Ukeoppgaver til INF 4130, 15. november 2011

Pass på å få tid til Oppgave 5, f.eks. ved å hoppe over Oppgave 3 og noe av Oppgave 4 i første omgang.

Oppgave 1
Løs oppgave 14.4 i boka (og grovplanlegg datastrukturen til Oppgave 14.5)

Oppgave 2
Løs oppgave 14.6 i boka

Oppgave 3
Anta |X| = |Y|. Vis så at om vi har funnet et uplukk S av X som er slik at |Gamma(S)| < |S|, så kan vi også lett finne et utplukk T av Y slik at |Gamma(T)| < |T|.

Oppgave 4
En liten utvidelse av den ungarske algoritmen er som følger (og her kan vi godt tenke oss at størrelsen av X og Y er forskjellige):

I stedet for bare å ”gro” ett tre fra en tilfeldig umatchet node i X (og gi helt opp om dette ikke fører fram) så gror vi trær fra *alle* umatchede noder i X, og gir bare opp om ingen av disse gir noen forbedringsvei. (NB: Man lar da allerede produserte trær fra umatchede noder i X stå, og går ikke inn i dem når flere trær bygges). Påstanden er at denne algoritmen vil finne den *størst mulige matching* (flest kanter) som kan lages i denne grafen (og da selvfølgelig også en perfekt en, om en slik finnes).

I programmet på side 422-423 tilsvarer dette grovt sett at man legger innholdet av else-grenen som starter nederst på side 422 inn i en løkke som gjør at variablen *r* gjennomgår alle umatchede noder i X, men stopper om en forbedringsvei blir funnet. Derfor må man fikse litt på avslutningene etc. Oppgaven er så å vise at om denne utvidede algoritmen stopper før den har en perfekt matching (eller en av størrelse *min*(|X|,|Y|)), så har den funnet en matching som har så mange kanter som mulig.

Nødvendig hint: Det er lett å vise at om man finner et utplukk av noder som "dekker" alle kanter (Fra NP-teorien: *Kantdekkende nodeutplukk*), så kan ingen matching være større (ha flere kanter) enn dette nodeutplukket. Når den utvidede algoritmen over stopper fordi vi ikke finner noen forbedringsvei, kan vi derfor forsøke å finne et slikt nodeutplukk som er like stort som den matchingen vi har (og som vi ikke klarte å utvide). I så fall vil vi være i mål. Studer derfor en slik sluttsituasjon, som vil bestå av flere trær av samme type som i figur 14.2, samt muligens et antall matchede kanter som ikke er i noe tre, og noen umatchede noder fra Y. Forsøk så å ut fra dette å angi et nodeutplukk som dekker alle kanter, og som er like stort som den matchingen man har.

Ekstaoppgave 4.1: I oppgave 14.6, fjern kantene x2-y3 og x3-y2, og vis at den utvidede ungarske algoritmen da stopper før den har en perfekt matching. Ut fra situasjonen når den stopper, angi et kantdekkende nodeutplukk som er like stort som matchingen, slik som vist over.

Ekstraoppgave 4.2: Det å finne det minste overdekkende nodeutplukket i en generell graf er, som vi har sett på i NP-teorien, NP-hardt. Vi så også på forelesningen at det å finne den maksimale matchingen i en generell graf kan løses i polynomisk tid (ved hjelp av snurping av odde løkker etc.).

Vi viste over at for bipartite grafer er den maksimale matchingen (altså et kant-utplukk) og den minimale overdekkende nodemengden (et nodeutplukk) like store. Forklar, ut fra betraktningene over (og antakelsen $P \neq NP$), hvorfor dette ikke kan gjelde for generelle grafer, og vis det også med et eksempel.

Oppgave 5
For å studere maksimal-flyt-algoritmen, følg eksempelet i Figur 14.9 i detalj. Se intro nederst side 439. NB: Det kryr av trykkfeil i *den gamle utgaven* av boka, men de fleste (alle?) er rettet i den nye (konsulter trykkfeillista).

Oppgave 6. Flyt i nettverk og matchinger i bipartite grafer (Mye repetisjon fra forelesningen)
Se på figur 14.10 på side 444 i boka. (Merk at det er en trykkfeil hvertfall i noen utgaver av boka: Kanten (x1,y2) i øverste graf skal fjernes). Vi skal se på sammenhengen mellom det å finne en maksimal matching i den øverste grafen, og det å finne en maksimal flyt i det nederste nettverket.

Spørsmål 6.1: Se først på følgende lemma, og forklar hvorfor det er riktig (Hint: Studér algoritmen):
Lemma: *I et nettverk med heltallige kapasiteter kan man alltid finne en flyt som både er maksimal og heltallig, og Ford/Fulkerson-algoritmen vil alltid finne en slik.*
Altså: Dersom kapasitetene er heltallige behøver man aldri dele opp en flyt slik at det f.eks. går flyt = ½ i en kant og ½ i en annen for å kunne oppnå maksimal flyt. Det betyr at dersom alle kapasitetene er 1 så kan man oppnå maksimal flyt ved å bare ha enten full flyt eller ingen flyt i hver kant. Dermed kan en slik flyt tolkes som et utplukk av kanter, nemlig de der det går full flyt.

Spørsmål 6.2: Forklar ut fra lemmaet over at det å finne en maksimal matching i den øverste grafen i figur 14.10 dermed er det samme som å finne en maksimal flyt i det nederste nettverket (med alle kapasiteter lik 1).

Spørsmål 6.3: Anta at du har matchingen (x2, y1), (x4, y3) og (x5, y5), og angi hvilken flyt *f* dette vil svare til i nederste figur.

Spørsmål 6.4: Tegn så N*(f)* (flytforberdrings-grafen = det *f*-avledede nettverket) for flyten *f* fra 7.3, og sjekk at det å lete etter en (flyt)forbedringsvei i denne fra s til t helt tilsvarer det å lete etter en (matching)forbedringsvei i øverste grafen, med den gitte matchingen.

Spørsmål 6.5: Bruk en forbedringsvei som blir funnet (f.eks. (x1, y1, x2, y4) i grafen og (s, x1, y1, x2, y4, t) i nettverket) til å øke matchingen/flyten, og se at dette blir helt tilsvarende operasjoner, og at du kommer til situasjonen angitt i flytnettverket nederst i fig 14.10 (der det er flytene som er angitt).

Spørsmål 6.6: Tegn så opp N*(f)* for denne nye flyten, og vis at flyten er maksimal ved å angi et kutt som har denne kapasiteten (= 4). Bruk så metoden fra Oppgave 4 (over) til å finne et nodeutplukk på fire noder som dekker alle kanter i den øverste grafen, og som dermed viser at matchingen er maksimal. Vis så at dette kuttet og dette nodeutplukket helt tilsvarer hverandre.