

# Woodworkers handbook on trees

Anton Augustsson

June 2020



# Innehåll

<b>1 basic-course-in-mathematics - Formel Samling</b>	<b>7</b>
1.1 Basic . . . . .	8
1.1.1 Mängder . . . . .	8
1.1.2 Potenslagar . . . . .	8
1.1.3 Logaritmer . . . . .	8
1.1.4 Tillvägagångs sätt . . . . .	8
1.1.5 Intervall . . . . .	9
1.1.6 Tillvägagångs sätt . . . . .	9
1.2 Komplexa tal . . . . .	10
1.2.1 Polärform . . . . .	10
1.3 Absolut Belopp . . . . .	12
1.3.1 Tillvägagångs sätt . . . . .	12
1.4 Summor . . . . .	13
1.4.1 Aritmetiska summor . . . . .	13
1.4.2 Geometriska summor . . . . .	13
1.4.3 Tillvägagångs sätt . . . . .	13
1.5 Kombinatorik . . . . .	14
1.5.1 Multiplikations principen . . . . .	14
1.5.2 Kombinationer . . . . .	14
1.5.3 Binomial satsen . . . . .	14
1.6 Funktioner och kordinatsystem . . . . .	15
1.6.1 Avståndsformeln . . . . .	15
1.6.2 Elipser . . . . .	15
1.7 Polynom division . . . . .	16
1.7.1 Tillvägagångs sätt . . . . .	16
1.8 Trigonometri . . . . .	17
1.8.1 Tillvägagångs sätt . . . . .	18
<b>2 PKD Sammanfatning</b>	<b>19</b>
2.1 Coding convention . . . . .	20
2.2 Design approach . . . . .	20
2.2.1 Process . . . . .	20
2.3 Recursion . . . . .	21
2.3.1 Recursion types . . . . .	21
2.4 Complexity . . . . .	21
2.4.1 Growth . . . . .	21
2.5 Recurrences . . . . .	21
2.5.1 Closed Form . . . . .	22
2.6 Higher-Order Function . . . . .	24
2.6.1 Higher-Order Functions on Lists . . . . .	24
2.7 Data types . . . . .	25
2.7.1 Basic . . . . .	25

2.7.2	Maybe Type . . . . .	25
2.7.3	New types and enumeration types . . . . .	25
2.7.4	Inductive Data Types . . . . .	28
2.7.5	Trees . . . . .	29
2.7.6	Other data types . . . . .	31
2.7.7	Graphs . . . . .	33
2.8	Important syntax . . . . .	36
2.8.1	Let . . . . .	36
2.8.2	IO . . . . .	36
2.9	Sorting Algorithms . . . . .	36
<b>3</b>	<b>Algebra 1 - Formel Samling</b>	<b>37</b>
3.1	logik . . . . .	38
3.1.1	värde tabeller . . . . .	38
3.1.2	Implikationer . . . . .	38
3.2	Mängder . . . . .	40
3.3	Bevis . . . . .	41
3.3.1	Induktions bevis . . . . .	41
3.3.2	Motsägelse bevis . . . . .	41
3.4	Delbarhet . . . . .	42
3.4.1	Största Gemensama Delaren (SGD) . . . . .	42
3.4.2	Primtal . . . . .	44
3.5	Diofantiska ekvationer . . . . .	45
3.6	Talbaser . . . . .	46
3.6.1	konvertera från decimal bass till annan bas . . . . .	46
3.6.2	konvertera från annan bas till decimal bas . . . . .	46
3.6.3	Andra exempel . . . . .	46
3.7	Functioner . . . . .	47
3.7.1	Inversen . . . . .	47
3.7.2	Relatioiner . . . . .	47
3.8	Summor . . . . .	48
3.8.1	Aritmetiska summor . . . . .	48
3.8.2	Geometriska summor . . . . .	48
3.9	Kongruensräkning . . . . .	49
3.10	Kardinalitet . . . . .	50
3.10.1	Uppräkneligamängder . . . . .	50
3.11	Polynom . . . . .	51
3.11.1	Polynom division . . . . .	51
3.11.2	Faktorsatsen . . . . .	52
<b>4</b>	<b>Single Variable Calculus 1 - Formel Samling</b>	<b>55</b>
4.1	Basic . . . . .	56
4.1.1	Mängder . . . . .	56
4.1.2	Intervall . . . . .	57
4.1.3	Funktion . . . . .	58
4.1.4	Trigonometri . . . . .	59
4.1.5	Exempel: Trigonometri . . . . .	60
4.2	Gränsvärden . . . . .	61
4.2.1	Kontinuitet . . . . .	62
4.3	Derivator . . . . .	63
4.3.1	Kjedje regeln . . . . .	63
4.3.2	L'Hôpital's rule . . . . .	64
4.3.3	Medelvärdessatsen . . . . .	64
4.3.4	Rolle . . . . .	64

4.3.5	växande funktioner . . . . .	65
4.3.6	Högreordnings deviator . . . . .	65
4.3.7	Impericit derivering . . . . .	65
4.3.8	invers funktioner . . . . .	65
4.3.9	exponetial och logaritm . . . . .	66
4.3.10	odefinerad form . . . . .	66
4.3.11	inversa trigometriska funktioner . . . . .	67
4.4	Grafritning . . . . .	68
4.5	Optemering . . . . .	70
4.6	Talfölder och serier . . . . .	71
4.6.1	Serier med varierande tecken . . . . .	73
4.6.2	Potensserier . . . . .	73
4.6.3	Taylor serier . . . . .	75
4.7	Integraler . . . . .	77
4.7.1	variabelsubstition . . . . .	79
4.7.2	Integration av rationella funktioner . . . . .	80
4.7.3	Partiell integration . . . . .	82
4.7.4	Generaliseringande integraler . . . . .	83
4.7.5	Volymberäkningar . . . . .	85
4.8	Differential ekvationer . . . . .	87
4.8.1	superabla ekvationer . . . . .	88
4.8.2	Linjära differentialekvationer av ordning 1 . . . . .	89
4.8.3	Linjära differentialekvationer av ordning 2 . . . . .	90
<b>5</b>	<b>Computer Architecture Cheat Sheet</b>	<b>95</b>
5.1	ISA1 . . . . .	96
5.1.1	MIPS instructions . . . . .	96
5.1.2	Sequencing . . . . .	98
5.1.3	Converge to Assambly code . . . . .	99
5.2	ISA2 . . . . .	99
5.2.1	MIPS type format . . . . .	99
5.2.2	Procedure . . . . .	99
5.2.3	Other ISA . . . . .	101
5.3	Arithmetic . . . . .	102
5.3.1	Binary numbers . . . . .	102
5.3.2	Negative integers . . . . .	102
5.3.3	Oporations . . . . .	102
5.3.4	Non integer numbers (Floating and fixed point) . . . . .	102
5.3.5	Overflow . . . . .	103
5.4	Logic . . . . .	104
5.5	processor control and datapath . . . . .	106
5.5.1	Clock . . . . .	108
5.5.2	Critical path . . . . .	108
5.6	pipline . . . . .	109
5.7	Pipline hazards . . . . .	111
5.7.1	Data hazards . . . . .	111
5.7.2	Control hazards . . . . .	111
5.7.3	Structural hazards . . . . .	112
5.8	Predicting Branches and Exceptions . . . . .	113
5.8.1	Static pridictors . . . . .	113
5.8.2	Dynamic pridictors . . . . .	114
5.8.3	Exceptions . . . . .	116
5.9	Input/Output . . . . .	117
5.10	Cache . . . . .	119

5.10.1 Memory hierarchy . . . . .	119
5.10.2 Cache misses 3C's . . . . .	120
5.11 Virtual Memory . . . . .	121
5.12 Parallism . . . . .	124
5.12.1 Multicore . . . . .	124
5.12.2 Paralel programing . . . . .	124
5.12.3 Syncronaziation . . . . .	124
5.12.4 Cache coherency . . . . .	125
5.12.5 ILP . . . . .	126

# Kapitel 1

## basic-course-in-mathematics - Formel Samling

## 1.1 Basic

### 1.1.1 Mängder

Naturliga tal:  $\mathbb{N} = \{1, 2, 3, \dots\}$

Heltal:  $\mathbb{Z} = \{\dots, -2, 1, 0, 1, 2, \dots\}$

Rationella tal:  $\mathbb{Q} = \left\{ \frac{a}{b} \mid a, b \in \mathbb{Z}, b \neq 0 \right\}$

Irrationella tal:  $\mathbb{P} = \mathbb{R} \setminus \mathbb{Q}$  eller  $\{x \mid x \in \mathbb{R}, x \notin \mathbb{Q}\}$

Reella tal:  $\mathbb{R} = \mathbb{P} \cup \mathbb{Q}$

### 1.1.2 Potenslagar

$$\begin{aligned} \left(\frac{a}{b}\right)^{-3} &= \left(\frac{b}{a}\right)^3 \\ \sqrt{a} &= a^{\frac{1}{2}} \end{aligned}$$

### 1.1.3 Logaritmer

#### Logaritmlagar:

Va vaksam över när  $a < 0$  då är logarytmen odefinerad.

$$(1): b = a^x \Leftrightarrow \log_a(b) = x \text{ för: } a > 0, b > 0, a \neq 1$$

$$(2): \log_a\left(\frac{b}{c}\right) = \log_a(b) - \log_a(c)$$

$$(3): \log_a(b * c) = \log_a(b) + \log_a(c)$$

$$(4): \log_a(b^d) = d \log_a(b)$$

$$(5): \log_a(b) = \frac{\log_f(b)}{\log_f(a)}$$

$$(6): \log_a(a) = 1$$

$$(7): \log_a(1) = 0$$

$$(8): a^{\log_a(x)} = x$$

$$(9): \log_{a^c}(b) = \frac{1}{c} \log_a(b)$$

### 1.1.4 Tillvägagångs sätt

$$\log_3(x) + \log_x\left(\frac{1}{27}\right) = 2 \tag{1.1}$$

**Lösning**

$$\begin{aligned}
 \log_3(x) - 3\log_x(3) &= 2 \\
 \log_3(x) - 3\frac{\log_3(3)}{\log_3(x)} &= 2 \\
 \log_3(x) - 3\frac{1}{\log_3(x)} &= 2 \\
 \log_3(x)^2 - 3 &= 2\log_3(x) \\
 y &= \log_3(x)^2 \\
 y^2 - 2y - 3 &= 0 \\
 \text{pq-formeln: } y &= 1 \pm \sqrt{4} = 1 \pm 2 \\
 \log_3(x) = 3 &\Leftrightarrow x = 3^3 = 27 \\
 \log_3(x) = -1 &\Leftrightarrow x = 3^{-1} = \frac{1}{3}
 \end{aligned}$$

**1.1.5 Intervall**

$$[a, b] = \{x | a \leq x \leq b\} \quad (1.2)$$

$$[a, \infty[ \quad (1.3)$$

$$]-\infty, \infty[ \quad (1.4)$$

**1.1.6 Tillvägagångs sätt**

$$\frac{2}{x-3} < \frac{5}{x} \quad (1.5)$$

**Lösning**

$$\begin{aligned}
 \frac{2}{x-3} - \frac{5}{x} &< 0 \\
 \frac{(x)2}{x(x-3)} - \frac{5(x-3)}{x(x-3)} &< 0 \\
 \frac{2x - 5x + 15}{x(x-3)} &< 0 \\
 \frac{-3x - 15}{x(x-3)} &< 0 \\
 \frac{-3(x+5)}{x(x-3)} &< 0 \\
 x \neq 0, x \neq 3
 \end{aligned}$$

Värde tabell:

	$x < 0$	$0 < x < 3$	$3 < x < 5$	$5 < x$
$x-5$	-	-	-	+
-3	-	-	-	-
x	-	+	+	+
$x-3$	-	-	+	+
hela	+	-	+	-

## 1.2 Komplexa tal

$$z = a + bi$$

$$\operatorname{Re}(z) = a$$

$$\operatorname{Im}(z) = b$$

$$\bar{z} = a - bi$$

$$|z| = \sqrt{a^2 + b^2}$$

Där  $b(\operatorname{Im}(z))$  är för y-axeln och  $a(\operatorname{Re}(z))$  är för x-axeln

Samma räkneregler för reella tal gäller för komplexa tal

### 1.2.1 Polärform

$$\arg(z) = \alpha + 2\pi * n$$

$\arg(z)$  är vinkeln mellan a (x-axeln) linjen  $|z|$ .  $n$  är heltal

$$\operatorname{Arg}(z) \in ]-\pi, \pi[$$

$$\operatorname{Arg}(z) \in \arg(z)$$

$$a = |z| \cos \alpha$$

$$b = |z| \sin \alpha$$

$$z = |z| \cos \alpha + i * |z| \sin \alpha$$

**Polärform:**

$$z = |z|(\cos \alpha + i * \sin \alpha) \quad (1.6)$$

**Eulers formel:**

$$z = |z|e^{i\alpha} \quad (1.7)$$

**Sats:**

$$|z * w| = |z| * |w|$$

$$\arg(z * w) = \arg(z) + \arg(w)$$

$$\arg\left(\frac{z}{w}\right) = \arg(z) - \arg(w)$$

**De Movre's formel:**

$$(\cos(\theta) + i * \sin(\theta))^n = \cos(n * \theta) + i * \sin(n * \theta)$$

### Binomisk ekvation

$$(\cos(\theta) + i * \sin(\theta))^n = \cos(n * \theta) + i * \sin(n * \theta)$$

### Tillvägagångs sätt

$$\text{Visa lösningarna i det komplexa tal planet } z^5 = \sqrt{3} + i \quad (1.8)$$

$$\begin{aligned} |\sqrt{3} + i| &= \sqrt{\sqrt{3}^2 + 1^2} = \sqrt{4} = 2 \\ \begin{cases} \sqrt{3} = 2 \cos(\alpha) \\ 1 = 2 \sin(\alpha) \end{cases} \\ \alpha &= \frac{\pi}{6} \\ |z|^5 (\cos(5 * \theta) + i * \sin(5 * \theta)) &= 2(\cos(\frac{\pi}{6}) + i * \sin(\frac{\pi}{6})) \end{aligned}$$

Vilket ger:

$$\begin{aligned} |z|^5 &= 2 \Leftrightarrow |z| = 2^{\frac{1}{5}} \\ 5 * \theta &= \frac{\pi}{6} + 2\pi n \Leftrightarrow \theta = \frac{\pi}{30} + \frac{2\pi n}{5} \quad n \in \mathbb{Z} \end{aligned}$$

I polärform blir det då:

$$z = 2^{\frac{1}{5}} \left( \cos \frac{\pi}{30} + \frac{2\pi n}{5} + i * \sin \frac{\pi}{30} + \frac{2\pi n}{5} \right)$$

Eulers formel:  $z = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30} + \frac{2\pi n}{5})}$

$$n = 0, 1, 2, 3, 4$$

$$z_1 = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30})}$$

$$z_2 = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30} + \frac{2\pi}{5})}$$

$$z_3 = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30} + \frac{4\pi}{5})}$$

$$z_4 = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30} + \frac{6\pi}{5})}$$

$$z_5 = 2^{\frac{1}{5}} e^{i(\frac{\pi}{30} + \frac{8\pi}{5})}$$

### 1.3 Absolut Belopp

#### 1.3.1 Tillvägagångs sätt

$$|2x - 8| + |1 - x| - 2|x - 3| = 8 + 3x \quad (1.9)$$

lösning

$$\left( \begin{array}{l} 2x - 8 = 0 \\ x = 4 \end{array} \right) \left( \begin{array}{l} 1 - x = 0 \\ x = 1 \end{array} \right) \left( \begin{array}{l} x - 3 = 0 \\ x = 3 \end{array} \right)$$

$$\text{I } \begin{cases} x \leq 1 \\ -(2x - 8) + (1 - x) + 2(x - 3) = 8 + 3x \end{cases}$$

$$\text{I } \begin{cases} x \leq 1 \\ -4x = 5 \end{cases} \quad \begin{cases} x \leq 1 \\ x = -\frac{5}{4} \end{cases} \text{ lösning}$$

$$\text{II } \begin{cases} 1 < x \leq 3 \\ -(2x - 8) - (1 - x) + 2(x - 3) = 8 + 3x \end{cases}$$

$$\text{II } \begin{cases} 1 < x \leq 3 \\ -2x = 7 \end{cases} \quad \begin{cases} 1 < x < 3 \\ x = -\frac{7}{2} \end{cases} \text{ Ej lösning}$$

$$\text{III } \begin{cases} 3 \leq x < 4 \\ -(2x - 8) - (1 - x) - 2(x - 3) = 8 + 3x \end{cases}$$

$$\text{III } \begin{cases} 3 \leq x \leq 4 \\ -(2x - 8) - (4x - 3) = 8 + 3x \end{cases} \quad \begin{cases} 3x < 4 \\ x = \frac{5}{6} \end{cases} \text{ Ej lösning}$$

$$\text{IV } \begin{cases} x \geq 4 \\ (2x - 8) - (1 - x) - 2(x - 3) = 8 + 3x \end{cases}$$

$$\text{IV } \begin{cases} x \geq 4 \\ x = -\frac{11}{2} \end{cases} \text{ Ej lösning}$$

## 1.4 Summor

### 1.4.1 Aritmetiska summor

$$s_n = a_1 + a_2 + a_3 + \dots + a_n = \frac{n(a_1 + a_n)}{2} \quad (1.10)$$

### 1.4.2 Geometriska summor

Börjar alltid med exponenten 0 och gör om summan så att den passar i följande talföljd:

$$s_n = a + ak + ak^2 + \dots + ak^{n-1} = \frac{a(k^n - 1)}{k - 1} \quad (1.11)$$

### 1.4.3 Tillvägagångs sätt

$$\sum_{k=n}^{2n} (2^k - k) \quad (1.12)$$

#### Lösning

sätter  $f = 0 = k - n$

$$\begin{aligned} \sum_{f=0}^n (2^{f+n} - (f+n)) &= 2^n * \sum_{f=0}^n (2^f) - \sum_{f=0}^n (f+n) \\ \frac{2^n(2^{n+1} - 1)}{2 - 1} - \frac{3n(n+1)}{2} &= 2^{2n+1} - 2^n - \frac{3n(n+1)}{2} \end{aligned}$$

## 1.5 Kombinatorik

### 1.5.1 Multiplikations principen

permetationer  $p(a,b)$  då ordningen spelar roll.

Antal sätt: (exemplet: antal sätt av måltider 7 företer 5 varmrätter 4 efterätter  $(7 \cdot 5 \cdot 4)$ )

$$n_1 \cdot n_2 \cdot \dots \cdot n_m \quad (1.13)$$

### 1.5.2 Kombinationer

Kombination  $c(a,b)$  då ordningen inte spelar roll.

Antal sätt: (exemplet: antal del-mängder två element ( $n =$  element  $k =$  antal element som kan väljas))

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad (1.14)$$

$$\binom{n}{k} = \frac{n \cdot (n-1)(n-2) \cdot \dots \cdot (n-(k-1))}{k!} \quad (1.15)$$

### Pascal triangel

$n$							
0	1						
1	1	1					
2	1	2	1				
3	1	3	3	1			
4	1	4	6	4	1		
5	1	5	10	10	5	1	
6	1	6	15	20	15	6	1
	0	1	2	3	4	5	6
					$k$		

### 1.5.3 Binomial satsen

$$(a+b)^n = \sum_{k=0}^n \binom{n}{k} a^k b^{n-k} \quad (1.16)$$

## 1.6 Funktioner och kordinatsystem

Kom ihåg att när det stor  $(x-a)$  förflytas den i x-axeln a steg till höger  $\rightarrow$  medans  $(x+a)$  förflytas den a steg till vänster  $<-$

### 1.6.1 Avståndsformeln

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (1.17)$$

### 1.6.2 Elipser

Förenkla till denna formeln:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (1.18)$$

Där a är avståndet på x-axeln och b är avståndet på y-axeln

### Hyperbol

Ser väldigt anerlunda ut från elipser

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad (1.19)$$

### Circkel

$$\frac{x^2}{a^2} + \frac{y^2}{a^2} = 1 \quad (1.20)$$

Där a = r

## 1.7 Polynom division

### 1.7.1 Tillvägagångs sätt

Man vet att ekvationen  $z^4 - 2z^3 - 7z^2 + 26z - 20 = 0$  har roten  $z = 2 + i$ . Lös ekvationen fullständigt. (1.21)

$$\begin{aligned} z &= 2 + i \quad \text{Är en läsning är också konjugatet en lösning enligt faktorsatsen } \bar{z} = a - bi \\ z &= 2 \pm i \end{aligned}$$

Vilket betyder att följande går att factorisera ut polynomet

$$(z - (2 + i))(z - (2 - i)) = z^2 - 4z + 5$$

**Långdivision (liggande stolen):**

$$\begin{array}{r} x^2 + 2x - 4 \\ \hline x^2 - 4x + 5) \overline{x^4 - 2x^3 - 7x^2 + 26x - 20} \\ \underline{-x^4 + 4x^3 - 5x^2} \\ \hline 2x^3 - 12x^2 + 26x \\ \underline{-2x^3 + 8x^2 - 10x} \\ \hline -4x^2 + 16x - 20 \\ \underline{4x^2 - 16x + 20} \\ \hline 0 \end{array}$$

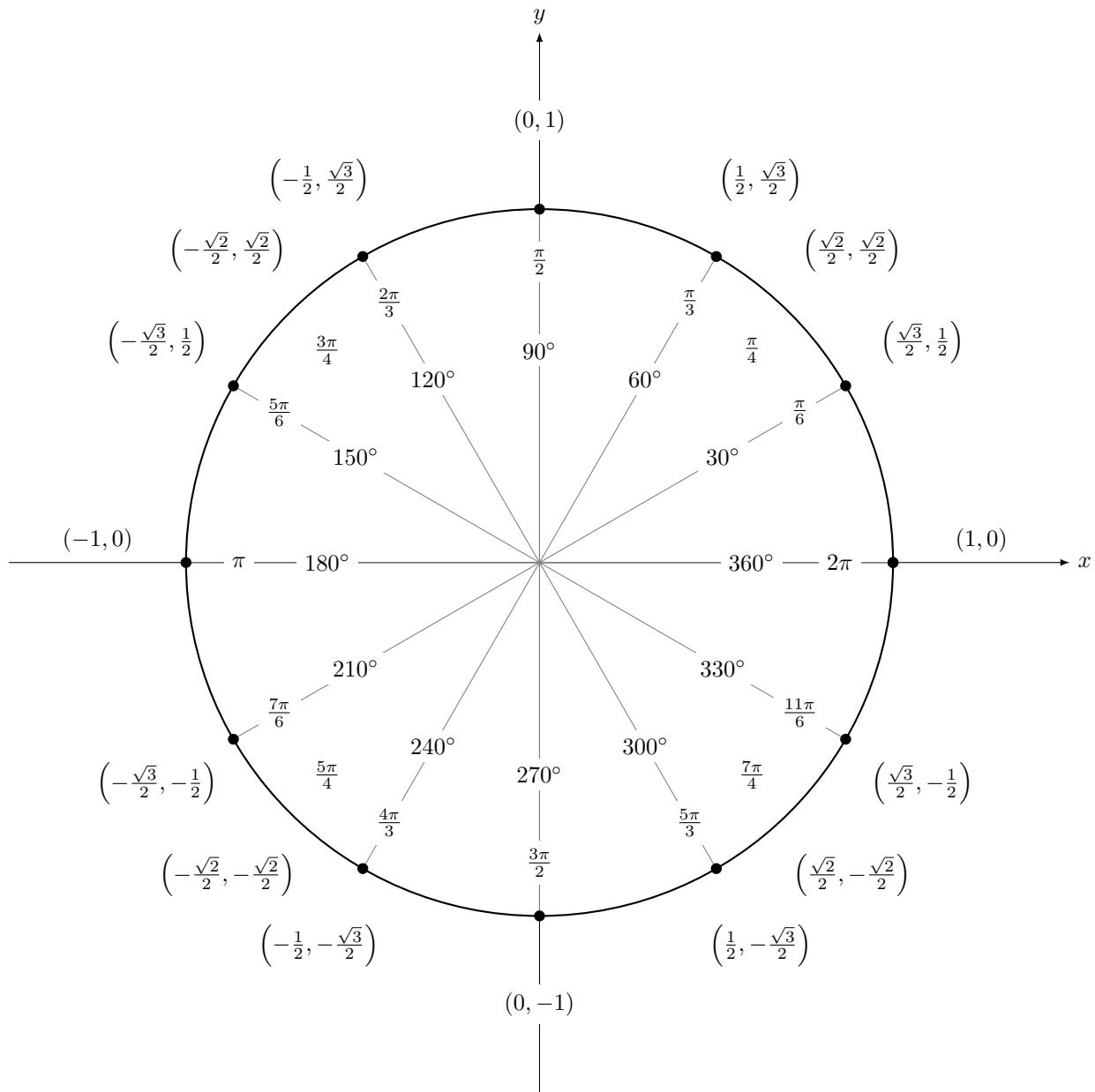
$z^2 + 2z - 4 = 0$  Är också en läsning som till slut ger följande

$$z = -1 \pm \sqrt{5}$$

$$z = 2 \pm i$$

Varge n grads polynom har alltid n stycken komplexa lösningar

## 1.8 Trigonometri



**Sats:**

$$360^\circ = 2\pi \text{rad}$$

$$v_g = v_r * \frac{180^\circ}{\pi}$$

$$v_r = v_g * \frac{\pi}{180^\circ}$$

**Sats:**

$$-1 \leq \sin t \leq 1$$

$$-1 \leq \cos t \leq 1$$

**Sats:**

$$\cos(-t) = \cos(t)$$

$$\sin(-t) = -\sin(t)$$

$$\tan(-t) = \frac{\sin(-t)}{\cos(-t)} = \frac{-\sin(t)}{\cos(t)}$$

**Additionsformlerna:**

$$\sin(\alpha + \beta) = \sin(\alpha)\cos(\beta) + \sin(\beta)\cos(\alpha)$$

$$\sin(\alpha - \beta) = \sin(\alpha)\cos(\beta) - \sin(\beta)\cos(\alpha)$$

$$\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\beta)\sin(\alpha)$$

$$\cos(\alpha - \beta) = \cos(\alpha)\cos(\beta) + \sin(\beta)\sin(\alpha)$$

**Trigonometriska ettan:**

$$(\sin t)^2 + (\cos t)^2 = 1$$

$$\sin^2 t + \cos^2 t = 1$$

### 1.8.1 Tillvägagångs sätt

$$\begin{aligned} \cos \frac{\pi}{12} &= \cos \left( \frac{\pi}{3} - \frac{\pi}{4} \right) = \cos \frac{\pi}{3} \cos \frac{\pi}{4} + \sin \frac{\pi}{3} \sin \frac{\pi}{4} \\ &= \left( \frac{1}{2} \right) \left( \frac{1}{\sqrt{2}} \right) + \left( \frac{\sqrt{3}}{2} \right) \left( \frac{1}{\sqrt{2}} \right) = \frac{1 + \sqrt{3}}{2\sqrt{2}} \end{aligned}$$

## Kapitel 2

# PKD Samanfatning

## 2.1 Coding convention

### VARIANT

A (recursive) function terminates if it has a variant. The variant need to follow all of the flowing rules

- Needs to decrease every recursive call
- All ways positive

### Side effects

All IO functions have side effects in order to separate pure Haskell function with impure functions that changes the state with is commonly the case with imperative and object oriented programming. Every IO function has a side effects.

### INVARIANT

A data types invariant is what value are allowed for the data type to work. Similar to preconditions for a function. An example is integer data type that can only use positive integers therefor the invariant is positive integers.

## 2.2 Design approach

- top-down design (Cheating): Is to break down a complex system in to subsystems to solve the problem.  
Most often is to write everything by scratch.
- bottom-up design (Stacking): Is to pieces existing system together to create a more complex system.  
little is programmed, most is copied.
- dodging: Get some code working more quickly, make progress with some part of the system and back-paddle to the dodged part later. The reason is to develop insight that will help solve the larger problem.

### 2.2.1 Process

1. Data Description
2. Data Examples
3. Function Description
4. Function Examples
5. Function Template
6. Code
7. Tests
8. Review and Refactor

**Programming to an Interface** More dynamic, can change models, less code to write and a layer of abstraction. ADT

## 2.3 Recursion

### 2.3.1 Recursion types

1. Simple recursion: There is at most one recursive call (in each branch) and the argument is decremented by one.
2. Complete recursion: Some argument becomes smaller in the recursive call, but not necessarily.
3. Multiple recursion: There are multiple recursive calls (in the same branch).
4. Mutual recursion: Two or more functions are defined in terms of each other.
5. Nested recursion: An argument to a recursive call is computed by a recursive call.
6. Recursion on a generalized problem: Sometimes, no suitable recursion scheme is obvious.

## 2.4 Complexity

### 2.4.1 Growth

1. Big  $\theta$  Notation: estimate of growth in intervals determine by constants Definition For non-negative functions  $f$  and  $g$ ,  $f(n) = \theta(g(n))$  if and only if there exist  $n_0 \geq 0$  and  $c_1, c_2 > 0$  such that for all  $n > n_0$   $c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$ .  $\theta(g(n))$  is the set of all functions  $f(n)$  that are bounded below and above
2. Big  $\Omega$  Notation: estimate of growth Lower bound
3. Big  $O$ : Notation: estimate of growth upper bound

### Relation

$$O(g(n)) \cap \omega(g(n)) = \theta(g(n))$$

## 2.5 Recurrences

### Example:

sumList [] = 0  
 sumList (x:xs) = x + sumList xs

1. pattern matching [] takes  $t_0$  time
2. pattern matching ( $x : xs$ ) takes  $t_1$  time
3. Adding  $x$  with recursive call takes  $t_{add}$
4. Then the recursive call takes  $T(n - 1)$

$$T(n) = \begin{cases} t_0 & \text{if } n = 0 \\ T(n - 1) + t_{add} + t_1 & \text{if } n > 0 \end{cases}$$

### 2.5.1 Closed Form

1. Use the substitution method to obtain a closed form for the following recurrence:

$$\begin{aligned}f(0) &= 5 \\f(n) &= f(n - 1) + n + 2, n > 0\end{aligned}$$

**Hint:**  $1 + 2 + 3 + \dots + n = \frac{n(n+1)}{2}$  — you do not need to prove this fact.

#### Expansion Method

$$\begin{aligned}f(0) &= 5 \\f(1) &= f(0) + n + 2 = n + 5 + 2 \\f(2) &= f(1) + n + 2 = 2n + 5 + 2 + 2 \\f(3) &= f(2) + n + 2 = 3n + 5 + 2 + 2 + 2 \\f(n) &= +n^2 + 5 + n \cdot 2\end{aligned}$$

#### Induction proof

Step1: test with base case sense the base case is predefine for 0 we do  $n = 1$

$$\begin{aligned}f(1)_{VL} &= 1^2 + 5 + 1 \cdot 2 = 8, \\f(1)_{HL} &= f(0) + 1 + 2 = 5 + 3 = 8 \\f(1)_{VL} &= f(1)_{HL}\end{aligned}$$

Step2: assumption for p  $f(p) = p^2 + 5 + p \cdot 2$

$$\begin{aligned}f(p+1)_{VL} &= f(p) + (p+1) + 2 = (p^2 + 5 + p \cdot 2) + (p+1) + 2 = \\&= (p^2 + (p+1)) + 5 + (p+1) \cdot 2 = (p^2 + 2p + 1) + 5 + (p+1) \cdot 2 \\f(p+1)_{HL} &= (p+1)^2 + 5 + (p+1) \cdot 2 = (p^2 + 2p + 1) + 5 + (p+1) \cdot 2 \\f(p+1)_{VL} &= f(p+1)_{HL}\end{aligned}$$

Conclusion: according to induction hypothesis the recurrence of the function is equal to

$$2n + 5 + \frac{n(1+n)}{2}$$

### Substitution Method

$$\begin{aligned}
f(n) &= f(n-1) + n + 2 \\
&= (f(n-2) + (n-1) + 2) + 1n + 2 \\
&= f(n-2) + (n-1) + n + 2 \cdot 2 \\
&= (f(n-3) + (n-2) + 2)(n-1) + n + 2 \cdot 2 \\
&= f(n-3) + (n-2) + (n-1) + n + 3 \cdot 2 \\
&\quad \vdots \\
&= f(n-k) + (n-(k-1)) + (n-(k-2)) + (n-(k-3)) + \dots + n + k \cdot 2 \\
&\quad \vdots \\
&= f(n-n) + (n-(n-1)) + (n-(n-2)) + (n-(n-3)) + \dots + n + n \cdot 2 \\
&= f(0) + 1 + 2 + 3 + \dots + n + n \cdot 2 = 5 + 1 + 2 + 3 + \dots + n + n \cdot 2
\end{aligned}$$

We can see the following patterns

$$2n + 5 + \sum_{k=1}^n (k) = 2n + 5 + \frac{n(1+n)}{2}$$

### Induction proof

Step1: test with base case sense the base case is predefine for 0 we do  $n = 1$

$$f(1)_{VL} = 2 \cdot 1 + 5 + \frac{1(1+1)}{2} = 2 + 5 + 1 = 8,$$

$$f(1)_{HL} = f(0) + 1 + 2 = 5 + 3 = 8$$

$$f(1)_{VL} = f(1)_{HL}$$

Step2: assumption for p  $f(p) = 2p + 5 + \frac{p(1+p)}{2}$

$$f(p+1)_{HL} = f(p) + (p+1) + 2 = (2p + 5 + \frac{p(1+p)}{2}) + (p+1) + 2 =$$

$$= 2(p+1) + 5 + \frac{p(1+p)}{2} + p = 2(p+1) + 5 + \frac{2(p+1) + p(1+p)}{2} =$$

$$= 2(p+1) + 5 + \frac{p^2 + 2p + p + 2}{2} =$$

$$f(p+1)_{HL} = 2(p+1) + 5 + \frac{(p+1)(p+2)}{2} = 2(p+1) + 5 + \frac{p^2 + 2p + p + 2}{2}$$

$$f(p+1)_{VL} = f(p+1)_{HL}$$

Conclusion: according to induction hypothesis the recurrence of the function is equal to

$$2n + 5 + \frac{n(1+n)}{2}$$

## 2.6 Higher-Order Function

### 2.6.1 Higher-Order Functions on Lists

```
map :: (a -> b) -> [a] -> [b]
filter :: (a -> Bool) -> [a] -> [a]
foldl :: (a -> b -> a) -> a -> [b] -> a
foldr :: (a -> b -> b) -> b -> [a] -> b
```

#### **map**

maps a function to each element in list.

#### **filter**

filters elements with a condition

```
filter :: (a -> Bool) -> [a] -> [a]
filter (<6) [6,3,0,1,8,5,9,3] = [3,0,1,5,3]
```

#### **foldl**

foldl recurses over a list “from the left,” i.e., it initially applies the given operation to the first list element and the given start value. starts from the left (first element) and apply the function to with each element. Similar to an accumulator. No one uses it since it is somewhat useless.

```
foldl :: (a -> b -> a) -> a -> [b] -> a
foldl (*) 1 [1,2,3,4] = 24
```

#### **foldr**

foldr recurses over a list “from the right,” i.e., it initially applies the given operation to the last list element and the given start value. starts from the right (last element) and apply the function to with each element. Similar to an accumulator.

```
foldr :: (a -> b -> b) -> b -> [a] -> b
foldr (*) 1 [1,2,3,4] = 24
```

## 2.7 Data types

### 2.7.1 Basic

```
String :: ['char'] --list of characters
List :: []          --undefined element types and elements
Tuple :: ()         --Predefine elements
Char :: ''          --single character
Int :: 1            --Hole number with define size
Integer :: 1         --Hole number with undefined size
Float :: 1.1         --Real number with double-precision
Double :: 1.1        --Real number with single-precision
```

### 2.7.2 Maybe Type

If the return is maybe “nothing” then the “Maybe type” is used, since it does not have to return a specific value. It is not polymorphic since you have to specify the type, however “Just” is at its self polymorphic function. If a operation that requires a specific type one needs to remove “Just”, for instance by a let function.

### 2.7.3 New types and enumeration types

New types: One creates more relevant names and formats of existing enumeration types. Overload is a problem with the use of the same operations that can not be used for the same data types. One needs to create new operations if it is not from the same type class.

Enumeration types: Instead of new types *type* enumeration types uses the operation call *data*. The difference is that enumeration types is independent from existing types therefor one becomes more flexible and precise.

#### example

```
data newTypeOfDataDerection = North | South | East | West
    deriving (Show) -- inorder to print

:t North
North :: newTypeOfDataDerection

-- We can use new types in pattern matching
oposit :: newTypeOfDataDerection -> newTypeOfDataDerection
oposit North = South
```

#### Type classes

A type class is a set of types that support certain related operations.

No function is applied by default to the new type, therefore you can write “deriving” the following functions are good to have

#### type classes to deriving for new data types

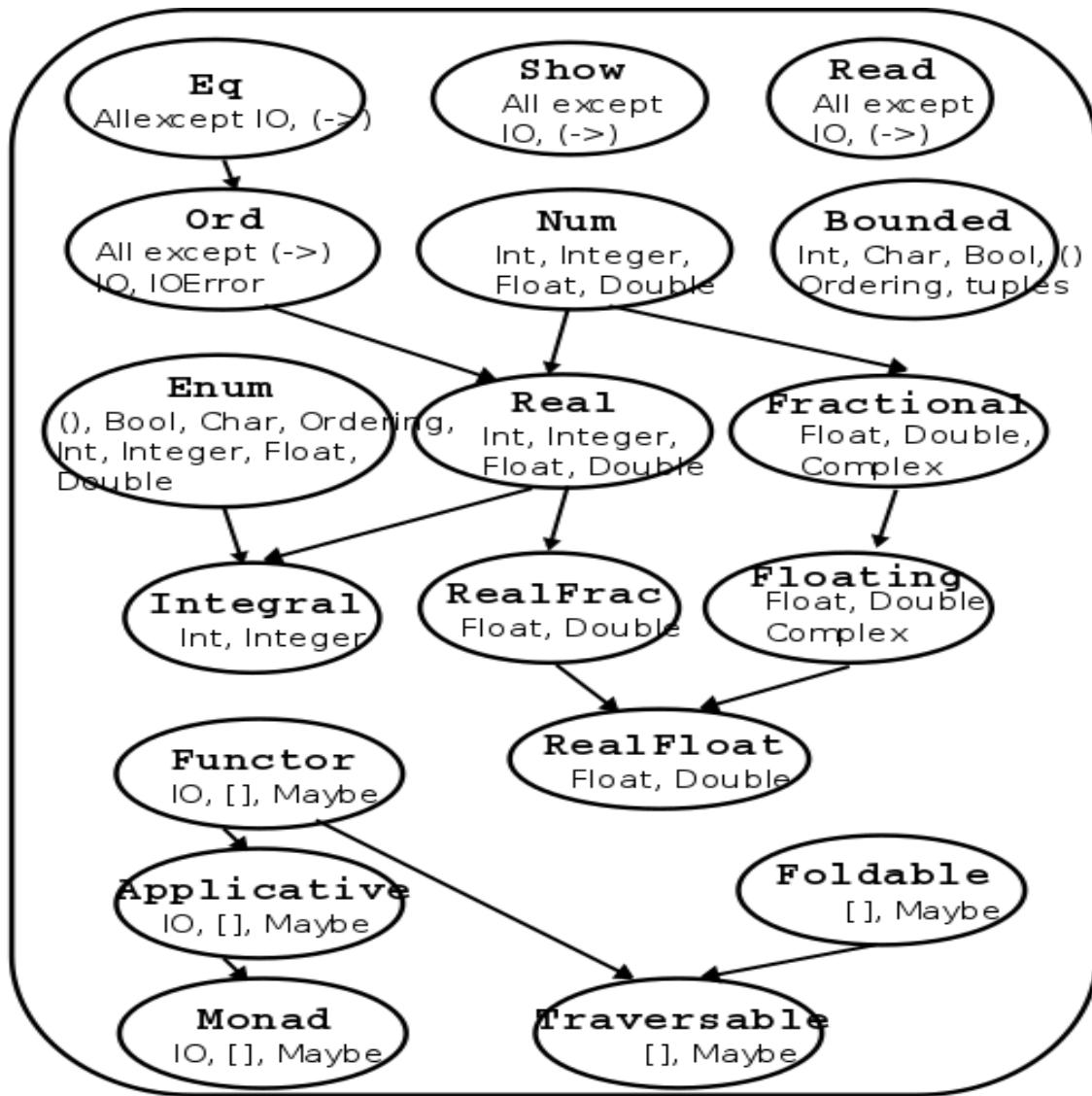
```
deriving (Show) -- in order to print in ghci and print normal
deriving (Eq) -- To test equality
deriving (Ord) -- the first one has the smallest value, order matter for comparison
```

#### New type classes

```
class Eq a where
  (==) :: a -> a -> Bool
```

**New type classes Instances**

```
instance Eq Colour where  
  (==) = eqColour
```



Figur 2.1: Type class

### 2.7.4 Inductive Data Types

Uses a base case then the inductive step as cases of. Multiple arguments. Type constructure

```
data AExp = Atom Int
          | Plus AExp AExp
          | Times AExp AExp

eval (Atom i)    = i
eval (Plus a b) = eval a + eval b
eval (Times a b) = eval a * eval b
```

### 2.7.5 Trees

#### Terminology

1. Search tree: All nodes on the left side is less than the parent and opposite on the right side
2. Out-degree: is how many children it has. Binary trees has out-degree 0 – 2
3. root Node: is a parent to any number of children at the top of the hierarchy
4. sub Node: is a parent to any number of children
5. Leaf: is a node that has no children
6. Node: every element
7. Height: most steps from the root node to a leaf

#### Representation

1. Inorder: (Left, Root, Right)
2. Pre-order: (Root, Left, Right)
3. Post-order: (Left, Right, Root)

```
data FBTree a = Leaf a
| Node (FBTree a) a (FBTree a)
deriving (Show)

Node (Leaf 1) 5 (Leaf 21)
rootValue :: FBTree a -> a
rootValue (Node _ a _) = a
rootValue (Node x) = x

height :: FBTree a -> Int
height (Leaf _) = 0
height (Node b a c) = 1 + max (height b) (height c)
```

#### Binary tree

Insertion:  $O(1)$  ( $O(h)$  if search tree)  
 Delition:  $O(n)$  ( $O(h)$  if search tree)  
 Search:  $O(n)$  ( $O(h)$  if search tree)  
 Height:  $O(n)$  (n nodes)

1. Full binary tree: has node out-degree either 2 or 0 worst-case complexity of  $O(\log n)$ .
2. Binary tree: each node has up to an out-degree of 2

#### Binary search tree

Insertion:  $O(h)$  ( $O(n)$  if search tree)  
 Delition:  $O(h)$   
 Search:  $O(n)$   
 Height:  $O(n)$  (n nodes)

### Red and black trees

Insertion:  $O(\log n)$

Deletion:  $O(\log n)$

Search:  $O(\log n)$

Element:  $O(2^r)$  (rank r)

Height:  $O(2 \cdot \log_2(n + 1))$  (n nodes)

Better then a normal binary tree since it balance the tree, therefor it becomes smaller and more efficient to search in. One should use red and black tree when there is a large number of nodes, say 50.

Definition: A red-black tree is a binary search tree where every node is colored either red or black, with the following balancing invariants:

1. No red node has a red parent.
2. Every path from the root to an empty subtree contains the same.
3. A red-black tree with n nodes has height at most  $2 \cdot \log 2(n + 1)$ .
4. there are 4 cases to rebalance (1;4) is similar so is (2;3).

### Algorithm

1. Perform a standard binary-search-tree insertion.
2. Color the new node red.
3. Rebalance the tree, if there is a red node with a red parent.

### Binomial Trees and heaps

Insertion:  $O(\log n)$

Search:  $O(\log n)$  (number of trees n)

Element:  $2^r$  (rank r, n=1 then r=0)

A heap can be used to implement a priority queue, where elements are added to a pool and assigned a priority. In a min-priority queue, extraction of an element yields an element with minimum priority. The smallest node is the root and every child is equal or larger then its parent.

Binomial Trees is a data structure by linking trees of rank  $r - 1$  together. A binomial heap (Vuillemin, 1978) is a list of binomial trees such that each tree satisfies the min-heap property (hence the root of each tree contains its minimum key); and the trees have strictly increasing ranks.

### Terminology

1. Link: putting together two trees.
2. Merge: putting together two heaps
3. Binomial Heap: a list (forest!) of Binomial Trees
4. Binomial Trees have the largest subtree to the left, while Binomial

### Binomial heaps

1. Heaps, extracting minimum element in worst case  $O(\log |h|)$
2. Binomial Trees, The height (here: number of edges on the longest branch) is  $r$ .
3. Binomial Trees, There are  $2^r$  nodes in the tree.
4. Binomial Trees, There are  $\binom{r}{k}$  nodes at level  $k$ . (Hence its name!)
5. Binomial Trees, The root has  $r$  subtrees of ranks  $r - 1, r - 2, \dots, 1, 0$ .
6. A binomial heap  $h$  has at most  $\lceil \lg |h| \rceil + 1$  binomial trees.
7. Inserting a binomial tree into a binomial heap is like addition with base 2.
8. merging is made with two cases either (case 1) when one is smaller or (case 2) when they are equal

`BinoTree = Node Int Int [BinoTree] — Node rank key (subtrees with decreasing rank)`

### 2.7.6 Other data types

#### Tables, Stacking and queuing

1. Table: a list of key-value pairs
2. Stacks: elements accessed in Last-In First-Out (LIFO) order
3. Queues: elements accessed in First-In First-Out (FIFO) order

#### Table operations

```
empty :: Table k v
insert :: Eq k => Table k v -> k -> v -> Table k v
exists :: Eq k => Table k v -> k -> Bool
lookup :: Eq k => Table k v -> k -> Maybe v — value from key
delete :: Eq k => Table k v -> k -> Table k v
iterate :: Table k v -> (b -> (k, v) -> b) -> b -> b — Foldr
keys :: Table k v -> (b -> k -> b) -> b -> b — all keys
values :: Table k v -> (b -> v -> b) -> b -> b — all values
```

#### Stack operations

— interface  
`newtype Stack a = StackImpl [a] — opaque!`

```
empty :: Stack a
isEmpty :: Stack a -> Bool
push :: a -> Stack a -> Stack a — insert
top :: Stack a -> a — the first value
pop :: Stack a -> (a, Stack a) — take out
```

### Queue operations

```
— interface
newtype Queue a = Q [a] — opaque

empty :: Queue a
isEmpty :: Queue a -> Bool
head :: Queue a -> a
enqueue :: Queue a -> a -> Queue a — take out element
dequeue :: Queue a -> Queue a — insert element
toList :: Queue a -> [a]
```

### Hastables

- Key value lookup (an index)
- Array is only define for small index, hash has no limit on available keys.
- Typically we have  $n$  possible keys from set  $U$  for a hashtable (which is an array) with  $m$  slots, where  $n \geq m$ .
- since there is infinitely many elements and limited amount of key there will be element with the same key, therefor a coalition is created.
- Worst-Case Retrieval: time complexity of retrieving a element.
- Load Factor: How much data is in the table  $\frac{\text{elements}}{\text{slots}}$
- Rehashing: make the hashtable more balanced.

### Collision Resolution by Chaining

- Most commonly used collision resolution
- Let each array slot (also called a bin) hold a list of elements (called a chain).
- In other words, When collision then add it to a list in that element

### Collision Resolution by Open Addressing

- Start with a table with each element is  $\perp$  previously used  $\Delta$ .
- Probing: is a function to insert items in a hashtable therefore resolves collations
- Types of probing:  
 Linear probing:  $f(i) = i$ .  
 Quadratic probing:  $f(i) = c_2 \cdot i^2 + c_1 \cdot i$ , where  $c_2 \neq 0$ .  
 Double hashing:  $f(i) = i \cdot h''(i)$ , where  $h''$  is another hash function.
- Inserting with Linear Probing: Insert it to the next available key
- Deleting with Linear Probing: Ignores  $\perp$  and  $\Delta$  idex will change.

### 2.7.7 Graphs

#### Types of graphs

- list
- tree
- forest
- Directed Acyclic Graph (DAG)

#### Terminology

- Node, vertex (plural: vertices)
- Edge connects two nodes.
- Self-loop edge from node to itself
- Adjacent nodes connected by an edge
- Degree number of edges from or to a node
- In-degree number of edges to a node
- Out-degree number of edges from a node

#### Representation

- **Adjacency Matrix** — a 2-dimensional array A of 0/1 values, with  $A[i, j]$  containing the number of edges between nodes i and j (undirected graph), or from node i to node j
  - + edge existence testing in  $\theta(1)$  time
  - finding next outgoing edge in  $O|V|$  time
  - + compact representation for dense graphs (when  $|E|$  is close to  $|V|^2$ )
- **Adjacency List** — a 1-dimensional array Adj of adjacency lists, with  $Adj[i]$  containing a list of the nodes adjacent to node i.
  - + finding next outgoing edge in  $\theta(1)$  time
  - edge existence testing in  $O(|V|)$  time
  - + compact representation for sparse graphs (when  $|E|$  is much smaller than  $|V|^2$ )
- **Edge List** — a list of tuples,  $(i, j)$ , for each edge  $(i, j)$  (plus a list of the nodes).
  - edge existence test in  $\theta(|E|)$ .
  - finding next outgoing edge in  $O(|E|)$  (unless appropriately sorted).
  - + compact representation for sparse graphs (when  $|E|$  is much smaller than  $|V|^2$ )

### topological sort

A topological sort is a linear ordering of all the nodes in a directed acyclic.

#### Algorithm

1. Select a node with in-degree 0.
2. Output it.
3. Remove it.
4. Repeat (from 1) until no nodes are left.

Total running time is  $\theta(|V| + |E|)$ .

### Graph Traversals

- Breadth-first search (BFS). Uses a queue for each grey node.  
Time complexity:  $\theta(|V| + |E|)$  — linear in the size of the graph.
- Depth-first search (DFS). Uses a stack for each (grey) node and has a rest of nodes (white).  
Time complexity:  $\theta(|V| + |E|)$  — linear in the size of the graph.

#### Breadth-First Search: Algorithm

Input: Some node A.

1. Paint A gray. Paint other nodes white. Add A to an initially empty FIFO queue of gray nodes. All grey nodes is in the queue. It is a queue not a stack so first in first out.
2. Dequeue head node, X. Paint its undiscovered (white) adjacent nodes gray and enqueue them. Paint X black. Repeat until queue is empty. For every black node add it to BFS order (black)

#### Depth-First Search: Algorithm

Input: Some node A.

#### DFS(G)

1. Paint all nodes white.
2. For each node v in G: if v is (still) white, DFS-Visit(G,v). Each subsequent call to DFS-Visit in line 2 is called a restart.

#### DFS(G)

1. Colour v gray.
2. For each node u adjacent to v: if u is white, DFS-Visit(G,u).

**Strongly-Connected Components**

- Strongly connected component (SCC): maximal set of nodes where there is a path from each node to each other node.
- Many algorithms first divide a digraph into its SCCs, then process these SCCs separately, and finally combine the sub-solutions. (This is not divide & conquer, since a different algorithm is run on each SCC!)
- An undirected graph can be decomposed into its connected
  - 1. Enumerate the nodes of G in DFS finish order, starting from any node
  - 2. Compute the transpose  $G^T$  (that is, reverse all edges)
  - 3. Make a DFS in  $G^T$ , considering nodes in reverse finish order from original DFS
  - 4. Each tree in this depth-first forest is a strongly connected component

## 2.8 Important syntax

### 2.8.1 Let

```

let x = 1 in x * 2 == 2
case x of
  1 -> "Hello"
  2 -> "H"
  3 -> "Hel"

input: 3 == "Hel"

# other examples
let f x y = x + 3 >= y + 3.1 in f 1 1 == False

f x = let g z = z+1 in g (g x)
f 1 == 3

:t div
div :: Integral a => a -> a

:t (/)
(/) :: Fractional a => a -> a

```

### 2.8.2 IO

**monads** Is used to return an IO and uses a operation ( $\gg=$ ) that replaces *do-notation*

```

class Monad m where
  ( $\gg=$ ) :: m a -> (a -> m b) -> m b
  return :: a -> m a

```

## 2.9 Sorting Algorithms

1. Insertion Sort: One at a time
2. Bubble Sort: sort in order two a time starting from left and work to the right side.
3. Merge Sort: Divide and Conquer, split into even pieces and sort them and then merge/sort again.
4. Quicksort: Divide and Conquer, takes a pivot to split smaller and larger.

## Kapitel 3

# Algebra 1 - Formel Samling

### 3.1 logik

Utsaga: ett påstående som erhåller antigen värderna sann(S) eller falsk(F) vilket ge en stluten utsaga eller ej ett sannings värde vilket kallas för öppna utsagor

Kombinerade utsagor:

Kunjuktion utsagor: består av två utsagor som vi kallar A och B

där and:  $\wedge (A \wedge B)$

Dissjunktion utsagor: består av två utsagor som vi kallar A eller B

or:  $\vee (A \vee B)$

icke utsaga A (motsatsen):  $\neg A$

Alla:  $\forall$

Minst en:  $\exists$

$\neg(\forall x : A) \Leftrightarrow \exists x : \neg A$

$\neg(\exists x : A) \Leftrightarrow \forall x : \neg A$

**Exempel:**

A: Alla reala tal x gäller att  $(x + 1)^2 = 0$

$\forall x : (x + 1)^2 = 0$

$\neg(\forall x : (x + 1)^2 = 0) = \exists x : (x + 1)^2 \neq 0$

$\neg A$ : Det finns reala tal x gäller att  $(x + 1)^2 \neq 0$

#### 3.1.1 värde tabeller

Kunjuktions värdetabel:

A	B	$A \wedge B$
S	S	S
S	F	F
F	S	F
F	F	F

Dissjunktions värdetabel:

A	B	$A \vee B$
S	S	S
S	F	S
F	S	S
F	F	F

#### 3.1.2 Implikationer

A medför B:  $A \Rightarrow B$  (implikation)

A och B medför varandra:  $A \Leftrightarrow B$  (ekvivalens)

**Implication värdetabell:**

A	B	$A \Rightarrow B$
S	S	S
S	F	F
F	S	S
F	F	S

**Ekvivalens värdetabel:**

A	B	$A \Leftrightarrow B$
S	S	S
S	F	F
F	S	F
F	F	S

**Exempel1:**

$$x = \sqrt{6 - x} \quad (3.1)$$

$$x = \sqrt{6 - x} \Rightarrow x^2 = 6 - x \Leftrightarrow x^2 + x - 6 = 0$$

$$\text{pq-formeln: } x = -\frac{1}{2} \pm \sqrt{\frac{1}{4} + \frac{24}{4}} \Leftrightarrow (x = -3) \vee (x = 2)$$

Eftersom det är en implikation är höger och inte ekvivalens så behöver inte rötterna vara sanna  
Testar för falska rötter:

$$2 = \sqrt{6 - 2} \text{ Sann}$$

$$2 = \sqrt{6 - (-3)} \text{ Falsk } 2 \neq \sqrt{6 - (-3)}$$

**Exempel2:**

$$(x + 2)(x + 1) = 2x(x + 1) \quad (3.2)$$

$$(x + 2)(x + 1) = 2x(x + 1) \text{ is not } \Rightarrow (x + 2) = 2x \text{ (x=0 is not allowed)}$$

Insted do ass following:

$$\begin{aligned} (x + 2)(x + 1) = 2x(x + 1) &\Leftrightarrow (x + 2)(x + 1) - 2x(x + 1) = 0 \Leftrightarrow (x + 1)(x + 2 - 2x) = 0 \Leftrightarrow \\ &\Leftrightarrow (x + 1 = 0) \vee (2 - x = 0) \end{aligned}$$

svar ej: x=1 och x=2

svar: x=1 eller x=2

svar:  $x = 1 \vee x = 2$

## 3.2 Mängder

Naturliga tal:  $\mathbb{N} = \{0, 1, 2, 3, \dots\}$

Heltal:  $\mathbb{Z} = \{\dots, -2, 1, 0, 1, 2, \dots\}$

Rationella tal:  $\mathbb{Q} = \left\{ \frac{a}{b} \mid a, b \in \mathbb{Z}, b \neq 0 \right\}$

Irrationella tal:  $\mathbb{P} = \mathbb{R} \setminus \mathbb{Q}$  eller  $\{x \mid x \in \mathbb{R}, x \notin \mathbb{Q}\}$

Reella tal:  $\mathbb{R} = \mathbb{P} \cup \mathbb{Q}$

$$A \cup B = \{x : (x \in A) \vee (x \in B)\}$$

$$A \cap B = \{x : (x \in A) \wedge (x \in B)\}$$

$$A \setminus B = \{x : (x \in A) \wedge (x \notin B)\}$$

$$A^\# = \{x : (x \in X) \wedge (x \notin A)\}$$

**Exempel:**

$$\text{Bevisa: } X \setminus (A \cup B) = (X \setminus A) \cap (X \setminus B) \quad (3.3)$$

$$\begin{aligned} x \in (X \setminus (A \cup B)) &\Rightarrow (x \in X) \wedge (x \notin (A \cup B)) \Rightarrow \\ &\Rightarrow (x \in X) \wedge (x \notin A) \wedge (x \notin B) \Rightarrow \\ &\Rightarrow (x \in X \setminus A) \wedge (x \in X \setminus B) \Rightarrow \\ &\Rightarrow x \in (X \setminus A) \cap (X \setminus B) \Rightarrow \\ &\Rightarrow X \setminus (A \cup B) \subseteq (X \setminus A) \cap (X \setminus B) \end{aligned}$$

### 3.3 Bevis

#### 3.3.1 Induktions bevis

steg 1: bevisa att det gäller för basfallet

Steg 2: bevisar att  $p \Rightarrow p$  a. Antar att det stämmer för  $p$  b. Vissar att med att stoppa in antagandet i  $p+1$  så blir  $hl = vl$  Tips: Förenkla  $hl$  först och sedan  $vl$  på klad papper fram och tillbaka **Exempel Recursion:**

$$a_1 = 2, a_{n+1} = \frac{7a_n}{7 - a_n}, n \in \mathbb{N}$$

$$\text{Vissa med induktion att } a_{n+1} = \frac{14}{7 - 2n}$$

Bevis med induktions

steg 1: visar att påståendet som vi kallar  $p$  gäller för basfallet ( $n=1$ )

$$VL_1 : a_{1+1} = a_2 = \frac{7 \cdot 2}{7 - 2} = \frac{14}{5}, HL_1 : a_{1+1} = a_2 = \frac{14}{7 - 2} = \frac{14}{5}$$

steg 2: visar att  $p_m \Rightarrow p_{m+1}$

steg 2a: antar att  $p_m$  gäller

$$a_{m+1} = \frac{14}{7 - 2m}$$

steg 2b: bevisar attt  $p_m \Rightarrow p_{m+1}$  genom att använda antagandet

$$\begin{aligned} VL_{m+1}a_{m+2} &= \frac{7a_{m+1}}{7 - a_{m+1}} = \frac{7 \frac{14}{7-2m}}{7 - \frac{14}{7-2m}} = \frac{\frac{7 \cdot 14}{7-2m}}{\frac{7(7-2m)-14}{7-2m}} = \\ &= \frac{7 \cdot 14}{7(7-2m+2)} = \frac{14}{7-2(m+1)} \\ HL_{m+1} &: \frac{14}{7-2(m+1)} \end{aligned}$$

Enlight induktionsprincipen är  $p_m$  sann för alla  $n = 1, 2, 3, \dots$  VSB

#### 3.3.2 Motsägelse bevis

Steg 1: Formulera utsagan och icke utsagan Steg 2: Hitta en motsägelse med utsagan Tips: Förenka båda led, tänk på teorin vi har, bättre att gå vaga moteveringar en inga alls

Bevis med motsägelse

Antar att motsatsen är sann

### 3.4 Delbarhet

a är delbar med b, altså kvoten ger ingen rest. Vi följand:  $a \mid b$  (3.4)

**Divitions algoritmen:**

$$\begin{aligned} a, b &\in \mathbb{Z} \\ a &\geq 0 \wedge b \geq 0 \\ a \mid b &\Rightarrow (q \in \mathbb{Z} : q \geq 0) \wedge (r \in \mathbb{Z} : 0 \leq r \leq a) \text{ Sådant att} \\ b &= qa + r \\ q &= \text{kvoten}, r = \text{resten} \end{aligned}$$

#### 3.4.1 Största Gemensama Delaren (SGD)

$$\begin{aligned} SGD(a, b) \\ a &= bq + r \\ 0 \leq r &\leq b \end{aligned}$$

**Euklides algoritm:**

$$\begin{aligned} SGD(a, b) \\ a &= bq_1 + r_1 \\ b &= r_1q_2 + r_2 \\ r_1 &= r_2q_3 + r_3 \\ r_2 &= r_3q_4 + r_4 \\ &\vdots \\ &\vdots \\ r_{k-3} &= r_{k-2}q_{k-1} + r_k \\ r_{k-2} &= r_{k-1}q_k + 0 \\ SGD(a, b) &= r_k \\ \text{Om } r_k &= 1 \Rightarrow a \in \text{primtal} \vee b \in \text{primtal} \end{aligned}$$

**Exempel:**

Förenkla  $\frac{114}{96}$

$$\begin{aligned} SGD(114, 96) : \\ 114 &= 1 * 96 + 18 \\ 96 &= 5 * 18 + 6 \\ 18 &= 3 * 6 + 0 \\ \frac{114}{6} &= 19 \\ \frac{96}{6} &= 16 \\ \frac{19}{16} \end{aligned}$$

**Lemma:**

$$\begin{aligned} a, b \in \mathbb{Z} \\ x, y \in \mathbb{Z} \\ SGD(a, b) = ax + by \end{aligned}$$

**Aritmetiska fundamentalsatsen:**

$$\begin{aligned} a \in \mathbb{Z} \wedge a \geq 2 \Rightarrow \\ \Rightarrow a \text{ kan endast primtalsfaktoriseras på ETT SÄTT} \end{aligned}$$

**Lemma 2.7:**

$$\begin{aligned} a \geq 2 \wedge a \notin \text{Primatal} \Rightarrow \\ \Rightarrow q \in \text{Primatal} \wedge q \mid a \wedge \text{a går att primtals faktoriera} \end{aligned}$$

### 3.4.2 Primtal

**Sats:**

För att bestäma om tal a är ett primtal

$$a \geq 1 \wedge a \in \text{Primatal}$$

om ett tall p finns som delar a gäller följande

$$2 \leq p \leq \sqrt{a} \leq a$$

**Exempel:**

Bestäm om 211 är ett primtal (3.5)

Om 211 är ett primtal så finns det inte en äkta delare a

$$2 \leq a \leq \sqrt{211}$$

$$\sqrt{211} \approx 16$$

$$a : \{\emptyset, \beta, \emptyset, \pi, \alpha, \alpha\}$$

a kan inte vara en äktadelare

**Euklides algoritm:**

Det finns oändligt många primtal

**Bevis:**

Motsägelsebevis

Antar att det finns ändligt många primtal

$$p_1, p_2, P_3, \dots, p_n$$

$$M = \prod_{k=1}^{n+1} n + 1 \Rightarrow M > p_j, j = 1, 2, 3, \dots, n$$

Vissar att M är ett primtal

$1 < b < M \wedge b \mid M$  Där b är minsta äkta delaren av M  $\Rightarrow$

$$\Rightarrow M = p_1 * b \Rightarrow p_1 \mid 1 \wedge p \geq 2$$

Detta är falskt eftersom båda utsägorna kan inte vara samtidigt

Eftersom motsatsen inte fungerar resulterar det i att satsen är sann

### 3.5 Diofantiska ekvationer

Sats:

$$\begin{aligned} ax + by = c \wedge a, b, c \in \mathbb{Z} & \quad a \neq 0, b \neq b \\ \Rightarrow & \\ ax + by = c \text{ Där } SGD(a, b) = 1 & \\ \text{Har den anmäla lösningen:} & \\ x = Cx_0 - nb \wedge y = Cy_0 + na & \end{aligned}$$

**Exempel:**

En lastbil lastas med 12kg packet och 20kg paket. Totalt väger lasten 296, hur många av varje packet? (3.6)

$$\begin{aligned} ax + by = C & \Leftrightarrow 12x + 20y = 296 \\ \text{steg1: Testar om } SGD(a, b) \mid c & \\ SGD(20, 12) = 4 \Rightarrow 4 \mid 296 & \\ \text{steg2: delar SGD med HL och VL} & \\ \frac{12x - 20y}{4} = \frac{296}{4} & \Leftrightarrow 3x - 5y = 74 \\ SGD(5, 3) : & \\ 5 = 1 \cdot 3 + 2 & \\ 3 = 1 \cdot 2 + 1 & \\ 2 = 2 \cdot 1 + 0 & \\ \text{steg3: Hjälp ekvation för att hittax}_0, y_0 & \\ 3x_0 - 5y_0 = 1 & \\ 1 = 3 - 1 * 2 & \\ 1 = 3 - (5 - 3) & \\ 1 = 2 \cdot 3 - 1 \cdot 5 & \\ x_0 = 2, y_0 = -1 & \\ \text{steg4: almäna lösningen } x = Cx_0 - bn, y = Cy_0 + an & \\ x = 74 \cdot 2 - 5n = 148 - 5n & \\ y = 74 \cdot (-1) + 3n = -74 + 3n & \\ \text{steg5: hittar godtyckliga lösningar} & \\ \text{Intervallet som n ligger i för x-termen:} & \\ n = 29, 28, 27, \dots, x = 148 - 5 \cdot 28 = 148 - 140 = 8 & \\ \text{Intervallet som n ligger i för y-termen:} & \\ n = 27, 28, 29, \dots, x = -74 + 3 \cdot 28 = 74 - 84 = 10 & \\ n = 28, x = 8, y = 10 & \\ 12 \cdot 8 + 20 \cdot 10 = 296 & \end{aligned}$$

## 3.6 Talbaser

### 3.6.1 konvertera från decimal bass till annan bas

$$175_8 = 1 \cdot 8^2 + 7 \cdot 8^1 + 5 \cdot 8^0$$

### 3.6.2 konvertera från annan bas till decimal bas

$$1609_{10} = (3 \cdot 8^3 + 1 \cdot 8^2 + 1 \cdot 8^1 + 1 \cdot 8^0) = 3111_8$$

eller så kan man använda euklides algoritm

skriv 517 i talbas 3

$$517 = 172 \cdot 3 + 1$$

$$172 = 57 \cdot 3 + 1$$

$$57 = 20 \cdot 3 + 0$$

$$20 = 6 \cdot 3 + 2$$

$$6 = 2 \cdot 3 + 0$$

$$2 = 0 \cdot 3 + 2$$

Svar:  $517_{10} = 202011_{tre}$

### 3.6.3 Andra exempel

**Exempel:** Skriv  $137_{10}$  i bass tre

$$\begin{aligned} 137_{10} &= 1 \cdot 9^2 + 3 \cdot 9^1 + 7 \cdot 9^0 = 1 \cdot 3^4 + 3 \cdot 3^2 + 7 \cdot 3^0 = \\ &= 1 \cdot 3^4 + 1 \cdot 3^3 + 0 \cdot 3^2 + 2 \cdot 3^1 + 1 \cdot 3^0 = 11021_{tre} \end{aligned}$$

## 3.7 Functioner

**Typer av funktioner:**

Injectio: alla element x har olika värden y  $f : A \rightarrow B, \{\forall x \in A : x_1 \neq x_2, f(x_1) \neq f(x_2)\}$

Surjekcio: mängd D är definitions mängden  $\{g : C \rightarrow D, g(x) = y, (\forall y \in D \wedge \exists x \in C)\}$

Bijekcio: Injectio  $\wedge$  Surjekcio

**Kareskapprodukten:**

$$A \times B = \{(a, b) : a \in A \wedge b \in B\}$$

$$\text{Låt } A = \{1, 2, 3\} \wedge B = \{x, y, z, w\}$$

$$AxB :$$

$$\{(1, x), (1, y), (1, z), (1, w)$$

$$(2, x), (2, y), (2, z), (2, w)$$

$$(3, x), (3, y), (3, z), (3, w)\}$$

### 3.7.1 Inversen

En funktions invers kan enda

$$f : A \rightarrow B \wedge \text{Bijektiv} \Rightarrow f^{-1}(x) \text{ Finns, där}$$

$$(1) x = f^{-1}(y) \Leftrightarrow y = f(x)$$

$$(2) D_{f^{-1}} = V_f \Leftrightarrow D_f = V_{f^{-1}}$$

$$(3) x = f^{-1}(f(x)), x \in D_f = V_{f^{-1}}$$

$$(3) y = f^{-1}(f(y)), y \in D_f = V_{f^{-1}}$$

### 3.7.2 Relatioiner

Relation:  $xRy$

Reflexiv:  $\forall x \in X : xRx$

Symetrisk:  $xRy \Rightarrow yRx, x \in X \wedge y \in X$

Transitiv:  $(xRy) \wedge (yRz) \Rightarrow xRz, \forall x, y, z \in X$

Ekvivalensrelation: Reflexiv och Symetrisk Transitiv

## 3.8 Summor

### 3.8.1 Aritmetiska summor

$$s_n = a_1 + a_2 + a_3 + \dots + a_n = \frac{n(a_1 + a_n)}{2} \quad (3.7)$$

### 3.8.2 Geometriska summor

Börjar alltid med exponenten 0 och gör om summan så att den passar i följande talföljd:

$$s_n = a + ak + ak^2 + \dots + ak^{n-1} = \frac{a(k^n - 1)}{k - 1} \quad (3.8)$$

**Exempel:**

$$\sum_{k=n}^{2n} (2^k - k) \quad (3.9)$$

sätter f = 0 = k - n

$$\begin{aligned} \sum_{f=0}^n (2^{f+n} - (f + n)) &= 2^n * \sum_{f=0}^n (2^f) - \sum_{f=0}^n (f + n) \\ \frac{2^n(2^{n+1} - 1)}{2 - 1} - \frac{3n(n + 1)}{2} &= 2^{2n+1} - 2^n - \frac{3n(n + 1)}{2} \end{aligned}$$

### 3.9 Kongruensräkning

Räkneregler:

$$\begin{aligned}a + b \pmod{n} &\equiv a \pmod{n} + b \pmod{n} \\a \cdot b \pmod{n} &\equiv a \pmod{n} \cdot b \pmod{n} \\a^b \pmod{n} &\equiv (a \pmod{n})^b\end{aligned}$$

**Exempel:** Vilket är det minsta positiva rest som kan erhållas vid division av  $19^{18}$  med 17?

$$19^{18} \equiv 2^{18} \pmod{17} \equiv 2^4 \cdot 2^4 \cdot 2^4 \cdot 2^4 \cdot 2^2 \pmod{17} \equiv (-1)^4 \cdot (-1)^4 \cdot (-1)^4 \cdot (-1)^4 \cdot 2^2 \pmod{17}$$

svar: resten blir 4

## 3.10 Kardinalitet

Kardinalitet eller "mäktighet" är ett sett att räkna med mängders sorlek och alla oändliga mängder har samma kardinalitet fast det är en delmängd. Naturliga tall har samma kardinalitet som reala tal trots att naturliga tal är en del mängd av reala talen

Låt A och B vara mängder. Vi sägger att A och B har samma kardinalitet då det finns en bijektion  
 $\exists f : A \rightarrow B, A \sim B$

Vi säger att A står i relation med B omm A och B har samma kardinalitet  $ARB$

### 3.10.1 Uppräkneligamängder

En mängd X sägs vara uppräknerlig omm X har samma kardinalitet som  $\mathbb{N}$

$\exists g : \mathbb{N} \rightarrow X$  där g är bijektiv

Exempel på uppräknerliga mängder är  $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \{1, 5, 78\}$

Exempel på ej uppräknerliga mängder  $\mathbb{R}, (0, 1)$

## 3.11 Polynom

### 3.11.1 Polynom division

Triviala delare: ej heltals kvot med delaren, har en kostant sådant  $\lambda \cdot f, \lambda \notin \mathbb{Z}$

Äkta delare: heltals kvot med delaren  $f(x), \exists a \in \text{polynom} : a | f(x)$

Irreducible: om polynomet endast har triviala delare det finns lösningar heltaslösningar  $f(x) = 0$

Reducible: om polynomet har äkta delare

Multiplisitet: vilken grad polynomet har

**Exempel:**

Man vet att ekvationen  $z^4 - 2z^3 - 7z^2 + 26z - 20 = 0$  har roten  $z = 2+i$ . Lös ekvationen fullständigt. (3.10)

$z = 2 + i$  Är en lösning är också konjugatet en lösning enligt faktorsatsen  $\bar{z} = a - bi$

$$z = 2 \pm i$$

Vilket betyder att följande går att factorisera ut polynomet

$$(z - (2 + i))(z - (2 - i)) = z^2 - 4z + 5$$

**Långdivision (liggande stolen):**

$$\begin{array}{r} x^2 + 2x - 4 \\ \hline x^2 - 4x + 5 ) \overline{x^4 - 2x^3 - 7x^2 + 26x - 20} \\ \quad - x^4 + 4x^3 - 5x^2 \\ \hline \quad 2x^3 - 12x^2 + 26x \\ \quad - 2x^3 + 8x^2 - 10x \\ \hline \quad - 4x^2 + 16x - 20 \\ \quad 4x^2 - 16x + 20 \\ \hline 0 \end{array}$$

$z^2 + 2z - 4 = 0$  Är också en lösning som till slut ger följande

$$z = -1 \pm \sqrt{5}$$

$$z = 2 \pm i$$

Varge n grads polynom har alltid n stycken komplexa lösningar

### 3.11.2 Faktorsatsen

$$f(x) = (x - \alpha)p(x)$$

**Exempel, Gemensam root hos två polynom:**

$p(x) = x^4 - x^3 + x^2 + 2 = 0$ ,  $g(x) = x^3 + 4x^2 + 4x + 3 = 0$  har en Gemensam root  
 Eftersom polynomen har en Gemensam root vet vi att det finns en gemensam  $(x - \alpha)$

$$f(x) = (x - \alpha)p_1(x)$$

$$g(x) = (x - \alpha)p_2(x)$$

euklides algoritm:

$$f(x) = (x - 5)g(x) + 17(x^2 + x + 1)$$

$$g(x) = \left(\frac{1}{17}(x + 3)\right)(17(x^2 + x + 1)) + 17(x^2 + x + 1) + 0$$

$$h(x) = x^2 + x + 1$$

$$f(x) = (x - 5)(x + 3)h(x) + 17h(x) = h(x)((x - 5)(x + 3) + 17) = h(x)(x^2 - 2x + 2)$$

$$g(x) = (x + 3)h(x)$$

$$f(x) = 0 \Leftrightarrow (h(x) = 0 \vee x^2 - 2x + 2 = 0)$$

$$h(x) = 0 \Rightarrow x = -\frac{1}{2} \pm \frac{\sqrt{3}}{2}i$$

$$x^2 + 2x + 2 = 0 \Rightarrow x = 1 \pm \sqrt{1 - 2} = 1 \pm i$$

**Exempel, Heltalslösning med okänd konstant:**

$$x^3 + bx^2 - 7x - 7 = 0 \text{ har en heltals lösning}$$

Eftersom ekvationen har en heltals lösning vet vi att  $\alpha \mid 7$

Kandidaterna är  $\{1, -1, 7, -7\}$

$$I(x = 1) 1 + b \cdot 1 - 7 - 7 = 0 \Rightarrow b = 13 \in \mathbb{Z}$$

$$II(x = -1) -1 + b \cdot -1 + 7 - 7 = 0 \Rightarrow b = 1 \in \mathbb{Z}$$

$$III(x = 7) 7 + b \cdot 7 - 49 - 7 = 0 \Rightarrow b = \underline{\quad} \notin \mathbb{Z}$$

$$IV(x = -7) 7 + b \cdot -7 + 49 - 7 = 0 \Rightarrow b = \underline{\quad} \notin \mathbb{Z}$$

svar: b=13, b=1

**Exempel, Rationell root:**

$g(x) = 2x^3 + 2x^2 + 2x + 3 = 0$  har en rationell root

Eftersom polynomet har en ratiotonal root vet vi att:

$$\frac{p}{q} \quad p, q \in \mathbb{Z}, \quad SGD(p, q) = 1, \quad p \mid 3 \wedge q \mid 2 \quad (a_0)$$

Det möjliga delarna är  $p = \pm 1, \pm 3, q = \pm 1, \pm 2$

$x = -\frac{3}{2}$  är den enda av kandidaterna som ger en sann root

Vilket vi löser genom polynom division

**Exempel, Renimaginär root:**

$z^4 + 4z^3 + 8z^2 + 12z + 15 = 0$  har en renimaginär root

Eftersom polynomet har en rentimaginär root vet vi att  $z = bi, b \in \mathbb{R}, b \neq 0$

$$0 = b^4 z^4 + 4b^3 z^3 + 8b^2 z^2 + 12bz + 15 = b^4 - 4b^3 i - 8b^2 + 12bi + 15 = (b^4 - 8b^2 + 15) + (12b - 4b^3)i = 0 = b^4 - 8b^2 + 15$$

$$0 = 12b - 4b^3 = 4b(3 - b^2) \Rightarrow b = 0, b = \sqrt{3}, b = -\sqrt{3}$$

$b = 0$  är ej en giltig lösning. Enligt faktorsatsen får vi följande

$$p(z) = (z - \sqrt{3}i)(z + \sqrt{3}i)Q(x) = (z^2 + 3)Q(z)$$

Vilket vi löser genom polynom division

**Exempel, Dubbel root:**

$p(t) = t^3 - 5t^2 + 3t + 9 = 0$  har en dubbel root

Eftersom polynomet har en dubbel root vet vi att derivatan ger os en root  $p'(t) = 0$

Att räkna ut derivatan gör man genm formeln  $ax^b \Rightarrow b \cdot ax^{b-1}$

$$p'(t) = 3t^2 - 10t + 3 = 0 \Rightarrow t_1 = 3, t_2 = \frac{1}{3} \text{ dock är sista en falsk root}$$

$$p(t) = x - 3^2 Q(t)$$

Vilket vi löser genom polynom division

**Exempel, SGD polynom:**

$$\text{Förenkla } \frac{x^4 + x^3 + 2x - 4}{x^4 - x^3 - 2x - 4}$$

$$SGD(x^4 + x^3 + 2x - 4, x^4 - x^3 - 2x - 4) :$$

$$x^4 + x^3 + 2x - 4 = 1 \cdot x^4 - x^3 - 2x - 4 + (2x^3 + 4x)$$

$$x^4 - x^3 - 2x - 4 = \frac{x}{2} - \frac{1}{2} \cdot (2x^3 + 4x) + (-2x^2 - 4) \text{ med polynom divition}$$

$$2x^3 + 4x = -x \cdot (-2x^2 - 4) + 0$$

Med SGD så kan vi ta ett polynom som är hjämt delbart ex:  $-2(x^2 + x)$

$$\text{polynom divition: } \frac{x^4 + x^3 + 2x - 4}{x^2 + x} = x^2 + x - 2$$

$$\text{polynom divition: } \frac{x^4 - x^3 - 2x - 4}{x^2 + x} = x^2 - x - 2$$

$$\frac{x^4 + x^3 + 2x - 4}{x^4 - x^3 - 2x - 4} = \frac{x^2 + x - 2}{x^2 - x - 2}$$

## Kapitel 4

# Single Variable Calculus 1 - Formel Samling

## 4.1 Basic

### 4.1.1 Mängder

Naturliga tal:  $\mathbb{N} = \{0, 1, 2, 3..\}$  or  $\{1, 2, 3..\}$

Heltal:  $\mathbb{Z} = \{.. - 2, 1, 0, 1, 2..\}$

Rationella tal:  $\mathbb{Q} = \left\{ \frac{a}{b} \mid a, b \in \mathbb{Z}, b \neq 0 \right\}$

Irrationella tal:  $\mathbb{P} = \mathbb{R} \setminus \mathbb{Q}$  eller  $\{x \mid x \in \mathbb{R}, x \notin \mathbb{Q}\}$

Reella tal:  $\mathbb{R} = \mathbb{P} \cup \mathbb{Q}$

$$A \cup B = \{x : (x \in A) \vee (x \in B)\}$$

$$A \cap B = \{x : (x \in A) \wedge (x \in B)\}$$

$$A \setminus B = \{x : (x \in A) \wedge (x \notin B)\}$$

$$A^\# = \{x : (x \in X) \wedge (x \notin A)\}$$

### 4.1.2 Intervall

Open intervall =  $(1, 4)$

Closed intervall =  $[1, 4]$

$$[a, b] = \{x | a \leq x \leq b\} \quad (4.1)$$

$$[a, \infty[ \quad (4.2)$$

$$]-\infty, \infty[ \quad (4.3)$$

**Exempel: Olikheter och intervall**

$$\frac{2}{x-3} < \frac{5}{x} \quad (4.4)$$

$$\begin{aligned} \frac{2}{x-3} - \frac{5}{x} &< 0 \\ \frac{(x)2}{x(x-3)} - \frac{5(x-3)}{x(x-3)} &< 0 \\ \frac{2x - 5x + 15}{x(x-3)} &< 0 \\ \frac{-3x + 15}{x(x-3)} &< 0 \\ \frac{-3(x-5)}{x(x-3)} &< 0 \\ x \neq 0, x \neq 3 \end{aligned}$$

Värde tabell:

	$x < 0$	$0 < x < 3$	$3 < x < 5$	$5 < x$
$x-5$	-	-	-	+
-3	-	-	-	-
x	-	+	+	+
$x-3$	-	-	+	+
hela	+	-	+	-

### 4.1.3 Funktion

$$f : A \rightarrow B$$

A: domain/definitionsmängd, B: Målmängd/kodomängd/värdemängd

Injectie: alla element x har olika värden  $f : A \rightarrow B, \{\forall x \in A : x_1 \neq x_2, f(x_1) \neq f(x_2)\}$

Surjektion: mängd D är definitions mängden  $\{g : C \rightarrow D, g(x) = y, (\forall y \in D \wedge \exists x \in C)\}$

Bijektion: Injectie  $\wedge$  Surjektion

$$f \circ g = f(g(x))$$

Begränsad: funktionen är inom ett intervall, altså nedåt och uppåt

Begränsat nedåt: functionen är endast begrensat nedåt

Begränsat uppåt: functionen är endast begrensat uppåt

Jämn function: altså är den symmetrisk, ex:  $x^2, f(-x) = f(x)$

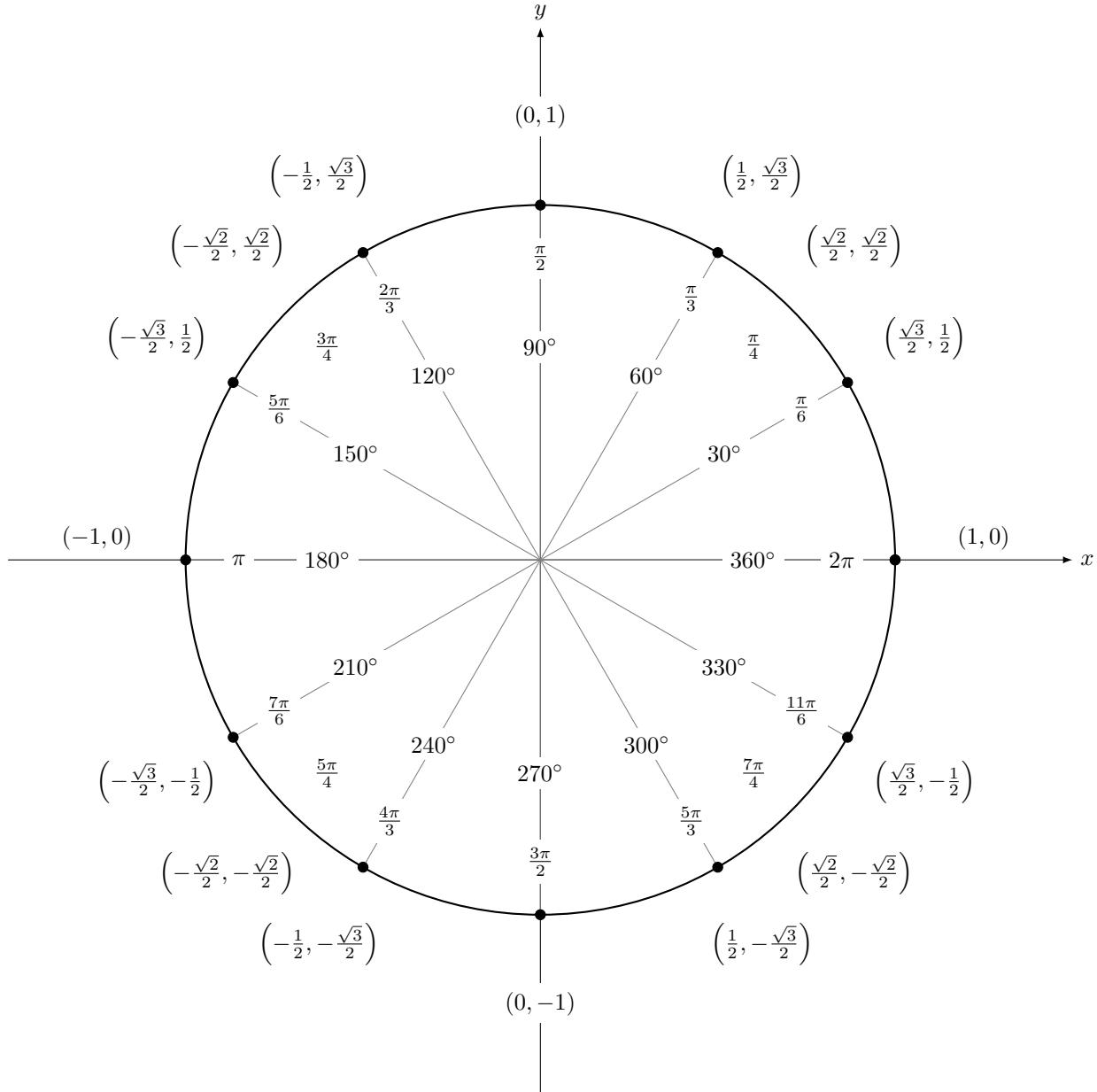
Udda function: altså är den spegelvänt symmetrisk, ex:  $x^3, f(-x) = -f(x)$

$$\text{Polynom}(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = \sum_{i=0}^n (a_i x^i)$$

$$\text{Rationell funktion: } R(x) = \frac{P(x)}{Q(x)}, \text{ ex: } x^{-1} = \frac{1}{x} \text{ ej polynom men rationell funktion}$$

#### 4.1.4 Trigonometri

$\sin$	0	$\frac{\sqrt{0}}{2}$	$\frac{\sqrt{1}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{4}}{2}$
$\cos$	$\frac{\sqrt{4}}{2}$	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{1}}{2}$	$\frac{\sqrt{0}}{2}$	



**Sats:**

$$360^\circ = 2\pi \text{rad}$$

$$v_g = v_r * \frac{180^\circ}{\pi}$$

$$v_r = v_g * \frac{\pi}{180^\circ}$$

**Sats:**

$$\begin{aligned} -1 &\leq \sin t \leq 1 \\ -1 &\leq \cos t \leq 1 \end{aligned}$$

**Sats:**

$$\cos(-t) = \cos(t)$$

$$\sin(-t) = -\sin(t)$$

$$\tan(-t) = \frac{\sin(-t)}{\cos(-t)} = \frac{-\sin(t)}{\cos(t)}$$

**Additionsformlerna:**

$$\sin(\alpha + \beta) = \sin(\alpha)\cos(\beta) + \sin(\beta)\cos(\alpha)$$

$$\sin(\alpha - \beta) = \sin(\alpha)\cos(\beta) - \sin(\beta)\cos(\alpha)$$

$$\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\beta)\sin(\alpha)$$

$$\cos(\alpha - \beta) = \cos(\alpha)\cos(\beta) + \sin(\beta)\sin(\alpha)$$

**Trigonometriska ettan:**

$$(\sin t)^2 + (\cos t)^2 = 1$$

$$\sin^2 t + \cos^2 t = 1$$

#### 4.1.5 Exempel: Trigonometri

$$\begin{aligned} \cos \frac{\pi}{12} &= \cos \left( \frac{\pi}{3} - \frac{\pi}{4} \right) = \cos \frac{\pi}{3} \cos \frac{\pi}{4} + \sin \frac{\pi}{3} \sin \frac{\pi}{4} \\ &= \left( \frac{1}{2} \right) \left( \frac{1}{\sqrt{2}} \right) + \left( \frac{\sqrt{3}}{2} \right) \left( \frac{1}{\sqrt{2}} \right) = \frac{1 + \sqrt{3}}{2\sqrt{2}} \end{aligned}$$

## 4.2 Gränsvärden

$$\lim_{x \rightarrow x_0} f = L \wedge \lim_{x \rightarrow x_0} g = M \Rightarrow \lim_{x \rightarrow x_0} (f + g) = L + M$$

Squeeze therum:  $h(x) \leq f(x) \leq g(x)$ ,  $\lim_{x \rightarrow x_0} h(x) = \lim_{x \rightarrow x_0} g(x_0) = L \Rightarrow \lim_{x \rightarrow x_0} f(x) = L$

$$\lim_{x \rightarrow 0} \frac{\sin(x)}{x} = 1$$

$\lim_{x \rightarrow \infty} \sin(x)$  är ej definierad

**Example: Limit för sin**

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow 0} \frac{\sin(2x^2)}{x^2} \\ & \lim_{x \rightarrow 0} \frac{2 \cdot \sin(2x^2)}{2x^2} = 2 \cdot \lim_{x \rightarrow 0} \frac{\sin(2x^2)}{2x^2} = 2 \end{aligned}$$

**Example: 0/0**

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow 0} \frac{x \cdot \cos(x) + \frac{\sin(x)}{x}}{x + x^2} \\ & \lim_{x \rightarrow 0} \frac{x \cdot \cos(x) + \frac{\sin(x)}{x}}{x(1 + x)} = \frac{1 + 1}{1} = 2 \end{aligned}$$

**Example: roten ur i nämnaren**

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow 0} \frac{x}{\sqrt{x+9} - 3} \\ & \lim_{x \rightarrow 0} \frac{x(\sqrt{x+9} + 3)}{(\sqrt{x+9} - 3)(\sqrt{x+9} + 3)} = \lim_{x \rightarrow 0} \frac{x(\sqrt{x+9} + 3)}{x+9-9} = \lim_{x \rightarrow 0} \sqrt{x+9} + 3 = 6 \end{aligned}$$

**Example: roten ur i nämnaren**

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow -\infty} \frac{2x-1}{\sqrt{3x^2+x+1}} \\ & \lim_{x \rightarrow -\infty} \frac{x(2-\frac{1}{x})}{\sqrt{x^2(3+\frac{1}{x}+\frac{1}{x^2})}} = \lim_{x \rightarrow -\infty} \frac{x(2-\frac{1}{x})}{x\sqrt{3+\frac{1}{x}+\frac{1}{x^2}}} = \frac{-2}{\sqrt{3}} \end{aligned}$$

**Example: infinity sin and squeeze**

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow \infty} \frac{\sin(x)}{x} \\ & \left| \frac{\sin(x)}{x} \right| = \frac{|\sin(x)|}{|x|} \leq \frac{1}{|x|} \Rightarrow \text{squeeze } \lim_{x \rightarrow \infty} \left| \frac{\sin(x)}{x} \right| = 0 \\ & \Rightarrow \lim_{x \rightarrow \infty} \frac{\sin(x)}{x} = 0 \end{aligned}$$

### 4.2.1 Kontinuitet

en funktion är kontinuerlig i punkt  $x_0$ ,  $\forall \varepsilon \wedge \exists \delta : |x - x_0| < \delta \Leftrightarrow |f(x) - f(x_0)| < \varepsilon$

Altså så finns det inga hopp i funktionen. Man kan dra penan på grafen utan att släppa om functionen bestor av flera uttryck är funktionen kontinuerlig om alla uttrycken är det

## 4.3 Derivator

Tangent: linjen som har samma lutning i en punkt som functionens derivata

$$y - f(x_0) = f'(x_0)(x - x_0) \Rightarrow y = ax + b$$

Samband: *Deriverbar  $\Rightarrow$  Kontinuerlig*

### Räkneregler

$f$	$f'$
$c$	0
$x$	1
$x^n$	$nx^{n-1}$
$a^x$	$a^x \ln a$
$e^{kx}$	$ke^x$
$\sin x$	$\cos x$
$\cos x$	$-\sin x$
$\tan x$	$1 + \tan^2 x \frac{1}{\cos^2 x}$
$\ln x$	$\frac{1}{x}$

$$(f + g)' = f' + g'$$

$$(cf)' = cf'$$

$$(fg)' = f'g + fg' \text{ Produktregeln}$$

$$\left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2} \text{ Kvotregeln, } g \neq 0$$

#### 4.3.1 Kjedje regeln

om functionen  $g$  är deriverbar i  $x$  och functionen är deriverbar i  $g(x)$

$$\Rightarrow h(x) = f \circ g = f(g(x)) \text{ deriverbar i } x$$

$$h'(x) = (f(g(x)))' = f'(g(x))g'(x)$$

#### Example: Kjedje regeln

Lös:  $(x^x)'$

$$(x^x)' = (e^{x \ln x})' = e^{(x \ln x)} (\ln x + x \frac{1}{x}) = e^{x \ln x} (1 + \ln x) = x^x (1 + \ln x)$$

### 4.3.2 L'Hôpital's rule

$$f, g : \mathbb{R} \rightarrow \mathbb{R} \wedge (\lim_{x \rightarrow x_0} \frac{f}{g} = \frac{0}{0} \vee \text{"} \lim_{x \rightarrow x_0} \frac{f}{g} = \frac{\pm\infty}{\pm\infty} \text{"}) \Rightarrow \lim_{x \rightarrow x_0} \frac{f}{g} = \lim_{x \rightarrow x_0} \frac{f'}{g'}$$

**Example:** L'Hôpital's rule

$$\begin{aligned} \text{Lös: } & \lim_{x \rightarrow 1} \frac{x^4 - 6x^3 + 5x^2}{x^3 - 8x^2 + 7x} \\ & \lim_{x \rightarrow 1} \frac{x^4 - 6x^3 + 5x^2}{x^3 - 8x^2 + 7x} = \text{"} \frac{0}{0} \text{"} \\ \text{L'Hôpital's rule} \Rightarrow & \lim_{x \rightarrow 1} \frac{4x^3 - 18x^2 + 10x}{3x^2 - 16x + 7} \\ & = \frac{-4}{-6} = \frac{2}{3} \end{aligned}$$

### 4.3.3 Medelvärdessatsen

Anta att  $f$  är kontinuerlig på  $[a, b]$  och deriverbar på  $(a, b) \Rightarrow \exists c \in (a, b)$

$$\text{så att } \frac{f(b) - f(a)}{b - a} = f'(c)$$

**Example:** Medelvärdessatsen

Bevisa att för varje  $a > b$  gäller följande  $|\cos a - \cos b| \leq |a - b|$

$f(x) = \cos x$  är kont och deriverbar i  $\mathbb{R} \wedge [a, b]$

$$\text{MVS } \Rightarrow \exists c \in (a, b) : f'(c) = \frac{f(b) - f(a)}{b - a} = \frac{\cos(b) - \cos(a)}{b - a} = -\sin(c)$$

$$\Rightarrow \left| \frac{\cos(b) - \cos(a)}{b - a} \right| = |- \sin(c)| \Rightarrow \frac{|\cos(b) - \cos(a)|}{|b - a|} = |- \sin(c)| \leq 1$$

$$\frac{|\cos(b) - \cos(a)|}{|b - a|} \leq 1 \Rightarrow |\cos(b) - \cos(a)| \leq |a - b|$$

### 4.3.4 Rolle

Anta att  $f$  är kontinuerlig på  $[a, b]$  och deriverbar på  $(a, b)$ . Om  $f(a) = f(b) \Rightarrow \exists c \in (a, b) : f'(c) = 0$

### 4.3.5 växande funktioner

- växande:  $x_1 < x_2 \Rightarrow f(x_1) \leq f(x_2)$
- stänkt växande:  $x_1 < x_2 \Rightarrow f(x_1) < f(x_2)$
- avtagande:  $x_1 < x_2 \Rightarrow f(x_1) \geq f(x_2)$
- stänkt avtagande:  $x_1 < x_2 \Rightarrow f(x_1) > f(x_2)$

### 4.3.6 Högreordnings derivator

$$\begin{aligned} \text{andragrads derivator: } (f')' &= f'' = \frac{d^2 f}{dx^2} \\ \text{tredjegrads derivator: } (f'')' &= f''' = \frac{d^3 f}{dx^3} \\ \text{ntregrads derivator: } f^{(n)} &= f^n = \frac{d^n f}{dx^n} \end{aligned}$$

### 4.3.7 Impericit derivering

deriverar båda sidorna om modifierat

#### Example: Impericit derivering

$$\begin{aligned} &\text{Bestäm tangenten till } x^3 + y^3 = 6xy \text{ i } (3,3) \\ &(3^3 + 3^3 = 6 \cdot 3 \cdot 3) \text{-sant} \\ &y - y_0 = f'(x_0)(x - x_0) \Rightarrow y - 3 = f'(3)(x - 3) \\ &\text{Implicit derivering: } 3x^2 + 3y^2 y' = 6y + 6xy' \\ &\Rightarrow 3x^2 y' - 6xy' = 6y - 6x^2 \Rightarrow y'(3x^2 - 6x) = 6y - 3x^2 \Rightarrow y' = \frac{6y - 3x^2}{3x^2 - 6x} \\ &\Rightarrow y'(3) = \frac{63 - 3 \cdot 3^2}{3 \cdot 3^2 - 6 \cdot 3} = -1 \\ &\Rightarrow y - 3 = -(x - 3) \Rightarrow x + y = 6 \vee y = -x + 6 \end{aligned}$$

### 4.3.8 invers funktioner

$$\begin{aligned} f : A \rightarrow B \wedge \text{Bijektiv} &\Rightarrow f^{-1}(x) \text{ Finns, där} \\ (1) \quad x &= f^{-1}(y) \Leftrightarrow y = f(x) \\ (2) \quad D_{f^{-1}} &= V_f \Leftrightarrow D_f = V_{f^{-1}} \\ (3) \quad x &= f^{-1}(f(x)), x \in D_f = V_{f^{-1}} \\ (3) \quad y &= f^{-1}(f(y)), y \in D_f = V_{f^{-1}} \end{aligned}$$

### 4.3.9 exponential och logaritm

$$\begin{aligned} \frac{a^{-3}}{b} &= \frac{b^3}{a} \\ \sqrt{a} &= a^{\frac{1}{2}} \\ a^{x+y} &= a^x a^y \\ (a^x)^y &= a^{xy} \\ a^{-x} &= \frac{1}{a^x} \\ a^{\frac{m}{n}} &= \sqrt[n]{a^m} \end{aligned}$$

(1):  $b = a^x \Leftrightarrow \log_a(b) = x$  för:  $a > 0, b > 0, a \neq 1$

$$(2): \log_a\left(\frac{b}{c}\right) = \log_a(b) - \log_a(c)$$

$$(3): \log_a(b * c) = \log_a(b) + \log_a(c)$$

$$(4): \log_a(b^d) = d \log_a(b)$$

$$(5): \log_a(b) = \frac{\log_f(b)}{\log_f(a)}$$

$$(6): \log_a(a) = 1$$

$$(7): \log_a(1) = 0$$

$$(8): a^{\log_a(x)} = x$$

$$(9): \log_{a^c}(b) = \frac{1}{c} \log_a(b)$$

### 4.3.10 odefinerad form

$$\frac{0}{0}, \infty \cdot 0, 1^\infty, \infty^0, 0^0$$

### 4.3.11 inversa trigometriska funktioner

#### arcsin

$$\cos : \left[ \frac{-\pi}{2}, \frac{\pi}{2} \right] \rightarrow [-1, 1]$$

$$\arcsin 1 = \frac{\pi}{2}$$

$$\arcsin 0 = 0$$

$\arcsin \pi$  = odefinerad

$$\sin(\arcsin(x)) = x$$

$$\arcsin(\sin(x)) = x$$

$$\begin{aligned} (\arcsin(x))' &= \frac{1}{\sin'(\arcsin(x))} = \frac{1}{\cos(\arcsin(x))} = \\ &\frac{1}{\sqrt{\cos^2(\arcsin(x))}} = \frac{1}{\sqrt{1 - \sin^2(\arcsin(x))}} = \frac{1}{\sqrt{1 - x^2}} \end{aligned}$$

#### arccos

$$\cos : [0, \pi] \rightarrow [-1, 1]$$

$$\arccos 1 = 0$$

$$\arccos 0 = \frac{\pi}{2}$$

$\arccos \pi$  = odefinerad

$$\cos(\arccos(x)) = x$$

$$\arccos(\cos(x)) = x$$

$$(\arccos(x))' = \frac{1}{\sqrt{1 - x^2}}$$

#### arctan

$$\cos : \left[ \frac{-\pi}{2}, \frac{\pi}{2} \right] \rightarrow \mathbb{R}$$

$$\tan(\arctan(x)) = x$$

$$\arctan(\tan(x)) = x$$

$$(\arctan(x))' = \frac{1}{1 + x^2}$$

**exempel**

$$\begin{aligned}\tan(\arccos x) &= \frac{\sin(\arccos x)}{\cos \arccos x} = \frac{\sqrt{1-x^2}}{x} \\ \cos(\arctan x) &= \cos \frac{\arcsin x}{\arccos x} = \frac{1}{1+x^2} \\ \sin(\arccos x) &= \sqrt{1-x^2} \\ \cos(\arcsin x) &= \sqrt{1-x^2}\end{aligned}$$

## 4.4 Grafritning

Ta reda på extrem punkterna och rita utifrån det

**Extremvärden**

- global minimum punkt:  $x_0, \forall x : f(x) \geq f(x_0)$
- lokal minimum punkt:  $x_0, \forall x \text{ nära } x_0 : f(x) \geq f(x_0)$
- global maximum punkt:  $x_0, \forall x : f(x) \leq f(x_0)$
- lokal maximum punkt:  $x_0, \forall x \text{ nära } x_0 : f(x) \leq f(x_0)$
- Kritisk punkt:  $f'(x) = 0$

**Komplexity**

- konvex: Tangent liger altid under funktionen  $y'' > 0$
- konkav: Tangent liger altid över funktionen  $y'' < 0$
- Inflektionspunkt: då funktionen byter från konvex till konkav eller tvärtom

**Exempel: Datan som behövs beräknas vid grafritning**

Rita funktionen  $y = (x^2 - 1)^3$

(1) Kollar om funktionen är Konternuelig:

Eftersom funkrionen består av polynom är den konternuerlig

(2) Extrem punkter:

(I) Derivatan: funktionen är deriverbar  $y' = 3(x^2 - 1)^2 \cdot 2x = 0$

$$\Rightarrow x = 0$$

(II) Singulär punkter:

eftersom funktionen är deriverbar i alla punkter i definitions mängden

Så finns det inga singulär punkter

(III) End värdet: Eftersom funktionen är ej definierad i ett intervall så finns det inga

(3) Komplexitet:

Andra derivatan avgör om funktionen är knvex eller konkav

$$y'' = (6x(x^2 - 1)^2)' = (6x(x^4 - 2x^2 + 1))' = (6x^5 - 12x^3 + 6x)' = 30x^4 - 36x^2 + 6 = 0$$

$$\Rightarrow t^2 - \frac{6}{5}t + \frac{1}{5} = 0 \Rightarrow t = \frac{6}{10} \pm \sqrt{\frac{36}{100} - \frac{20}{100}} = \frac{6}{10} \pm \frac{4}{10}$$

$$t = 1 \vee t = \frac{1}{5} \Rightarrow x = \pm 1 \vee x = \pm \frac{1}{\sqrt{5}}$$

(4) Asymptoter:

(I) lodräta asymptoter:

$$\lim_{x \rightarrow +\infty} y = +\infty$$

$$\lim_{x \rightarrow -\infty} y = +\infty$$

(II) vågräta asymptoter:

Eftersom funktionen inte är en kvot finns det inga sådana asymptoter

$f'$	$x < -1$	$x = -1$	$-1 < x < -\frac{1}{\sqrt{5}}$	$x = -\frac{1}{\sqrt{5}}$	$\dots$
$f''$	—	—	—	—	...
$f$	avtagande kokav	0 avtagande inflektions punkt	+ avtagande konvex	0 avtagande inflektions punkt	...

## 4.5 Optemering

Sats: om funktionen  $f(x)$  är knternuerlig på det slutna intervallet  $[a, b]$   
 så antar det sitt största värde och minsta värde där  $\exists x_1, x_2 \in [a, b]$   
 så att  $f(x_1) \leq f(x) \leq f(x_2)$

### Exempel: Max/Min värde

Låt  $f(x) : [-1, 2] \rightarrow \mathbb{R}$ ,  $f(x) = x^3 - 3|x|$

- (1) Konternuerlig:  
 $f$  är konternuerlig i intervallet

- (2) Extrem punkter:

(I) Derivatan: funktionen är deriverbar i intervallet förutom då  $x = 0$

låt  $f$  bestå av  $f_1(x) = x^3 - 3x, x \geq 0 \wedge f_1(x) = x^3 + 3x, x < 0$

$$\Rightarrow f'_1(x) = 3x^2 - 3, x \geq 0 \wedge f'_2(x) = 3x^2 + 3, x < 0$$

$$f'_1(x) = 0 \Rightarrow x = 1 \in [1, 2], f(1) = -2$$

$$f'_2(x) = 0 \Rightarrow x = \text{Odefinerad}$$

(II) Singulär punkter:

$f$  är inte deriverbar då  $x = 0, f(0) = 0$

(III) End punkterna: Eftersom funktionen är ej definerad i ett intervall så finns det inga

$$f(-1) = -4$$

$$f(2) = 2$$

- (3) Största och minsta värdet:

$$f(-1) < f(1) < f(0) < f(2)$$

Svar: Max är 2, min är -4

## 4.6 Talfölder och serier

**Def:** Talföljder

En talföjd är en funktion  $a : \mathbb{N} \rightarrow \mathbb{N}$

Vi skriver  $a_n$  istället för  $a(n)$ ,  $a_2 = a(2)$

Vi säger att  $a_n \rightarrow a \in \mathbb{N}$  så är den **konvergent**

Vi säger att  $a_n \rightarrow \infty$  så är den **divergent** eller ej existerande

Konvergent kan vara begränsad uppåt eller nedåt

Talföljed kan vara växande eller avtagande

$$a_n \rightarrow a \vee b_n \rightarrow b \text{ Omm } a \text{ och } b \text{ existerar } a_n + b_n \rightarrow a + b$$

**Exempel:** Derivatan av serier

$$\text{Ange } f'(2), \quad f(x) = \sum_{n=1}^{\infty} \frac{(x-2)^n}{n^2 2^{2n}}$$

$$\text{Anger den första termen } (\frac{x-2}{4})' = \frac{1}{4}$$

$$f'(x) = \frac{1}{4} + \sum_{n=2}^{\infty} \frac{n(x-2)^{n-1}}{n^2 2^{2n}} \text{ Inre och yttre derivatan}$$

$$f'(2) = \frac{1}{4} + \sum_{n=2}^{\infty} \frac{n(2-2)^{n-1}}{n^2 2^{2n}} = \frac{1}{4} + 0 = \frac{1}{4}$$

**Sats:**

Om  $(a_n)^\infty$  är Konvergent  $\Rightarrow (a_n)$ ,  $n = 1$  är begränsad

**Sats:**

Låt  $a > 0$

(I)  $a^n \rightarrow 0$  om  $a < 1$

(II)  $a^n \rightarrow +\infty$  om  $a > 1$

$$(a_n)_{n=1}^{\infty} = \sum_{k=0}^n a_n = a_1 + a_2 + a_3 + \dots$$

**Sats:** Geometrisk serie

$$s_n = a + ak + ak^2 + \dots + ak^{n-1} = \frac{a(k^n - 1)}{k - 1}$$

$$\sum_{n=0}^{\infty} r^n \Rightarrow$$

Konvergent om  $|r| < 1$

Divergent om  $|r| > 1$

**Sats:**

$$\sum_{n=0}^{\infty} a_n \text{ konvergent} \Rightarrow a_n \rightarrow 0$$

**Sats: p-serie**

$$\sum_{n=0}^{\infty} \frac{1}{n^p} \Rightarrow$$

konvergent om  $\Rightarrow p > 1$   
divergent om  $\Rightarrow p \leq 1$

**Sats:**

Antag att  $0 \leq a_n \leq b_n$  för varje  $n \in \mathbb{N}$

$$\sum_{n=1}^{\infty} a_n \text{ konvergent} \Rightarrow a_n \rightarrow 0$$

$(I) \sum_{n=1}^{\infty} b_n \text{ konvergerar} \Rightarrow \sum_{n=1}^{\infty} a_n \text{ konvergerar}$

$(II) \sum_{n=1}^{\infty} a_n \text{ divergerar} \Rightarrow \sum_{n=1}^{\infty} b_n \text{ divergerar}$

**Sats:**

Antag att  $a_n > 0, b_n > 0$  för varje  $n \in \mathbb{N}$  och antag att  $\frac{a_n}{b_n} \rightarrow L \neq 0 \wedge L < \pm\infty$

Då gäller att serien  $\sum_{n=1}^{\infty} a_n$  och  $\sum_{n=1}^{\infty} b_n$

är båda divergenta

**Sats: kvotkriterium**

$$\text{Låt } a_n > 0 \wedge \lim_{n \rightarrow \infty} \frac{a_{n+1}}{a_n} \rightarrow L \Rightarrow$$

$(I) 0 \leq L \leq 1 \Rightarrow \text{Konvergerar } \sum_{n=1}^{\infty} a_n (a_n \rightarrow 0)$

$(II) L > 1 \Rightarrow \sum_{n=1}^{\infty} a_n \text{ divergerar } (a_n \rightarrow +\infty)$

**Exempel:** Måste kunna

$$\sum_{n=1}^{\infty} \frac{5^n n!}{n^n}$$

Låt  $a_n = \frac{5^n n!}{n^n} \Rightarrow \frac{a_{n+1}}{a_n} = \frac{5^{n+1}(n+1)!/n^n}{5^n n!/n^n} = 5 \frac{(n+1)n^n}{(n+1)^{n+1}} = 5 \left(\frac{n}{n+1}\right)^n = 5 \frac{1}{\left(1 + 1/n\right)^n} \rightarrow \frac{5}{e} > 1 \Rightarrow$  derigerar

**Sats:** Rotkriteriet (Ovanlig)

$$\text{Låt } a_n > 0 \wedge \lim_{n \rightarrow \infty} \sqrt[n]{a_n} \rightarrow L \Rightarrow$$

(I)  $0 \leq L \leq 1 \Rightarrow$  Konvergerar  $\sum_{n=1}^{\infty} a_n (a_n \rightarrow 0)$

(II)  $L > 1 \Rightarrow \sum_{n=1}^{\infty} a_n$  derigerar ( $a_n \rightarrow +\infty$ )

#### 4.6.1 Serier med varierande tecken

**Sats:** leibinz

Antag att  $\sum_{n=1}^{\infty} a_n$  är en alternerande serie  
och att om  $|a_n| \rightarrow 0 \wedge |a_n| \geq |a_{n+1}| \Rightarrow$  konvergent

**Def:** absolut konvergent

Endast om  $\sum_{n=1}^{\infty} |a_n|$  konvagerar  $\Rightarrow$  då är den absolutkonvergent  
om seriern är absolutkonvergent så är den också konvergent behöver inte vara tvärtom

#### 4.6.2 Potensserier

**Def:** Potensserier

En potensserier kring  $x_0$  är en serie på formen  $\sum_{n=1}^{\infty} a_n (x - x_0)^n$ ,  $a_n \in \mathbb{R}$  (koefficienten)  
 $x$  är en variabel

**Def: Konvergens radie**

$$T(x) = \sum_{n=1}^{\infty} \frac{f(x_0)^{(n)}}{n!} (x - x_0)^n$$

vid exempelvis grad 2 så blir det andra derivatan man räknar ut

$$R = L^{-1} = \frac{1}{L} \quad (\text{L är där serien konvergerar, R är konvergens radie}) \sum_{n=1}^{\infty} a_n (x - x_0)^n, \quad a_n \in \mathbb{R} \quad (\text{koefficienten})$$

$x$  är en variabel

Om konvergens radien är noll så finns det inte några intervar för konvergens eller divergens

**Sats: Potensserier**

En potensserie kring  $x_0$  är en serie på formen  $\sum_{n=1}^{\infty} a_n (x - x_0)^n$   $a_n \in \mathbb{R}$   $a_n$  koficienten  $x$  är en variabel

När det står ex  $(3x + 2)^n$  så måste det stå med i  $a_n, 3^n$

**Exempel: Potensserier**

$\sum_{n=1}^{\infty} \frac{(x - 7)^n}{n(n+1)}$  är en potensserie där  $x_0 = 7$ ,  $a_n = \frac{1}{n(n+1)}$

$$\left| \frac{a_{n+1}}{a_n} \right| = \left| \frac{1/((n+1)(n+2))}{1/(n(n+1))} \right| = \left| \frac{n}{n+2} \right| = \left| \frac{n}{n(1+2/n)} \right|$$

$$\Rightarrow \left| \frac{1}{1} \right| = 1, \quad \text{då } n \rightarrow \infty$$

$$\Rightarrow R = \frac{1}{1} = 1 \Rightarrow \text{vi får följande intervall av potenserien}$$

Absolutkonverget för  $|x - 7| < 1 \Leftrightarrow 6 < x < 8$

Divergent för  $|x - 7| > 1 \Leftrightarrow 8 < x \vee x < 6$

Testar edge fallen  $x = 6, x = 8$

$$\sum_{n=1}^{\infty} \frac{(8 - 7)^n}{n(n+1)} \Rightarrow \left| \sum_{n=1}^{\infty} \frac{(1)^n}{n(n+1)} \right|, \quad \left| \frac{(1)^n}{n(n+1)} \right| \rightarrow 0 \Rightarrow \text{absolutkonvergerar}$$

$$\sum_{n=1}^{\infty} \frac{(6 - 7)^n}{n(n+1)} \Rightarrow \left| \sum_{n=1}^{\infty} \frac{(-1)^n}{n(n+1)} \right|, \quad \left| \frac{(-1)^n}{n(n+1)} \right| \rightarrow 0 \Rightarrow \text{absolutkonvergerar}$$

**Svar:**

Absolutkonverget för  $6 \leq x \leq 8$

Divergent för  $8 < x \vee x < 6$

### 4.6.3 Taylor serier

**Def:** Talorpolynom

Taylorpolynom av grad  $n \in \mathbb{N}$ , kring  $x = x_0$

$$f(x) = P_n(x) = \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k, \quad 0! = 1, f^{(0)} = f$$

När  $x \approx x_0 \Rightarrow f(x) \approx p_n(x)$

**Sats:** Talor sats

Antag att  $f \in C^{n+1}$ .

$f(x) = p_n(x) + R_{n+1}(x)$  Där felet är  $R_{n+1}(x)$

$$R_{n+1}(x) = \frac{f^{(n+1)}(c)}{(n+1)!} (x - x_0)^{n+1}, \quad x \vee x_0 \leq c \leq x_0 \vee x$$

**exempel:** felet

Upskata felet hos  $f(x) = \sin x$ , med grad 5

Sista termen är  $\frac{x^5}{5!} \Rightarrow$  Felet  $\frac{f^{(6)}(c)}{6!} x^6, (0 < c < 1)$

$$\left| \frac{f^{(6)}(c)}{6!} x^6 \right| \leq \frac{|f^{(6)}(c)|}{6!} = \frac{|\sin c|}{6!} \leq \frac{1}{6!} = \frac{1}{720}$$

**Exempel:** grad n vid specificerad punkt

Lös: Hitta Taylorpolynomet, ording  $2n - 1$ , för  $\sin 2x$  vid  $x = \frac{\pi}{2}$

$$f(x) = \sin(2x)$$

$$f\left(\frac{\pi}{2}\right) = 0$$

$$f'(x) = -2 \cos(2x)$$

$$f\left(\frac{\pi}{2}\right) = -2$$

$$f''(x) = -2^2 \sin(2x)$$

$$f\left(\frac{\pi}{2}\right) = 0$$

$$f^{(3)}(x) = -2 \cos(2x)$$

$$f\left(\frac{\pi}{2}\right) = 2^3$$

$$f^{(4)}(x) = -2^2 \sin(2x)$$

$$f\left(\frac{\pi}{2}\right) = 0$$

$$f^{(5)}(x) = -2^4 f'(x)$$

$$f\left(\frac{\pi}{2}\right) = -2^5$$

$$p_{2n-1}(x) = -2\left(x - \frac{\pi}{2}\right) + \frac{2^3}{3!}\left(x - \frac{\pi}{2}\right)^3 - \frac{2^5}{5!}\left(x - \frac{\pi}{2}\right)^5 + \dots + \frac{(-1)^n \cdot 2^{2n-1}}{(2n-1)!}\left(x - \frac{\pi}{2}\right)^{(2n-1)}$$

**Def: maclaurin polynomial**

Taylorpolynom av grad  $n \in \mathbb{N}$ , då  $x_0 = 0$

Big-O notation: Vi säger att  $g(x) = O(f(x))$  när  $x \approx x_0$ , värsta fall

Man kan också säga istället för Big-O notation en rest  $R_{n+1}(x) = x^{n+1}H(x)$  där  $H(x)$  begränsad i intervallet

$$e^x = \sum_{k=0}^n \frac{x^k}{k!} + O(x^{n+1})$$

$$\sin x = \sum_{k=0}^n \frac{(-1)^{k-1}}{(2k-1)!} x^{2k-1} + O(x^{2n})$$

$$\cos x = \sum_{k=0}^n \frac{(-1)^k}{(2k)!} x^{2k} + O(x^{2n+1})$$

$$\ln(1+x) = \sum_{k=0}^n \frac{x^k}{k} + O(x^{n+1})$$

## 4.7 Integraler

**Regler no.1**

$$\int_a^a f(x)dx = 0$$

**Regler no.2**

$$\int_a^b f(x)dx = - \int_b^a f(x)dx$$

**Regler no.3 (Linjearitet)**

$$A, B \in \mathbb{R}$$

$$\int_a^b (A \cdot f(x)dx + B \cdot g(x)) = A(\int_a^b f(x)dx) + B(\int_a^b g(x)dx)$$

Fungerar på subtraction men INTE multiplicatione eller divition!

**Regler no.4**

$$c \in [a, b]$$

$$\int_a^b f(x)dx = \int_a^c f(x)dx + \int_c^b f(x)dx$$

**Regler no.5**

$$f(-x) = -f(x) \text{ är udda}$$

$$\int_{-a}^a f(x)dx = 0$$

**Regler no.6**

$$f(-x) = f(x) \text{ är jämn}$$

$$\int_{-a}^a f(x)dx = 2 \int_0^a f(x)dx$$

**Regler no.7**

$$\left| \int_a^b f(x)dx \right| \leq \int_a^b |f(x)|dx$$

**Regler no.8**

$$f(x) \leq g(x) \Rightarrow \int_a^b f(x)dx \leq \int_a^b g(x)dx$$

**Sats:** Medelvärdessatsen för integraler

$$f : [a, b] \rightarrow \mathbb{R} \text{ konternuelig } \wedge \exists c \in (a, b) \Rightarrow \int_a^b f(x)dx = f(c)(b - a)$$

**Sats:** Fanalysens huvudsats, Fundemetal theorem of calculus

Satsen är den vi använder för att lösa integraler utan geometrisk tolkning

$$f : [a, b] \rightarrow \mathbb{R} \text{ konternuelig } \Rightarrow F(x) = \int_a^x f(t)dt, a \leq x \leq b$$

**Sats:**

$$\begin{aligned} f : [a, b] \rightarrow \mathbb{R} \text{ konternuelig } &\Rightarrow F \text{ är e primitiv funktion till } f(F'(x) = f(x)) \\ \int_a^b f(x)dx &= F(b) - F(a) \\ \text{Vi skriver } \int_a^b f(x)dx &= [F(x)]_a^b = F(b) - F(a) \end{aligned}$$

**Exempel:** arean mellan två grafer

$$\begin{aligned} \int_a^b (f(x) - g(x))dx &\text{ där f är översta funnktionen och g är understa} \\ \int_0^1 (x - x^2)dx &= [x^2/2 - x^3/3]_0^1 = 1/2 - 1/3 = 1/6 \end{aligned}$$

**Primitiva funktioner (Obestämda integralen)**

$\int f dx$	$F$
$\int 1 dx$	$x + c$
$\int n^n dx$	$x^{n+1}/(n+1) + c$
$\int 1/xdx$	$\ln x  + c$
$\int \sin x dx$	$-\cos x + c$
$\int \cos x dx$	$\sin x + c$
$\int e^x dx$	$e^x + c$
$\int 1/\sqrt{1-x^2} dx$	$\arcsin x + c$
$\int 1/(x^2+1) dx$	$\arctan x + c$
$\int 1/\cos^2 x dx$	$\tan x + c$

**Exempel:**

$$\begin{aligned} \int_a^b e^x dx &= [e^x]_a^b = e^b - e^a \\ \int_e^x dx &= e^x + c, c \in \mathbb{R} \end{aligned}$$

**Def:** medelvärdet

$$Avgf = \frac{1}{b-a} \int_a^b f dx$$

### 4.7.1 variabelsubstitution

$$\int f'(g(x))g'(x)dx = f(g(x)) + c$$

$$\int_e^x dx = e^x + c, \quad c \in \mathbb{R}$$

**Exempel:**

$$\int e^{x^2} 2x dx$$

Låt  $u = x^2 \Rightarrow du = 2x dx \Rightarrow \int e^{x^2} 2x dx = \int e^u du =$

$$= e^u + c = e^{x^2} + c, \quad c \in \mathbb{R}$$

### Integraler som följer mönster

Integralles som inehåller  $\sqrt{x^2 + a^2}, \quad a > 0$

$$x = a \tan u \Rightarrow u = \arctan x, \quad -\pi/2 < u < \pi/2$$

$$\Rightarrow \sqrt{x^2 + a^2} = \sqrt{a^2 + a^2 \tan^2 u} = \sqrt{a^2(1 + \tan^2 u)} = a \sqrt{\frac{\sin^2 u + \cos^2 u}{\cos^2 u}} = a \frac{1}{\cos u}$$

Exempel:  $\int \frac{1}{(1 + 25x^2)^{3/2}} dx$

$$\int \frac{dx}{\sqrt{1 + 25x^2}^3} = \int \frac{dx}{\sqrt{25(1/25 + x^2)}^3} = \frac{1}{5^3} \int \frac{dx}{\sqrt{1/205x^2}}$$

Låt  $x = 1/5 \tan u \Leftrightarrow u = \arctan 5x$

$$\Rightarrow dx = \frac{1}{5 \cos^2 u} du$$

$$\Rightarrow \frac{1}{5^3} \int \frac{dx}{\sqrt{1/205x^2}} = \frac{1}{5^3} \int \frac{\cos^3 u}{1/5^3} \frac{1}{5 \cos^2 u} du = \frac{1}{5} \int \cos u du =$$

$$= \frac{1}{5} \sin u + c = \frac{1}{5} \sin \arctan 5x + c \quad \text{Kan nu förenkla med trigonometriska regler}$$

och få:  $\frac{x}{\sqrt{1 + 25x^2}} + c$

Integralles som inehåller  $\sqrt{x^2 - a^2}, \quad a > 0$

$$x = \frac{a}{\cos u} \Rightarrow u = \arccos \frac{a}{x}$$

$$\Rightarrow \sqrt{x^2 - a^2} = \sqrt{\frac{a^2}{\cos^2 u} - a^2} = \sqrt{\frac{a^2 - a^2 \cos^2 u}{\cos^2 u}} = \sqrt{\frac{a^2 \sin^2 u - a^2 \cos^2 u}{a^2 \cos^2 u}} = a |\tan u|$$

Integralles som innehåller  $\sqrt{ax + b}$

$$ax + b = u^2$$

Exempel:  $\int \frac{dx}{2 + \sqrt{x}}$

Låt:  $u^2 = x \Rightarrow 2udu = dx$

$$\Rightarrow \int \frac{2udu}{2 + u} = 2 \int \frac{u+2-2}{2+u} du = 2\left(\int du - 2 \int \frac{du}{2+u}\right) = 2u - 4(\ln 2 + u) + c$$

### 4.7.2 Integration av rationella funktioner

Om det står beräkna generaliseringen så ska man beräkna med detta

Rationella funktioner:  $R(x) = \frac{P(x)}{Q(x)}$ ,  $P, Q$  är polynom

Algoritm

1. Polynom division om grad av täljare är större än grad av nämnare
2. Faktorisera nämnaren i reala faktorer
3. Partiella bråk och sedan integrera dem, (dela upp i mindre lösbara integraler)

#### Partiella bråk

$x - a$	$\frac{A}{x-a}$
$(x - a)^k$	$\frac{A_1}{x-a} + \frac{A_2}{x-a^2} + \dots + \frac{A_k}{x-a^k}$
$x^2 + bx + c, (b^2 - 4c < 0)$	$\frac{Ax+B}{x^2+bx+c}$
$x^2 + bx + c, (b^2 - 4c < 0)$	$\frac{A_1x+B_1}{x^2+bx+c} + \frac{A_2x+B_2}{(x^2+bx+c)^2} + \dots + \frac{A_kx+B_k}{(x^2+bx+c)^k}$

$$\int \frac{1}{ax+b} dx = \frac{1}{a} \ln |ax+b| + c$$

$$\int \frac{x}{x^2+a^2} dx = \frac{1}{2} \ln |x^2+a^2| + c$$

$$\int \frac{x}{x^2-a^2} dx = \frac{1}{2} \ln |x^2-a^2| + c$$

$$\int \frac{dx}{x^2+a^2} = \frac{1}{a} \tan^{-1} \frac{x}{a} + c$$

$$\int \frac{dx}{x^2-a^2} = \frac{1}{2a} \ln \frac{x-a}{x+a} + c$$

$$\int \frac{x^3 + 2}{x^2 - 5x + 4} dx$$

1. Polynomdivision

$$P(x) = x^3 + 2, Q(x) = x^2 - 5x + 4$$

Med polynom division så får vi kvot  $x + 5$  och rest  $21x - 18$

$$x^3 + 2 = (x + 5)(x^2 - 5x + 4) + 21x - 18$$

2. Faktorisera nämnaren i reala faktorer

$$\Rightarrow \int \frac{x^3 + 2}{x^2 - 5x + 4} dx = \int \frac{(x + 5)(x^2 - 5x + 4) + 21x - 18}{x^2 - 5x + 4} dx = \int (x + 5)dx + \int \frac{21x - 18}{x^2 - 5x + 4}$$

3. Integrerar partiella bråk

$$\frac{21x - 18}{x^2 - 5x + 4} = \frac{21x - 18}{(x - 4)(x - 1)} = \frac{A}{x - 4} + \frac{B}{x - 1} \Rightarrow \frac{21x - 18}{(x - 4)(x - 1)} = \frac{Ax - A + Bx - 4B}{(x - 4)(x - 1)}$$

$$\Rightarrow A + B = 21 \wedge -A - 4B = -18 \Rightarrow A = 22 \wedge B = -1$$

$$\int (x + 5)dx + \int \frac{22}{x - 4} dx - \int \frac{1}{x - 1} dx$$

Svar:  $x^2/2 + 5x + 22 \ln|x - 4| - \ln|x - 1| + c, c \in \mathbb{R}$

### 4.7.3 Partiell integration

Om det är integraler med två funktioner så hjälper oftast partiell integration

#### Formel

$$\int_a^b f' g dx = [fg]_a^b - \int_a^b fg' dx$$

$$\int f' g dx = fg - \int fg' dx$$

#### Bevis

$$(f(x)g(x))' = f'(x)g(x) + f(x)g'(x) \Rightarrow$$

$$\int_a^b (fg)' dx = \int_a^b f' g dx + \int_a^b fg' dx \Rightarrow \int_a^b f' g dx = \int_a^b (fg)' dx - \int_a^b fg' dx$$

$$\int_a^b f' g dx = [fg]_a^b - \int_a^b fg' dx$$

#### Exempel: polynom \* trigonomotrik

$$\int_0^{\pi/2} x \sin x dx$$

$$\int_0^{\pi/2} (-\cos x)' x dx = [-x \cos x]_0^{\pi/2} + \int_0^{\pi/2} \cos x (x)' dx =$$

$$[-x \cos x]_0^{\pi/2} + [\sin x]_0^{\pi/2} = 1$$

#### Exempel: polynom \* exponensiel

$$\int e^x x^2 dx$$

$$\int (e^x)' x^2 dx = [e^x x^2] - \int e^x 2x dx = e^x x^2 - 2 \int (e^x)' x dx = e^x x^2 - 2e^x x + 2 \int (e^x)' dx = e^x x^2 - 2e^x x + 2e^x + c$$

#### Exempel: exponensiel \* trigonomotrik

$$\int e^x \cos x dx$$

Låt  $I = \int e^x \cos x dx = \int (e^x)' \cos x = e^x \cos x - \int e^x \sin x dx = e^x \cos x + e^x \sin x - \int e^x \cos x dx \Rightarrow$

$$I = e^x \cos x + e^x \sin x - I \Rightarrow I = \frac{e^x \cos x + e^x \sin x}{2} + c$$

#### Exempel: bonus ln

$$\int_1^2 \ln x dx$$

$$\int_1^2 x' \ln x dx = \dots = 2 \ln 2 - 1$$

#### 4.7.4 Generaliseringar av integraler

**Def:**

Antag att  $f$  är kontinuerlig i  $a, b$  och att  $\lim_{x \rightarrow a^+} f(x) = \infty$

Vi definierar den generaliserade integralen  $\int_a^b f(x)dx = \lim_{\varepsilon \rightarrow 0^+} \int_{a+\varepsilon}^b f(x)dx, \dots, konstingst$

Om gränsvärdet existerar och är ändlig säger vi att integralen är konvergent, annars är den divergent

#### Beteckning

$\varepsilon$  Andvänds för små tal  $\lim_{\varepsilon \rightarrow 0^+}$

$M$  Andvänds för stora tal  $\lim_{M \rightarrow +\infty}$

$c$  Andvänds för konstanter  $+c$

#### Sats:

$$\int_a^{+\infty} \frac{dp}{x^p}, \quad a > 0$$

$p > 1 \Rightarrow$  Konvergerar

$p \leq 1 \Rightarrow$  Divergerar

#### Sats: Jämförelsesatsen

Anta att  $f$  och  $g$  är konternuerliga och  $0 \leq f(x) \leq g(x), (a \in [-\infty, +\infty), b \in (-\infty, \infty])$

(I) om integralen är konvergent  $\int_a^b g(x)dx \Rightarrow \int_a^b f(x)dx$  är också konvergent

(II) om integralen är divergent  $\int_a^b f(x)dx \Rightarrow \int_a^b g(x)dx$  är också divergent

#### Sats:

Om  $f(x)$  är positiv konternuerlig och avtagande i intervallet  $x \geq N$ ,

så är serien  $\sum_{n=N}^{\infty} f(n)$  konvergent

precis när  $\int_N^{\infty} f(x)dx$  är konvergent

**Exempel:**

Betämm om serien konvergerar eller divergerar  $\sum_{n=10}^{\infty} \frac{1}{n \ln n (\ln(\ln n))^2}$

$$f(x) = \frac{1}{x \ln x (\ln(\ln x))^2}$$

är positiv, konternuerlig och avtagande då nämnaren är stängt positivtökande för  $x \geq 10$

$$\int_{10}^{+\infty} f(x) dx = \lim_{M \rightarrow +\infty} \int_{10}^M f(x) dx, (u = \ln \ln x \Rightarrow du = 1/\ln x \cdot 1/x dx) \Rightarrow \lim_{M \rightarrow +\infty} \int_{\ln \ln 10}^{\ln \ln M} \frac{1}{u^2}$$

$$\lim_{M \rightarrow +\infty} [-1/u]_{\ln \ln 10}^{\ln \ln M} = \lim_{M \rightarrow +\infty} (-1/\ln \ln M + 1/\ln \ln 10) = 1/\ln \ln M$$

$$\sum_{n=10}^{\infty} \frac{1}{n \ln n (\ln(\ln n))^2} \text{ konvergerar}$$

**Sats:**

Antag att  $a_n > 0, b_n > 0$  för varje  $n \in \mathbb{N}$  och antag att  $\frac{a_n}{b_n} \rightarrow L \neq 0 \wedge L < \pm\infty$

Då gäller att serien  $\int_{n=1}^{\infty} a_n$  och  $\int_{n=1}^{\infty} b_n$  är båda divergenta

### 4.7.5 Volymberäkningar

$$V = \int_a^b A(x)dx \text{ Rotationsvolymer runt x-axeln} \Leftrightarrow \pi \int_a^b ((g(x))^2 - (f(x))^2) dx \text{ g är övre f är undre}$$

$$V = 2\pi \int_a^b xf(x)dx \text{ Rotationsvolymer runt y-axeln} \Leftrightarrow 2\pi \int_a^b x(g(x) - f(x))dx \text{ g är övre f är undre}$$

**exempel**

Beräkna volymen av den kropp som uppstår när området som begränsas av kurvan  $y = 4x - x^2 - 3$  och x-axeln roteras kring y-axeln

$$\begin{aligned} y = 0 &\Leftrightarrow 4x - x^2 - 3 = 0 \Leftrightarrow x = 1 \vee x = 3 \\ &\wedge y > 0, x \in [1, 3] \\ \Rightarrow V &= 2\pi \int_1^3 x(4x - x^2 - 3)dx = 2\pi \int_1^3 (4x^2 - 3x - x^3)dx \\ &= 2\pi [4/3x^3 - 3/2x^2 - x^4/4]_1^3 = 16\pi/3 \end{aligned}$$

### Kurvängd

$$\begin{aligned} L &= \|\vec{x}_0 - \vec{x}_1\| = \sqrt{(a-c)^2 + (b-d)^2} \\ \Rightarrow L &= \int_a^b \sqrt{1 + (f'(x))^2} dx \end{aligned}$$

**exempel**

Find the length of curve  $y = x^3/12 + 1/x$  from  $x = 1$  to  $x = 4$

$$\begin{aligned} f(x) &= x^3/12 + 1/x \\ L &= \int_1^4 \sqrt{1 + (f'(x))^2} dx = \int_1^4 \sqrt{1 + (x^2/4 - 1/x^2)^2} dx \\ &= \int_1^4 \sqrt{1 + x^4/16 - 1/2 + 1/x^4} dx = \int_1^4 \sqrt{x^4/16 + 1/x^4 + 1/2} dx \\ &= \int_1^4 \sqrt{(x^2/4 + 1/x^2)^2} dx = \int_1^4 (x^2/4 + 1/x^2) dx \\ &= [x^2/4 + 1/x^2]_1^4 = 6 \end{aligned}$$

**Rotationsarea**

$$A = \int_a^b 2\pi|f(x)|\sqrt{1 + (f'(x))^2}dx$$

**exempel: Gabriel's Horn/Torricell's trumpet**

Bestäm volymen och arean av den kropp som uppstår när området som begränsas av kurvan  $y = 1/x, x \geq 1$  och x-axeln roteras kring x-axeln

$$V = \pi \lim_{M \rightarrow +\infty} \int_1^M \left(\frac{1}{x}\right)^2 dx = \pi \lim_{M \rightarrow +\infty} [-1/x]_1^M = \pi \cdot 1 = \pi$$

$$\begin{aligned} A &= \lim_{M \rightarrow +\infty} \int_1^M 2\pi|1/x|\sqrt{1 + (-1/x^2)^2}dx = 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{\sqrt{1 + 1/x^4}}{x} dx \\ &= 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{x^2 \sqrt{1 + 1/x^4}}{x^3} dx = 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{\sqrt{x^4 + 1}}{x^3} dx \\ &> 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{\sqrt{x^4}}{x^3} dx = 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{x^2}{x^3} dx \\ &= 2\pi \lim_{M \rightarrow +\infty} \int_1^M \frac{1}{x} dx \text{ Diveigerar enligt p-satsen till } \infty \end{aligned}$$

Eftersom integralen har en undre begränsning som divergerar även divergerar också integralen till  $+\infty$

## 4.8 Differential ekvationer

Tangent plan (Slope field): Andvänds för att se hur kurvan ser ut utefrån olika start värden.

grad: högsta graden på derivatan  $ty'''(t) - 4y'(t) + 5t^2y(t) = e^t$  har grad 3

Linjär diffirential ekvation: diff funktionerna har ingen upphöjning

$y'' - 4t^2t' + e^t y = 0$  är linjär

$t^2y''' + 5ty' - 4y^2 = 5$  är inte linjär

Homogen:  $omh(t) = 0 \Rightarrow$  ODE är homogen

Inhomogen:  $omh(t) \neq 0 \Rightarrow$  ODE är inhomogen

$y''' - \sin^2(t)y' = 5y$  är homogen

$e^{t^2}y^{(5)} - y'' + 4ty(t) = e^t + t^3$  är inhomogen

### Sats: superposition principle

Låt  $a_n y^{(n)} + a_{n-1} y^{(n-1)} + \dots + a_1 y' + a_0 y = 0$

Om  $y_1(x), y_2(x)$  är lösningar till differentials ekvanationen

$\Rightarrow Ay_1(x) + By_2(x)$ ,  $A, b \in \mathbb{R}$  är en lösning till differentials ekvanationen

### 4.8.1 superabla ekvationer

**Def:** superabla ekvationer

En differentialekvation är separabel om den kan skrivas på formeln

$$\frac{dy}{dx} = f(x)g(y)$$

**Exempel: Vilka är separabel**

(I)  $y' = x + y$  är inte separabel

(II)  $\frac{dy}{dx} = 1 + e^y$  är separabel

**Exempel: beräkning**

$$\text{Lös } y'(x) = (1 + e^{-x})(y^2 - 1)$$

$$y = \pm 1$$

$$\begin{aligned} \frac{dy}{y^2 - 1} &= (1 + e^{-1})dx \Rightarrow \int \frac{dy}{y^2 - 1} = \int (1 + e^{-1})dx (*) \\ \int \frac{1}{(y+1)(y-1)} dy &= \int \frac{((y+1) - (y-1))}{2(y+1)(y-1)} dy = \int \frac{1}{2(y-1)} dy - \int \frac{1}{2(y+1)} dy \\ (*) \Rightarrow \int \frac{1}{2(y-1)} dy - \int \frac{1}{2(y+1)} dy &= \int (1 + e^{-1})dx \\ \Rightarrow 1/2 \ln |y-1| - 1/2 \ln |y+1| &= x - e^{-x} + c \Rightarrow \ln \left| \frac{y-1}{y+1} \right| = 2x - 2e^{-x} + 2c \\ \Rightarrow e^{1/2 \ln \left| \frac{y-1}{y+1} \right|} &= e^{2x - 2e^{-x} + 2c} \\ \Rightarrow y = \frac{1 + e^{2x - 2e^{-x}} + 2c}{1 - e^{2x - 2e^{-x}} + 2c} \vee y &= \frac{1 - e^{2x - 2e^{-x}} + 2c}{1 + e^{2x - 2e^{-x}} + 2c} \vee y = \pm 1 \end{aligned}$$

### 4.8.2 Linjära differentialekvationer av ordning 1

#### Methodology

$$y' + p(x)y = q(x)$$

$\text{Om } g(x) = 0 \Rightarrow \text{homogen och därmed separable}$

$\text{Om } g(x) \equiv 0 \text{ Multiplisera med } e^{M(x)}$  För att kuna använda produktregeln så vi kan slå ihop  $y, y'$

$$M(x) = \int p(x)dx \text{ är en primitiv funktion till } p(x) \Rightarrow c = 0$$

$$\Rightarrow e^{M(x)}y' + e^{M(x)}p(x)y(x) = e^{M(x)}q(x) \text{ Antifunktionen slår ut } p(x)$$

$$\Rightarrow e^{M(x)}y' + (e^{M(x)})'y(x) = e^{M(x)}q(x) \text{ VL kan vi använda produktregeln bakvänt}$$

$$\Rightarrow (e^{M(x)}y(x))' = e^{M(x)}q(x) \text{ Integrerar båda led}$$

$$\Rightarrow \int (e^{M(x)}y(x))' dx = \int (e^{M(x)}q(x)) dx \text{ Antiderivatan slår ut derivatan}$$

$$\Rightarrow e^{M(x)}y(x) = \int (e^{M(x)}q(x)) dx \text{ Får } y \text{ ensamt}$$

$$\Rightarrow y(x) = e^{-M(x)} \int (e^{M(x)}q(x)) dx$$

#### Exempel

$$\text{Lös } (1+t^2)y' + ty = \frac{1}{\sqrt{1+t^2}}$$

$$y' + \frac{t}{(1+t^2)y} = \frac{1}{(1+t^2)\sqrt{1+t^2}} \text{ Linjör ode, ordning 1}$$

$$p(t) = \frac{t}{(1+t^2)y}, q(t) = \frac{1}{(1+t^2)\sqrt{1+t^2}}$$

$$M(t) = \int \frac{t}{(1+t^2)y} dt, \text{ Låt } u = 1+t^2 \Rightarrow dt = du/2t$$

$$\Rightarrow M(t) = \int \frac{1}{2u} du = \frac{1}{2} \ln |u| = \frac{1}{2} \ln t^2 + 1$$

$$e^{M(t)} = e^{\frac{1}{2} \ln t^2 + 1} = \sqrt{t^2 + 1}$$

$$\Rightarrow \sqrt{t^2 + 1}y' + \frac{t\sqrt{t^2 + 1}}{t^2 + 1}y = \frac{1}{1+t^2}$$

$$\Rightarrow \sqrt{t^2 + 1}y' + \frac{t}{\sqrt{t^2 + 1}}y = \frac{1}{1+t^2}$$

$$\Rightarrow (\sqrt{t^2 + 1}y)' = \frac{1}{1+t^2} \Rightarrow \sqrt{t^2 + 1}y = \int \frac{1}{1+t^2}$$

$$\Rightarrow \sqrt{t^2 + 1}y = \arctan(t) + c$$

$$\Rightarrow y = \frac{\arctan(t)}{\sqrt{t^2 + 1}} + \frac{c}{\sqrt{t^2 + 1}}, c \in \mathbb{R}$$

### 4.8.3 Linjära differentialekvationer av ordning 2

#### Methodology

Det finns 3 metoder för att lösa Linjära differentialekvationer av ordning 2

$ay'' + by' + cy = 0$  vilket är den skäneräla formeln. Antag att lösningen är på formen

$$y = e^{rx}, \quad y' = re^{rx}, \quad y'' = r^2e^{rx} \Rightarrow ar^2e^{rx} + bre^{rx} + ce^{rx} = 0$$

$(ar^2 + br + c)e^{rx} = 0 \Rightarrow ar^2 + br + c = 0$  användar pq-formeln och får följande

$$r = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

(i) Om  $b^2 - 4ac > 0 \Rightarrow r_1, r_2 \in \mathbb{R} \wedge r_1 \neq r_2 \Rightarrow y_1 = e^{r_1 x}, y_2 = e^{r_2 x}$

är lösningen till  $ay'' + by' + cy = 0 \Rightarrow y_k = c_1 e^{r_1 x} + c_2 e^{r_2 x}, c_1, c_2 \in \mathbb{R}$  superposition

(ii)  $b^2 - 4ac = 0 \Rightarrow r_1 = r_2 \in \mathbb{R}$  dubbelrot  $y_1 = e^{r_1 x}, y_2 = xe^{r_1 x}$

(iii) Om  $b^2 - 4ac < 0 \Rightarrow r_1 \neq r_2 \in \mathbb{C}$

$r_1 = k + li, r_2 = k - li$  Koplextals konjugat enligt Euler formel, alla lösningar är

$$y = c_1 e^{r_1 x} + c_2 e^{r_2 x}, c_1, c_2 \in \mathbb{C}$$

$$y = c_1 e^{r_1 x} + c_2 e^{r_2 x} = c_1 e^{(k+li)x} + c_2 e^{(k-li)x} = c_1 e^{kx} e^{lix} + c_2 e^{kx} e^{-lix}$$

$$c_1 e^{kx} (\cos lx + i \sin lx) + c_2 e^{kx} (\cos -lx + i \sin -lx)$$

$$c_1 e^{kx} (\cos lx + i \sin lx) + c_2 e^{kx} (\cos lx - i \sin lx) = e^{kx} ((c_1 + c_2) \cos lx + (c_1 - c_2)i \sin lx)$$

$$e^{kx} (\vec{c}_1 \cos lx + \vec{c}_2 \sin lx)$$

$$y = e^{kx} (c_1 \cos lx + c_2 \sin lx), c_1, c_2 \in \mathbb{R}$$

Vi för ett svar som är reellt, men vi använder genväg med komplexa tal

#### Exempel Positiv kvot

Lös  $y''(x) - 4y'(x) + 3y(x) = 0$

$$r^2(x) - 4r(x) + 3 = 0 \Rightarrow r_1 = 1, r_2 = 3 \Rightarrow y(x) = c_1 e^x + c_2 e^{3x}, c_1, c_2 \in \mathbb{R}$$

#### Exempel noll kvot

Lös  $y''(t) - 4y'(t) + 4$

$$r^2 - 4r + 4 = 0 \Rightarrow (r - 2)^2 = 0 \Rightarrow r = 2$$
 dubbel rot

$$y(t) = c_1 e^{2t} + c_2 t e^{2t}, c_1, c_2 \in \mathbb{R}$$

#### Exempel koplex kvot

Lös  $y''(t) + 25y(t) = 0$

$$r^2 + 25 = 0 \Rightarrow r^2 = -25 \Rightarrow r = \pm 5i$$

$$y(t) = c_1 \cos(5t) + c_2 \sin(5t), c_1, c_2 \in \mathbb{R}$$

## Partikulär lösning

### Methodology

Partikulär lösning är för att gissa lösningen till det ike homogena lösningarna  
 $ay'' + by' + cy = h(t)$ ,  $y(t) = y_h(t) + y_p(t)$

If  $f(x) = P_n(x)$ , try  $y_p = x^m A_n(x)$

If  $f(x) = P_n(x)e^{rx}$ , try  $y_p = x^m A_n(x)e^{rx}$

If  $f(x) = P_n(x)e^{rx} \cos(kx)$ , try  $y_p = x^m e^{rx}(A_n(x) \cos(kx) + B_n(x) \sin(kx))$

If  $f(x) = P_n(x)e^{rx} \sin(kx)$ , try  $y_p = x^m e^{rx}(A_n(x) \cos(kx) + B_n(x) \sin(kx))$

För att ta reda på t eller k så kan tänka att den homogena lösningen är ej en lösning  
 så börja med k=0 sen gå up tills det är sant

### Exempel trigonometrisk

Lös  $y'' + 9y = \sin 3x$

Sats: diff ekv kan skrivas på följande formel  $y = y_h + y_p$

Homogena lösningen

$$r^2 + 9 = 0 \Rightarrow r = \pm 3i \Rightarrow y_h = A \sin 3x + B \cos 3x$$

Particulöra lösningen

$$\text{Sats } y_p = x^m(A_1 \sin 3x + A_2 \cos 3x), m = 1$$

är godtaglig då det är första ekv som inte kan skrivas på homogen formel

$$y_p = A_1 x \sin 3x + A_2 x \cos 3x$$

$$y'_p = A_1(\sin 3x + 3x \cos 3x) + A_2(\cos 3x - 3x \sin 3x)$$

$$y''_p = A_1(3 \cos 3x + 3 \cos 3x - 9x \sin 3x) + A_2(-3 \sin 3x - 3 \sin 3x - 9x \cos 3x)$$

$$\Rightarrow y''_p = A_1(6 \cos 3x - 9x \sin 3x) + A_2(-6 \sin 3x - 9x \cos 3x) + 9(A_1 x \sin 3x + A_2 x \cos 3x) = \sin 3x$$

$$6A_1 \cos 3x - 6A_2 \sin 3x = \sin 3x \Rightarrow A_1 = 0, A_2 = -1/6$$

$$\Rightarrow y_p = -x/6 \cos 3x$$

Den almäna lösningen blir på följande

$$y(x) = A \sin 3x + B \cos 3x - x/6 \cos 3x$$

**Exempel polynom och Exponensiel**

Lös  $y'' + y' - 2y = 4e^2x + x^2$

Sats: diff ekv kan skrivas på följande formel  $y = y_h + y_p$

Homogena lösningen

$$r^2 + r - 2 = 0 \Rightarrow r_1 = 1, r_2 = -2$$

$$y_h = c_1 e^x + c_2 e^{-2x}, c_1, c_2 \in \mathbb{R}$$

Particulöra lösningen

$$\text{Sats } y_p = y_{p1} + y_{p2}$$

$$y'' + y' - 2y = 4e^{2x}$$

$$y_{p1} = x^m(Ae^{2x}), m = 0 \Rightarrow y'_{p1} = 2Ae^{2x} \Rightarrow y''_{p1} = 4Ae^{2x}$$

$$\Rightarrow 4Ae^{2x} + 2Ae^{2x} - 2Ae^{2x} = 4e^{2x} \Rightarrow A = 1 \Rightarrow y_{p1} = e^{2x}$$

$$y_{p2} = Ax^2 + Bx + C \Rightarrow y'_{p2} = 2Ax + B \Rightarrow y''_{p2} = 2A$$

$$\Rightarrow (2A) + (2Ax + B) - (2(Ax^2 + Bx + C)) = x^2$$

$$\Rightarrow -2A = 1, 2A - 2B = 0, 2A + B - 2C = 0 \Rightarrow A = -1/2, B = -1/2, C = -3/4$$

$$\Rightarrow y_{p2} = -1/2x^2 - 1/2x - 3/4$$

$$\Rightarrow y(x) = c_1 e^x + c_2 e^{-2x} + e^{2x} - 1/2x^2 - 1/2x - 3/4$$

**Resonans****Exempel Resonans**

$$\text{Lösy}'' + y = \sin 2t$$

$$r^2 + 1 = 0 \Rightarrow r = \pm i$$

$$y_h = c_1 \cos t + c_2 \sin t$$

$$y_p = A \sin t + B \cos t$$

$$y'_p = 2A \cos 2t - 2B \sin 2t$$

$$y''_p = -4A \sin 2t - 4B \cos 2t$$

$$y''_p + y_p = \sin 2t \Rightarrow -4A \sin 2t - 4B \cos 2t + A \sin t + B \cos t = \sin 2t$$

$$\Rightarrow A = -1/3, B = 0 \Rightarrow y_p = -1/3 \sin 2t \Rightarrow y(t) = c_1 \cos t + c_2 \cos t - 1/2 \sin 2t$$

**Serielösningar (ej på tenta)**

Antag att serien  $\sum_{n=0}^{\infty} c_n(x - x_0)^n$  konvergerar för alla  $n = 0$

$x \in x_0 - R, x_0 + R$  Då är funktionen  $f(x) = \sum_{n=0}^{\infty} c_n(x - x_0)^n$  deriverbar i

$(x_0 - R, x_0 + R) \wedge f'(x) \sum_{n=0}^{\infty} n c_n (x - x_0)^{n-1}$  Den sista serien konvergerar också i intervallet  $(x_0 - R, x_0 + R)$



## Kapitel 5

# Computer Architecture Cheat Sheet

## 5.1 ISA1

### 5.1.1 MIPS instructions

The instructions that a assembly language use for the MIPS proses.

#### Terminology

- Register file: 32 registers from R0-R31, each is 32 bits.
- Memory: Large each adders stores a word (4 x Registers). Address is always increments of 4. Memory is segmented into 4 8-bits of data to create a word.
- PC (Project counter): Increase with 4 each time to go to the next memory address.
- Instruction Registers: Retrieve the current instructions.
- Control: Inserts data to the register file and add the operation to ALU (Compute).
- ALU (Compute): Calculates the value.
- Memory Address: Call the value from a specific address. From ex “lw”.
- Data Register: Retries and directs value from memory to register file.

*note:* memory operations like (sw R1, 0(R2)) stores the word in poison R2+0 where 0 is the increment since if it is a loop it is needed. The value at that position is R1.

### The operations most used

Function	Instruction	Effect
<b>add</b>	add R1, R2, R3	R1 = R2 + R3
<b>sub</b>	sub R1, R2, R3	R1 = R2 - R3
<b>add immediate</b>	addi R1, R2, 145	R1 = R2 + 145
<b>multiply</b>	mult R2, R3	hi, lo = R2 * R3
<b>divide</b>	div R2, R3	low = R2/R3, hi = remainder
<b>and</b>	and R1, R2, R3	R1 = R2 & R3
<b>or</b>	or R1, R2, R3	R1 = R2   R3
<b>and immediate</b>	andi R1, R2, 145	R1 = R2 & 14
<b>or immediate</b>	ori R1, R2, 145	R1 = R2   145
<b>shift left logical</b>	sll R1, R2, 7	R1 = R2 << 7
<b>shift right logical</b>	srl R1, R2, 7	R1 = R2 >> 7
<b>load word</b>	lw R1, 145(R2)	R1 = memory[R2 + 145]
<b>store word</b>	sw R1, 145(R2)	memory[R2 + 145] = R1
<b>load upper immediate</b>	lui R1, 145	R1 = 145 << 16
<b>branch on equal</b>	beq R1, R2, 145	if (R1 == R2) go to PC + 4 + 145*4
<b>branch on not equal</b>	bne R1, R2, 145	if (R1 != R2) go to PC + 4 + 145*4
<b>set on less than</b>	slt R1, R2, R3	if (R2 < R3) R1 = 1, else R1 = 0
<b>set less than immediate</b>	slti R1, R2, 145	if (R2 < 145) R1 = 1, else R1 = 0
<b>jump</b>	j 145	go to 145
<b>jump register</b>	jr R31	go to R31
<b>jump and link</b>	jal 145	R31 = PC + 4; go to 145

Figur 5.1: MIPS operation tabel

### 5.1.2 Sequencing

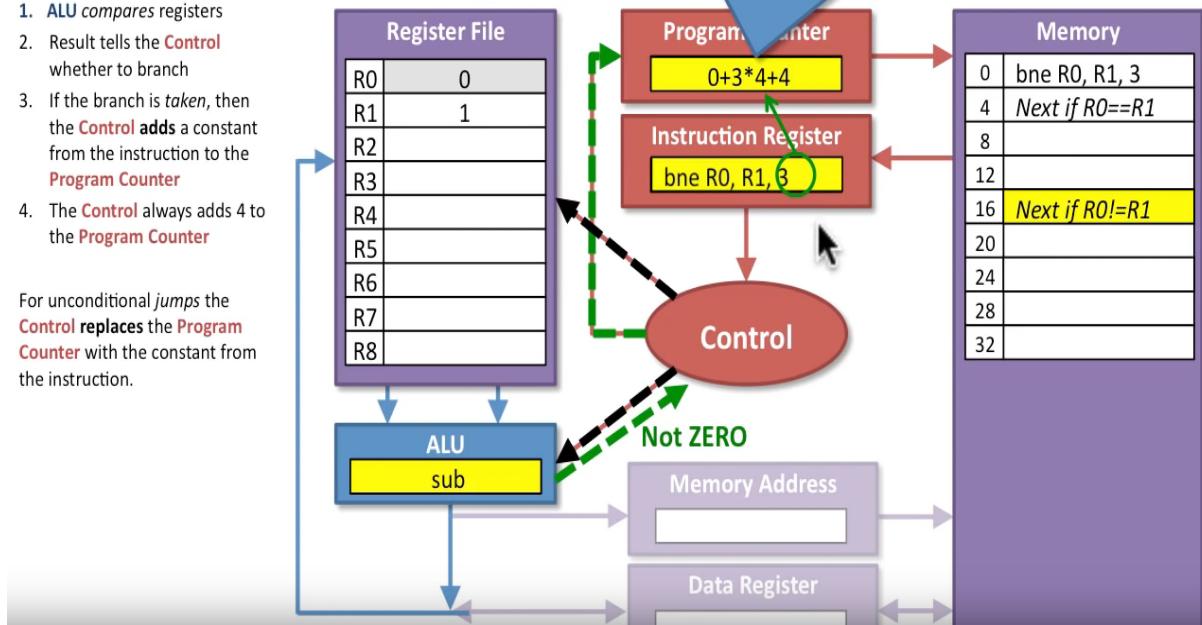
## Sequencing in detail

46

1. ALU compares registers
2. Result tells the Control whether to branch
3. If the branch is taken, then the Control adds a constant from the instruction to the Program Counter
4. The Control always adds 4 to the Program Counter

For unconditional jumps the Control replaces the Program Counter with the constant from the instruction.

The label constant is in instruction words, so it needs to be multiplied by 4 to convert to byte address.



Figur 5.2: Sequencing

### 5.1.3 Converge to Assambly code

If Else example

```
R1 = i; R2 = j; R3 = h

if (i==j)      bne R1, R2, DoElse
    h = i+j;   add R3, R1, R2
                j SkipElse
else           DoElse:
    h = i-j;   sub R3, R1, R2
                SkipElse:
```

Figur 5.3: Assambly exempl

## 5.2 ISA2

### 5.2.1 MIPS type format

- R-format, Aretmetic and lodgical
  - op: ex add
  - rs: first register
  - rt: second register
  - rd: destination register
  - shmt: shift amount
  - funct: Function selector (add = 32, sub =34)
- I-format, Load/store, branch and intemidiats
  - op: ex addi, beq
  - rs: first register
  - rt: second register
  - rd: inttediate value
- J-format, Jump op: ex j skipelse
  - rd: address

A instruction allwas ends with 00 it means that when a instruction is done we update the program couner to by 4 or more it changes the posistion of the 16bit intermidjet for i-format and can. (bne, beq)

### 5.2.2 Procedure

Procedure is how a function call is handeld. It is devided into two parts Caller and Callee, the caller call the procedure (callee). The orperation for controlling procedure is called jump-and-link (jal). It stores the return address (PC+4) in \$ra (R31). Where (ra = register address (updates automatically)) (PC = project counter) (R31 = value). jr \$ra returns the value.

Name	Bit Fields						Notes (32 bits total)
	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	
R-Format	op	rs	rt	rd	shmt	funct	Arithmetic, logic
I-format	op	rs	rt	address/immediate (16)			Load/store, branch, immediate
J-format	op	target address (26)				Jump	

Figur 5.4: MIPS Types

### Stacking

To avoid conflicts with writing over the callers register files, one uses staking in predefined ranges to allow store data independent from the callers. (R29 = store stack data \$sp) When a register file is added the stack pointer is moved and when it is done it returns with (jr) and then reset the previously used register files. when it has used what it needs it removes them when leaded one value at a time

Notice that the register files for the callee save is \$sx and for the caller \$tx.

```

integer_array_sum:
    addi $sp, $sp, -4      # increase stack-pointer by one word
    sw $s0, 0($sp)        # save array index i to stack
    addi $sp, $sp, -4      # increase stack-pointer by one word
    sw $s1, 0($sp)        # save element n to stack

    addi $v0, $zero, 0      # Initialize Sum to zero.
    add $s0, $zero, $zero   # Initialize i to zero.

ia_loop:
    beq $s0, $a1, ia_done  # Done if i == N
    lw $s1, 0($a0)         # n = A[i]
    add $v0, $v0, $s1       # Sum = Sum + n
    addi $a0, $a0, 4        # address = ARRAY + 4*i
    addi $s0, $s0, 1        # i++
    j ia_loop              # next element

ia_done:
    lw $s1, 0($sp)        # loads i from stack-pointer
    lw $s0, 4($sp)         # loads n from stack-pointer
    addi $sp, $sp, 8        # restore stack-pointer

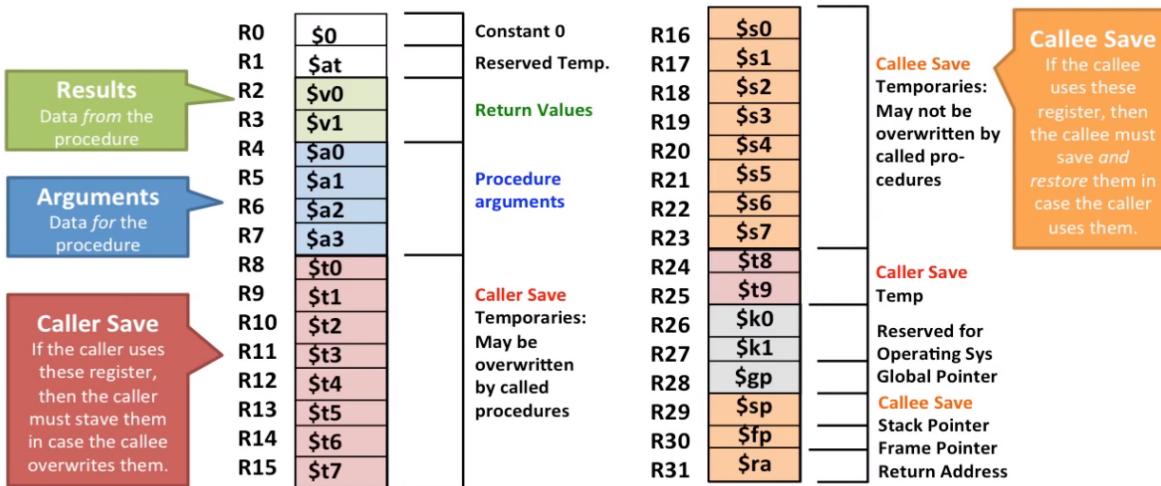
    jr    $ra                # return to caller

```

Figur 5.5: code ex

### 5.2.3 Other ISA

## MIPS register names and conventions



Figur 5.6: MIPS register

## 5.3 Arithmetic

### 5.3.1 Binary numbers

- Binary numbers reduce noise and disregard the exact voltage to result in on/off mode.
- In computer science octal and hexadecimals are also often used because of its properties. every octal digit represents 3 in decimal every hex digit represents 4 in decimal
- msb=most significant bit
- lsb=least significant bit
- Optimization:  $0010 * 0101$  2 operations  $\rightarrow 0101 * 0010$  1 operation
- Multiplication: (fixed point need to shift decimal point)  $0010 * 0101 = 0010 + (0010 \text{ shifting}[00]) = 1010$   
 $0011.010 * 0001.110 =$
- Addition: (Same for fixed points and integers)  $1110 + 1000 = \text{carry}[1]0110$  Overflow  $0101 + 0001 = 0110$   
 Not overflow

### 5.3.2 Negative integers

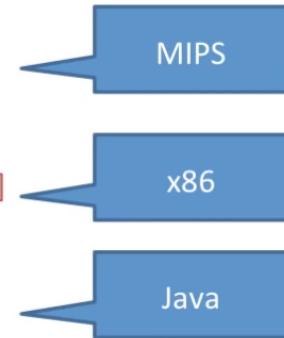
- *Signed magnitude*: msb is the sign  $1011_2 = -(2+1)_{10} = -3_{10}$
- *tw's complement*: if msb is 1 then negative number every other digit after is positive  $1011_2 = -8 + 2 + 1_{10} = -5_{10}$

### 5.3.3 Operations

#### 5.3.4 Non integer numbers (Floating and fixed point)

- Converting binary to decimal:  $0.111 = 1/2 + 1/4 + 1/8 = 0.875$

- **Accumulator** (1 register)
  - 1 address      **add A**       $acc \leftarrow acc + mem[A]$
- **General purpose register file** (load/store)
  - 3 addresses      **add Ra Rb Rc**       $Ra \leftarrow Rb + Rc$
  - load Ra Rb**       $Ra \leftarrow Mem[Rb]$
- **General purpose register file** (Register-Memory)
  - 2 address      **add Ra addressB**       $Ra \leftarrow Mem[addressB]$
- **Stack** (not a register file but an operand stack)
  - 0 address      **add**       $tos \leftarrow tos + next$   
 $tos = top\ of\ stack$



Figur 5.7: Other ISA

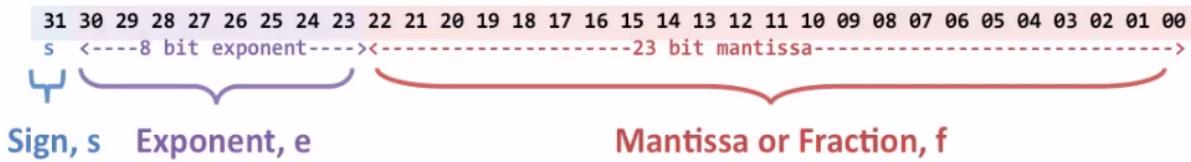
- Fixed point: for defined ranges 1.001 and 0.100 one can chose large and small representations
- Floating point: IEEE standard, se following image
- Standard floating point is in binary so FFFF is at most 1111 and with two's complement 0111

### 5.3.5 Overflow

- input: msb = 1 for both numbers and output: msb = 0 (overflow)
- input: msb = 0 for both numbers and output: msb = 1 (overflow)
- input: msb = 0 for both numbers and output: msb = 0 (Not overflow)
- input: msb = 1 for both numbers and output: msb = 1 (Not overflow)
- input: msb = 1 and msb = 0 (May overflow)

Stack	Accumulator	Register (Register-memory)	Register (Load-store)
Push A	Load A	Load R1, A	Load R1, A
Push B	Add B	Add R1, B	Load R2, B
Add	Store C	Store C, R1	Add R3, R2, R1
Pop C			Store C, R3
JVM	PDP-8, 8008, 8051	x86	MIPS, PPC, ARM, SPARC

Figur 5.8: Other ISA instructions



$$(-1)^s \times (1.f) \times 2^{(e - 127)}$$

Figur 5.9: Floating Point

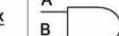
## 5.4 Logic

- truth table, is used to represent all possible inputs and then the corresponding output.
- To determine the possible schematics one can write for the input A and B “!A and B” not A and B. One often gets unnecessary complexity, one can often simplify the schematics. It is only done with the inputs that have an output of one.
- De Morgan’s Law  $!(A + B) = !A \cdot !B$  and  $\cdot$  works the same way but are different.
- karnaugh map is used to easily determine the schematics. It is a matrix. It is possible to use more inputs than two. In the matrix one looks at when output is one.
- a cabal with 4 wires can take 4 bits. Overflow needs one extra wire on output

### Building blocks

- Building blocks are the first layer of abstraction since one can not see the logical gates constructed of. For example add is a building block.

## Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																																																																
Alg. Expr.	$\bar{A}$	$AB$	$\bar{AB}$	$A+B$	$\bar{A}+\bar{B}$	$A \oplus B$	$\bar{A} \oplus \bar{B}$																																																																																																
Symbol																																																																																																							
Truth Table	<table border="1"><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	0	0	1	0	1	0	0	1	1	1	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	1	1	0	1	1	1	0	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	0	0	1	0	1	0	0	1	1	0	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	1	0	1	1	1	0	1	1	1	0	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	B	A	X	0	0	0	0	1	1	1	0	1	1	1	0	<table border="1"><tr><th>B</th><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	B	A	X	0	0	1	0	1	0	1	0	0	1	1	1
A	X																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
B	A	X																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					

Figur 5.10: Floating Point

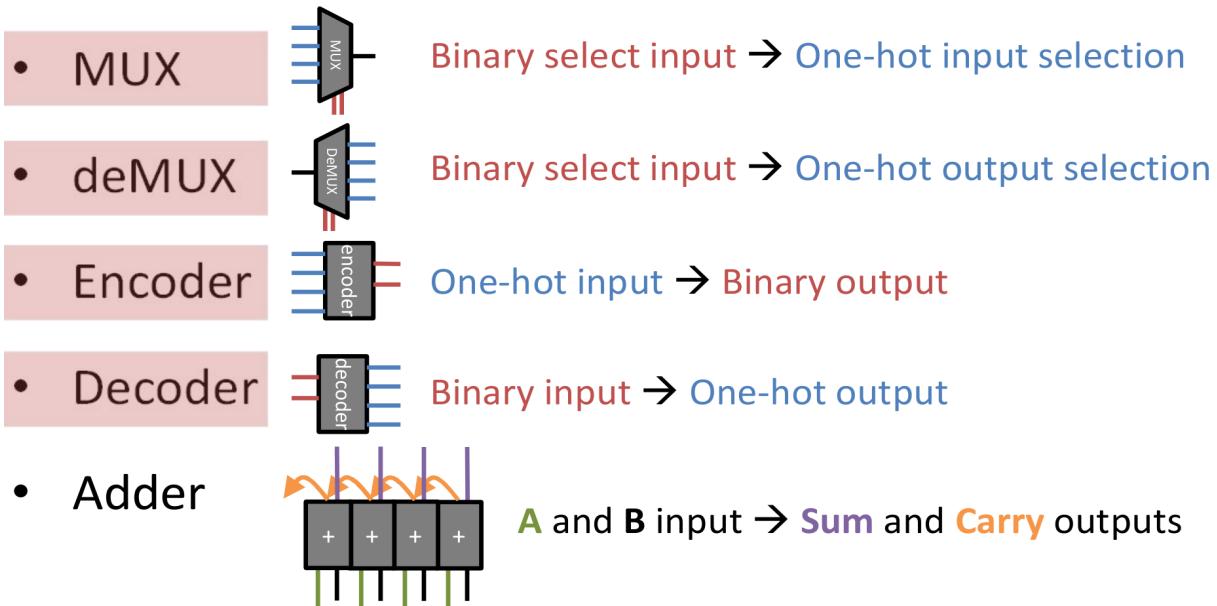
- Two types of logic, combinational and sequential. combinational: output just depends on input sequential: output depends on the state. State is stored in memory update on clock.
- MUXes (Multiplexers): choose input from a bus/ routing signal. 2-bit decision for input 4-bit selection. Starts from position 0.
- DEMUXes (Demultiplexers): opposite, take one signal and decides where it wants to be sent to Bus is a multi-bit signal “wire”
- Decoder: binary to hot, (adds a zero or one)? in arrays it can show position
- Encoder: hot to binary, (remove a zero or one)?
- Adders: adds carries on extra bit to handle overflow

### Latches

- One use is for counters with a clock state triggers the count. So when input is one the output is 0 and when the clock clicks it is one then when the input is 0 the output is still 1.
- flip flop: is a edge triggered latch has a clock trigger to open or close latch and then can save it to SRAM cell

### Memory

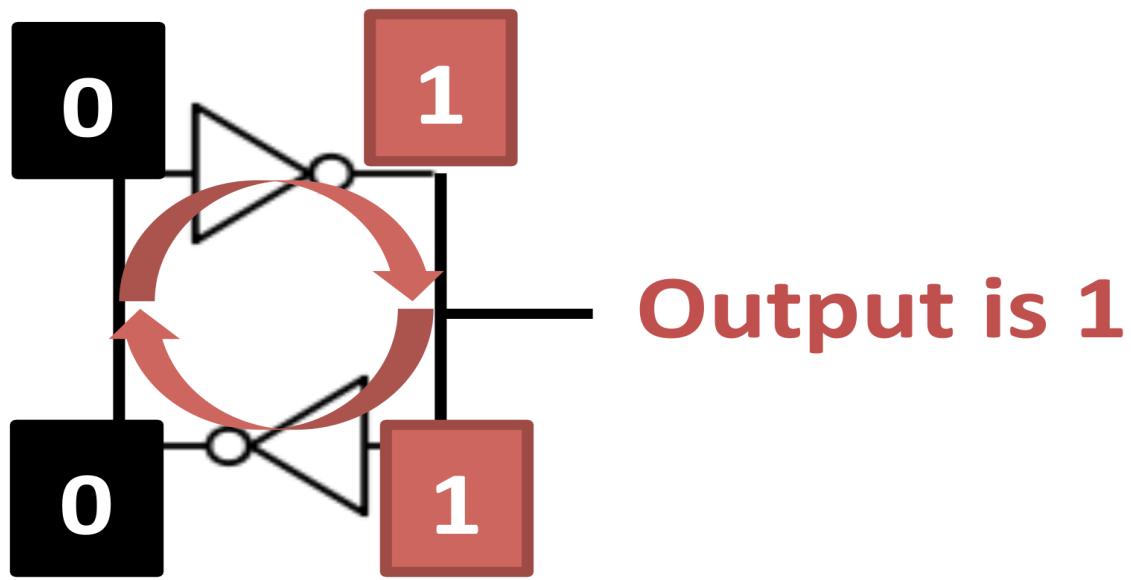
- SRAM (Static Random Access Memory) Big, fast and expensive. Loops through value in order to store it. To write to memory one uses a switch to update value.
- DRAM (Dynamic Random Access Memory) Smaller, Slower and cheaper. Uses a transistor to store value and a capacitor to store the charge so the data does not disappear. That is why it is slower because the capacitor needs to be recharged



Figur 5.11: Floating Point

## 5.5 processor control and datapath

There is 3 main part of the mips datapath as seen in the following image:



Figur 5.12: latche

An overview of the mips datapath with j-format instruction:  
Think about the controller like a decoder that decodes the op code from the different formats.

- **ALU operations (add, or, etc.)**
  - Load the instruction
  - Calculate the next PC
  - Read the register file
  - Do the ALU operation
  - Write data back to the register file
- **Memory access (load/store)**
  - Load the instruction
  - Calculate the next PC
  - Read the register file
  - Calculate the address
  - Read/Write the data memory
  - Write data back to the register file
- **Instruction fetch (branch)**
  - Load the instruction
  - Calculate the next PC
  - Read the register file
  - Calculate the branch address
  - Do a branch comparison
  - Update the PC

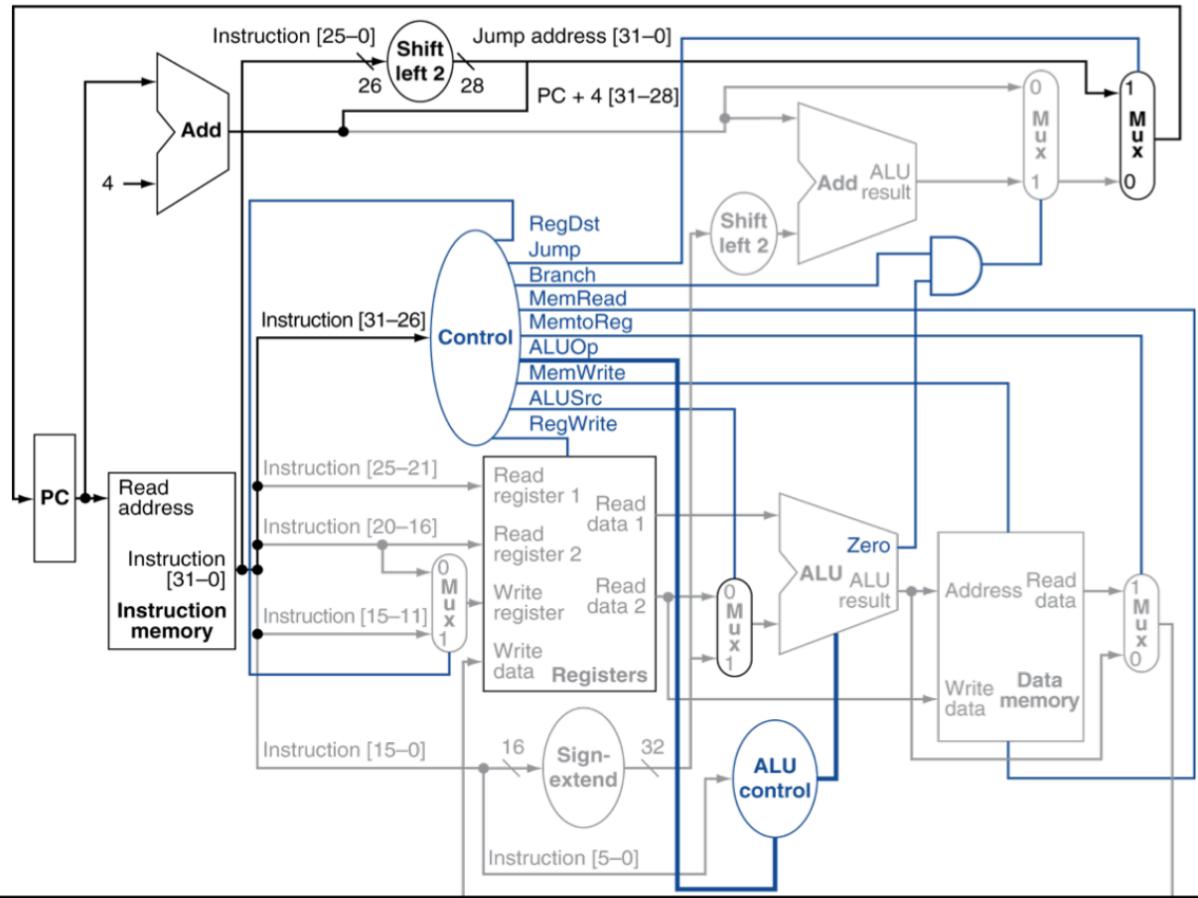
Figur 5.13: mips datapath 3 parts

### 5.5.1 Clock

A clock is used to update the state and continu with the other instructions. Every state element uses a clock. In mips prosesor clock gose to memory, pc, rf and dm. The clock unit is in (MHz) converting from (ns) is simple. Ex 10ns:  $1/10\text{ns} = 0.1\text{MHz}$ .

### 5.5.2 Critical path

The longest path of the datapath is the critical path. Often PC is the fastest so one then calculates the amount of time it takes for the instruciton has traveld the data path and refernd to know what the critical path is. The longest part is data memory sinze it is big and slow. One often say read and write takes constant time regardless of the amout of diffrent read and write.



Figur 5.14: mips datapath

## 5.6 pipeline

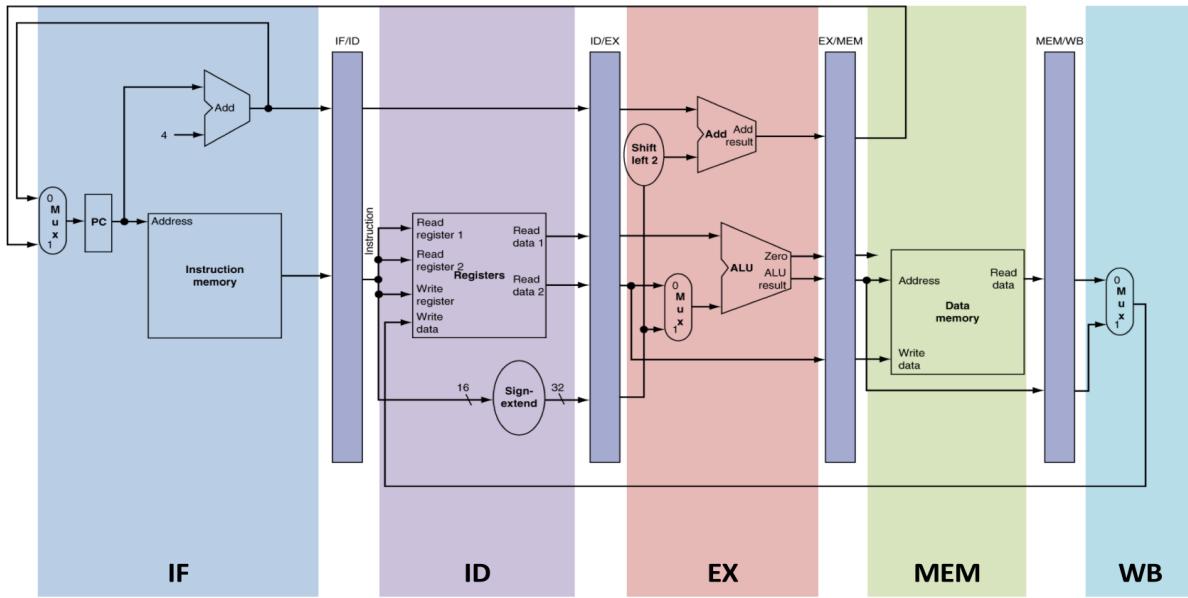
The purpose is to split a instruction cycle into multiple stages to run instructions in parallel to improve performance. The stage has its own pipeline registerfile for storing the necessary registers. Pipeline registers have a performance reduction since it takes time, therefore more pipeline stages do not equal greater performance over a certain amount. One issue of performance is balancing the stages inorder to get a good clock frequency. Not all stages are needed for each operation or instruction

### Mips stages

- IF= Instruction fetch
- ID= Decode and RF Read
- Ex= ALU Execute
- MEM= Memory
- WB= RF Write back

### Terminology

- Bubbles= Is detected by the hardware where there is no instructions



Figur 5.15: Pipeline

- Nop= Is a sudo instruction for a stall type instruction (do not do anything)
- Delay slots= a stall type for branches. Try to fill in those slots with usful instructions.
- Interference= Can not read and write at the same time.
- Double pumping= Spiting write to first clock cycle and the second cycle is read.
- Forwarding= Getting data from a different pipeline stage from a register file.

### Calculating time coplexity

- overhead:  $(\text{instructions time (ex 100ns)}) / (\text{stages (ex 1000)}) + (\text{register overhead (1ns)}) = 1.1\text{ns}$   
Total time:  $1.1 * 1000$
- Latancy:  $(\text{stages} * (\text{penalty per stage}) + \text{original})$   
Latancy:  $(\text{Time per instruction (ex 100ns)}) + (\text{pipeline register's (ex 100-1)} * (\text{penalty per stage (ex 2ns)})) = 300\text{ns}$  (3x slower)
- Trouput:  $(\text{Time per instruction (ex 100ns)}) / (\text{Stages (ex 100-1)}) + (\text{Time pipline registes (ex 2ns)}) = 3\text{ns}$  (every instuction is 33x faster)

## 5.7 Pipeline hazards

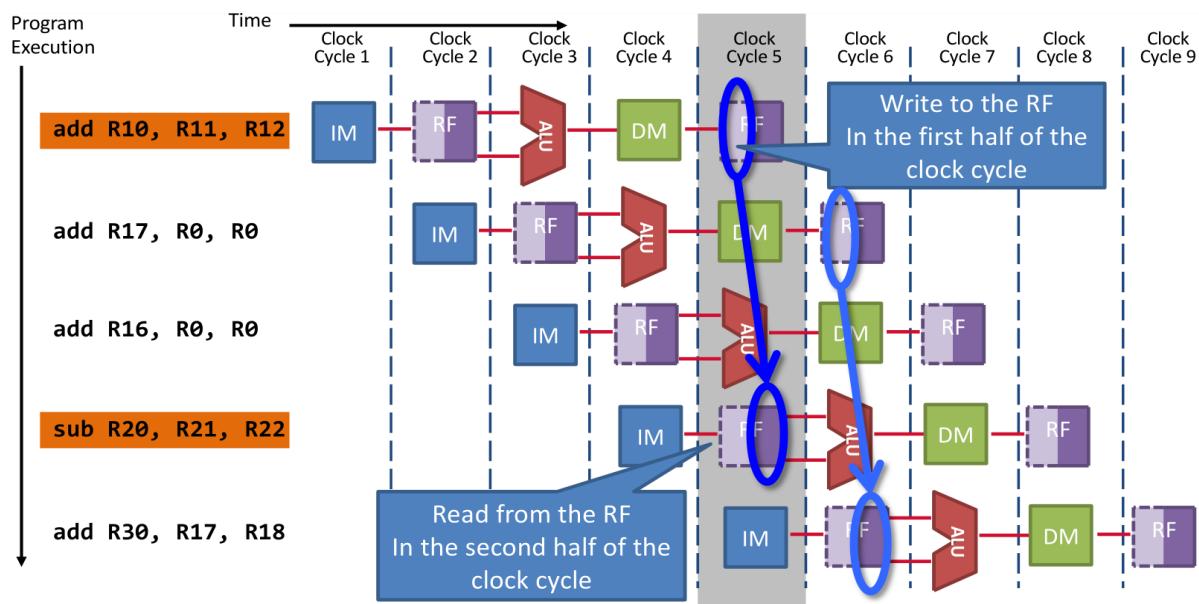
### 5.7.1 Data hazards

#### The issue

- Data is not available where we need it (later in the pipeline; not written back yet).
- Data is not available when we need it (need to read memory first).

#### Fixing the issue

- Forward the data to where we need it.
- Stall if data is not ready yet (NOPs or Bubbles).



Figur 5.16: Double pump

### 5.7.2 Control hazards

#### The issue

- Don't know which instruction is next when we need to fetch it.

#### Fixing the issue

- Calculate branch as early as possible.
- Stall with a branch delay slot.

### 5.7.3 Structural hazards

#### The issue

- Can't do the instruction because the hardware is busy.

#### Fixing the issue

- Build more hardware (double-pumped register file). Double-pumped write at the first half cycle and read at the other half

#### Calculating time complexity

- Branch delay  
$$\text{(How many branches (ex 20\%)) / (How many useful instructions can be filled in (ex 50\%))} = 20\% / 0.5 \\ = 10\%$$
- Cycles per instructions (CPI):  $\text{(total cycles (with nop)) / (useful work (without nop))}$

## 5.8 Predicting Branches and Exceptions

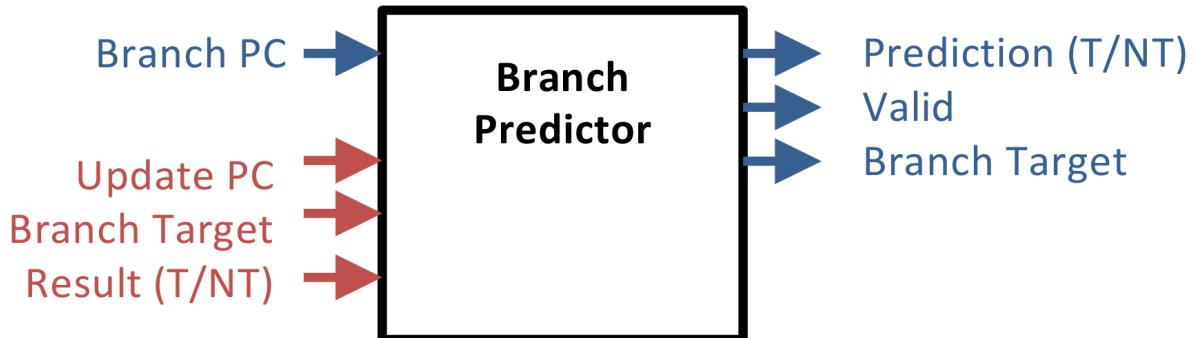
When the prediction is wrong, clean up is needed. – “Kill” or “Squash” so they don’t execute (they were wrong) – Prevent them from writing: disable RegWrite, MemWrite in the pipeline – Turn them into NOPs: change opcode to add R0, R0, R0 in the pipeline

### 5.8.1 Static predictors

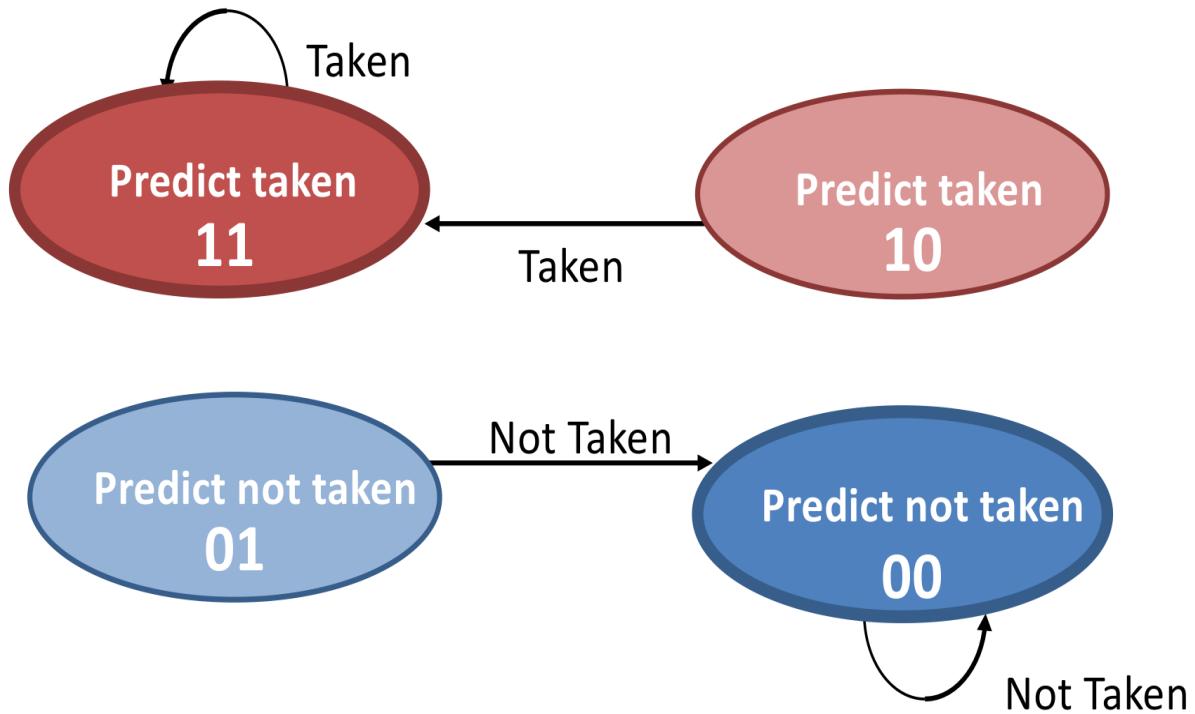
- Predict always not taken
- Predict always taken
- Backwards-Taken, Forward-Not-Taken (BTFTNT)

### 5.8.2 Dynamic predictors

The implementation of branch predictors look somthing like this:



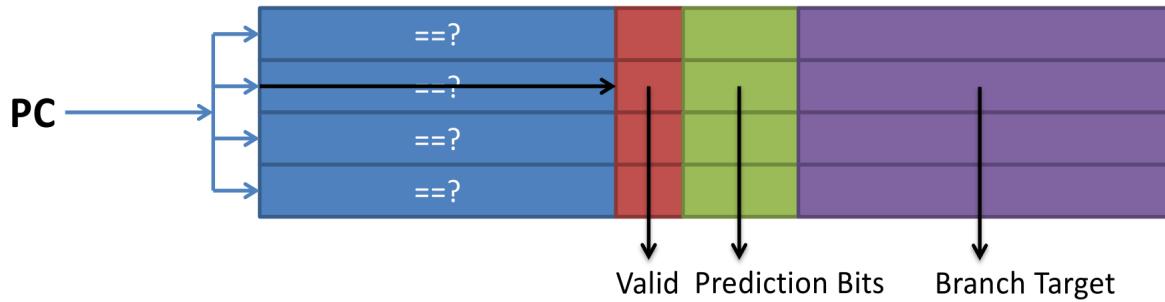
Figur 5.17: branch-predictor



Figur 5.18: 2-bit predict

**BTB**

- Branch Target Buffer (BTB)
- Save a table with PC onorder to have history that we can predict on



Figur 5.19: btb

### 5.8.3 Exceptions

Exceptions are non-normal events that interrupt the normal flow of instructions

- Divide by zero
- Misaligned memory access
- Page fault
- Memory protection violation

Interrupts are external events that interrupt the normal flow of

- **Synchronous vs. asynchronous**
  - Synchronous: occur at the same place every time a program executes
  - Asynchronous: caused by external devices and happen at different times
- **User requested vs. coerced**
  - Coerced are hardware events the user can't control
  - User requested are from the user
- **User maskable vs. nonmaskable**
  - Can the user disable the exception/interrupt?
- **Within vs. between instructions**
  - Does the event prevent the current instruction from completing, or interrupt after it?
- **Resume vs. terminate**
  - Can the event be handled (corrected) by the OS or program, or must the program be terminated?

Figur 5.20: exceptions

## 5.9 Input/Output

### Terminology

- nvrrom (flash): Similar to ram but it saves data when there is no power
- Busses: Parallel: many bits at once (e.g., 32 bits together in one clock) Used to be used everywhere Still used inside chips
- Serial links: Serial: one bit at a time (e.g., 32 clock cycles to send 32 bits) Used to be used only where distances were long (e.g., networks) Now used for most off-chip communications
- memory-mapped I/O: Manual handling of I/O devices Map portions of the address space to I/O devices Read and write to those addresses to access the
- direct memory access (DMA): Hardware who mange I/O devices (dynamicly)
- Polling: The device puts its status somewhere The OS repeatedly checks for it to change
- Interrupt: When the device is ready, it gets the processor's attention by signaling an interrupt The OS then jumps to an interrupt handler to handle the event
- Throughput: x/s The read and write speed
- Latency: s/x Accessing time.
- Overhead: any combination of excess or indirect computation time, memory, bandwidth, or other resources that are required to perform a specific task

**Data transfers: manual copy**

```
add R2, R0, R0      // Counter starts at 0
loop:
    lw R4, 0x12f0(R0) // Read the I/O device
    sw R4, 0xfe00(R2) // Store the results
    addi R2, R2, 4     // Next location
    bne R3, R2, loop
done:
```

**Data transfers: DMA**

```
addi R2, R0, 0xfe00 // destination
addi R3, R0, 230400 // number of words to copy
sw R1, 0x100b(R0) // set the device address
sw R2, 0x1008(R0) // set the destination address
sw R3, 0x1004(R0) // set the length and start
wait:
    lw R2, R0(0x1000) // Wait for it to be done
    bne R2, R0, wait
done:
```

## 5.10 Cache

### 5.10.1 Memory hierarchy

• Registers	3 accesses/cycle	32-64
• Cache	1-10 cycles	8kB-256kB
• Cache	40 cycles	4-20MB
• DRAM	200 cycles	4-16GB
• Flash	1000+ cycles	64-512GB
• Hard Disk	1M+ cycles	2-4TB

Figur 5.21: memory-hierarchy

#### 3-types of cache types

- Fully-associative: Have to search all blocks, but very flexible
- Direct-mapped: Only one place for each block, no flexibility
- Set-associative: Only have to search one set for each block, flexible because each set can store multiple blocks in its ways

#### Write policies

- Write-through: slow, simple
- Write-back: fast (keeps the data just in the cache), more complex

#### Terminology

- Data: What is being stored
- Tag: What address the data has
- Index: What we use to determine where we want to put the data
- Cache line (block) size: How many tag's there are
- Valid bit: If the data is correct
- Dirty bit: The data has been changed and we need to write it to memory
- Type of cache: Fully-associative (FA), Set-associative (SA), Direct-mapped (DM)
- Replacement policy: Write-through, Write-back

### Cache blocks

- tag has 30bits and the 2 other bits is to determine what data 63 bits for each entry 32bit data 30bit tag 1bit valid-bit
- larger block of data -> fewer tags -> more efficient storage
- 1 word cache block we have 1 32bit data
- 2 word cache block we have 2 32bit data
- 4 word cache block we have 4 32bit data
- we load more data when we have space for it we will then have spatial locality
- Last 2: determine the byte within the word
- Next N: determine which word in the cache block
- Remaining 32-N-2: tag

### Calculate overhead

- Data = Cache-Lines \* Byte-Lines
- Overhead = Cache-Lines \* (Tags-per-line + valid-bit)
- % data = Overhead / Data

### Calculate Address (Tag Index Offset)

- Direct mapped:  
Offset = log<sub>2</sub> (number of data blocks) (+ 2-bit (4-bit cache block size if 64 then log 64))  
Index = log<sub>2</sub> (number of cache lines)  
Tags = 32 (bit processor) - (Index + Offset)
- x-set associative:  
Offset = log<sub>2</sub> (number of data blocks) (+ 2-bit determines how the address looks like)  
Index = log<sub>x</sub> ((number of cache lines) / x)  
Tags = 32 (bit processor) - (Index + Offset)
- fully associative:  
Offset = log<sub>2</sub> (number of data blocks) (+ 2-bit determines how the address looks like)  
Index = 0  
Tags = 32 (bit processor) - (Offset)

### Calculate Average memory access time

- Average-cycles = CacheL1\*Cycles + CacheL2\*Cycles + Dram\*Cycles

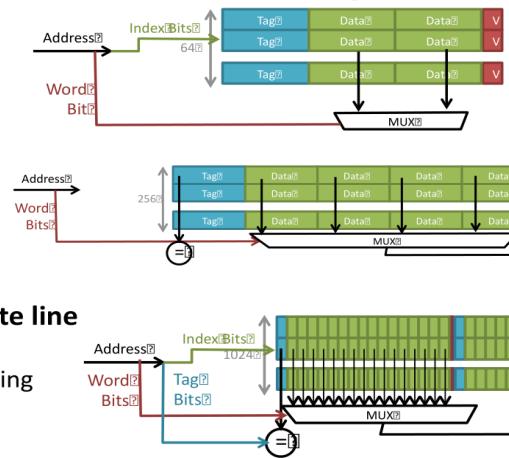
### Minimum associativity

- (page size) / ((Cache size) / (bytes per line))  
ex: 8kB page size 64KB cache and 32bytes per line.  $\log_2(2^{13}/(2^{10}/32)) = 8$  therefore 8-way

## 5.10.2 Cache misses 3C's

- cold/compulsory miss: When the program first begins it doesn't have anything in the cache
- conflict miss: too small to store the necessary data
- capacity miss: maps at the same block and then overwrites unnecessarily

- Determine which bits in a 32-bit address are used for selecting the **byte (B)**, selecting the **word (W)**, indexing the **cache (I)**, and the cache **tag (T)**, for each of the following caches:
- 64-line, direct-mapped, write-through, 8 byte line**
  - 8 byte line = 2 words per line, need 1 word bit
  - 64 lines, need 6 bits for indexing
  - TTTT TTTT TTTT TTTT TTTI IIII IWBB**
- 256-line, fully-associative, write-back, 16 byte line**
  - 16 byte line = 4 words per line, need 2 word bits
  - 256 lines, but fully associative! 0 bits for index (data can go anywhere)
  - TTTT TTTT TTTT TTTT TTTT TTTT TTTT WWBB**
- 4096-line, 4-way set-associative, write-back, 64 byte line**
  - 64 byte line = 16 words per line, need 4 word bits
  - 4096 lines/4-ways = 1024 sets, need 10 bits for indexing
  - TTTT TTTT TTTT TTTT IIII IIII IIWW WWBB**



Figur 5.22: cache-format

## 5.11 Virtual Memory

### Why use VM

- Map memory to disk (“unlimited” memory)
- Keep programs from accessing each other’s memory (security)
- Fill holes in the RAM address space (efficiency)

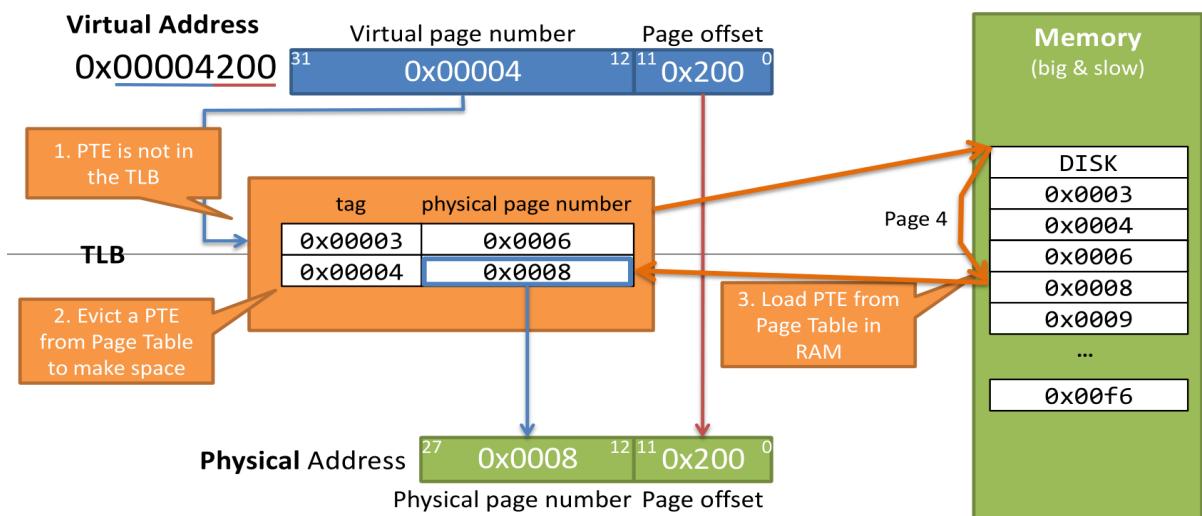
### Termonology

- Translation= map address
- Page tables= for each program keep track of all translations
- Fine grain= page table with spesific address
- Coarse grain= page table with adress mape ranges
- Page Table Entries (PTEs)= number of translation
- Translation Lookaside Buffer TLB= All of the pages, fast translation via hardware
- VA= Virtual program addresses
- PA= Physical RAM addresses
- Page offset= point to a range and then use the offset to determine where
- Translation Lookaside Buffer (TLB)= page table cache (Faster)
- Multilevel page table translation= page table point to other page tables (inception)

### Combining TLB and cache

- Physical caches: slow and needs the translation first and then save get the PA also known as Physical-Index, Physical-Tagged (PIPT)
- Virtual caches: fast uses only virtual addresses, no translation, difficult for protection also known as Virtual-Index, Virtual-Tagged (VIVT):
- Virtual-Index, Physically-Tagged (VIPT): VA for index PA for tags, fast does translation and fetch from cache at the same time, most commonly used. We need a comparator to see if the PA tag from cache matches the TLB PA only then we can say if we had a hit or a miss. The VA offset is what the PA tag is selected and the VA tag (number of virtual pages) is what the TLB uses. Mostly used for L1 cache not L2.

### TLB



Figur 5.23: tlb

**useful conversion:**  $2^n = kB$

0	1kB	20	1MB	30	1GB
1	2B	11	2kB	31	2GB
2	4B	12	4kB	32	4GB
3	8B	13	8kB		
4	16B	14	16kB		
5	32B	15	32kB		
6	64B	16	64kB		
7	128B	17	128kB		
8	256B	18	256kB		
9	512B	19	512kB		

Figure 5.24: conversion

#### Calculating Page sizeing VIPT

- Number-of-Virtual-Pages=  $2^{32}/(pages)$  (in a 32bit address)
- Bits-used-for-Page-Offset= log(pages)
- Bits-used-for-VPN= 32(bit processor)- Bits-used-for-Page-Offset

#### Calculating TLB size

- TLB-size=Entries\*Pages

## 5.12 Parallism

### 5.12.1 Multicore

**powerwall**

$$P = CV^2f$$

$C$  = capasiter, smaller transistors smaller capasiters

$V$  = voltage, decreasing makes it slower

$f$  = frequency, reducing clockspeed

### 5.12.2 Paralel programing

Average processors

Calculate how many cores are used in average we need to know

how chunks (work) there is in total

how many time units (cycles think of a reverse pyramid)

there are 16 input data and we have 8-cores, we want to calculate the total sum

what is the average processor being used

15 chunks, 4 time units  $\Rightarrow 15/4 = 3.75$

**parallel issues**

- Most programs can not utilize parallelism, need to divide the program to different executions.
- We also need to share cache and therefore have performance issue since we can't use the entire cache.

**How much faster**

75% parallel, 25% nonparallel we have hundred thousand cores

$\Rightarrow$  Parallelism takes  $(0.75/100000 + 0.25 * 1)$

Singular takes  $(0.75 * 1 + 0.25 * 1) \Rightarrow 4 * \text{faster}$

$$\text{Speedup with Amdahl's law } \text{Speedup} = \frac{1}{(1 - P) + P/S}$$

$P$  = Parallel action

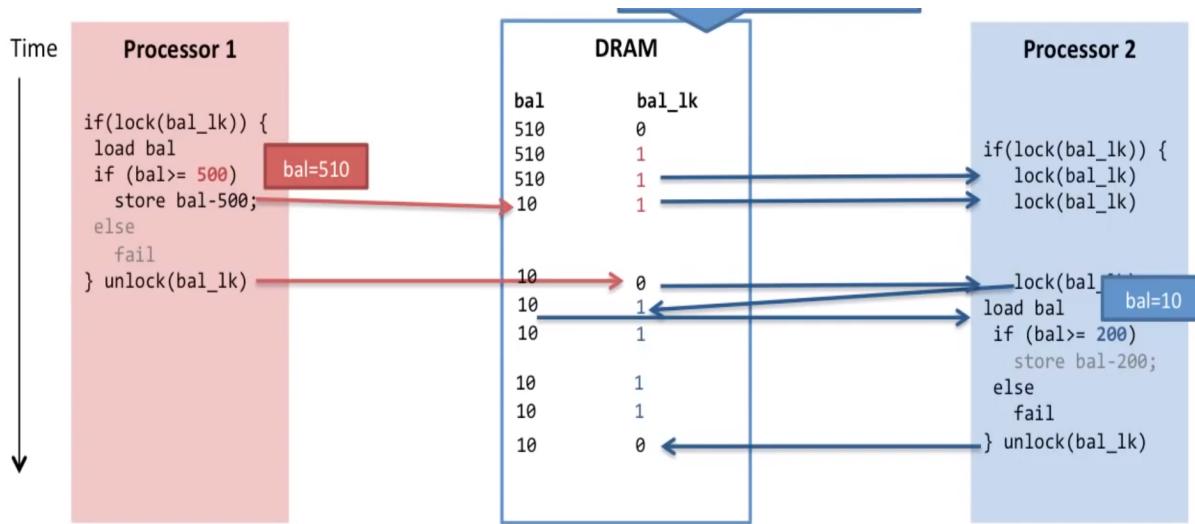
$S$  = Speed up of the parallel part

$$\Rightarrow \frac{1}{(1 - 0.75) + 0.75/100000} = \frac{1}{0.25 + 0} = 4$$

### 5.12.3 Syncronaziation

- Fix the issue with 2 processors accessing the same value at the same time.
- We can use atomic swap to first set lock to 1 then check the data.

locks

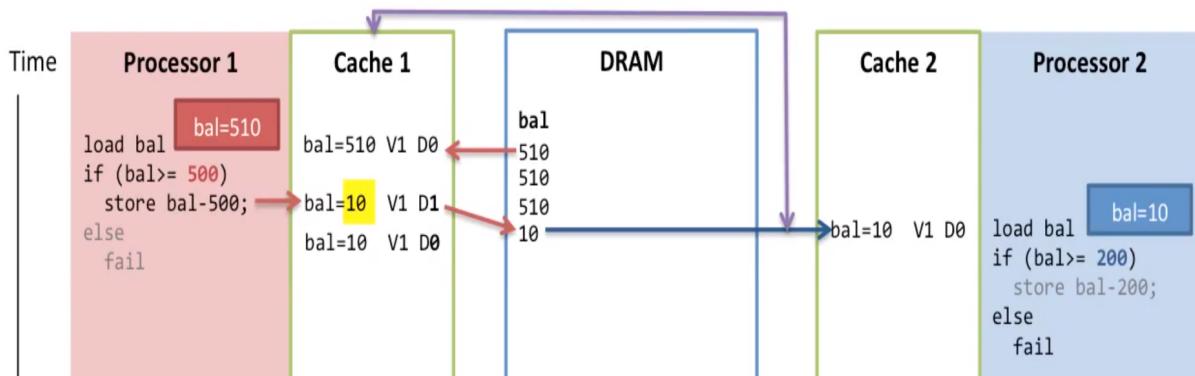


Figur 5.25: locks

#### 5.12.4 Cache coherency

- How we use caches to save memory
- Locks: if the data has been accessed. (not a guarantee of protection)
- Snooping: Look what the other processors are storing in their private cache

snooping



Figur 5.26: locks

### 5.12.5 ILP

- Instruction level parallelism (ILP)
- better to have out-of-order execution since we can find independent instructions
- We use Dual-issue pipeline so we can have one for memory instructions and one for non memory instruction
- It makes ISA promises with inorder execution
- Issue with data hazards, more complexity

**dual issue pipeline**

- **Regular Path**

- **Ld/St Path**

- Added:

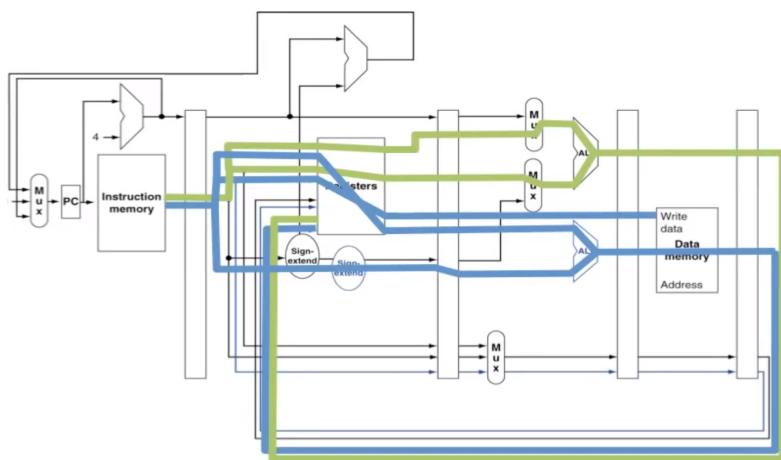
- More RF ports
- 2<sup>nd</sup> instruction fetch
- 2<sup>nd</sup> sign-extension
- 2<sup>nd</sup> ALU
- More forwarding logic and paths

- Now we can issue both a ld/st and any other instruction at the same time!

```

1: add r1, r2, r3
1: ld r4, r5
2: sub r7, r1, r4
2: st r8, r9
3: or r5, r8, r9
3: nop

```



Figur 5.27: dual-issue-pipeline

### Simultaneous Multithreading (SMT)

- Threads: each process has its own thread so it can be more easily run in parallel
- SMT: allows threads when stalled to switch to another thread and run instead