kanter i G til de positive reelle tallene.

Vi har altså bruk både for ordnede par og for funksjoner for å gi en skikkelig definisjon.

Vektede grafer

Eksempel

Vektede grafer

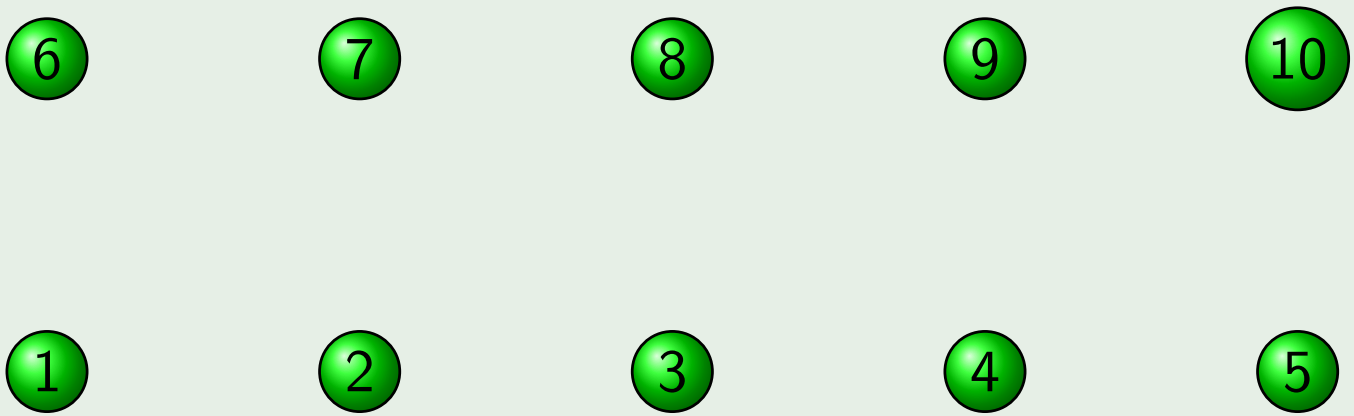
Eksempel

La oss se på et eksempel:

Vektede grafer

Eksempel

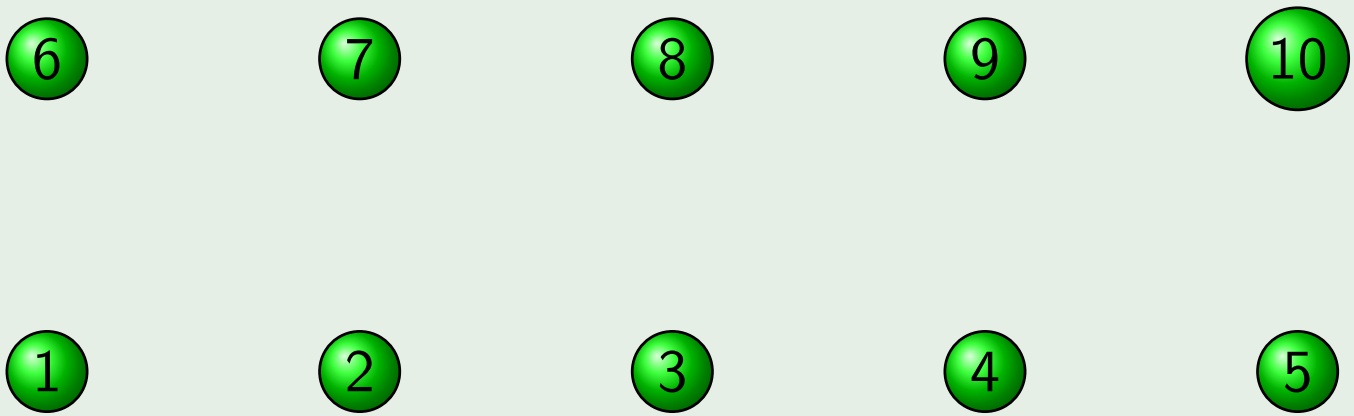
La oss se på et eksempel:



Vektede grafer

Eksempel

La oss se på et eksempel:

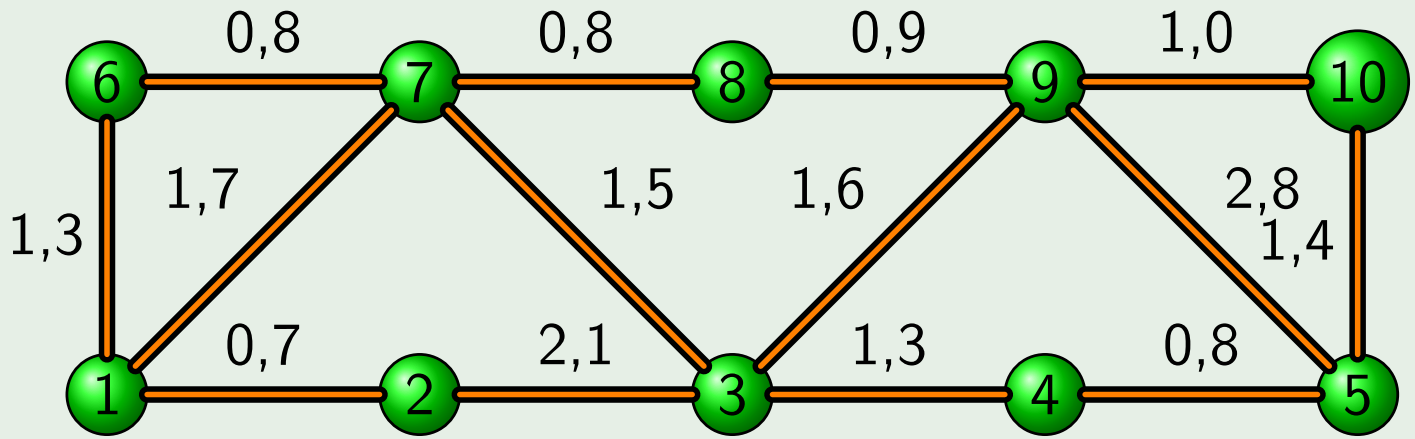


- Det er mulig å trekke kabler mellom disse skjematisk tegnede byene, hvor kostnaden målt i antall NOK 10^7:

Vektede grafer

Eksempel

La oss se på et eksempel:

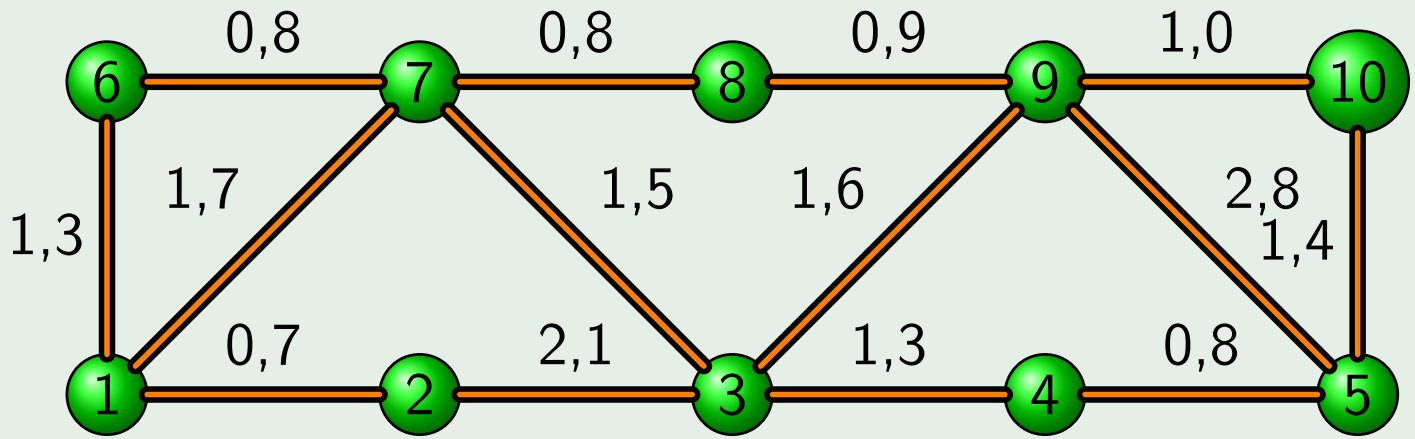


- Det er mulig å trekke kabler mellom disse skjematisk tegnede byene, hvor kostnaden målt i antall NOK 10^7:

Vektede grafer

Eksempel

La oss se på et eksempel:



- Det er mulig å trekke kabler mellom disse skjematisk tegnede byene, hvor kostnaden målt i antall NOK 10^7:

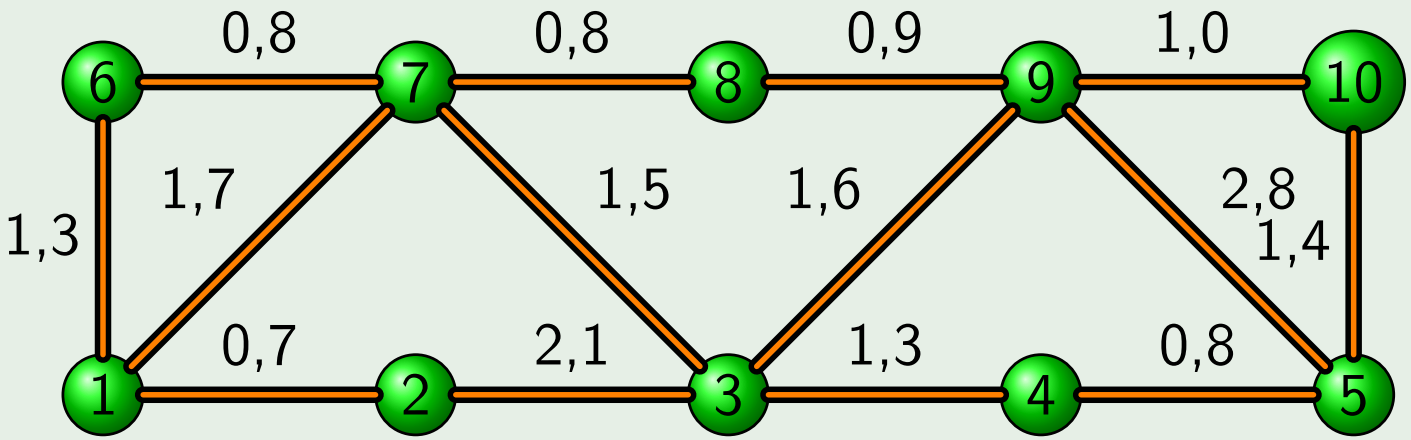
Kan vi fjerne noen av kantene slik at anleggskostnadene blir minst mulig, men vi fortsatt forbinder alle byer med kabler?

Vektede grafer

Eksempel (Fortsatt)

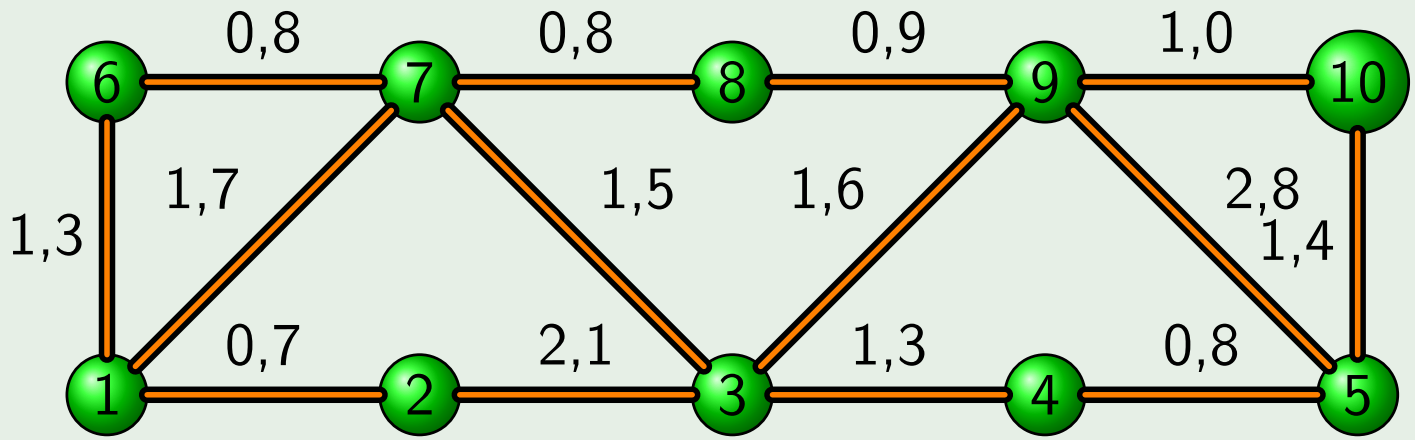
Vektede grafer

Eksempel (Fortsatt)



Vektede grafer

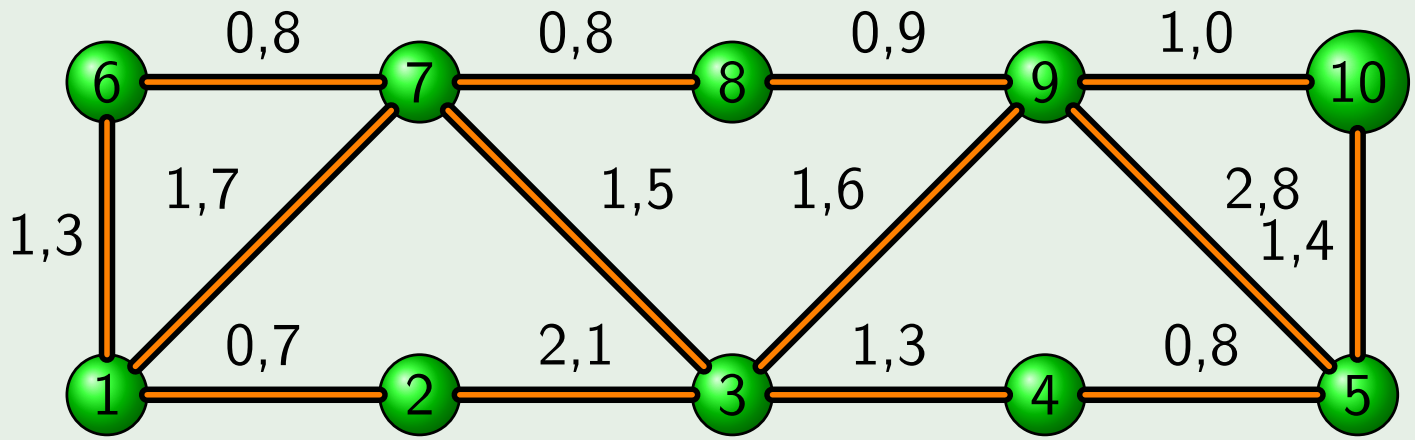
Eksempel (Fortsatt)



Så lenge grafen inneholder kretser, må det være greit å ta bort en kant i kretsen.

Vektede grafer

Eksempel (Fortsatt)

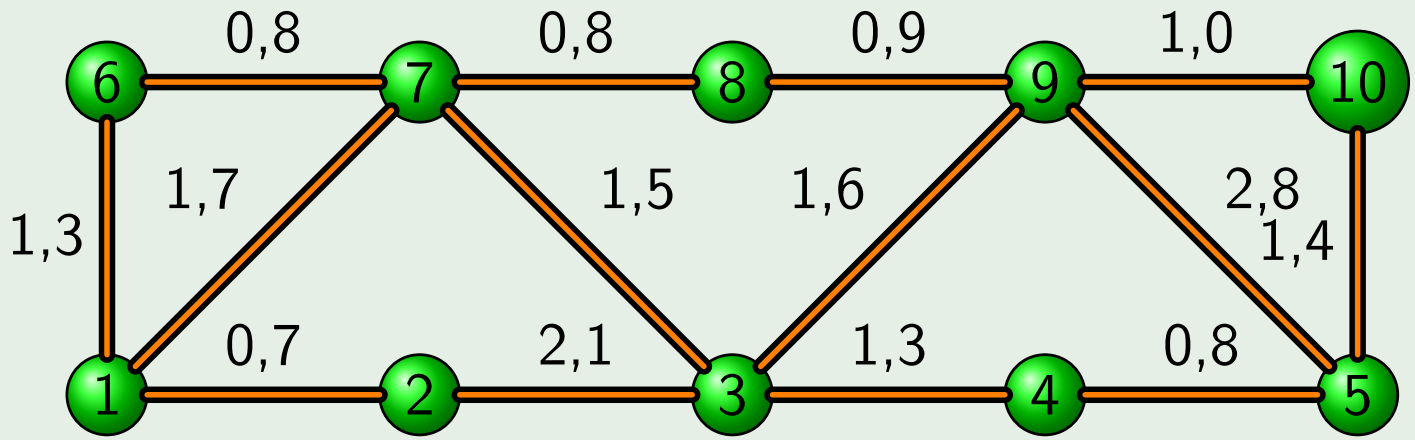


Så lenge grafen inneholder kretser, må det være greit å ta bort en kant i kretsen.

Vi bør derfor finne det mest kostnadseffektive deltreet som når over alle nodene.

Vektede grafer

Eksempel (Fortsatt)



Så lenge grafen inneholder kretser, må det være greit å ta bort en kant i kretsen.

Vi bør derfor finne det mest kostnadseffektive deltreet som når over alle nodene.

Vi skal komme tilbake til dette eksemplet når vi har diskutert algoritmen som ligger bak.

Vektede grafer

Definisjon

Vektede grafer

Definisjon

La G være en sammenhengende graf, og la T være et deltre av G. Det betyr her at T og G har de samme nodene, alle kantene i T er kanter i G, men noen kanter i G kan mangle i T.

Vektede grafer

Definisjon

La G være en sammenhengende graf, og la T være et deltre av G. Det betyr her at T og G har de samme nodene, alle kantene i T er kanter i G, men noen kanter i G kan mangle i T.

Vi sier at T spenner ut G hvis alle nodene i G ligger inntil en kant i T. (Tegning på tavla).

Vektede grafer

Definisjon

La G være en sammenhengende graf, og la T være et deltre av G. Det betyr her at T og G har de samme nodene, alle kantene i T er kanter i G, men noen kanter i G kan mangle i T.

Vi sier at T spenner ut G hvis alle nodene i G ligger inntil en kant i T. (Tegning på tavla).

Husk at et tre er en sammenhengende graf, så dette betyr at alle par av forskjellige noder i G kan forbindes med en (og bare en) sti i T.

Vektede grafer

Definisjon

La G være en sammenhengende graf, og la T være et deltre av G. Det betyr her at T og G har de samme nodene, alle kantene i T er kanter i G, men noen kanter i G kan mangle i T.

Vi sier at T spenner ut G hvis alle nodene i G ligger inntil en kant i T. (Tegning på tavla).

Husk at et tre er en sammenhengende graf, så dette betyr at alle par av forskjellige noder i G kan forbindes med en (og bare en) sti i T.

- Hvis G er et vektet tre, er problemet å finne et tre T som spenner ut G slik at summen av vektene på kantene i T blir minst mulig.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.
- I læreboka står det en pseudokode for Prims algoritme.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.
- I læreboka står det en pseudokode for Prims algoritme.
- Her vil vi beskrive algoritmen litt mer uformelt.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.
- I læreboka står det en pseudokode for Prims algoritme.
- Her vil vi beskrive algoritmen litt mer uformelt.
- Det viser seg at hvis man bygger opp et tre ved i hvert skritt å gjøre det som i øyeblikket virker mest fornuftig, så kommer man frem.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.
- I læreboka står det en pseudokode for Prims algoritme.
- Her vil vi beskrive algoritmen litt mer uformelt.
- Det viser seg at hvis man bygger opp et tre ved i hvert skritt å gjøre det som i øyeblikket virker mest fornuftig, så kommer man frem.
- Korrekthetsbeviset for Prims algoritme skal vi ikke legge vekt på, men det forventes at man kan praktisere den på eksempler.

Vektede grafer

- Prims algoritme gir en metode for å finne det minimale utspennende treet til en vektet graf.
- I læreboka står det en pseudokode for Prims algoritme.
- Her vil vi beskrive algoritmen litt mer uformelt.
- Det viser seg at hvis man bygger opp et tre ved i hvert skritt å gjøre det som i øyeblikket virker mest fornuftig, så kommer man frem.
- Korrekthetsbeviset for Prims algoritme skal vi ikke legge vekt på, men det forventes at man kan praktisere den på eksempler.
- Vi beskriver Prims algoritme litt anderledes enn den er formulert i læreboka, men effekten er den samme, vi får det samme treet bygget opp i den samme rekkefølgen.