Diskret matematikk 4140

”Discrete Mathematics and Its Applications” (Rosen)

Oppsummering

Del 1 – Logikk og bevis, mengder og funksjoner

1.1 Logikk

Utsagn	Utsagn er en påstand (p) som enten er sann eller usann. Sannhetsverdi 0 eller 1, T eller F. Kan være primitive eller sammensatte
Sammensatte utsagn	Satt sammen av flere utsagn vha. logiske operatorer
Konjunksjon:	p og q. $p \wedge q$
Disjunksjon:	p eller q $p \vee q$
Eksklusiv eller:	$p \oplus q$. Utsagnet er sant bare når p eller q er sanne, ikke når begge er sanne
Negasjon:	"ikke p". $\neg$ p
Implikasjon	$p \rightarrow q$. p medfører q. Utsagnet er usant når hypotesen *p* er sann og konklusjonen *q* usann, ellers er det sant.
Bikondisjonal	$p \leftrightarrow q$ er utsagnet som er sant når *p* og *q* har samme sannhetsverdier, og ellers usant. *p* hvis og bare hvis *q*.
Konvers:	$q \rightarrow p$ er konversen (det omvendte) til $p \rightarrow q$
Kontrapositiv	$\neg q \rightarrow \neg p$ er det kontrapositive til $p \rightarrow q$ og har samme sannhetsverdier – det er usant bare når p er sann og q er usann.
Invers	$\neg p \rightarrow \neg q$ er inversen til $p \rightarrow q$
Boolske variable/søk	Bruker de logiske operatorene AND, OR og NOT til å søke etter like utsagn.
Bitstrenger	Sekvens av en eller flere bits(0110)
Presedens for kvantorer:	$\neg$, ^, v, →, ←→

1.2 Proporsjonale ekvivalenser

Tautologi	Et sammensatt utsagn som alltid er sant.
Selvmotsigelse	Et sammensatt utsagn som aldri er sant.
Tilfredsstillbare utsagn:	Utsagn som verken er tautologi eller selvmotsigelse
Logisk ekvivalente utsagn:	Utsagn som alltid har samme sannhetsverdi. *p* og *q* er logisk ekvivalente hvis og bare hvis *p* ←→ *q* er en tautologi $(p \leftrightarrow q) \equiv \left[(p \rightarrow q) \wedge (q \rightarrow p) \right]$
De Morgans lover:	$\neg(p \vee q) \equiv \neg p \vee \neg q$ og $\neg(p \wedge q) \equiv \neg p \wedge \neg q$

1.3 Kvantorer

Universell kvantifisering:	P(x) er sant for alle x i universet (definisjonsområdet). $\forall xP(x)$
Eksistensiell kvantifisering:	P(x) er sant for minst en x i universet. $\exists xP(x)$
Univers:	”definisjonsområdet” – område der funksjonen er gyldig. (Universe of discourse, domain)
Negasjon:	Det eksisterer ikke…
Motbevis	$\neg\forall xP(x)\Leftrightarrow\exists x\neg P(x)$ og $\neg\exists xP(x)\Leftrightarrow\forall x\neg P(x)$ For å motbevise et utsagn som alltid skal være sant, trenger vi altså bare å finne et moteksempel.

1.4 Nestede kvantorer

Kvantorer som opererer innenfor andre kvantorer, kan beskrives som sirkelreferanser. $!\exists$ er den unike kvantoren, uttrykker at det finnes en og bare en.

1.5 Bevismetoder

Teorem:	Påstand man kan bevise er sann
Bevis:	Serie med logisk følgeriktige påstander som til sammen utgjør et argument.
Aksiomer og postulater:	Underliggende matematiske antakelser
Logisk følgeriktighet:	Fremgangsmåte for å trekke logiske følger som utgjør stegene på vei mot et bevis.
Lemma:	Enkelt teorem brukt til å bevise andre teoremer.
Korollar:	Utsagn som kan sluttes direkte fra et teorem som har blitt bevist.
Konjunksjon:	Påstand man ikke kan vite om er sant eller usant

Regler for inferens

Regler for begrunnelse av logiske resonnement..

Modus ponens: ”Law of detachment”. Tautologien $(p \wedge (p \rightarrow q)) \rightarrow q$ er grunnlaget . Sier at dersom både en implikasjon og dens hypotese er sann så er konklusjonen at implikasjonen er sann.

$$\begin{array}{l} p \\ \underline{p \rightarrow q} \\ \therefore q \end{array}$$

Addisjon:

$$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array} \quad p \rightarrow (p \vee q)$$

Forenkling

$$\begin{array}{l} \underline{p \wedge q} \\ \therefore p \end{array} \qquad (p \wedge q) \rightarrow p$$

Konjunksjon:

$$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array} \quad ((p) \wedge (q)) \rightarrow (p \wedge q)$$

Modus tollens

$$\begin{array}{l} \neg q \\ \underline{p \rightarrow q} \\ \therefore \neg p \end{array} \quad [\neg q \wedge (p \rightarrow q)] \rightarrow \neg p$$

Hypotetisk syllogisme

$$\begin{array}{l} p \rightarrow q \\ \underline{q \rightarrow r} \\ \therefore p \rightarrow r \end{array} \qquad [(p \rightarrow q) \wedge (q \rightarrow r)] \rightarrow (p \rightarrow r)$$

Disjunktiv syllogisme

$$\begin{array}{l} p \vee q \\ \underline{\neg p} \\ \therefore q \end{array} \qquad [(p \vee q) \wedge \neg p] \rightarrow q$$

Resulosjon

$$\begin{array}{l} p \vee q \\ \underline{\neg p \vee r} \\ \therefore q \vee r \end{array} \quad [(p \wedge q) \wedge (\neg p \rightarrow r)] \rightarrow (q \vee r)$$

Inferensregler for kvantorer

Universell instansering: P© er sann for en c innenfor universet, gitt $\forall xP(x)$

Universell generalisering $\forall xP(x)$ er sann gitt at forutsetningen P© er sann for alle elementer c i universet.

Eksistensiell instansering: Gitt at $\exists xP(x)$ er sann, vet vi at det finnes et element c slik at P© er sann.

Eksistensiell generalisering: $\exists xP(x)$ er sann når vi kjenner et element c slik at P© er sann.

Metoder for å bevise teoremer

Direkte bevis: $p \to q$ vises direkte ved å vise at når p er sann må også q være sann.

Indirekte bevis: $\neg q \to \neg p$. Viser at det kontrapositive er sant.

Selvmotsigelser: $p \to (r \wedge \neg r)$. Kan skrives om til et direkte bevis ved å anta p og $\neg q$ og vise $\neg q \to \neg p$. Dette fører da til $p \wedge \neg p$

Vaccous proof: Hvis p er usant, kan vi konkludere at implikasjonen $p \to q$ er sann.

Trivielt bevis: Hvis konklusjonen *q* er sann, vet vi at *p* er sann.

Bevis ved tilfeller: Vi kan vise at $(p_1 \vee p_2 \vee p_3 \vee ... \vee p_n) \to q$ er sant ved å bevise $(p_1 \to q) \wedge (p_2 \to q) \wedge ... \wedge (p_n \to q)$.

Bevis for ekvivalens: Vi beviser "hvis og bare hvis" $p \leftrightarrow q$ ved $(p \to q) \wedge (q \to p)$

Bevis ved induksjon
i) Vis a P er sann for n = k
ii) Anta et P er sann for n = k og vis at dette medfører at P er sann for n = k+1. P vil da være sann for alle n.

Feil i bevis

En argumentrekke er sann hvis og bare hvis alle innbefattede hypoteser er sanne.

Feilslutning (fallacie) Resultat av å trekke ugyldige slutninger

Sirkel-resonnering

1.6 Mengder (sets)

Mengde: En mengde er en samling uordnede elementer

Like mengder: To mengder er like dersom de inneholder de samme elementene. $\forall x\left(x \in A \leftrightarrow x \in B\right)$. Skrives også $A \subseteq B \wedge B \subseteq A$

Delmengder: A er en delmengde av B hvis og bare hvis alle elementene i A også er med i B. $A \subseteq B$

Kardinalitet: Antall elementer i mengden S denoteres $\left|S\right|$.

Potensmengder: Mengden av alle delmengder, dvs. alle uordnede par. P(S). Har kardinalitet $\left|P(S)\right| = 2^{|S|}$

$P(A)=\{B|B\subseteq A\}$

Kartesisk produkt: Mengden av alle ordnede par (a,b).

$AxB = \left\{(a,b)\middle|a \in A \wedge b \in B\right\}$. Kardinalitet $\left|AxB\right| = \left|A\right|x\left|B\right|$

Den tomme mengde: $\varnothing$

1.7 Mengdeoperasjoner

Union: Mengden som inneholder elementer som er i A, B eller begge.

$A \cup B = \left\{x\middle|\left(x \in A\right) \vee \left(x \in B\right)\right\}$

Snitt: Mengden som inneholder elementer som er både i A og B.

$A \cap B = \left\{x\middle|\left(x \in A\right) \wedge \left(x \in B\right)\right\}$

Inklusjon/eksklusjon prinsippet: $\left|A \cup B \cup C\right| = \left|A\right| + \left|B\right| + \left|C\right| - \left|A \cap B\right| - \left|A \cap C\right| - \left|B \cap C\right| + \left|A \cap B \cap C\right|$

Mengdedifferanse: $A - B = A / B\left\{x\middle|x \in A \wedge x \notin B\right\}$ Kalles også komplementet til A med hensyn på B.

Komplement: $\overline{A} = \left\{x\middle|x \notin A\right\}$

Disjunkt: To mengder er disjunkte hvis snittet er den tomme mengden.

De Morgans lov: $\overline{A \cup B} = \overline{A} \cap \overline{B}$ og $\overline{A \cap B} = \overline{A} \cap \overline{B}$

Medlems-tabeller: Kan brukes til å vise identiteter.

Unioner/snitt av flere mengder: $A_1 \cup A_2 \cup ... \cup An = \bigcup_{i=1}^{n} A_i$ og $A_1 \cap A_2 \cap ... \cap An = \bigcap_{i=1}^{n} A_i$

1.8 Funksjoner

Funksjon: Tilordning av nøyaktig ett element fra en mengde til en annen, $A \to B$

Domene/ kodomene: Hvis $f(a) = b$ så er *b* bildet av *a*, og *a* er før-bildet av *b.*

Verdimengde: Mengden av alle bilder

En-til-en/injektiv: Alt i definisjonsmengden treffer noe unikt i verdimengden.
$\left(f(x) = f(y)\right) \to x = y$, $\forall x \forall y (x \neq y \to f(x) \neq f(y))$

På/surjektiv: Alt i verdimengden treffes. $\forall b \exists a \left(f(a) = b\right)$ der $b \in B$ og $a \in A$

Bijektiv: Funksjoner som både er injektive og surjektive. En funksjon kan inverteres hvis og bare hvis den er bijektiv.

Invers: En invers funksjon er slik at når $f(a) = b$ så er $f^{-1}(b) = a$

Komposisjon: Komposisjonen til to funksjoner er $\left(f \circ g\right)(a) = f\left(g(a)\right)$.
Forutsetter at verdimengden til *g* er en del av domenet til *f*.

Gulvfunksjonen: Reelt tall som er det største heltallet mindre eller lik *x*.
$\lfloor x \rfloor = n$ hvis og bare hvis $n \leq x < n + 1$

Takfunksjonen: Reelt tall som er det minste heltallet større eller lik *x*.
$\lceil x \rceil = n$ hvis og bare hvis $n - 1 < x \leq n$

Graf: Grafen til en funksjon *f* er mengden ordnede par
$\left\{(a,b), a \in A, f(a) = b\right\}$

Del 2 – Algoritmer, tall og matriser

2.1 Algoritmer

En algoritme er et endelig antall instruksjoner for å løse et problem.
Viktige begreper er *input, output, presisjon, korrekthet, endelighet, effektivitet, generalitet*.

Søkealgoritmer

Metoder som brukes for å finne elementer i en mengde.

Lineært søk: Sammenligner *x* og a_1. Fortsetter så med a_2 osv. helt til den finner en $a_n = x$.

Binært søk: Brukes på lister som er sortert i stigende rekkefølge. Algoritmen begynner å sammenligne med elementet midt i listen, så splitter den igjen de to dellistene i nye lister osv.

Sorteringsalgoritmer

Ordner elementene i en liste i ønsket rekkefølge.

Bubble sort: Sorterer ved å sammenligne to og to elementer og endre rekkefølgen på disse hvis den er feil. L^2 operasjoner

Innsetting: Begynner med det andre elementet, sammenligner dette med det første og plasserer det foran dersom det er mindre.

2.2 Funksjonsvekst

Big-O: En funksjon f(x) er O(g(x)) dersom det finnes konstanter *C*, *k* > 0, slik at $|f(x)| \leq C|g(x)|$ for alle *x* > *k*.

Dersom

$f_1(x)$ *er* $O(g_1(x))$ og $f_2(x)$er $O(g_2(x))$, så har vi $(f_1 + f_2)$ er $O(\max(g_1(x), g_2(x)))$

og $(f_1 f_2)(x)$ er $O((g_1(x)g_2(x))$

Big-omega $f(x)\Omega(g(x))$ dersom det finnes konstanter *C*, *k* > 0, slik at $C|g(x)| \leq |f(x)|$ for alle *x* > *k*.

Big-theta: En funksjon er $\Theta(g(x))$ dersom den er $O(g(x))$ og $\Omega(g(x))$. *f* er av orden *g*.

2.3 Kompleksiteten til algoritmer

Tidskompleksitet
Beregningskompleksitet
Plasskompleksitet

2.4 Tall og divisjon

Deling: *a* deler *b* hvis det finnes et heltall *c* slik at *b* = *ac*. Når *a* deler *b* sier vi at *a* er en faktor i *b* og *b* er et multiplum av *a.*

(1) Hvis $a|b$ og $a|c$ gir $a|(b+c)$

(2) Hvis a|b, da har vi a|bc for alle heltall c

(3) Hvis a|b og b|c da a|c

Bevis 1

Anta at a|b og b|c. Da finnes det heltall *s* og *t* med *b* = *as* og c = at. Altså b + c = as + at = a(s+t).
Dermed a|(b+c)

Primtall: Tall som kun er delelig med 1 og seg selv

Aritmetikkens fundamental-teorem Alle positive heltall over 1 har entydig primtallsfaktorisering.

Primtalls-faktor Dersom *n* er et sammensatt heltall, vil det ha en primtallsfaktor $\leq\sqrt{n}$
For å finne primtallsfaktoriseringen av et tall kan man altså begynne med roten av tallet.

Bevis:

Hvis *n* er et sammensatt tall har det en faktor *a* med 1<*a*<*n*. Altså

n= *ab* der både *a* og *b* er positive tall større enn 1. Vi ser at

$a\leq\sqrt{n}$ eller b$\leq\sqrt{n}$ ellers ville vi hatt ab<$\sqrt{n}$*$\sqrt{n}$=n. Altså *n* har en positiv divisor mindre enn $\sqrt{n}$, og ved aritmetikkens fundamentalteorem er det enten ett primtall eller har en primtallsdivisor mindre eller lik $\sqrt{n}$.

Metode for primtallsfaktorisering:

Dersom 2^m-1 er et primtall så er *m* et primtall.

Test opp til $\sqrt{n}$ og finn $p|n$

Test opp til $\sqrt{p}$ og finn $q|p$

Test opp til $\sqrt{q}$ til q er et primtall

Divisjons-algoritmen: Gitt to heltall kan man skrive det ene tallet som det andre som går et antall ganger opp i det første, pluss en rest.
$a = dq + r$ og $0 \leq r < d$

Største felles divisor: Det største tallet *d* som er slik at *d*|*a* og *d*|*b* er den største felles divisoren til *a* og *b*. Skrives *gcd(a,b) sfd(a,b)* eller bare *(a,b)=(b,a)*.

$\gcd(a,b)=p_1^{\max(a_1,b_1)}p_2^{\max(a_2,b_2)}\ldots p_n^{\max(a^n,b^n)}$ der *p* er primtallene til *a* og *b*.

Relativt primiske: Heltallene *a* og *b* er relativt primske hvis *gcd(a,b) = 1.*
$a_1, a_2, \ldots, a_n$ er parvis relativt primiske hvis g*cd(a_i,a_j) = 1*.

Minste felles multiplum: Det minste heltallet som deles av *a* og *b*. *lcm(a,b) = mfm(a,b)*

$lcm(a,b)=p_1^{\min(a_1,b_1)}p_2^{\min(a_2,b_2)}\ldots p_n^{\min(a^n,b^n)}$ der *p* er primtallene til *a* og *b*.

*gcd * lcm:* $ab=\gcd(a,b)\cdot lcm(a,b)$

Modulær aritmetikk:
Når *a* og *b* er heltall og *m* er et positivt heltall, da er *a* kongurent med *b* modulo *m* hvis *m* deler *a-b*.
Skrives $a \equiv b \pmod{m}$.

$a \equiv b \pmod{m} \Leftrightarrow a \bmod m = b \bmod m$

Bevis:
Heltallene *a* og *b* er kongurente modulo *m* hvis og bare hvis det finnes et tall *k* slik at $a = b + km$. Dvs. at de har samme rest.
Bevis:
Hvis a≡ b (mod m) da m|(a-b). Dette betyr at det finnes et heltall k slik at

a-b = km, og a = b + km. Tilsvarende hvis det finnes et heltall k slik at

a = b+km, da vil km = a-b. Altså deler m (a-b) slik at a|b(mod m)

Kongurenser kan adderes og multipliseres, samt deles med et tall *c*
dersom *gcd(c,m) = 1*.
Mengden av alle heltall kongurent med et heltall a kalles kongurensklassen til *a* modulo *m*.

2.5 Tall og algoritmer

Eurklids algoritme: Metode til å finne gcd. *La* $a = bq + r$, hvor a, b, q og r er heltall. Da er $\gcd(a,b) = \gcd(b,r)$

Bevis:
Må vise at felles divisor for *a* og *b* er det samme som felles divisorer for *b* og *r*, siden begge par da vil ha den samme største felles divisor. Anta at *d* deler både *a* og *b*. Da følger det at *d* også deler *a – bq = r*. Det vil si at en felles divisor for *a* og *b* også er en felles divisor for *b* og *r*. Tilsvarende anta at *d* deler både *b* og *r*. Da deler *d* også *bq + r = a*. Altså bil en felles divisor for *b og* r også være felles divisor for *a* og *b*.

Den største felles divisor finnes som den siste divisor som ikke er null i Euklids algoritme.
Eks. finn gcd(414,662):

$\gcd(662,414)$:

662 = 414 *1 +248

414=248*1+166

248=166*1+82

166=82*2+2

82=41*2+0

Dvs gcd(414,662) = 2

2.6 Anvendelse av tallteori

Lineær kombinasjon: Hvis *a* og *b* er positive heltall, finnes det heltall *s* og *t* slik at *gcd(a,b) = sa + tb.*
Brukes ved å utføre Euklids algoritme baklengs

Deling: Hvis *a*, *b* og *c* er positive heltall slik at gcd(a,b) = 1 og *a*|*bc*, da har vi *a*|*c.*

Generalisering mhp primtall: Hvis *p* er et primtall og $p|a_1a_2\cdots a_n$ hvor alle *ai* er et heltall, da har vi $p|a_i$ for *i*.

Invers: Hvis *a* og *m* er relativt primiske finnes det et tall $\bar{a}$ slik at $a\bar{a}\equiv 1 (\mathrm{mod} m)$. Inversen finnes ved lineære kombinasjoner og Euklids algoritme, og kan brukes til å løse ligninger på formen $ax\equiv b(\mathrm{mod} m)$, hvor en løsning er garantert dersom *gcd(a,m) = 1*. En invers er garantert dersom a og m er relativt primske og m>1.

Det kinesiske restteoremet: Et system av kongurenser:

$$x \equiv a_1 (\mathrm{mod}\, m_1)$$

$$x \equiv a_2 (\mathrm{mod}\, m_2)$$

$$\vdots$$

$$x \equiv a_n (\mathrm{mod}\, m_n)$$

har en unik løsning modulo *m* hvis m_1, m_2, …, m_n er parvis relativt primiske. Løsningen er på formen $x=a_1M_1y_1+a_2M_2y_2+\cdots+a_nM_ny_n$ der y_n er inverser av M_n modulo m_n, samt at $M_n = m/m_n$

Fermats lille setning: Hvis *p* er et primtall og *a* er et heltall som ikke deles av *p*, har vi:

$$a^{p-1} \equiv 1 (\mathrm{mod}\, p)$$

$$a^{p} \equiv a (\mathrm{mod}\, p)$$

2.7 Matriser

Matrise En *m x n matrise* er en tabell med *m* rader og *n* kolonner. To matriser er like dersom de har samme antall rader og kolonner, og tilsvarende innhold på hver posisjon.

$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}$$

Summering

La $A = \left[a_{ij}\right]$ og $B = \left[b_{ij}\right]$ være m x n matriser. Da er summen av A og B m x n matrisen som har $a_{ij} + b_{ij}$ som sitt (i,j) element. Dvs. $A + B = \left[a_{ij} + b_{ij}\right]$

Multiplikasjon

La A være en m x k matrise og B en k x n matrise. Produktet av A og B (AB) er m x n matrisen med innholdet i (i,j) tilsvarende summen av produktene av de korresponderende elementene fra den i'te raden i A og den j'te kolonnen i B.

Dvs. $AB = \left[c_{ij}\right]$ der $c_{ij} = a_{i1}b_{1j} + a_{i1}b_{2j} + \cdots + a_{ik}b_{kj}$

$$\begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} x \begin{pmatrix} m'_{11} & m'_{12} \\ m'_{21} & m'_{22} \end{pmatrix} =$$

$$\begin{pmatrix} (m_{11} * m'_{11} + m_{12} * m'_{21}) & (m_{11} * m'_{12} + m_{12} * m'_{22}) \\ (m_{21} * m'_{11} + m_{22} * m'_{21}) & (m_{21} * m'_{12} + m_{22} * m'_{22}) \end{pmatrix}$$

Del 3 – Matematiske resonnementer, induksjon og rekursjon

3.1 Bevisstrategi

a. Skriv om det som skal bevises ved å erstatte med definisjonene.
b. Analyser hva hypotesen og konklusjonen innebærer
c. Bruk en av bevismetodene i 1.5 til å bevise utsagnet. Begynn med direkte bevis.

Forlengs eller *baklengs resonnering*

3.2 Sekvensering og summasjon

Sekvens: Diskret struktur som brukes for å representere en ordnet liste. En sekvens er en funksjon fra en delmengde av heltall til en mengde *S.*

$a: \mathbb{N} \to S,\ \mathbb{Z}^+ \to S$

Geometrisk progresjon: En sekvens på formen $a, ar, ar^2, \cdots, ar^n$ der *a* og *r* er reelle tall.

Aritmetisk progresjon: En sekvens på formen $a, a+d, a+2d, \cdots, a+nd$ der *a* og *r* er reelle tall.

Summasjon: Hvis *a* og *r* er reelle tall og $r \neq 0$ da

$$\sum_{j=0}^{n} ar^j = \begin{cases} \dfrac{a\left(1-r^{n+1}\right)}{1-r} & \text{for r} \neq 1 \\ (n+1)a & \text{for r=1} \end{cases}$$

Kardinalitet: Mengdene *A* og *B* har samme kardinalitet hvis og bare hvis det er en en-til-en korrespondanse fra *A* til *B*.

Tellbarhet: En mengde som enten er endelig eller som har kardinalitet lik mengden av positive heltall kalles *tellbar*.

$\{\mathbb{R}\}$ *ikke tellbar:* Kantors bevis. Mengden av reelle tall er ikke tellbar. Bevises med motbevis, antar at mengden er tellbar og finner et tall som ikke er med i den oppsatte listen.

3.3 Matematisk induksjon

Induksjon: Vis først *P(n)* for en spesifikk *n*, vis deretter at dette medfører *P(n+1)* for alle *n*.

Sterk induksjon: Viser $[P(1) \wedge P(2) \wedge \cdots \wedge P(n)] \to P(n+1)$

Kalles også det andre prinsippet i matematisk induksjon.

Bevis for induksjon:

Anta at vi vet at *P(1)* er sann, og at utsagnet $P(k) \to P(k+1)$ er sant for alle positive heltall *k*. For å vise at *P(n)* er sann for alle heltall, anta at det er minst en *n* som gjør *P(n)* usann. Da vil mengden *S* av positive heltall hvor *P(n)* er usann, ikke være tom. Vi kaller elementet i *S* som *m*. Vi vet at *m* ikke kan være 1, siden *P(1)* er sann. Siden *m* er positiv og større enn en, så er *(m-1)* et positivt heltall. Og siden *(m-1)* er mindre enn *m*, finnes den ikke i *S*, så *P(m-1)* må være sann. Siden implikasjonen $P(m-1) \to P(m)$ også er sann, da må *P(m)* være sann. Dette er en selvmotsigelse av *m*. Altså er *P(n)* sann for alle positive heltall *n*.

Velordnings-prinsippet: Alle ikke-tomme mengder har et minste element

3.4 Rekursive definisjoner og strukturell induksjon

Rekursjon: Et objekt kan defineres rekursivt ut fra seg selv

For eksempel kan et tall være definert av de foregående tallene i en tallfølge. Får *f(0)* og prosedyre for *f(n)*.

Fibonacci-tallene: $f_0, f_1, f_2, \cdots,$ er definert ved ligningene $f_0 = 0, f_1 = 1$ og $f_n = f_{n-1} + f_{n-2}$ for $n > 2$

Lamés teorem: Antall steg brukt av Euklids algoritme for å finne *gcd(a,b)* er mindre eller lik fem ganger desimallengden av *b (*a<b).
Eks. *b=10*, steg: 5.

Trær med rot: Grunnsteg: rot, rekursive steg for greiner

Utvidet binært tre med rot: Grunnsteg: $\varnothing$, rekursivt steg: høyre og/eller venstre deltre

Fullt binært tre: Alle indre hjørner har to greiner.

3.5 Rekursive algoritmer

Rekursiv algoritme: En algoritme som løser et problem ved å redusere det til en utgave med lavere innverdier.

Iterasjon: Starter på *n - n -1* og jobber seg oppover til *n*.

Del 4 – Telling

4.1 Grunnleggende telling

Produktregelen: Hvis en oppgave kan brytes ned i to oppgaver, der det er n_2 måter å gjøre den andre oppgaven etter den første, som kan gjøres på n_1 måter, er totalt antall måter å utføre hele oppgaven på $n_1 n_2$.

Sumregelen: Hvis to oppgaver kan utføres på hhv. n_1 og n_2 måter, og de ikke kan utføres samtidig, er måter å gjøre oppgaven på n_1+n_2. Dette kan utvides til å gjelde flere oppgaver.

Inklusjon-eksklusjon-prinsippet: Ser på kardinaliteten til mengder: $|A \cup B| = |A| + |B| - |A \cap B|$

4.2 Skuffeprinsippet

Skuffeprinsippet: Hvis $k+1$ objekter plasseres i k skuffer vil en skuff minst inneholde to objekter.

Det generaliserte skuffeprinsippet: Hvis n objekter plasseres i k skuffer vil det være en skuff som inneholder minst $\lceil n/k \rceil$ objekter.

Viktig skufferesultat: Hver følge av n^2+1 unike reelle tall vil ha en delfølge av lengde $n+1$ som enten er strengt voksende eller strengt avtakende.

4.3 Permutasjoner og kombinasjoner

En permutasjon på en mengde distinkte objekter er et ordnet utvalg av disse objektene. Et ordnet utvalg av r elementer av en mengde kalles en r-permutasjon.

Ordnede utvalg $P(n,r) = n(n-1)(n-2)\cdots(n-r+1) = \frac{n!}{(n-r)!}$

Uordnede utvalg $\frac{\text{Antall ordnede utvalg}}{\text{Antall måter å sortere}} = \frac{P(n,r)}{P(r,r)} = \binom{n}{r} = nCr = \frac{n!}{r!(n-r)!}$

Kombinatorisk bevis: Bruker telleteknikker for å bevise et teorem, i motsetning til for eksempel algebraiske teknikker

4.4 Binomialkoeffisienter

Binomialkoeffeisent: Antall r-kombinasjoner fra en mengde med n elementer $\binom{n}{r}$ dukker opp som koeffisienter i utvidelsen av potenser i binomiale uttrykk som $(a+b)^n$

$$(x+y)^n = \sum_{j=0}^{n} \binom{n}{j} x^{n-j} y^j = \binom{n}{0} x^n + \binom{n}{1} x^{n-1} y + \binom{n}{2} x^{n-2} y^2 + \cdots + \binom{n}{n} y^n$$

Binomialt uttr.: Sum av to argument, for eksempel $(x+y)$

Del 6 – Avanserte tellemetoder

6.1 Rekursjonsrelasjoner

Rekursjonsrelasjonen for en følge $\{a_n\}$ er en ligning som uttrykker a_n ved et eller flere av de foregående leddene.

La $\{a_n\}_{n=1}^{\infty}$ være en følge. En rekurensrelasjon for $\{a_n\}$ er en ligning som uttrykker a_n som et "uttrykk" i $a_1, a_2, ..., a_{n-1}$. En følge som tilfredsstillerrekursjonsrelasjonen blir kalt en løsning.

Tårnet i Hanoi Løses ved $H_n = 2H_{n-1} + 1$ steg

6.2 Løsning av rekursjonsrelasjoner

En lineær homogen rekursjonsrelasjon av grad *k* med konstante koeffisienter er en relasjon på formen: $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$.

Disse løses på følgende måte:

Karakteristisk ligning $r^2 - c_1 r - c_2 = 0$ har to distinkte røtter r_1 og r_2. Da er sekvensen $\{a_n\}$ en løsning til rekurensrelasjonen $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ hvis og bare hvis $an = \alpha_1 r_1^n + \alpha_2 r_2^n$ for $n \in \mathbb{N}$ for α_1 og α_2 konstanter

6.5 Prinsippet for inklusjon/eksklusjon

La A_1, A_2,… , A_n være endelige mengder. Da

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{1 \le i \le n} |A_i| - \sum_{1 \le i \le j \le n} |A_i \cap A_j| +$$

$$\sum_{1 \le i \le j \le k \le n} |A_i \cap A_j \cap A_k| - \cdots + (-1)^{n+1} |A_1 \cap A_2 \cap \cdots \cap A_n|$$

Del 7 – Relasjoner

7.1 Relasjoner og deres egenskaper

Binær relasjon — *A* og *B* er mengder. En binær relasjon fra *A* til *B* er delmengden av *A* x *B*. Altså mengden ordnede par. Denne kan representeres ved grafer eller matriser.

Refleksiv — En relasjon *R* på en mengde *A* er refleksiv dersom det for alle $a \in A$ så er $(a,a) \in R$. Ellers *irrefleksibel*

Symmetrisk — En relasjon R på en mengde *A* er symmetrisk dersom $(a,b) \in R$ når $(b,a) \in R$.

$\forall (a,b) \exists (b,a)$

Antisymmetrisk — En relasjon *R* på en mengde *A* er antisymmetrisk dersom $(a,b) \notin A$ når $(b,a) \in A$, unntatt hvis $a = b$

$\exists (a,b) \nexists (b,a) \mid a = b$ ok

Asymmetrisk — En relasjon *R* på en mengde *A* er asymmetrisk dersom $(a,b) \in R$ medfører $(b,a) \notin R$

$\exists (a,b) \nexists (b,a) \mid a \neq b$

Transitiv — En relasjon *R* på en mengde *A* er transitiv dersom $(a,b),(b,c) \in R$ medfører $(a,c) \in R$

$\forall (a,b) \forall (b,c) \exists (a,c)$

$R^n \subseteq R \mid n \in 1,2,\ldots,n$

Kombinasjoner av relasjoner — To relasjoner *R* fra *A* til *B* og *S* fra *B* til *C* kan kombineres til en ny relasjon $S \circ R$ fra *A* til *C*. For paret (2,3) i *R* og (3,1) i *S* gir dette (2,1) i $S \circ R$.

Eksponering av relasjoner — $R^1 = R$, og $R^{n+1} = R^n \circ R$

7.2 n-ære relasjoner

n mengder

$R = M_1 \times M_2 \times \cdots \times M_n$ er en *n*-ær relasjon. *n* er graden og M_1, M_2 er definisjonsområdet/domenet til relasjonen. Brukes mye i databaser.

Projection — Plukker ut elementer fra de opprinnelige mengdene

Join — Sammenligningsoperatoren

7.3 Representasjon av relasjoner

Matriser $M_R = \left[m_{ij} \right]$. Rutenett med *i* rader og *j* kolonner

Måte å skrive informasjon om en relasjon.
R er refleksiv hvis diagonalen i grafen kun har 1'ere.
R er symmetrisk hvis og bare hvis $M_R = \left(M_R\right)^t$ - flip'n'check

Grafer En *rettet* graf består av en mengde *V* hjørner og *E* kanter
R en relasjon på *A*. *A* vises som hjørner, *R* som kanter.
R er refleksiv hvis det er en *løkke* på hvert hjørne
R er symmetrisk hvis det er en pil i begge retninger fra alle *V*.
R er antisymmetrisk hvis det er pil kun en vei
R er transitiv hvis man kan hoppe to steg med en pil fra et *V*.

7.4 Tillukninger av relasjoner

Refleksiv tillukning $R \cup \Delta$ der $\Delta = \{(a,a) | a \in A\}$ Legger til de par som mangler

Symmetrisk *tillukning* $R^{-1} = (b,a) | (a,b) \in R$

Sti i grafer Rekkefølge av kanter. Det er en vei av lengde *n* fra *a* til *b* bare hvis $(a,b) \in R^n$

Sammenkoblings *relasjon:* Oppretter en sti fra *a* til *b*. $R^* = \bigcup_{n=1}^{\infty} R^n$

Transitiv tillukning: Unionen av alle mulige veier i relasjonen. Lik R^*

7.5 Ekvivalensrelasjoner

R som er refleksiv, symmetrisk og transitiv. Delelighet gir alltid ekvivalensrelasjoner.

Ekvivalens-klassen Mengden av alle elementer som er relatert til et element $a \in A$
Skrives $[a]_R$ eller $[a]$
aRb

Ekvivalente utsagn: $[a]=[b]$,
$[a] \cap [b] \neq \varnothing$

7.6 Delvise ordninger

R er refleksiv, antisymmetrisk og transitiv

Delvis ordnet mengde En mengde *S* sammen med en delvis ordning *R* kalles en delvis ordnet mengde – poset, og denoteres *(S,R).* Delvis ordnet mengde $(A, \preceq)$

Kompatible Elementer *a* og *b* i en delvis ordnet mengde er kompatible dersom $a \preceq b$ eller $b \preceq a$

Total ordning Alle mengder er kompatible
$(\mathbb{Z}, \leq)$ er totalt ordnet siden $a \leq b \vee b \leq a \mid a,b \in \mathbb{Z}$
$(\mathbb{Z}^+, |)$ er ikke totalt ordnet siden det inneholder ikkekompatible elementer som 5 og 7

Velordnet $(S, \prec)$ er velordnet dersom det har delvis ordnet mengde slik at $\prec$ er den delvise ordningen og at alle ikke-tomme mengder av S har et minste element

Minimalt element: $a \in A$ er et minimalt element dersom det ikke finnes $b \in A$ slik at

$b \prec a$.

Minste element $a \in A$ er det minste elementet hvis $a \preceq b$ for alle $b \in A$.

Øvre/nedre skranke Et element som er større/mindre enn alle andre elementer i en delmengde A av den delvise ordningen $(S, \prec)$

Største/minste øvre/nedre skranke Element som er større/mindre enn alle andre øvre/nedre skranker for A.

Hasse-diagram : Forenkling av den rettede grafen. (1) Fjern alle løkker. (2) Fjern alt som er med på grunn av transitivitet. (3). Fjern alle piler

Gitter/lattice En delvis ordnet mengde hvor alle par har både en minste øvre skranke og en største nedre skranke.

Del 8 – Grafer

8.1 Grafer

Enkel graf $G=(V,E)$ har ingen rettede kanter (piler), løkker eller parallelle kanter.

Mulitgraf: Som enkel graf, men parallelle kanter tillates. Altså funksjon *f* fra *E* til $\{\{u,v\}|u,v\in V\wedge u\neq v\}$

Psudograf Som multigraf, men tillatter løkker. Altså *f* til E der kravet om $u\neq v$ bortfaller.

Rettet graf: Alle kanter har retning. Ingen løkker eller paralleller ok.

Rettet mulitgraf: Kanter med retning, paralleller ok.

8.2 Grafterminologi

Nabohjørner: Hjørnene *u* og *v* er naboer dersom det er en kant mellom dem

Grad Graden til et hjørne *v (deg v)* er antall kanter som berører *v*. Løkker teller dobbelt.

La G = (V,E) være en urettet graf med e kanter. Da har vi:

Håndhilsingsteoremet: $2e=\sum_{v\in V}\deg(v)$

Urettet graf En urettet graf har et partall antall hjørner av odd grad.

Rettet graf For rettede grafer har vi *inn-grad (deg⁻ v)* og *ut-grad (deg⁺ v)*.

Komplette grafer: *n* hjørner, og alle par hjørner er bundet sammen av en kant

Sykler C_n $n\geq 3$

Hjul W_n Fremkommer ved å legge til et hjørne (i midten) av en syklisk graf.

Todelte grafer *(*Bipartite graphs)
En enkel graf *G* som kan partisjoneres i to disjunkte mengder V_1 og V_2 Slik at alle kanter i grafen binder sammen et hjørne i V_1 og et hjørne i V_2 (slik at ingen kanter i G binder sammen hjørner i V_1 eller V_2.)

8.3 Representasjon av grafer. Isomorfi

Isomorfi To grafer er isomorfe hvis det finnes en bijeksjon *f* fra V_1 til V_2 slik at *a* og *b* er naboer i G_1 hvis og bare hvis *f(a)* og *f(b)* er naboer i G_2. Sjekk: samme antall hjørner og kanter, samme grader.

Krav: samme antall kanter og hjørner, samt tilsvarende grader.

Nabomatriser: Angir hvilke hjørner som er naboer med hverandre. For enkle grafer må denne være symmetrisk. For multigrafer får vi ikke 0-1-matriser.

Berøringsmatriser: en matrise $M=[m_{ij}]$ der

$$m_{ij}=\begin{cases}1 \text{ når kanten } e_j \text{ berører hjørnet } v_i\\ 0 \text{ ellers}\end{cases}$$

8.4 Koblinger

Veier	La G=(V,E,f) være en ikkerettet graf og la $n \in N$. En vei av lengde n mellom u og v er en følge av kanter e_1, e_2, slik at $f(e_1)=\{u_1,v_1\}$, $f(e_2)=\{u_2, v_2\}$, $f(e_n)=\{u_{n-1},v\}$ En liste av hjørner i en graf. Veien er enkel hvis den ikke inneholder samme hjørnet flere ganger.
Sammenkoblet G	Det finnes en vei mellom alle hjørner i *G*.
Sterk sammenkoblet	Det finnes veier i begge retninger mellom alle hjørner.
Svakt sammenkoblet	*G* er sammenkoblet som en rettet graf.

8.5 Euler- og Hamilton veier

Eulerkrets	Enkel krets som inneholder alle kanter i en graf En koblet multigraf har en Eulerkrets hvis og bare hvis alle hjørner har partall grad.
Eulervei	Enkel vei som inneholder alle kanter i en graf En koblet multigraf har en Eulervei hvis og bare hvis den har nøyaktig to hjørner av odd grad
Hamilton-krets	Krets som inneholder hvert hjørne kun en gang
Hamilton-vei	Vei som inneholder hvert hjørne kun en gang

Del 9 – Trær

9.1 Trær

Tre — En sammenkoblet urettet graf uten enkle kretser. En urettet graf er et tre hvis og bare hvis det er en unik vei mellom to hjørner. Et tre har *foreldre, barn, blader, forfedre, etterfølgere* og *interne hjørner.*

m-ært tre — Tre med rot der hvert interne hjørne mark har *m* barn.

n hjørner — Et tre med *n* hjørner har *n-1* kanter

Fullt m-ært tre — Dersom hvert hjørne har nøyaktig *m* barn.
i interne hjørner og $n = mi + 1$ kanter
n hjørner gir $i = (n-1)/m$ interne hjørner og $l = [(m-1)n+1]/m$ løv
i interne hjørner gir $n = mi+1$ hjørner og $l=(m-1)i+1$ løv
l løv har $n = (ml-1)(m-1)$ hjørner og $i = (l-1)/(m-1)$ interne kanter

Høyde og antall løv — Et *m*-ært tre av høyde *h* har maksimalt m^h løv.

9.3 Traversering av trær

Universell adresse:
1. Roten har adresse 0, hvert barn på nivå 1 adresseres 1, 2, 3, …, k
2. For hvert hjørne *v* på nivå *n* med adresse *A*, adresseres barna fra venstre til høyre med *A.1, A.2, …, A.k.*

Preorder traversal — Besøker roten, så subtre T_1,så T_2 i preorder helt til T_n.

Inorder traversal — Besøker subtre T_1 helt til venstre, så rot, så andre subtrær fra venstre mot høyre.

Postorder traversal — Besøker subtrær fra venstre mot høyre, til slutt roten

9.4 Overspennende trær

Overspennende trær — La *G* være en enkel graf. Et overspennende tre til *G* er et tre som inneholder alle hjørnene i *G*. Dvs. multiple kanter fjernes fra enkle kretser.

Del 10 – Boolsk algebra

10.1 Boolske funksjoner

Boolsk algebra Gir operasjoner og regler for arbeidet med mengden $\{0,1\}$.

Komplement Komplementet til $\bar{0}=1$ og $\bar{1}=0$

Boolsk sum $1+1=1,\ 1+0=1,\ 0+1=1,\ 0+0=0$

Boolsk produkt $1*1=1,\ 1*0=0\ ,0*1=0,\ 0*0=0$

Boolsk funksjon En funksjon fra B^n til B har grad n, der $B=\{0,1\}$

Antall boolske funksjoner av grad n er 2^{2^n}

Dualer Dualer F^d av F ved å bytte + med * og 0 med 1

Boolske identiteter:

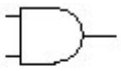
Loven om det dobbelte komplement	$\bar{\bar{x}} = x$	
Idempotentlover	x+x = x	x*x = x
Identitetslover	x + 0 = x	x * 1 = x
Dominasjonslover	x + 1 = 1	x * 0 = 0
Kommutative lover	x+y = x+y	xy=yx
Assosiative lover	x+(y+z) = (x+y)+z	x(yz) = (zy)x
Distributive lover	x+yz=(x+y)(x+z)	x(y+z) = xy + xz
De Morgans lover	$\left(\overline{xy}\right) = \bar{x} + \bar{y}$	$\overline{(x+y)} = \bar{x}\bar{y}$
Absorbsjonslover	x+xy = x	x(x+y) = x
Enhetsegenskap	$x+\bar{x} = 1$	
Nullegenskap	$x\bar{x} = 0$	

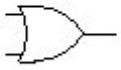
10.2 Representasjon av boolske funksjoner

Literal Boolsk variabel eller dens komplement

Minterm Boolsk produkt av n literaler.

10.3 Logiske porter

OG-port  x,y → xy

ELLER-port  x,y → x+y

Inverter  $x \rightarrow \bar{x}$

104. Minimalisering av kretser

Må finne frem til boolske uttrykk med færre operatorer, dvs. minimere de boolske funksjonene.

Del 11 – Funksjonsmaskiner

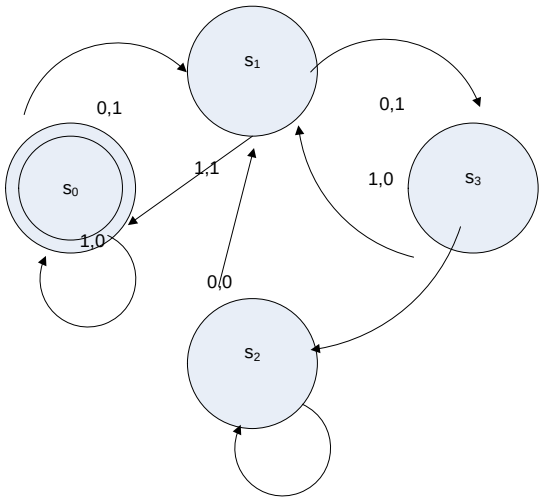
11.1 Språk og grammatikker

Grammatikk (G = V, T, S, P) og består av vokabular, terminaler, startsymbol og produksjonsregler
Tom streng: λ

Språk Mengden ord dannet av en grammatikk

11.2 Endelige tilstandsmaskiner (med output)

En endelig tilstandsmaskin med output $M=(S,I,O,f,g,s_0)$ består av tilstander *S*, starttilstand s_0, inputalfabet *, outputalfabet O*, inputfunksjon *f,* outputfunksjon *g* samt en startfase s_0.



Tilstand	f	g
	Input 0 1	Output 0 1
s0	s_1 s_0	1 0
s1	s_3 s_0	1 1
s2	s_1 s_2	0 1
s3	s_2 s_1	0 1

1.1.3 Endelige tilstandsautomater (uten output)

En endelig tilstandsmaskin *M = (S, I, f, s_0, F)* består av tilstander *S*, inputalfabet *I*, inputfunksjon *f*, starttilstand s_0, og endelige tilstander *F*.

Deterministisk Gir ut en endelig tilstand *F*

Ikke-deterministisk Kan gi ut en mengde tilstander (også den tomme mengde)
Språket *L* som gjenkjennes av en ikke-deterministisk endelig tilstandsmaskin vil også bli gjenkjent av en deterministisk tilstandsmaskin.

11.4 Språkgjenkjenning

Regulære uttrykk Regulære uttrykk over et alfabet *I* er definert som:
$\varnothing$ er et regulært uttrykk
λ er et regulært uttrykk
x er et regulært uttrykk når $x \in I$
(AB), (A U B) og A^* er alle regulære uttrykk

En mengde er regulær hvis og bare hvis den kan gjenkjennes av en endelig tilstandsautomat.