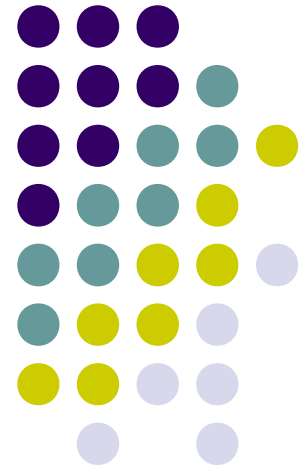


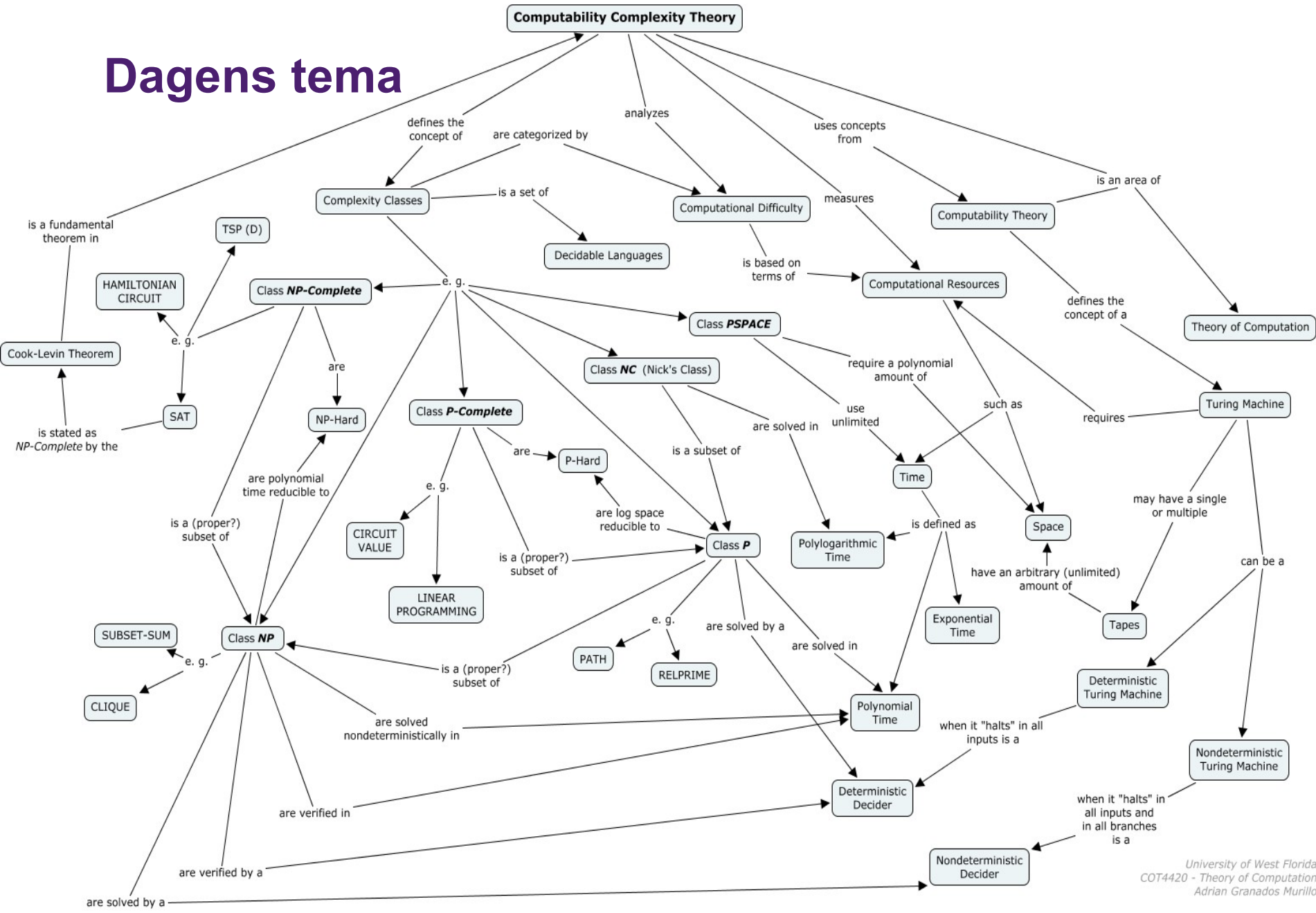
Øvingsforelesning 11

P vs NP

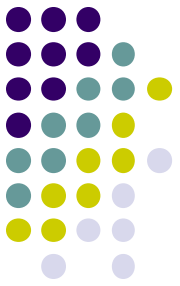
Håkon Jacobsen
hakoja@stud.ntnu.no



Dagens tema

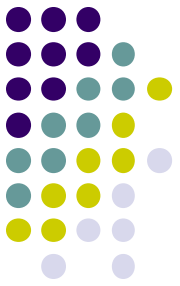


Dagens tema



- Neste ukes øvinger
- Introduksjon og motivasjon for problemet
- Polynomreduksjoner
- Kompleksitetsklasser
 - P, NP og NPC
- Eksamenseksemppler
- Denne ukens øvinger

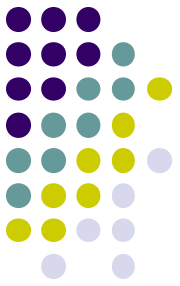
Introduksjon



Dagens datamaskiner er veldig raske...

Men enkelte problemer tar allikevel
alt for lang tid til at vi kan løse dem

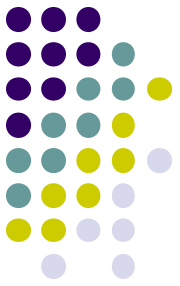
Eksempel



$$7 \times 13 = ?$$

"Multiplikasjonsproblemet"
(Løsning: 91)

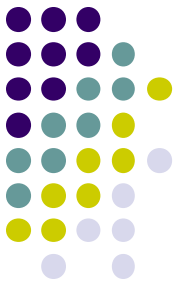
Eksempel



$$? \times ? = 91$$

"Faktoriseringsproblemet"
(Løsning: 7 x 13)

Et litt større multiplikasjonseksempel



$$\begin{array}{r} 33987174230284385545 \\ 30123627613875835633 \\ 98649596959742349092 \\ 9302771479 \end{array} \times \begin{array}{r} 6264200187401285096 \\ 1516549482644422193 \\ 0203717862350901911 \\ 1660653946049 \end{array} = ?$$

Løsning:

21290246318258757547497882016271517497806703963277
21627823338321538194998405649591136657385302191831
6783107387995317230889569230873441936471

Tok under ett sekund å utføre på WolframAlpha

Et litt større faktoriseringsseksempel

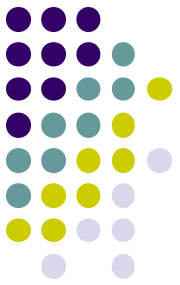


$$\begin{array}{r} ? \times ? = \\ 212902463182587575474978820162715 \\ 174978067039632772162782333832153 \\ 819499840564959113665738530219183 \\ 167831073879953172308895692308734 \\ 41936471 \end{array}$$

$$\begin{array}{r} 339871742302843855 \\ 453012362761387583 \\ 563398649596959742 \\ 3490929302771479 \end{array} \times \begin{array}{r} 626420018740128509 \\ 615165494826444221 \\ 930203717862350901 \\ 9111660653946049 \end{array}$$

Brukte 2000 MIPS-år i 1999

RSA-212

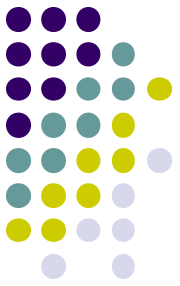


74037563479561712828046796097429573142593188889231289084936
23263897276503402826627689199641962511784399589433050212758
53701189680982867331732731089309005525051168770632990723963
80786710086096962537934650563796359

\$30 000

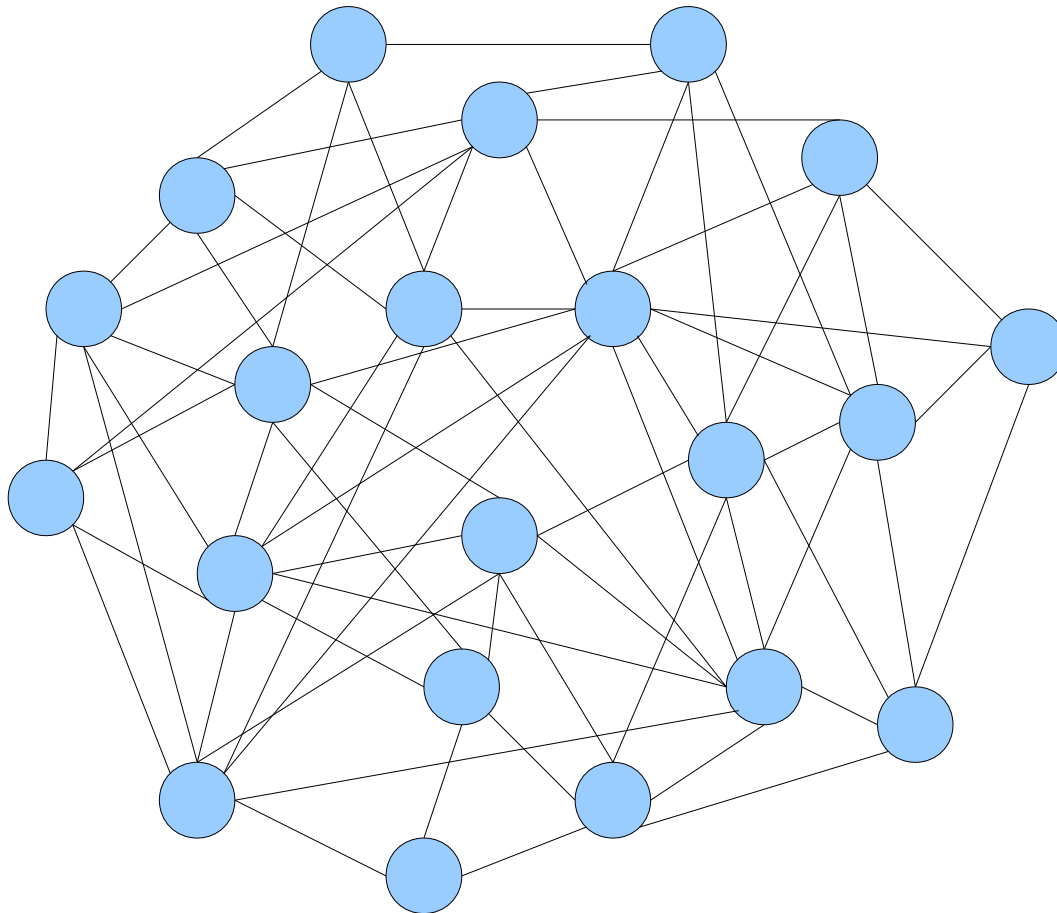
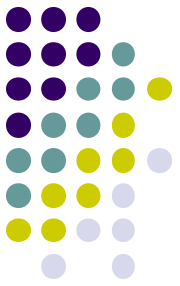
Ingen har foreløpig klart dette og fortsatt
gjenstår: RSA-220, RSA-230, ..., RSA-617

Hvorfor er faktorisering så vanskelig?

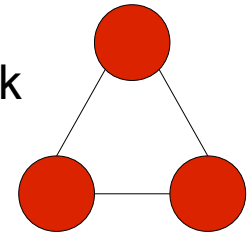


- Del med 2, 3, 5, 7, 11, ... helt til du finner en faktor.
- Brute-force-søking er tregt når mulighetsrommet er stort.
- Men er søking nødvendig?

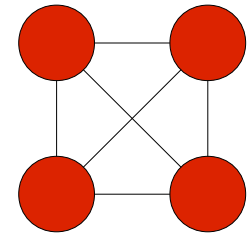
Clique-problemet



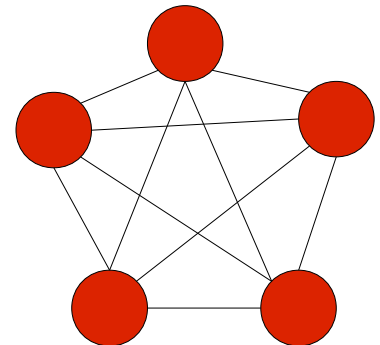
3-klikk



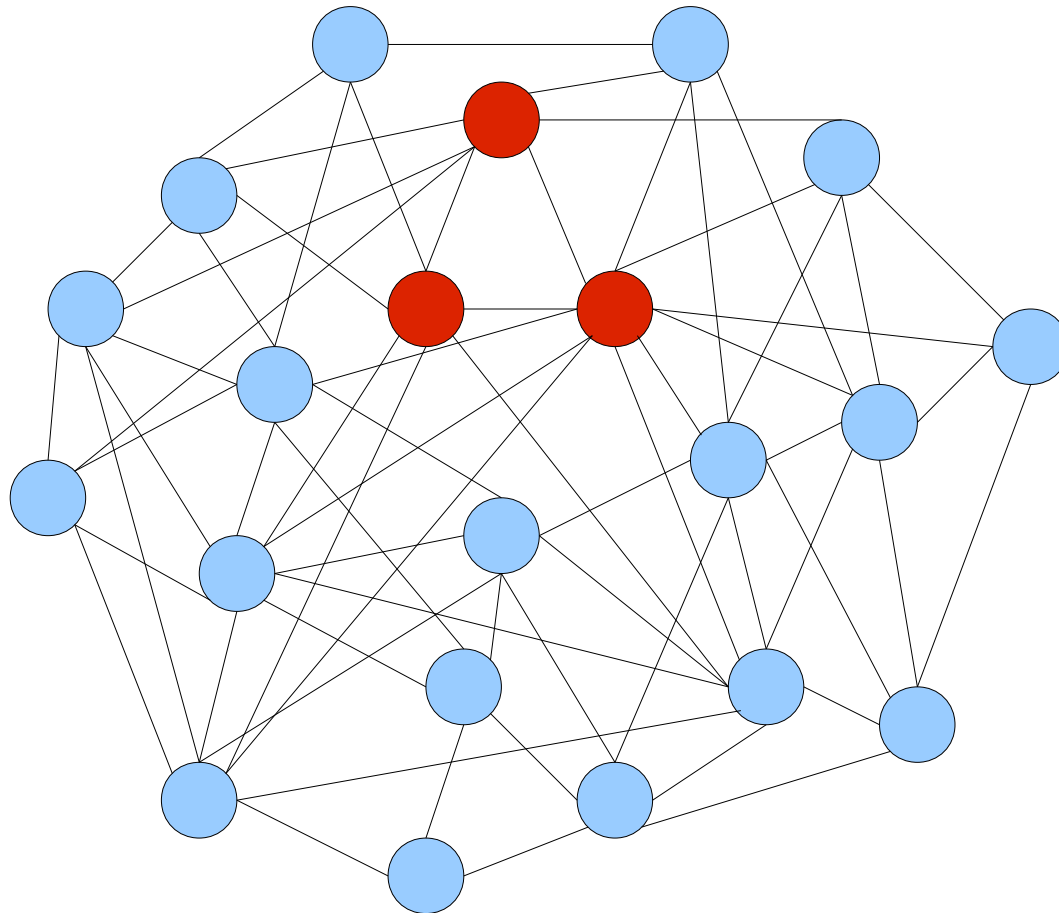
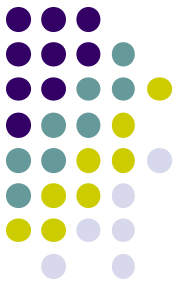
4-klikk



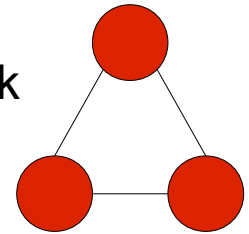
5-klikk



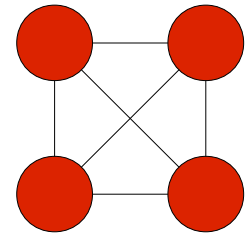
Clique-problemet



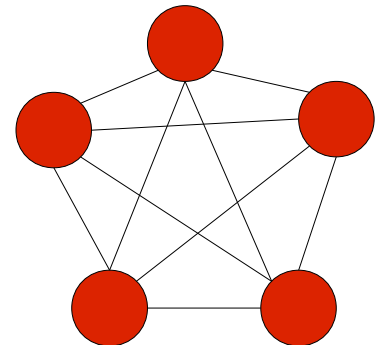
3-klikk



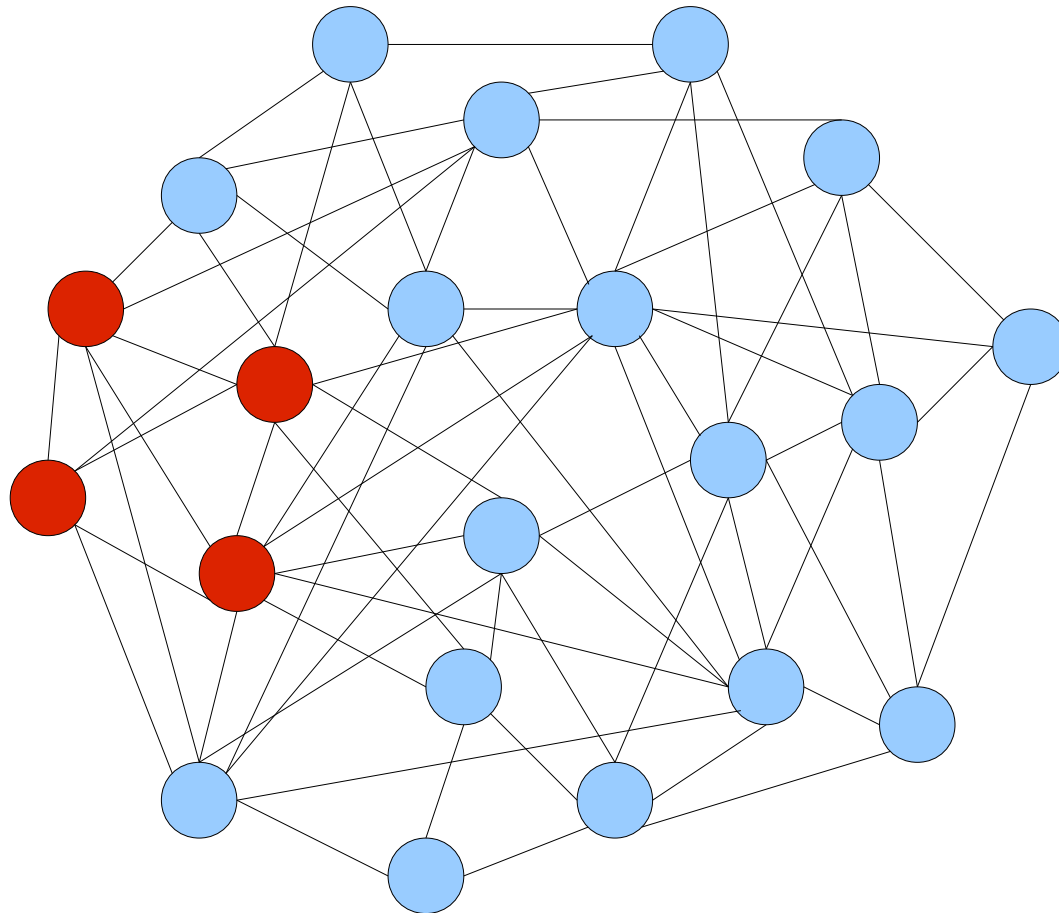
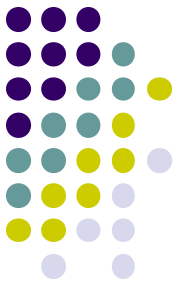
4-klikk



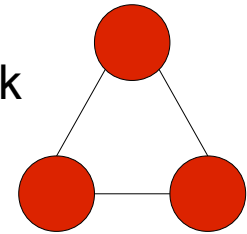
5-klikk



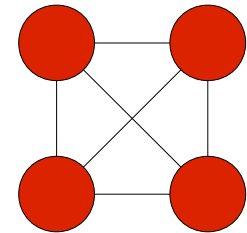
Clique-problemet



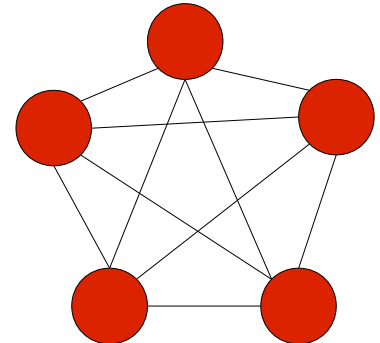
3-klikk



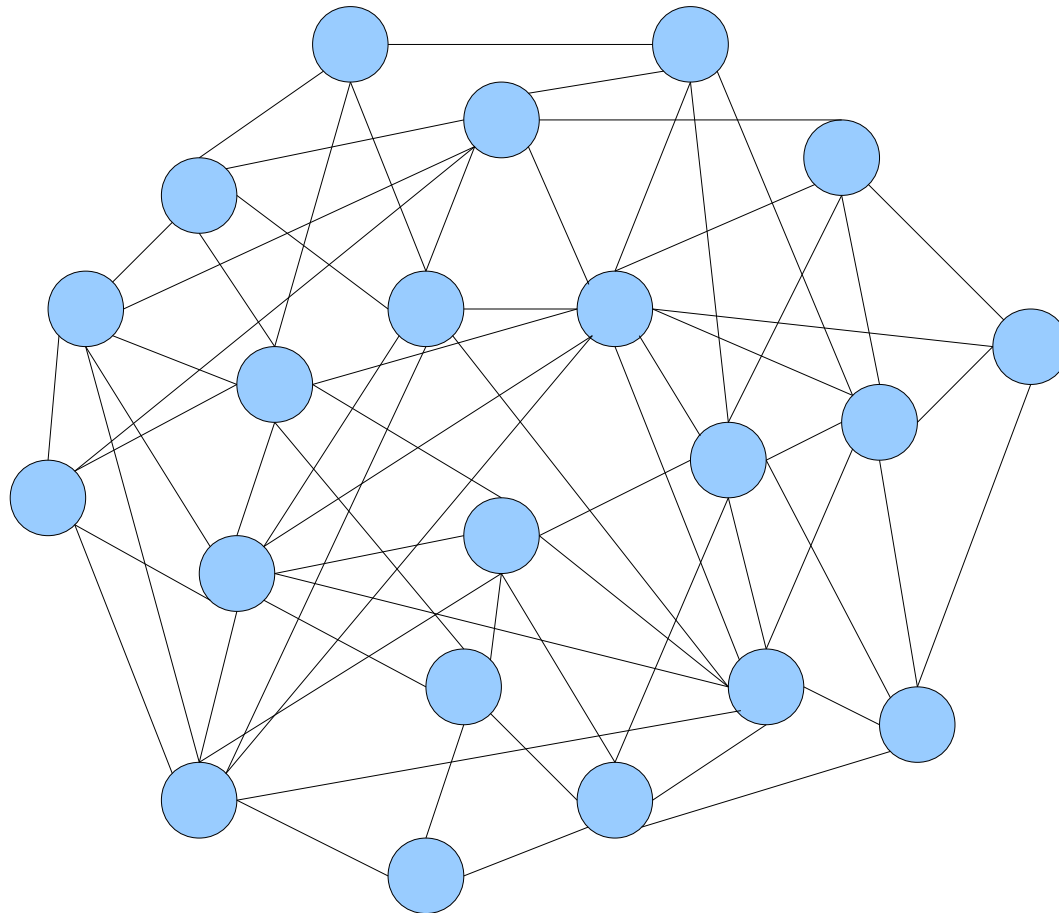
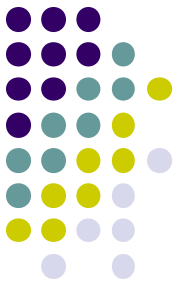
4-klikk



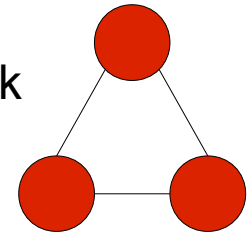
5-klikk



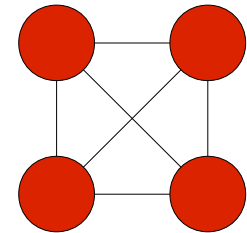
Clique-problemet



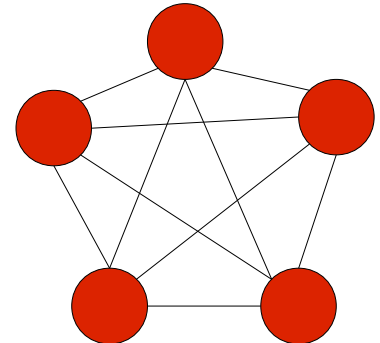
3-klikk



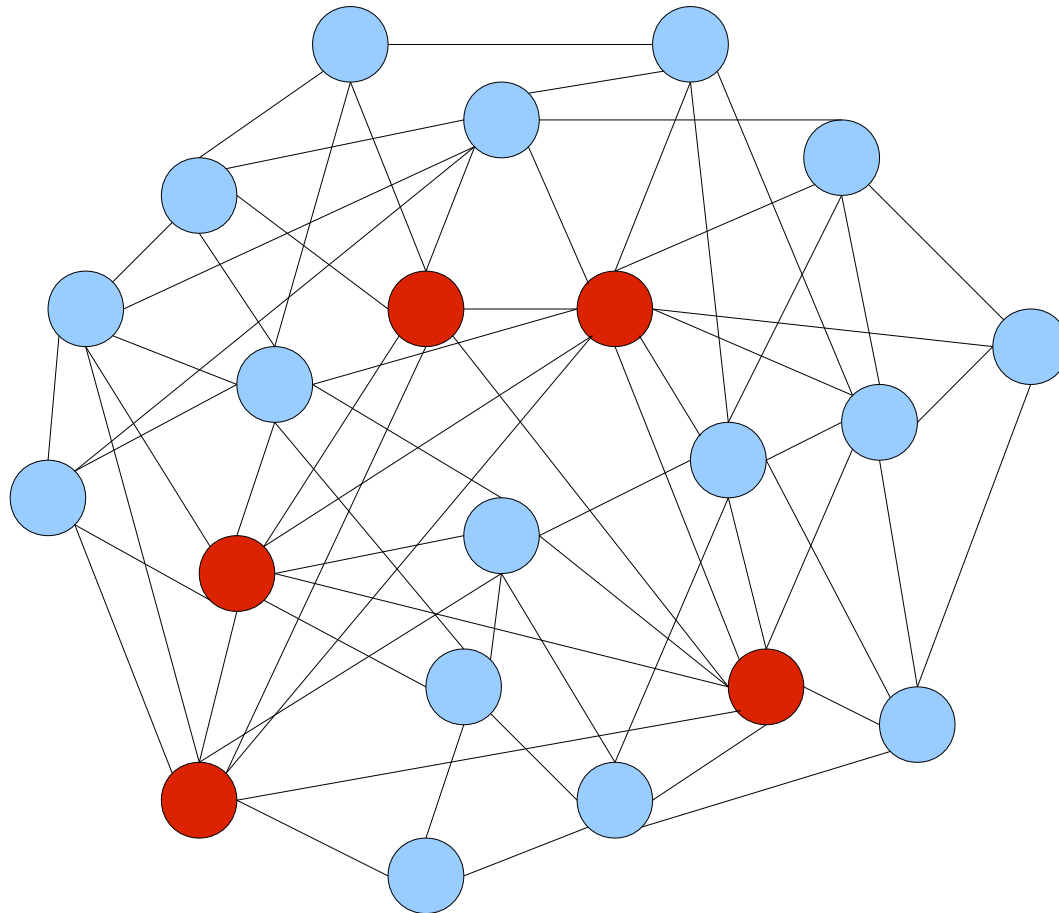
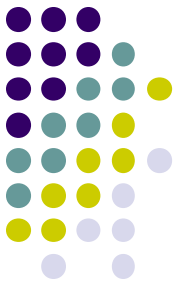
4-klikk



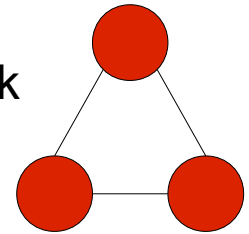
5-klikk



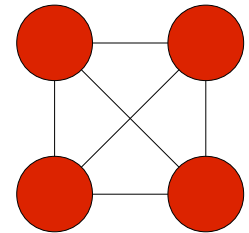
Clique-problemet



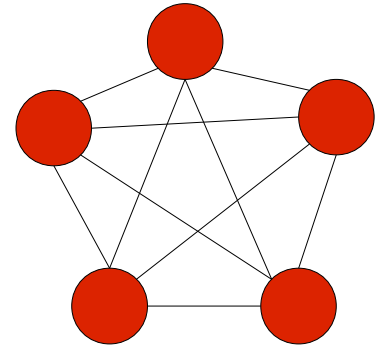
3-klikk



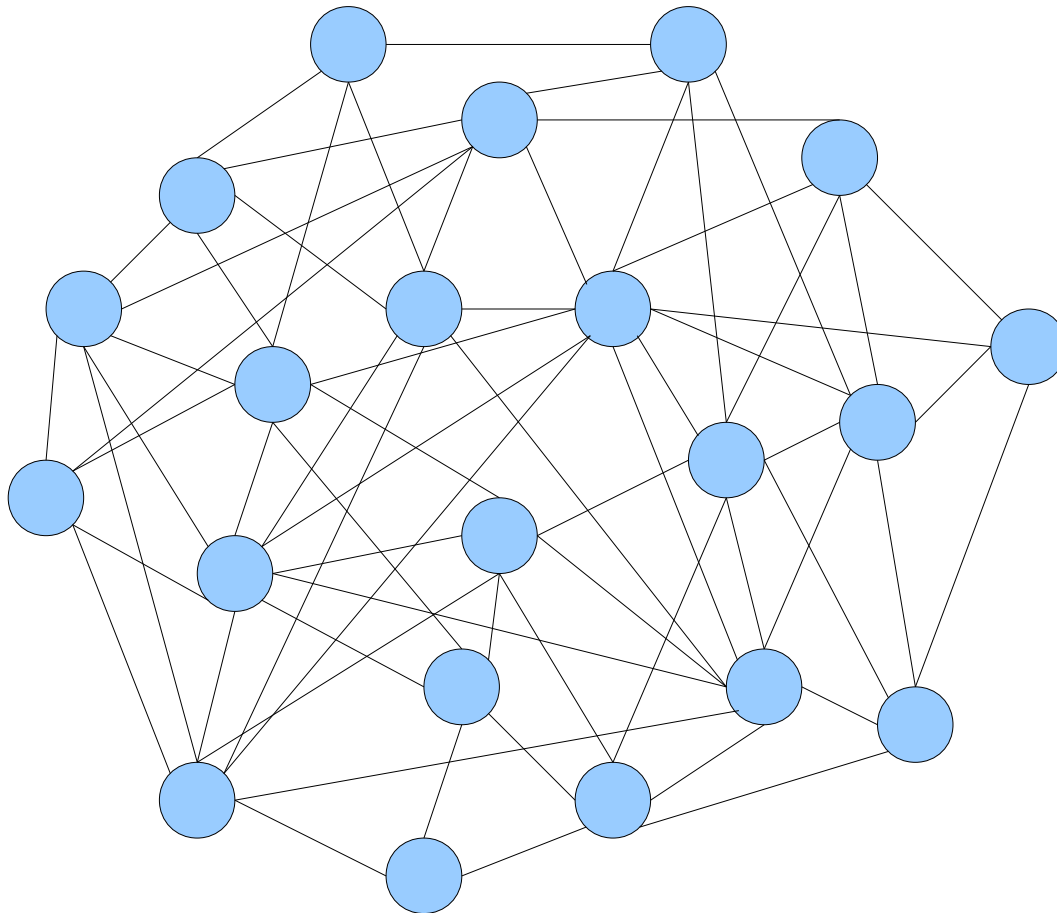
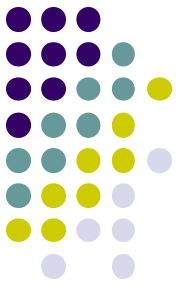
4-klikk



5-klikk



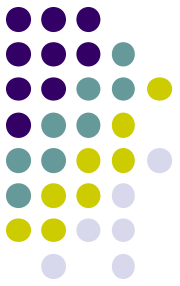
Clique-problemet



Å finne den **største** klikken i en graf som inneholder over 100 noder kan ta flere århundrer å finne ved brute-force-søking

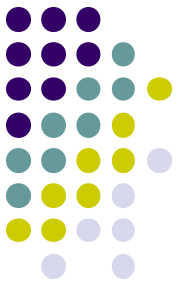
Men er det nødvendig?

Andre søkeproblemer



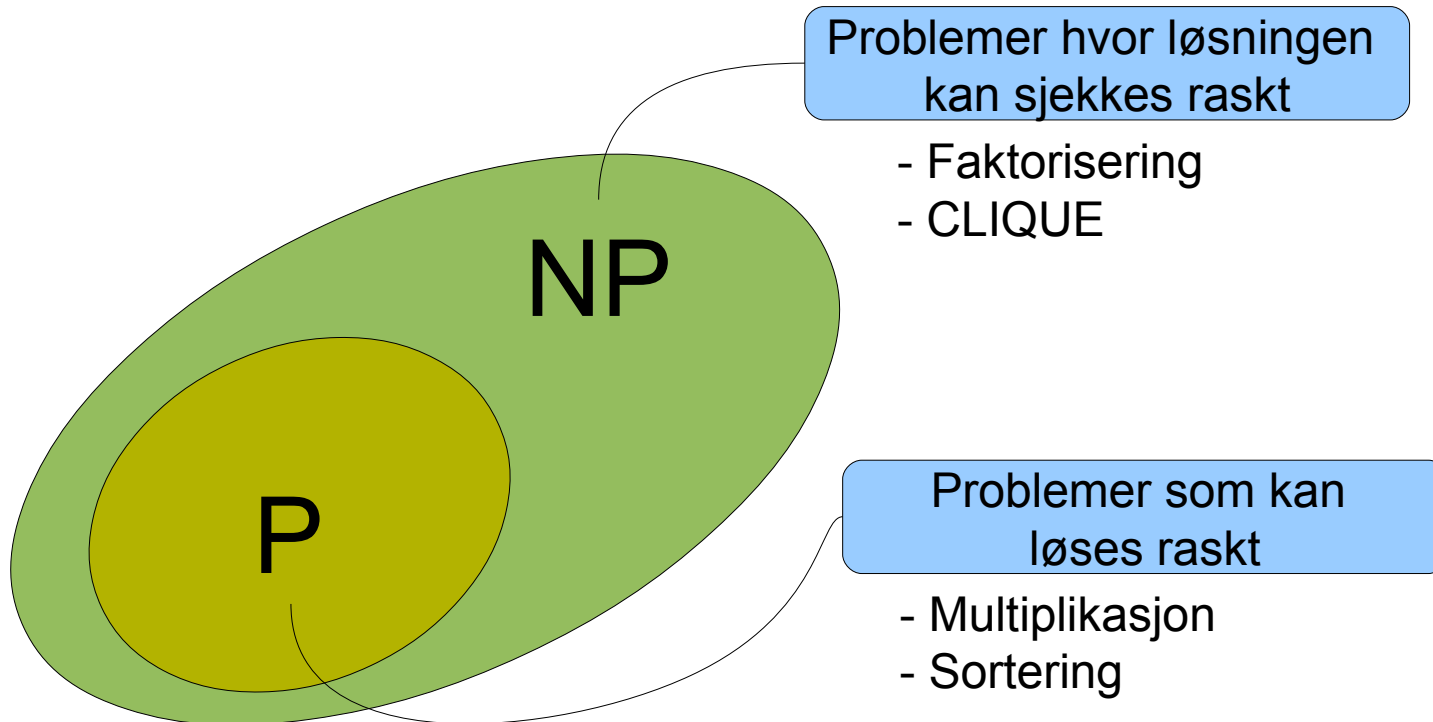
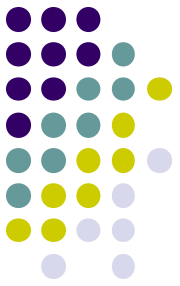
- Scheduling (planlegging)
- Fargelegging av kart
- Proteinfolding
- Teorem-beviser
- Sudoku
- Stabling av esker
- Design av microchiper
- Traveling salesman
- + veldig mange flere...

P og NP



- P – "Polynomial time"
 - Problemer som kan *løses* raskt
- NP – "Nondeterministic Polynomial time"
 - Problemer hvor en løsning kan *verifiseres* raskt
 - Inkluderer søkeproblemene

P og NP som problemklasser



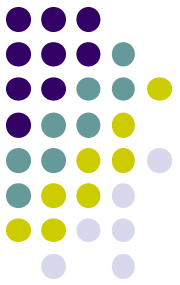
Problemer hvor løsningen kan sjekkes raskt

- Faktorisering
- CLIQUE

Problemer som kan løses raskt

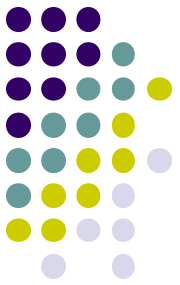
- Multiplikasjon
- Sortering

Det store spørsmålet



$P = NP$
eller
 $P \neq NP?$

Problemlassen NPC



212902463182587575474
978820162715174978067
039632772162782333832
153819499840564959113
665738530219183167831
073879953172308895692
30873441936471

Transformer
 \leq_P

Faktoriserings-
problemet

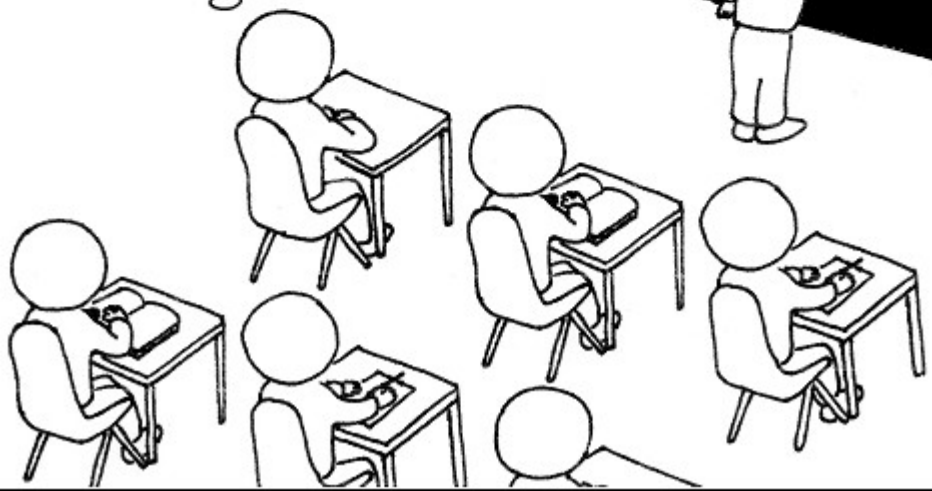
CLIQUE-
problemet



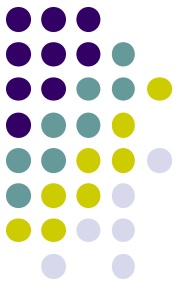
THUS, FOR ANY NONDETERMINISTIC TURING MACHINE M THAT RUNS IN SOME POLYNOMIAL TIME $p(n)$, WE CAN DEVISE AN ALGORITHM THAT TAKES AN INPUT w OF LENGTH n AND PRODUCES $E_{M,w}$. THE RUNNING TIME IS $O(p^2(n))$ ON A MULTITAPE DETERMINISTIC TURING MACHINE AND...

WTF, MAN. I JUST WANTED TO LEARN HOW TO PROGRAM VIDEO GAMES.

SIPSER CH7

$$N_i = (A_{i0} \vee B_{i0}) \wedge (A_{i1} \vee B_{i1}) \wedge \dots \wedge (A_{i,p-1} \vee B_{i,p-1}) \wedge (A_{i,p} \wedge B_{i,p})$$
$$N = N_0 \wedge N_1 \wedge \dots \wedge N_{p-1}$$


全吉博



Litt historie



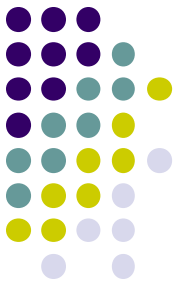
Alan Turing, 1912-1954



Alonzo Church, 1903-1995

- På 30-tallet begynner man å tenke over de fundamentale begrensningene en *regnemaskin* må ha
- Lager en teoretisk modell for datamaskinen slik vi kjenner den i dag
 - Turingmaskiner
 - λ -calculus
- Formaliserer begrepet *algoritme*
- Church-Turing-tesen

Litt historie - kompleksitetsteori

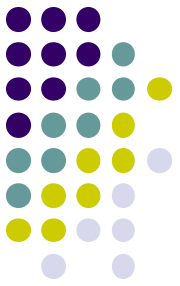


- 1960-tallet: Kompleksitetsteorien dannes, P og NP defineres
 - Hvor effektive er algoritmene våre?
- 1970-tallet: NP-kompletthet defineres
 - Edmonds, Rabin, Karp, Levin, Cook...
- 1956: Uoppdaget brev fra Gödel til von Neumann
 - Gödel kjente allerede til den moderne formuleringen av problemet



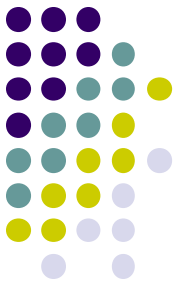
Kurt Gödel, 1906-1978

Clay Mathematics Institute – Millenium Prize Problems



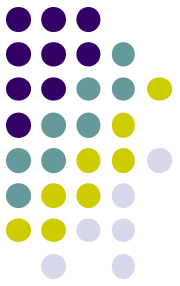
- ✓ Poincaré-formodningen
- P vs NP
- Navier-Stokes' problem
- Riemann hypotesen
- Birch- og Swinnerton-Dyer-formodningen
- Yang-Mills teori
- Hodge-formodningen

\$1 000 000



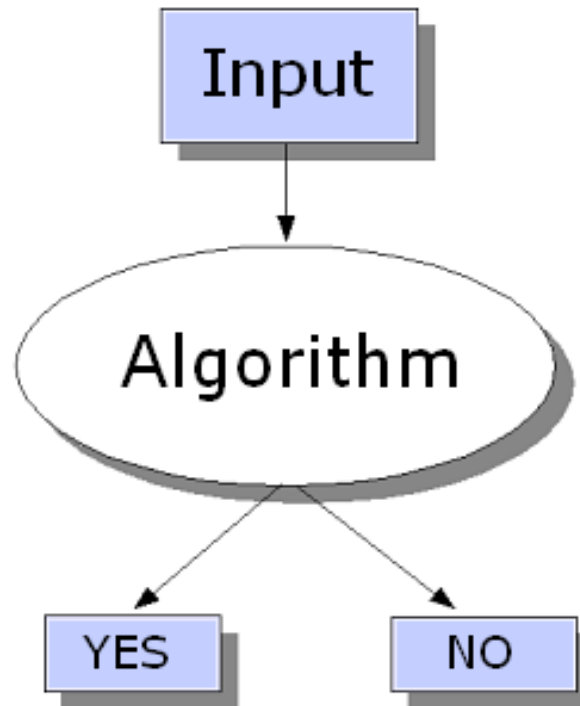
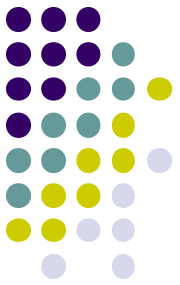
Problemreduksjoner

Reduksjoner

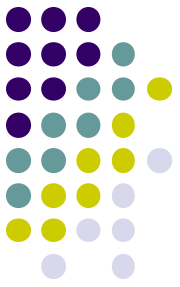


- Transformerer en type problem til et annet.
- Kan brukes til å etablere en **relativ** vanskelighetsgrad mellom to typer problemer.
- Behøver ikke vite hvor vanskelig problemene *faktisk* er.

Avgjørelsesproblemer

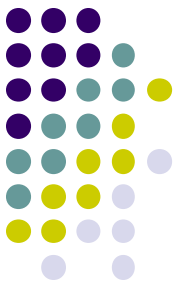


Optimaliseringsproblemer vs avgjørelsesproblemer



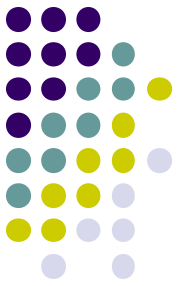
- $\text{SHORTEST-PATH}(u, v, G)$ (Optimaliseringsproblem)
 - Finn stien med **færrest kanter** fra u til v i en graf G .
- $\text{PATH}(u, v, G, k)$ (Avgjørelsesproblem)
 - **Eksisterer** det en sti fra u til v med maks k kanter?

Optimaliseringsproblemer vs avgjørelsesproblemer



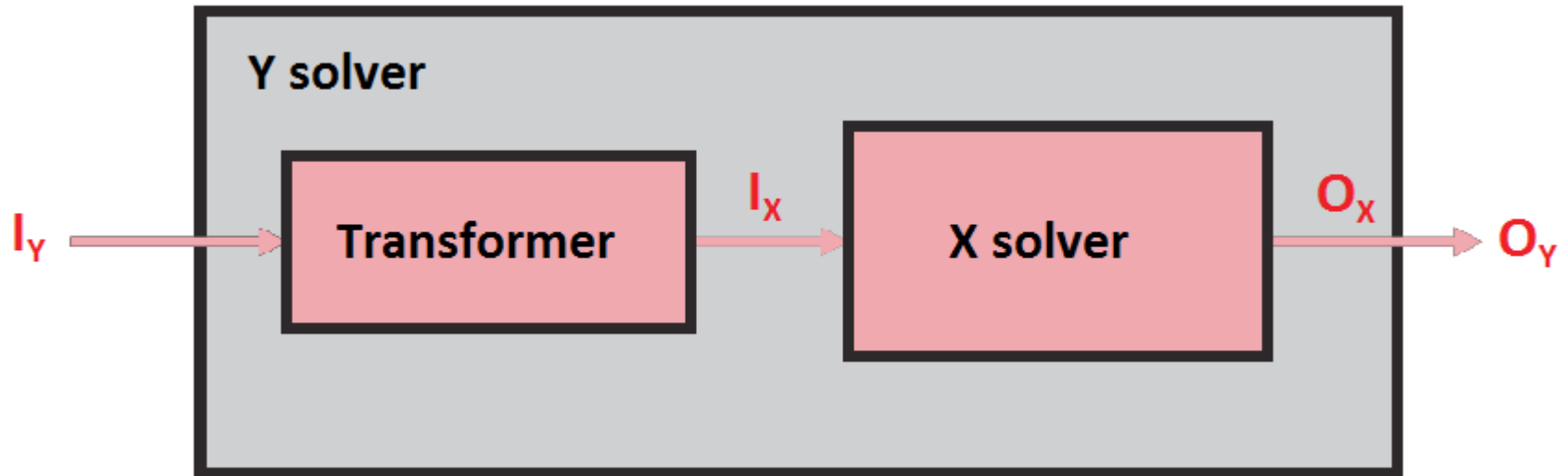
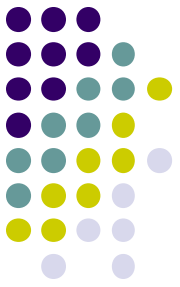
- Anta vi kan løse SHORTEST-PATH kjapt:
 - Sammenlign svaret med parameteren k i PATH.
 - Kan dermed løse PATH vha SHORTEST-PATH.
- Anta motsatt, at vi kan løse PATH kjapt:
 - Finnes det en sti fra u til v med $V - 1$ kanter?
 - Nei: **return** "Ingen sti finnes"
 - Ja: Finnes det en sti med $(V - 1) / 2$ kanter?
 - Nei: Finnes det en sti med $3(V - 1) / 4$ kanter?
 - etc...
 - Ja: Finnes det en sti med $(V - 1) / 4$ kanter?
 - etc...
 - Kan dermed løse SHORTEST-PATH vha PATH.

Reduksjoner



- Ønsker å vise:
 - "Problem X er minst like vanskelig som problem Y".
 - Vet ikke hvor vanskelig noen av dem *faktisk* er.
- Hvordan gjør vi det?

Reduksjon



Reduksjoner

- Javel...og så?

X kan løses effektivt

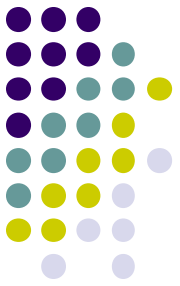


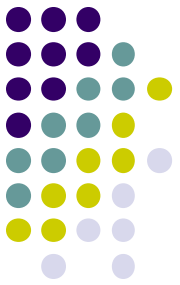
Y kan løses effektivt

Y kan IKKE løses effektivt



X kan IKKE løses effektivt





Polynomreduksjon

kjøpingsproblem

Y



ukjent problem

X

Y

\leq_P

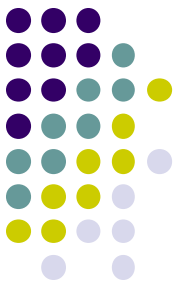
X

"X er minst like vanskelig som Y"

X kan løses effektivt
↓
Y kan løses effektivt



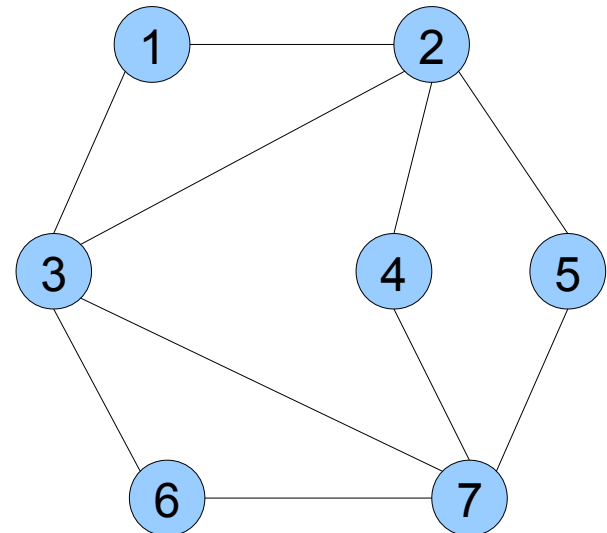
X kan IKKE løses effektivt
↑
Y kan IKKE løses effektivt

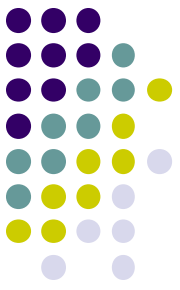


Independent Set

- Et independent **set** i en graf er en samling noder hvor ingen av nodene har kanter til hverandre.
- INDEPENDANT-SET(G, k): Finnes det et independent set i en graf G , større eller lik k ?

- $k = 4$?
 - Ja: $\{1, 4, 5, 6\}$
- $k = 5$?
 - Nei

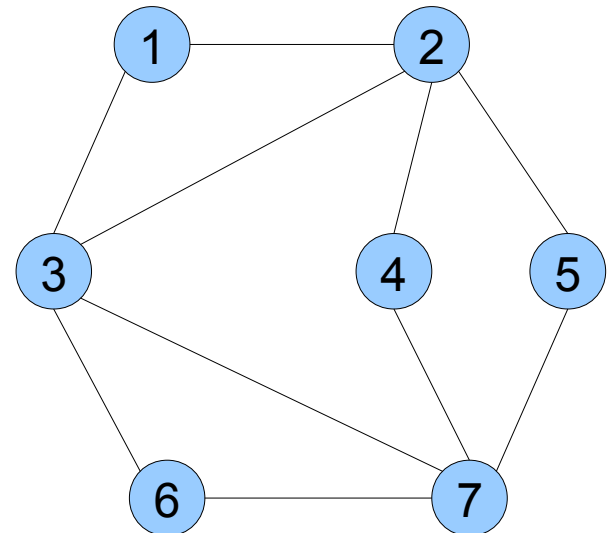


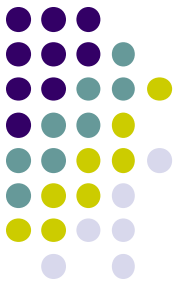


VERTEX-COVER

- Et **vertex-cover** i en graf er en samling noder, slik at alle kantene i grafen er knyttet til minst én av disse nodene.
- VERTEX-COVER(G, k): Har grafen G et vertex cover bestående av *maks* k noder?

- $K = 3$?
 - Ja: $\{2, 3, 7\}$
- $K = 2$?
 - Nei.

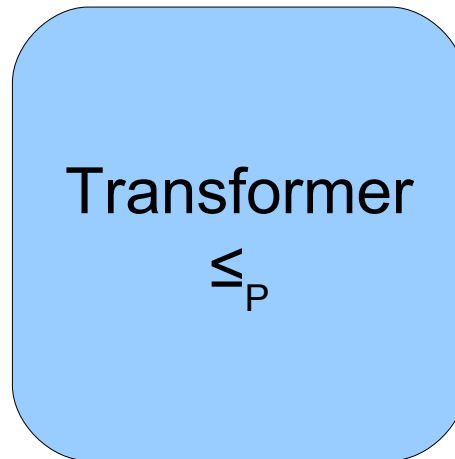
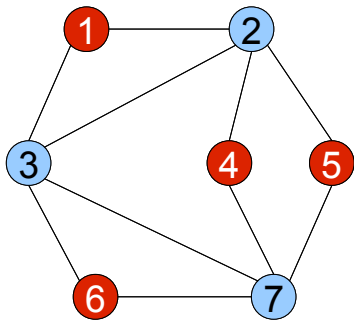


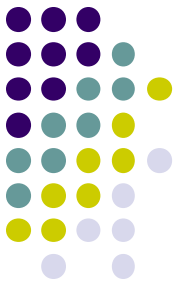


INDEPENDENT-SET \leq_P VERTEX-COVER

Har grafen et independent set ≥ 4 ?

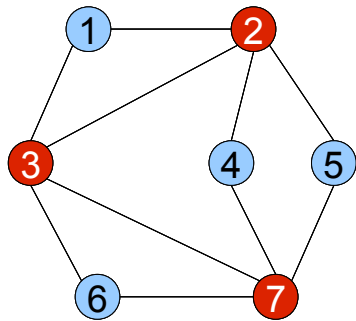
Har grafen et vertex-cover $\leq 7 - 4 = 3$?



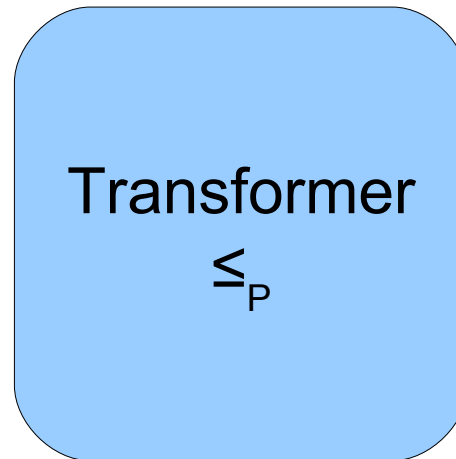


VERTEX-COVER \leq_P INDEPENDENT-SET

Har grafen et
vertex-cover ≤ 3 ?

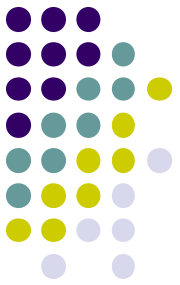


Har grafen et
independent set $\geq 7 - 3 = 4$?



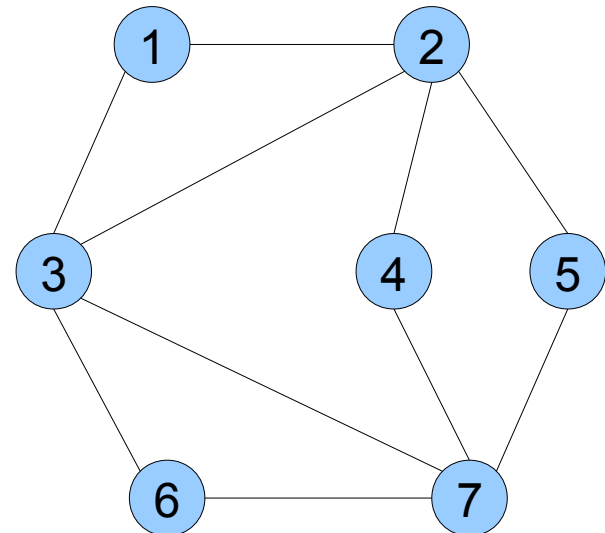
Gitt grafen $G = (E, V)$.

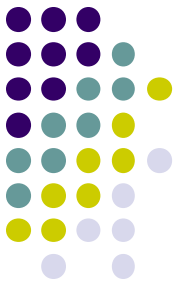
S er et independent set $\leftrightarrow V - S$ er et vertex cover.



CLIQUE

- En **klikk** er en samling noder i en graf som utgjør en *komplett* subgraf.
- CLIQUE(G, k): Finnes det en klikk i grafen G med minst k noder?
- $k = 3$?
 - Ja: $\{1, 2, 3\}$
- $k = 4$?
 - Nei.

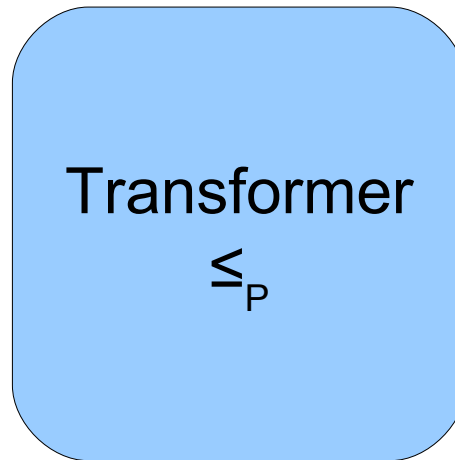
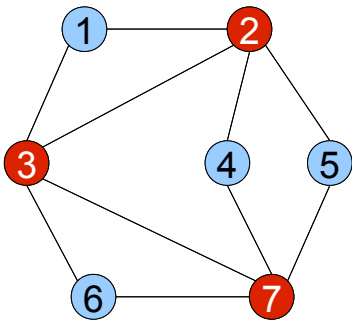




VERTEX-COVER \leq_P CLIQUE

Har grafen et vertex-cover ≤ 3 ?

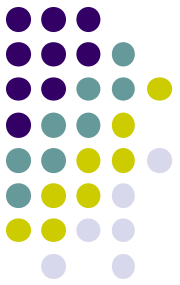
Har grafen en klikk $\geq 7 - 3 = 4$?



G har et vertex cover av størrelse k



\bar{G} har en klikk av størrelse $V - k$



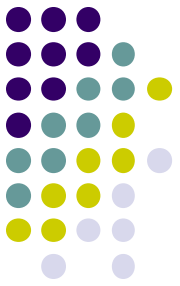
3-CNF-SAT

- 3-CNF-SAT: Gitt
 - $z_i \in \{x_1, x_2, \dots, x_n, \bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$,
 - $C_t = (z_i \vee z_j \vee z_k), i \neq j \neq k$
 - $\phi = C_1 \wedge C_2 \wedge \dots \wedge C_k$,

Hvilke verdier må z_1, z_2, \dots, z_k ha for at ϕ skal være **sann**?

- Eksempel: $\phi = (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$
 - $x_1 = 1$
 - $x_2 = 0$
 - $x_3 = 1$

3-CNF-SAT \leq_P INDEPENDENT-SET



Hvilke verdier av x_1, x_2, x_3
må til for å tilfredsstille Φ ?

$$\phi = C_1 \wedge C_2 \wedge C_3$$

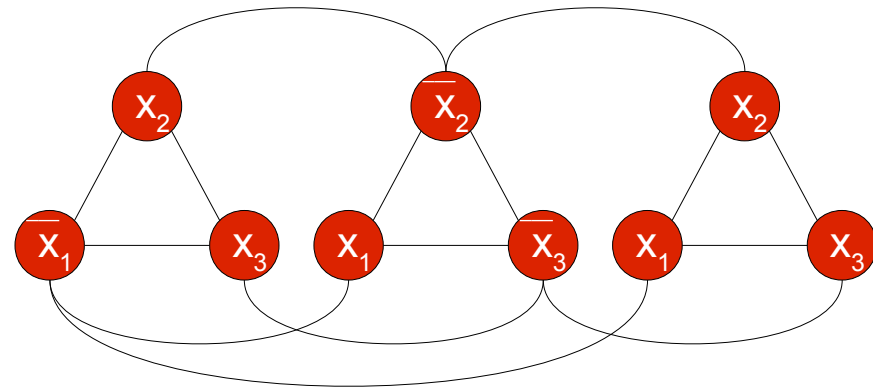
$$C_1 = (\bar{x}_1 \vee x_2 \vee x_3)$$

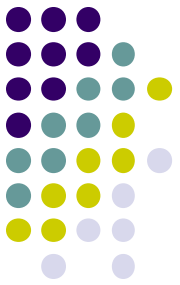
$$C_2 = (x_1 \vee \bar{x}_2 \vee \bar{x}_3)$$

$$C_3 = (x_1 \vee x_2 \vee x_3)$$

Transformer
 \leq_P

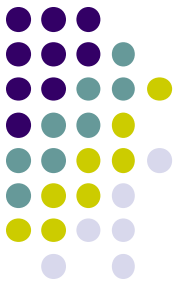
Klarer vi å finne et independent
set av størrelse 3?





Kompleksitetsklasser (Problemklasser)

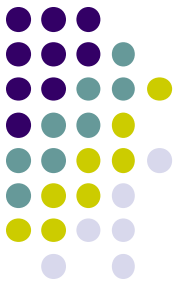




Problemlassen P

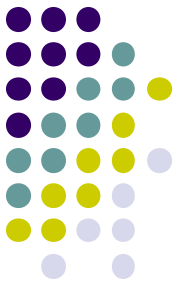
P: Mengden av *avgjørelsesproblemer* som kan løses i polynomisk tid

- Polynomisk tid: $O(n^k)$
 - Eksempel: $O(1)$, $O(\log n)$, $O(n)$, $O(n^2)$, $O(n^{100})$
- Problemer i P kalles ofte **tractable** (på godt norsk) og algoritmene som løser dem **effektive**.



Noen problemer i P

- Sortering
- Korteste vei
- Maks flyt
- Traversering
- Minimale spenntrær
- Primtallstesting
 - Først bevist i 2002
- ...omtrent hele pensum

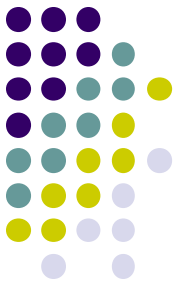


Problemlassen NP

NP: Mengden av *avgjørelsesproblemer* der en **ja-løsning** kan verifiseres i polynomisk tid

- Blir vi gitt en ja-løsning på problemet kan vi verifisere at den er korrekt i polynomisk tid.
- NP står for *Nonderministic Polynomial time*.

COMPOSITES vs PRIMES



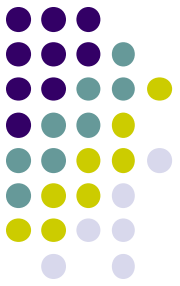
- Er dette et sammensatt (composite) tall?

15226050279225333605356183781326374297180
68114961380688657908494580122963258952897
654000350692006139

- Ja, fordi den har følgende faktor:

37975227936943673922808872755445627854565
536638199

COMPOSITES vs PRIMES

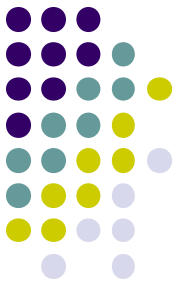


- Er dette et sammensatt tall?

3347807169895689878604416984821269081770479498371
3768568912431388982883793878002287614711652531743
087737814467999489

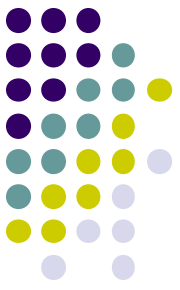
- Nei...

???



Problemlassen NP

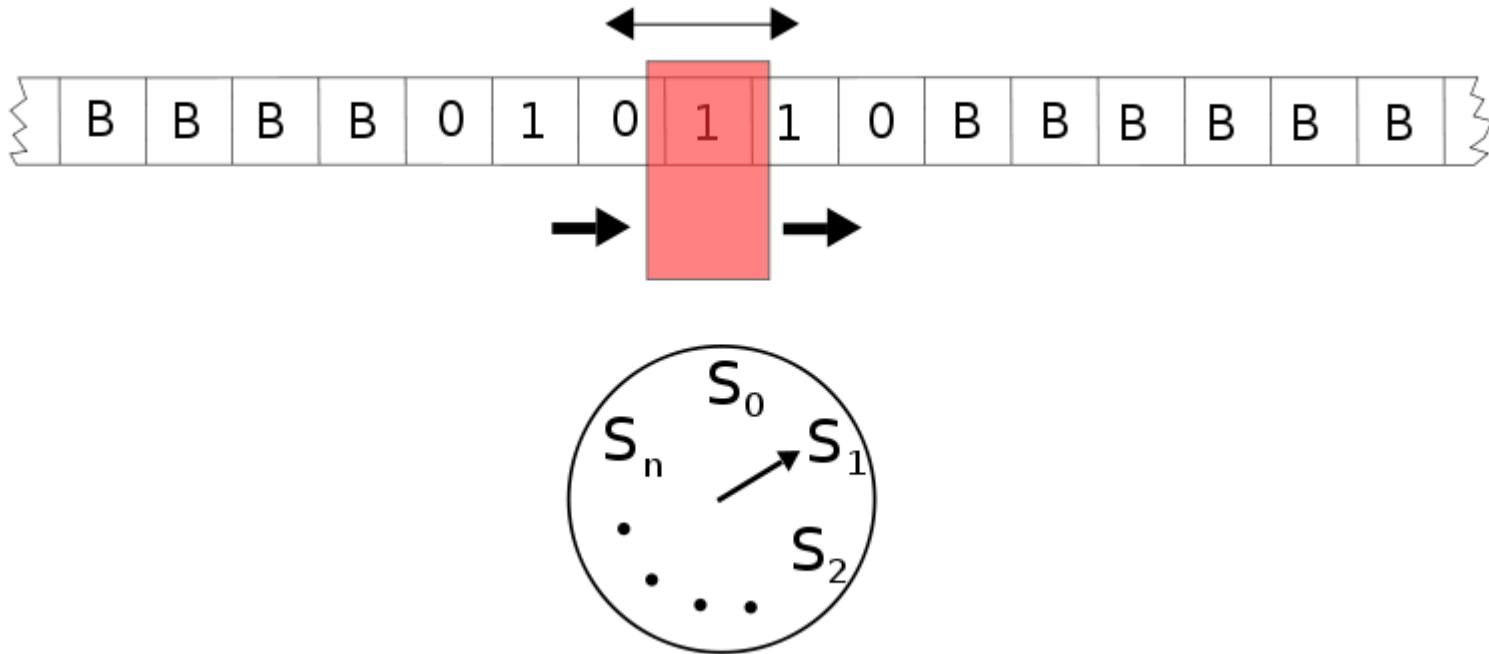
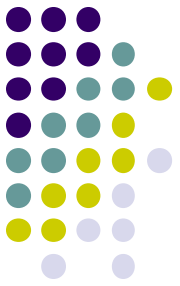
- Åpenbart at P ligger i NP.
 - Beregn en korrekt løsning i polynomisk tid.
- Alle problemer i NP kan løses i eksponentiell tid vha. brute-force-søk:
 - Generer alle de mulige løsningene.
 - Verifiser dem.
- Men kan vi klare bedre?
- Det vet vi ikke!

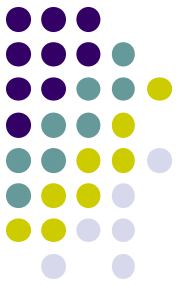


P vs NP

- Spørsmålet om $P=NP$ eller $P \neq NP$ er det viktigste åpne spørsmålet innen Computer Science
- Hvis $P=NP$:
 - Et hav av problemer lar seg løse effektivt
 - Nesten all kryptografi må revurderes
 - Matematiske beviser kan automatiseres
 - Kreativitet gjøres "overflødig"
 - Like lett å *lage* LF som å *koke* LF
- Hvis $P \neq NP$:
 - Omtrent som før

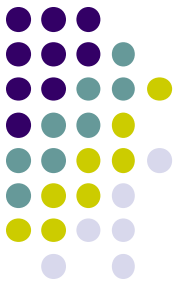
Digresjon – Nondeterministic Turingmaskin





NP-komplette problemer

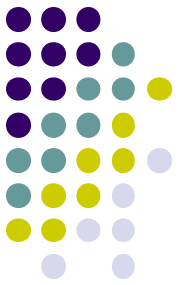




NP-komplette problemer

- Alle problemer i NP kan reduseres til noen spesielle problemer – de **NP-komplette problemene**.
- Klarer du å løse *ett* NP-komplett problem i polynomisk tid, kan du løse **ALLE** problemene i NP i polynomisk tid.
 - På grunn av reduksjonsegenskapen over.
 - Avhenger av at reduksjonen er polynomisk.
- Spørsmålet om $P = NP$ er redusert til *ett* problem.

Problemklassen NPC (NP-complete)

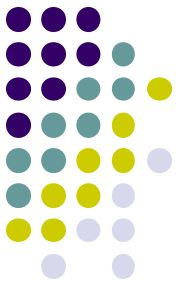


Et problem X er **NP-komplett** hvis

1) X er i NP

2) Ethvert problem Y i NP kan transformeres til X i polynomisk tid (betegnet $Y \leq_p X$)

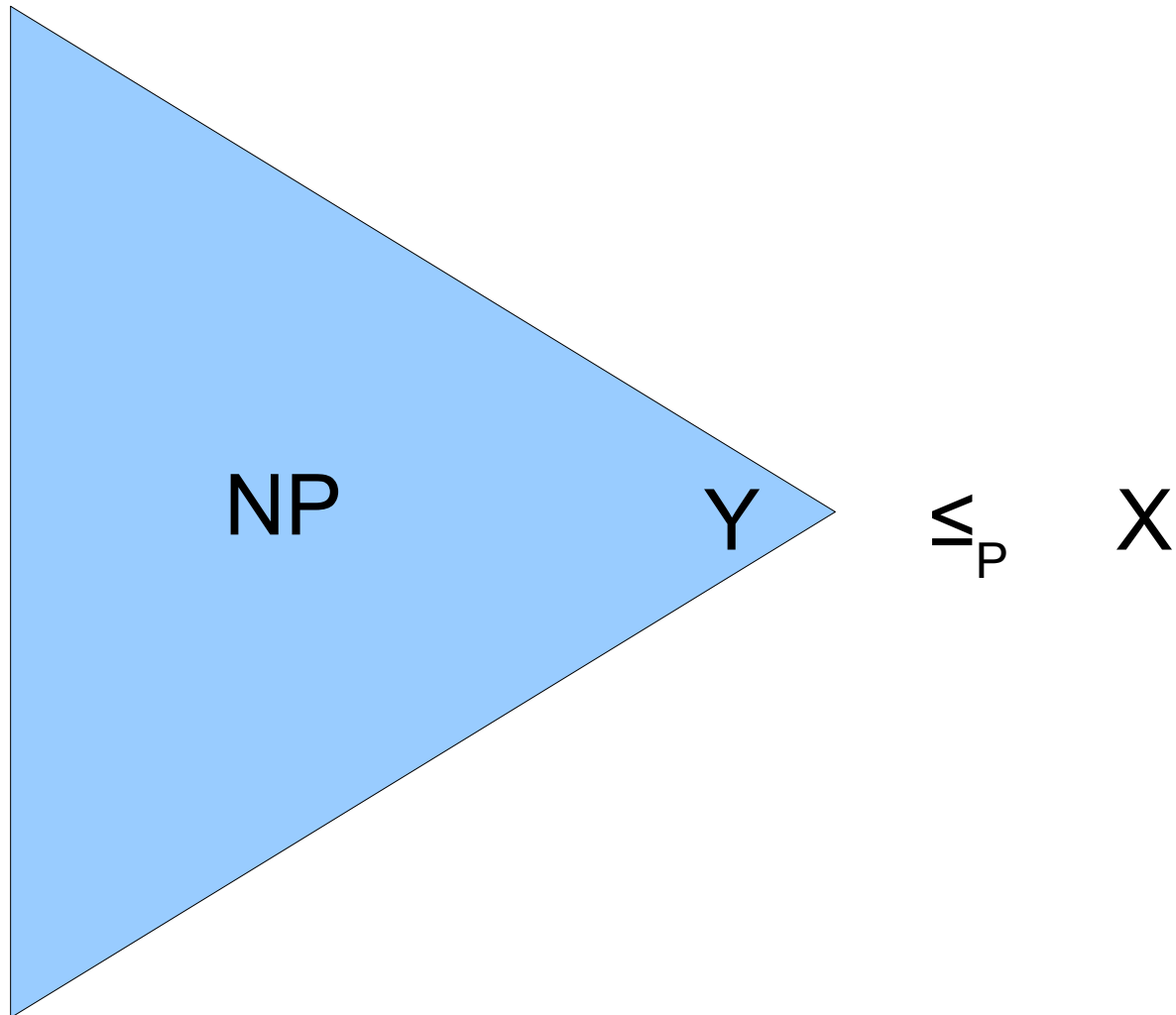
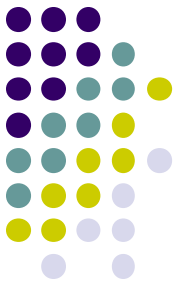
Mengden av alle NP-komplette problemer kalles **NPC**.
Et problem kalles **NP-hard** om det oppfyller 2).

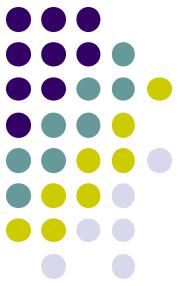


Vise at et problem er i NPC

- Gitt et ukjent problem X .
 - 1) Vis at et ja-svar kan verifiseres i polynomisk tid.
 - 2) Finn et problem Y , allerede vist til å ligge i NPC, og reduser det til X .
- Nå kan **alle** problemer i NP reduseres til X .
- X er mao, minst like vanskelig som **alle** andre NP-problemer.

Vise at et problem er i NPC





Det første NPC-problemet

- Hvordan beviste man det første NPC-problemet?
- Cook-Levin-Teoremet (1970)
- (Circuit)-SAT er NP-komplett

4/20 Notes

#	starting config				#
#	0	q ₁	1	0	0
#	:	:	:	:	:
#	q accept				#

Input is a Turing Machine and input string
Output is a satisfiability instance

num rows = n^k n - input size
 $k > 0$ Satisfiability is NP complete

$\Phi = \Phi_{\text{cell}} \wedge \Phi_{\text{start}} \wedge \Phi_{\text{move}} \wedge \Phi_{\text{accept}}$

Φ_{cell} : Guarantee that each table cell is valid $O(n^{2k})$
 Φ_{start} : iff start configuration accurate $O(n^k)$
 Φ_{move} : each transition is valid $O(n^{2k})$
 Φ_{accept} : there is an accepting configuration $O(n^{2k}) = O(n^{2k})$

$C = Q \cup \Gamma \cup \{ \# \}$
(a valid character in the table)

$X_{i,j,s} = \begin{cases} 1 & \text{cell}[i,j] = s \\ 0 & \text{otherwise} \end{cases}$

$\Phi_{\text{cell}} = \bigwedge_{1 \leq i,j \leq n^k} \left[\left(\bigvee_{s \in C} X_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,t \in C \\ s \neq t}} \overline{X_{i,j,s} \vee X_{i,j,t}} \right) \right]$

requires that one of these variables is set to true. make sure only one is true (there is only one character per square)

size = $O(n^{2k})$

$\Phi_{\text{start}} = X_{1,1,\#} \wedge X_{1,2,q_0} \wedge X_{1,3,w_0} \wedge X_{1,4,w_0} \wedge \dots$
size = $O(n^k)$

$\Phi_{\text{accept}} = \bigvee_{1 \leq j \leq n^k} X_{j,j,q_{\text{accept}}}$
size = $O(n^{2k})$

valid \rightarrow

a	q ₁	b
q ₁	a	e

 $S(q_1, a) = \{ (q_1, b, e) \}$
 invalid \rightarrow

a	q ₁	b
q ₂	b	c

 $S(q_1, b) = \{ (q_2, c, L), (q_2, a, R) \}$

$\Phi_{\text{move}} = \bigwedge_{1 \leq i,j \leq n^k} \left[\bigvee_{\substack{a_1, \dots, a_4 \\ \text{is a legal window}}} \left(X_{i-1,j,a_1} \wedge X_{i,j,a_2} \wedge X_{i+1,j,a_3} \wedge X_{i,j+1,a_4} \wedge X_{j+1,i,a_5} \wedge X_{i+1,j+1,a_6} \right) \right]$
size = $O(n^{2k})$

(i-1,j)	(i,j)	(i+1,j)
(i-1,j+1)	(i,j+1)	(i+1,j+1)

 Q.E.D.

Karp (1972): Reducibility Among Combinatorial Problems

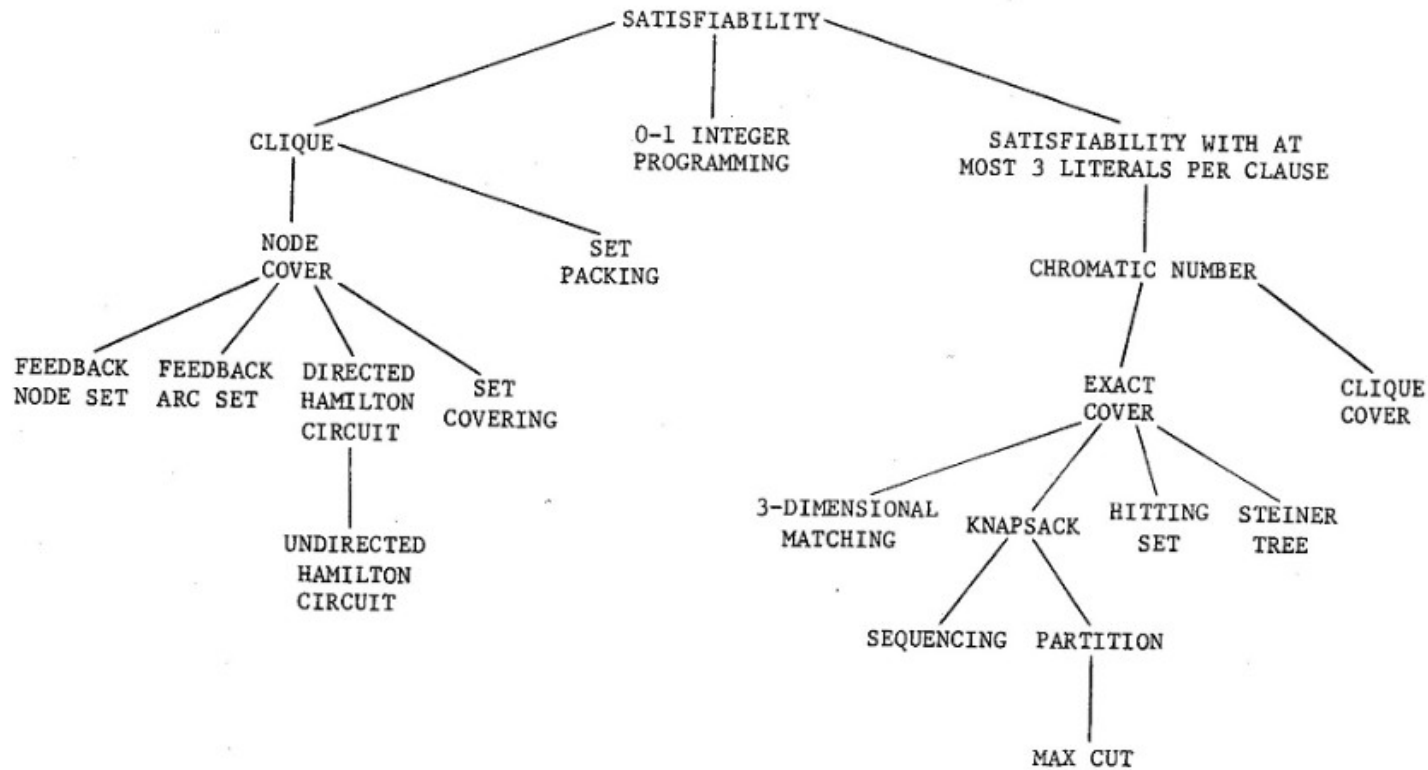
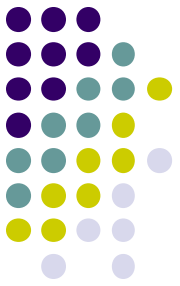
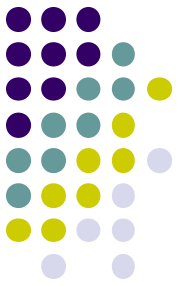


FIGURE 1 - Complete Problems

Slik tror vi det er...

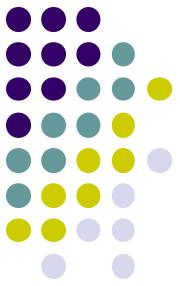


NP

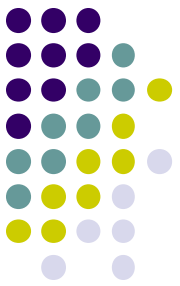
P

NPC

...eller er det slik?

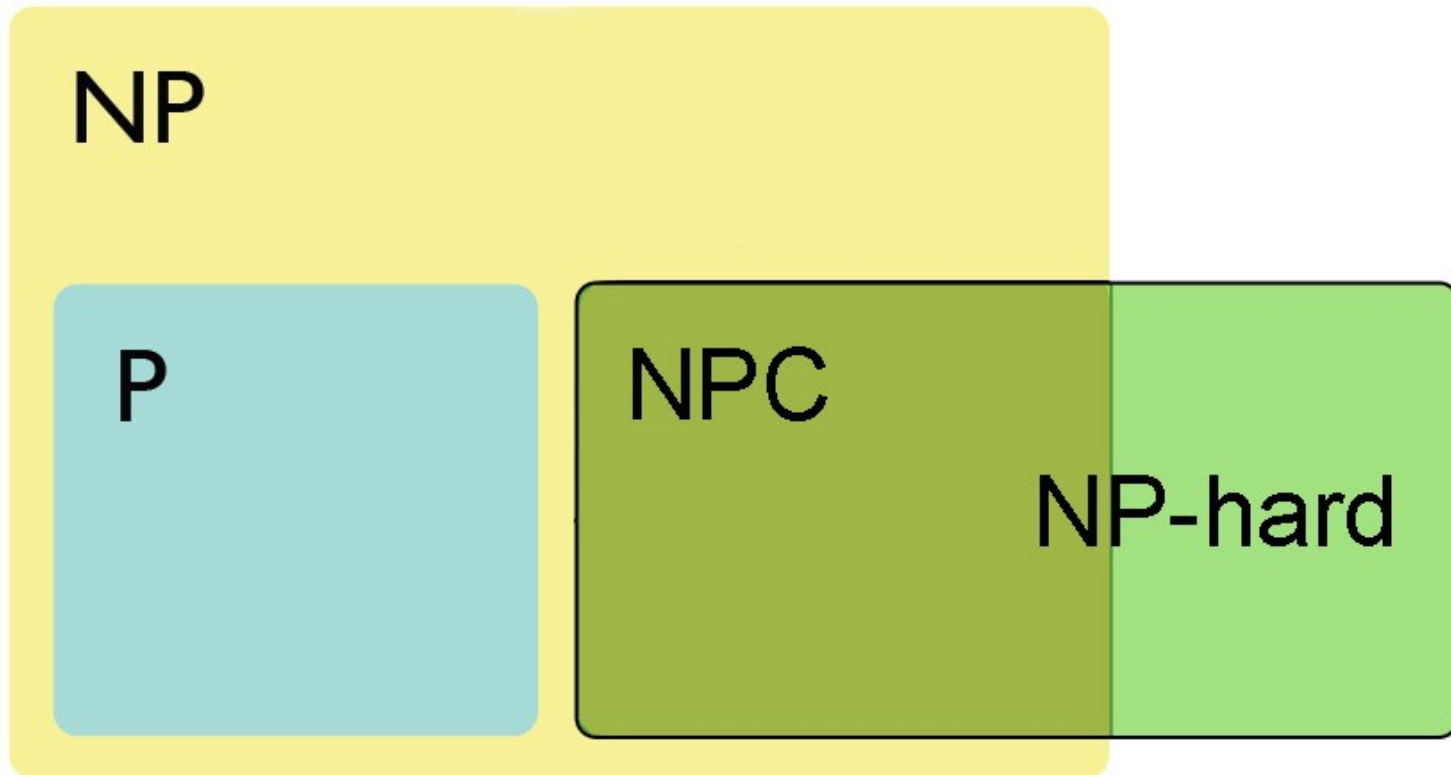


$P = NP = NPC$

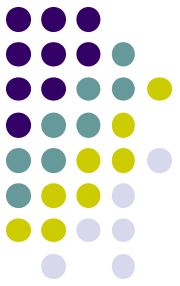


Fullstendig venndiagram

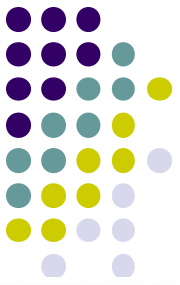
- Slik er det hvis $P \neq NP$



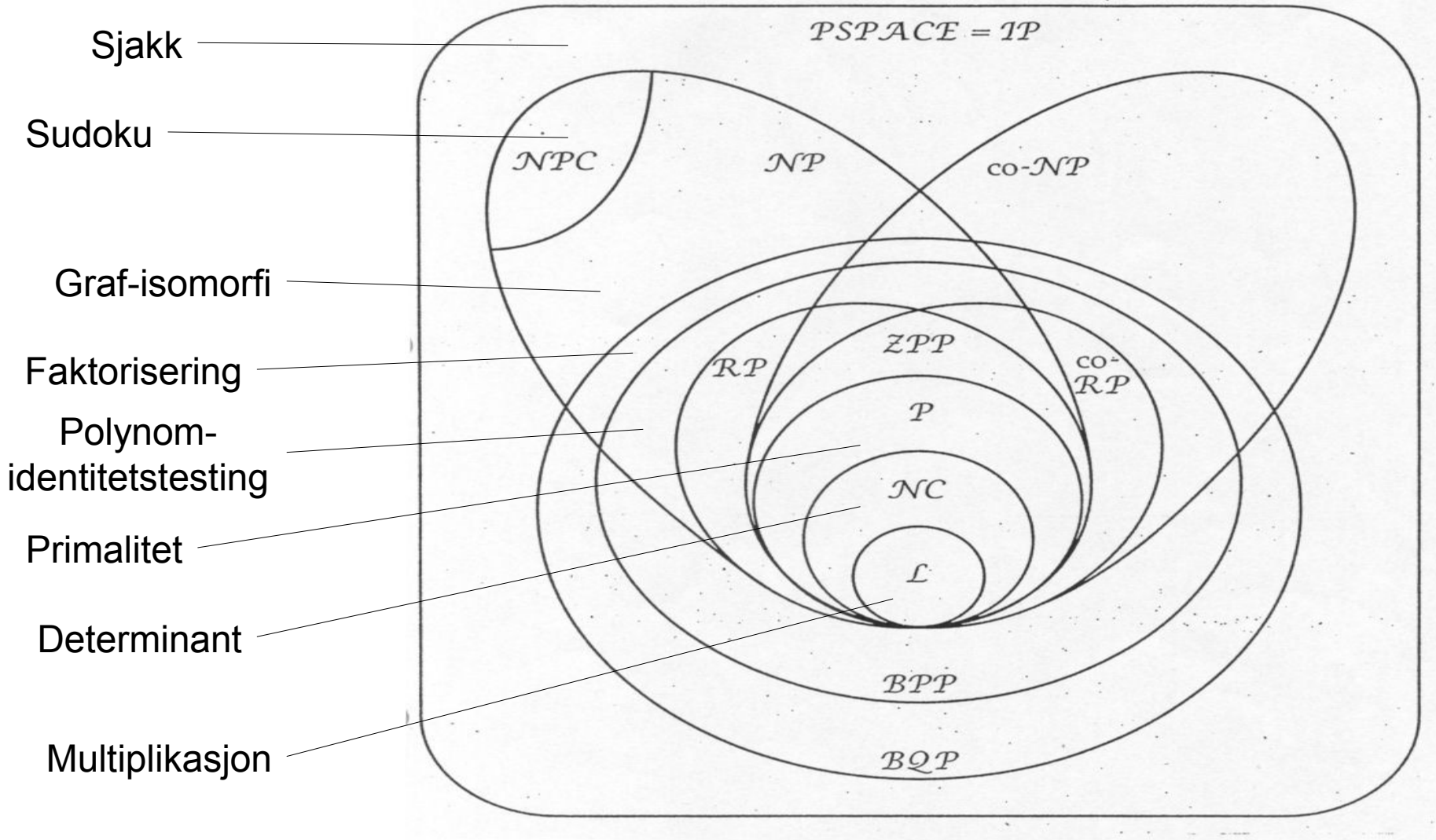
Problemer i NP som vi IKKE tror er i NPC

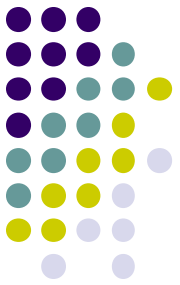


- Vi vet ikke om så mange naturlige problemer...
 - Faktorisering
 - Grafisomorfi
 - Stokastiske spill
 - Diskrete logaritmer
- ...men hvis $P \neq NP$ sier Ladners teorem (1971) at det finnes uendelig mange av dem



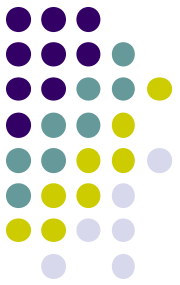
Flere problemklasser





Noen NPC-problemer

- SAT (Satisfiability)
- CIRCUIT-SAT
- 3-CNF-SAT
- SUBSET-SUM
- CLIQUE
- VERTEX-COVER
- HAM-CYCLE
- TSP (Travelling Salesman Problem)
- GRAPH K-COLORABILITY
- KNAPSACK
- 0-1 INTEGER-PROGRAMMING
- SUBGRAPH-ISPMORPHISM

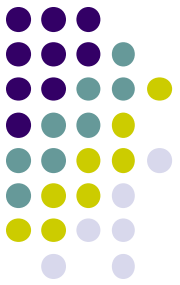


SUBSET-SUM

- Vi har en sekvens av tall, og vil finne ut om noen av disse tallene kan legges sammen til en gitt sum
- Eks:

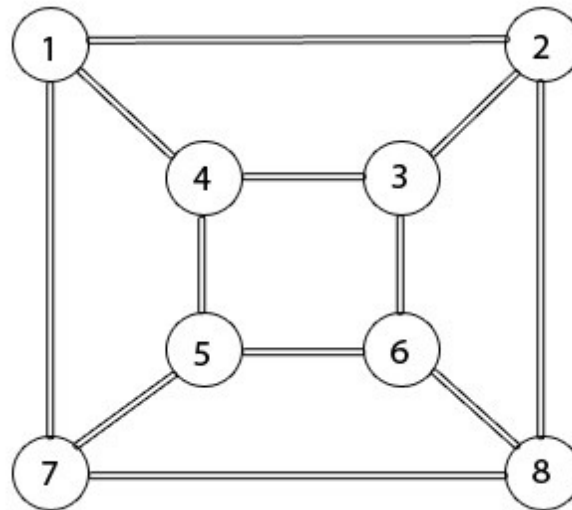
$$\{8, 13, 4, 7, -17, 23, 2\}$$

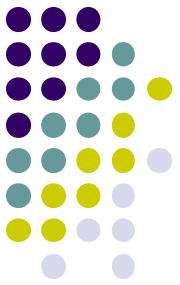
- Finner vi summen 39 her?
 - Nei
- Finner vi summen 40 her?
 - Ja, $23 + 13 + 4 = 40$



HAM-CYCLE

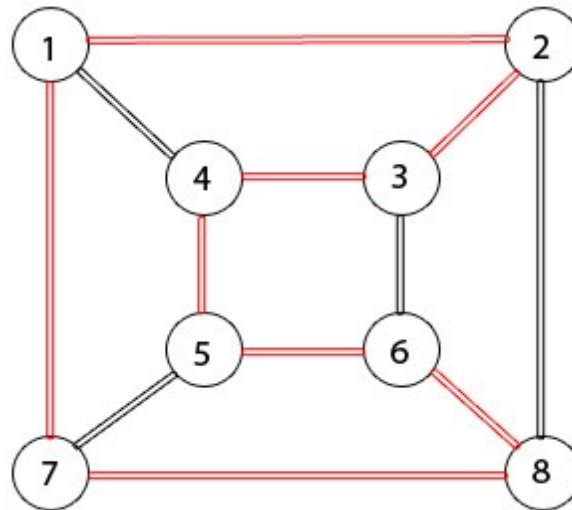
- En **hamiltonsk sykel** i en urettet graf G , er en simpel sykel som inneholder alle noder i G .
- Finnes det en Hamiltonsk sykel i grafen?

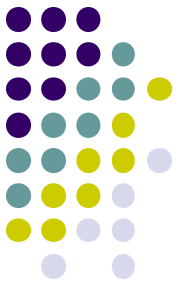




HAM-CYCLE

- En **hamiltonsk sykel** i en urettet graf G , er en simpel sykel som inneholder alle noder i G
- Finnes det en Hamiltonsk sykel i grafen?



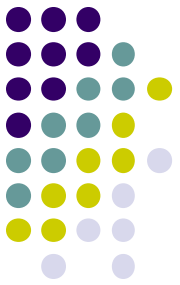


TSP (Travelling Salesman Problem)

- I en vektet, urettet graf, finn billigste sykel som besøker alle nodene nøyaktig én gang.
- Eks.
 - den korteste veien gjennom de 15 største byene i Tyskland

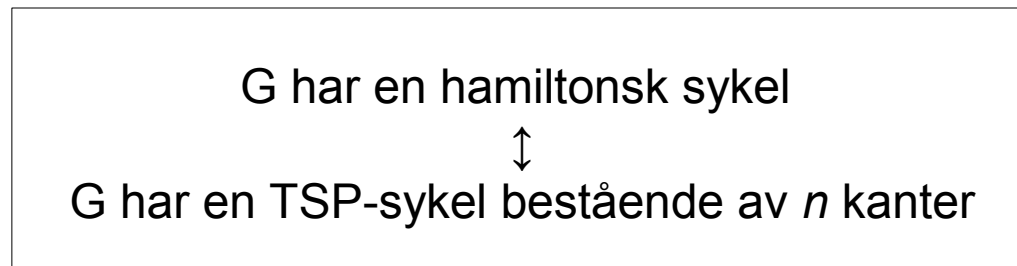
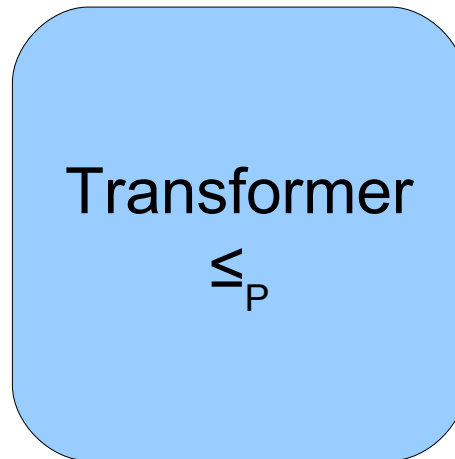
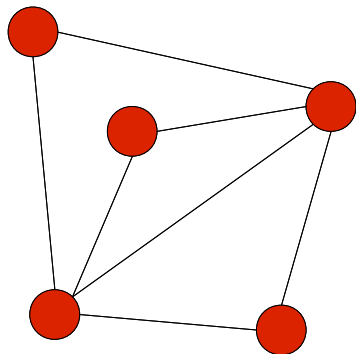


HAM-CYCLE \leq_P TSP

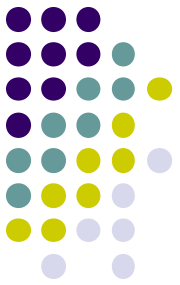


Finnes det en hamiltonsk sykel i grafen?

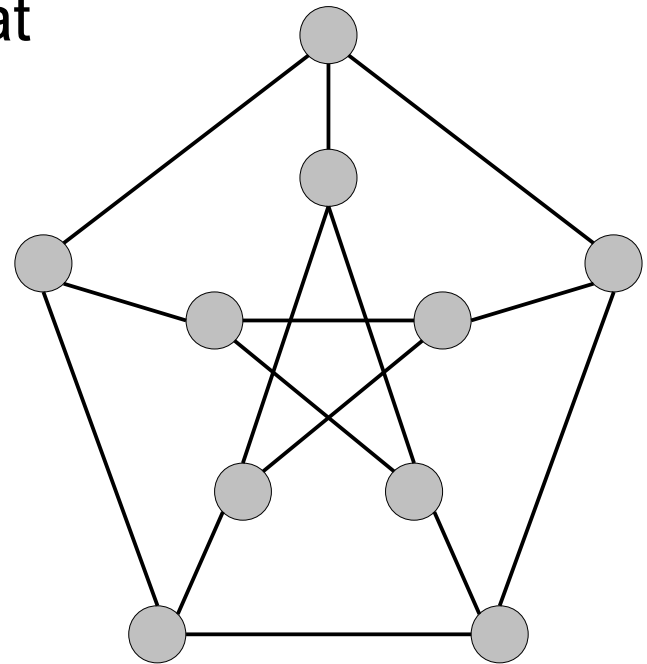
Finn billigste sykel bestående av 5 kanter



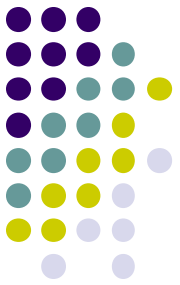
GRAPH K-COLORABILITY



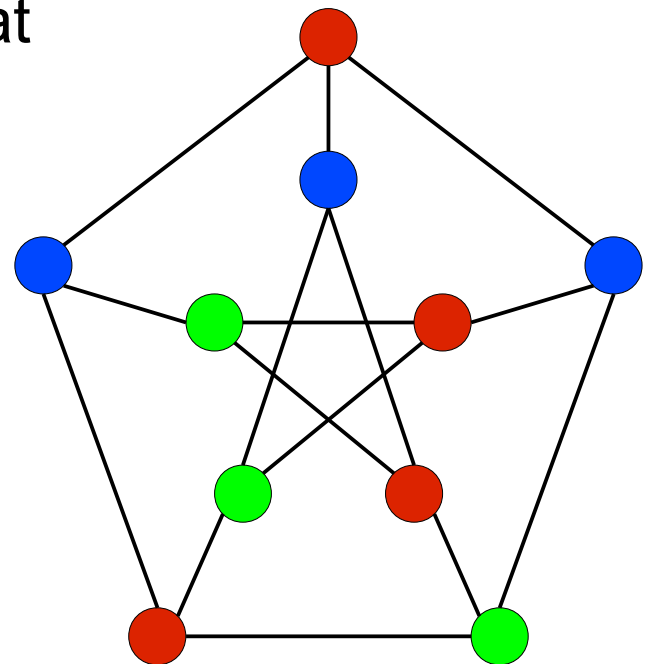
- Gitt k forskjellige farger, kan vi fargelegge nodene i en graf slik at ingen naboer har samme farge?
- Eks:
 - Hvis $k = 3$?
 - Hvis $k = 2$?

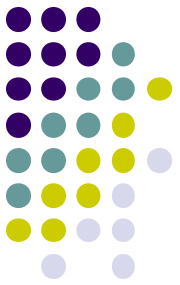


GRAPH K-COLORABILITY



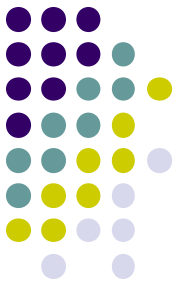
- Gitt k forskjellige farger, kan vi fargelegge nodene i en graf slik at ingen naboer har samme farge?
- Eks:
 - Hvis $k = 3$?
 - Ja.
 - Hvis $k = 2$?
 - Nei
 - Obs! Problemet når $k = 2$ kan løses i polynomisk tid.





Noen tidligere eksamensoppgaver

Eksamen desember 2008



Oppgave 1

Du står overfor de tre problemene A, B og C. Alle tre befinner seg i mengden NP. Du vet at A er i mengden P og at B er i mengden NPC. Anta at du skal bruke polynomiske reduksjoner mellom disse problemene til å vise ulike egenskaper.

Merk: Enkelte av egenskapene kan selvfølgelig vises på andre måter. Du kan se bort fra det i denne oppgaven.

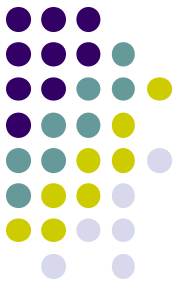
f) Fullfør følgende utsagn

For å bevise at C er i P må C reduseres til A i polynomisk tid.

For å bevise at C er i NPC må B reduseres til C i polynomisk tid.

Hvis B kan reduseres til A i polynomisk tid, så er $P = NP$.

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

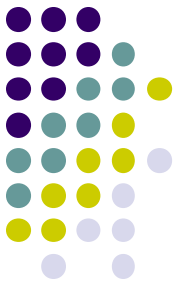
4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

Ja Nei

5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

[] 1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

[] [] 2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

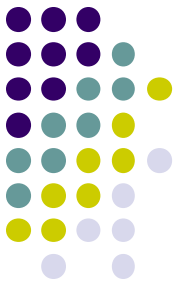
[] [] 3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

[] [] 4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

[] [] 5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

[] 1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

[] 2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

[] [] 3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

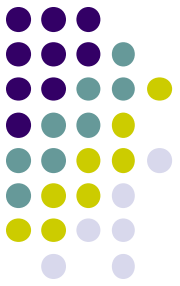
[] [] 4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

Ja Nei

[] [] 5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

[] 1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

[] 2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

[] 3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

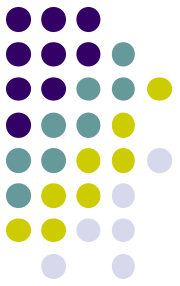
[] [] 4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

Ja Nei

[] [] 5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

[] 1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

[] 2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

[] 3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

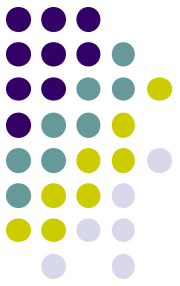
[] 4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

Ja Nei

[] [] 5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei

Eksamen mai 2006



Oppgave 1

d) Anta at du har to NP-komplette problemer A og B. Anta også at du har et problem C fra mengden P.

[] 1. Hvis A befinner seg i P vil B også gjøre det.

Ja Nei

[] 2. Hvis du kan redusere problemet C til problemet A i polynomisk tid så følger det at C er NP-komplett.

[] 3. Hvis $P = NP$ vil det likevel finnes problemer i NP som ikke er NP-komplette.

[] 4. Hvis $P \neq NP$ så er NP-komplette problemer ikke beregnbare.

Ja Nei

[] 5. 1-0-ryggsekkproblemet (kjøretid $O(nW)$) er NP-komplett.

Ja Nei