Data och Information

Dr. Johan Hagelbäck

johan.hagelback@lnu.se

http://aiguy.org

Data eller information?

- I den verkliga världen har vi information, till exempel en bok eller ett stycke musik
- Denna information kodas till en representation som kan sparas digitalt
- Det vi har sparat är data – det har ingen mening
- Det finns flera problem med kodningen beroende på vad för typ av information som ska lagras
- Hur kan vi lagra en text eller musik i digital form?

Data eller information?

- Termerna data och information överlappar varandra, och ibland blandas de ihop
- Kortfattat kan man beskriva skillnaden som:
  - Data är kvantiteter, symboler eller tecken som kan lagras digitalt
  - Information är kunskap om ett ämne, en händelse eller fakta om något
- Information fås genom bearbetning av data

Data eller information?

- Information är fakta eller kunskap om den verkliga världen
- Data är kodning av information så att den kan manipuleras av ett datorsystem
- Information har en mening

Data eller information?

- Ett kreditkortsnummer är information – vi använder numret för att betala saker vi handlar
- Själva siffrorna som utgöra numret kan lagras i t.ex. ett RFID chip eller en streckkod
- Sekvensen av siffror har i sig självt ingen mening
- Det är när vi hämtar sekvensen av siffror och använder dem som vi får information – sekvensen har ett användningsområde, en mening

Konvertera information till data

- Det finns flera utmaningar med att konvertera information till data som kan lagras
- Hur kan vi lagra en bild av vår katt eller en låt med vår favoritartist?
- Det finns två typer av data: *kontinuerlig* och *diskret*
- Data är kontinuerlig om det finns oändligt antal möjliga värden
- Data är diskret om det finns ett ändligt antal möjliga värden

Kontinuerlig eller diskret?

- Kontinuerlig data kommer från t.ex. mätningar av något i den verkliga världen, musik, film, …
- Diskret data associeras till saker som kan räknas, t.ex. antal studenter som får högsta betyg på min kurs

- Exempel: är vikten på en apelsin kontinuerlig eller diskret?

Kontinuerlig eller diskret?

- Om vi har tur kan vår apelsin väga exakt 200 gram
- Den kan också väga 229,3 gram,
- … eller 229,31533 gram,
- … eller 229,31533480185993 gram
- Apelsinens vikt är kontinuerlig eftersom det finns oändligt antal värden som en apelsin kan väga

Kontinuerlig eller diskret?

- Om vi i stället ska ange betyg på vår favoritfilm finns det ett ändligt antal möjliga värden:
  - 1, 2, 3, 4 eller 5
  - 1 till10
- Betyget är ett exempel på diskret data

Kontinuerlig eller diskret?

- Ett enskilt betyg är diskret data
- Om vi i stället ska beräkna medelvärdet på flera betyg för en film får vi dock ett kontinuerligt värde – oändligt antal decimaler

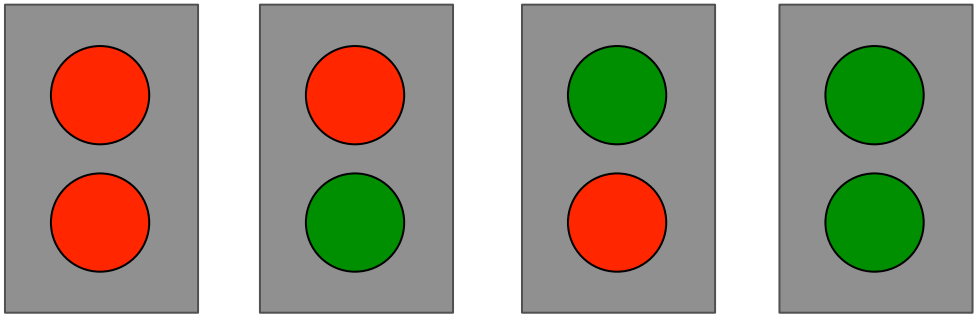
Binära tal

Binära tal

- I moderna datorer lagras som sagt värden digitalt
- Kontinuerliga värden är analoga, diskreta värden digitala
- I digitala system är den minsta enheten ett *binärt tal*, ofta kallad *bit*
- En bit har endast två möjliga värden: 0 eller 1, På eller Av, Sant eller Falskt, …
- Sekvenser av binära tal, ofta kallade bit-strängar, är det enda en dator kan lagra
- Därför måste all information kodas till sekvenser av nollor och ettor

Binära tal

- En enskild bit kan bara ha två möjliga värden
- Hur många kombinationer kan en sekvens av två bitar anta?



Binära tal

- En bit = 2 kombinationer
- Två bitar = 4 kombinationer
- Tre bitar = ?

- För varje bit vi läggar till dubbleras antalet kombinationer
- Tre bitar har alltså 8 kombinationer, fyra bitar 16 kombinationer, …
- Antalet kombinationer för N bitar blir då 2^N

Kodning

- All information från den verkliga världen kodas till en specifik bit-sträng
- Vi kan till exempel koda färgen röd till 100, grön till 010 och blå till 001
- Bokstaven A kan kodas till 0001, B till 0010, C till 0011, …

Datakapacitet

- För att kunna koda information måste vi veta hur många bitar som krävs
- Om vi till exempel ska koda alla tecken på ett tangentbord, räcker det med 4 bitar eller krävs kanske 8 bitar?
- Om vi bara använder 3 bitar finns det totalt 8 olika kombinationer på bit-strängen, så då kan vi bara koda bokstäverna A till H…

Datakapacitet

- Antalet bitar som krävs är proportionerligt mot antal möjliga värden informationen kan anta!
- Ska vi spara veckodag räcker 3 bitar då det finns sju dagar i veckan
- Ska vi spara månad krävs 4 bitar (16 kombinationer och vi har 12 månader)

Datakapacitet

Typ av information	Antal möjliga värden	Antal bitar som krävs
Veckodag	7	3
Månad	12	4
Dag på månad	31	5
Tecken på tangentbord	≈104	7
Dag på år	365	9

Datakapacitet

- Datakapacitet i ett datorsystem är mängden information som kan kodas av systemet
- Denna är direkt proportionerlig mot antalet bitar vi kan lagra, och mäts alltså i antal bitar
- Standardenheten är inte bitar, utan *byte*
- En byte är en bit-sträng med 8 bitar = 2^8 = 256 möjliga kombinationer

Word

- En annat mätvärde är word (översätts oftast inte till ord)
- Detta beror på hårdvaran hos datorsystemet
- Det motsvarar antalet bitar systemets processor kan hantera i en enhet
- Idag är datorsystemen oftast 64 bitar
- För 15 år sedan var 32 bitar vanligast, och för 30 år sedan 8 bitar

Prefix

- Prefix anges för större datakapaciteter: MB för megabyte, GB för gigabyte, …

Prefix	Symbol	Binär	Decimal
Kilo	K	2^{10}	$\approx 10^3$ (tusen)
Mega	M	2^{20}	$\approx 10^6$ (miljon)
Giga	G	2^{30}	$\approx 10^9$ (miljard)
Tera	T	2^{40}	$\approx 10^{12}$
Peta	P	2^{50}	$\approx 10^{15}$

Datakapacitet

Typ av information	Datakapacitet (bytes)
Tecken på tangentbord	1 B
10 sidors text	40 KB
Fem minuter MP3 musik	5 MB
Digitalt foto	5 MB (beroende på upplösning hos kameran…)
CD skiva	800 MB
DVD skiva	8,5 GB
Hela Wikipedia	≈ 6 TB
Lomonosov table (alla möjliga drag i schack med 7 pjäser kvar)	140 TB

Datatyper och kodning

Datatyper och kodning

- All digital data lagras som en sekvens av bitar
- Varje unik kombination av bitar motsvarar ett möjligt värde på den data som ska lagras
- Olika typer av data kodas på olika sätt: siffror, tecken, bilder, ljud, film, …

- Vi ska se på några exempel hur digital data lagras

Tal

Numerära system

- Ett numerärt system bestämmer hur tal representeras i skriftlig form
- I verkliga värden använder vi oftast decimalsystemet med basen 10
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- I datorvärlden används ofta det binära systemet med basen 2
  - 0, 1
- Ett annat vanligt system i datorvärlden är det hexadecimala systemet med basen 16
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Numerära system

- Siffran 10 kan betyda olika sak beroende på vilket numerärt system vi syftar på
- För att förtydliga kan vi ange basen på det numerära systemet i nedsänkt stil: 10_{10} syftar på det decimala systemet

Tal	Numerärt system	Decimalvärde
10_{10}	Decimal	10
10_{2}	Binärt	2
10_{16}	Hexadecimalt	16

Heltal

- Heltal representeras som en bit-sträng som översätts till decimalsystemet
- Högsta och lägsta möjliga talet beror på antalet bitar som används, och om vi behöver både positiva och negativa tal eller bara positiva tal
- Antalet möjliga heltal är 2^N-1 eftersom första talet alltid är 0

Heltal

Bit-sträng	Heltal i decimalform
00000000	0
00000001	1
00000010	2
00000011	3
00000100	4
…	…
11111110	254
11111111	255

Unsigned integer

Bit-sträng	Heltal i decimalform
00000000	-128
00000001	-127
00000010	-126
…	…
10000000	0
…	…
11111101	125
11111110	126
11111111	127

Signed integer

Decimaltal

- Att representera heltal med bit-strängar är ganska rakt på sak, men hur kan vi representera decimaltal som 2,31 eller 3,141592653589793?
- Vi måste då avsätta ett antal bitar till höger av bit-strängen för att hantera decimaler
- Resten av bitarna hanterar heltal
- Decimalerna blir mindre och mindre ju längre ifrån decimalavdelaren vi kommer

Decimaltal

- Antag att vi har talet $1{,}101_2$
- De tre bitarna längst till höger representerar alltså decimaldelen i talet
- I decimalform blir talet:

$$
\begin{aligned}
&1,101_2 = 1 \cdot 2^2 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = \\
&= 1 \cdot 1 + 1 \cdot \tfrac{1}{2} + 0 \cdot \tfrac{1}{4} + 1 \cdot \tfrac{1}{8} = \\
&= 1 + \tfrac{1}{2} + \tfrac{1}{8} = \\
&= 1,625
\end{aligned}
$$

Datatyper i Java

Datatyp	Bitar	Omfång
Byte	8	-128 till 127
Short	16	-32768 till 32767
Integer	32	-2147483648 till 2147483647 -2^{31} till $2^{31}-1$
Long	64	-2^{63} till $2^{63}-1$
Float	32	32-bit IEEE 754 floating point
Double	64	64-bit IEEE 754 floating point

Precision

- Ett problem med att koda decimaltal i digital form är *precision* – antalet decimaler vi kan lagra
- Om vi till exempel lagrar Pi, lagras det som 3,141 eller 3,14159265358979323846?
- Oavsett vad kommer vi aldrig att kunna lagra Pi eller 1/3 exakt
- Alla decimaltal med oändligt antal decimaler kommer att lagras som en approximation
- Datorer producerar därför ofta inkorrekta, dock väldigt exakta, decimaltal
- Om ett datorsystem använder 16 bitar för att lagra decimaltal, säger vi att systemet använder 16 bitars precision

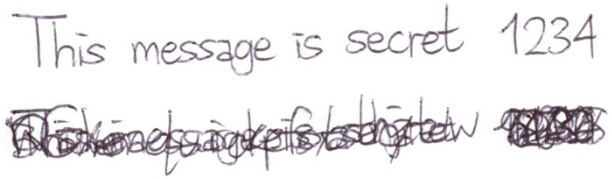
Underflow och Overflow

- Precision på decimaltal är en källa till fel i datorsystem
- Två andra är *underflow* och *overflow*
- Ett heltal som lagras med 8 bitar kan ha värden mellan 0 och 255
- Vad händer om vi beräknar 255+10?

Underflow och Overflow

- Resultatet ska bli 265, men eftersom 8 bitar inte kan lagra så höga tal kan beräkningen generera ett fel eller, ännu värre, börja om på 0 och bli 10!
- Om en beräkning resulterar i ett värde som ligger utanför vad vi kan lagra blir det overflow
- Underflow händer om vi försöker lagra ett decimaltal som är så litet att antalet bitar i decimaldelen inte räcker till, och lagras därför som 0

Text



Text

- När vi skriver text på en dator är vi vana vid att vi kan ändra font, storlek, typ (fet, normal, kursiv) och färg:

  Q **Q** Q *Q* **Q**

- Hur kan vi lagra detta som bit-strängar?
- En bokstav som visas på skärmen är en bild!
- Bokstäver kodas som heltal enligt specifika scheman som kallas teckenkodning

Teckenkodning

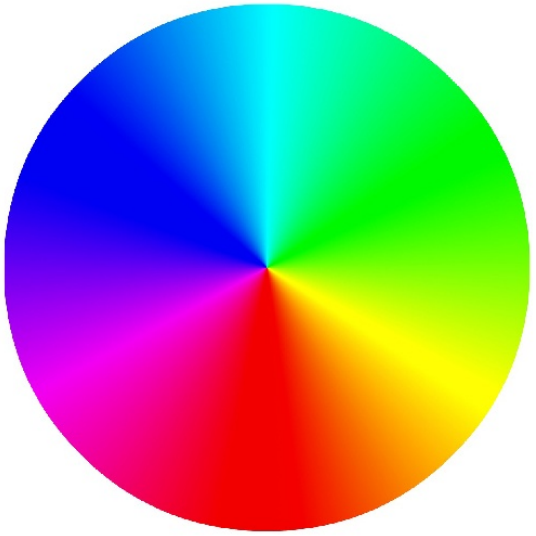
- Ett av de vanligaste formaten är ASCII, med plats för 128 tecken (7 bitar), som kom ut 1963

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29	30	31	32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39
(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59
< 60	= 61	> 62	? 63	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79
P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	^ 94	_ 95	` 96	a 97	b 98	c 99
d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119
x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	127	128	129	130	131	132	133	134	135	136	137	138	139

Teckenkodning

- Problemet med ASCII är att det inte finns plats för tecken som används i andra språk än engelska
- Det är också användbart med fler symboler
- ASCII har därför i de flesta datorsystem ersatts av Unicode, som stödjer alla språk
- Unicode har 21 bitar
- Eftersom 21 bitar inte får plats i en byte, används ofta varianterna UTF-8 (8 bitar) eller UTF-16 (16 bitar)
- UTF-8 är standardkodningen på Internet

Färger



Färger

- Våra ögon består av tappar och stavar
- Stavar kan bara uppfatta ljusstrålningens intensitet (ljusstyrka)
- Tappar kan uppfatta både intensitet och våglängd
- Det finns en typ av tappar för rött ljus, en annan för grönt ljus och en tredje för blått ljus
- Dessa blandas så att vi kan uppfatta alla färger
- Från biologin kan vi alltså dra slutsatsen att färger är 3-dimensionella

RGB

- Den vanligaste modellen för att representera färger i ett datorsystem är RGB
- Det använder sig av tre färgkanaler för huvudfärgerna rött, grönt och blått
- De tre kanalerna kan blandas för att visa andra färger
- Varje kanal har vanligtvis ett värde mellan 0 och 255, vilket resulterar i drygt 16 miljoner möjliga färger
- Varje kanal behöver 8 bitar = totalt 24 bitar

RGB

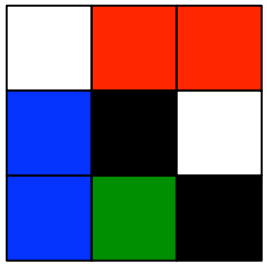
Färg	Namn	Bitar Röd Grön Blå	Decimal
	Röd	111111110000000000000000	(255,0,0)
	Grön	000000001111111100000000	(0,255,0)
	Blå	000000000000000011111111	(0,0,255)
	Gul	111111111111111100000000	(255,255,0)
	Svart	000000000000000000000000	(0,0,0)
	Vit	111111111111111111111111	(255,255,255)
	Grå	000011110000111100001111	(128,128,128)

Bilder

Bilder

- Den vanligaste kodningen för bilder är ett 2-dimensionellt nät av pixlar
- Varje pixel representerar en färg enligt RGB modellen
- Varje pixel lagras alltså med 24 bitar

Bilder



255	255	255
0	0	255
0	0	0

255	0	0
0	0	255
0	255	0

255	0	0
255	0	255
255	0	0

Bilder

- 248x400 pixlar = 99200 pixlar
- Varje pixel kräver 24 bitar
- Bildens storlek blir då 2380800 bitar ≈ 2 MB

- Bilder har också en header med lite information som dess bredd och höjd
- Dess storlek är dock försumbar

Storlek

- En liten bild som 248x400 pixlar behöver 2 MB för att lagras
- En 10 megapixel bild från en digitalkamera behöver cirka 30 MB
- En bild med bara gråskalor kräver en tredjedel så mycket utrymme som en färgbild (en grå kanal från 0 till 255)

Ljud

Ljud

- Ljud är ett fysiskt fenomen som uppstår när vågformer färdas genom luft
- Vågformerna får örats trumhinna att vibrera – vi hör ett ljud
- Komplexa ljud som tal eller musik består av kombinationer av ljudvågor med olika frekvenser
- En ljudvåg är analog, och för att lagras digitalt behöver den kodas med en process som kallas *sampling*

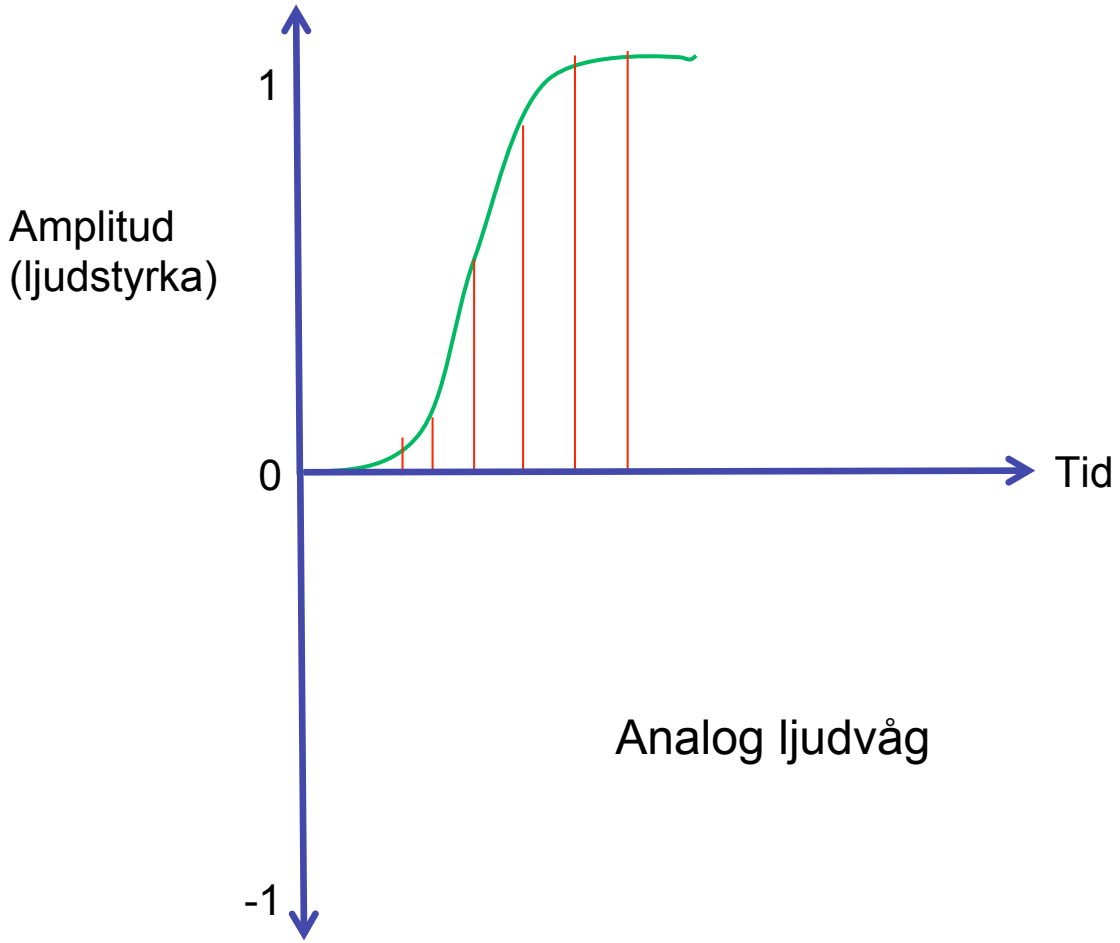
Frekvens

- Frekvens är hur ofta ljudvågor svänger
- Lägre frekvenser ger låga toner
- Höga frekvenser ger höga toner
- Frekvensen mäts i *hertz*, antal svängningar per sekund
- Människor hör ungefär mellan 20 Hz och 20 kHz

Sampling

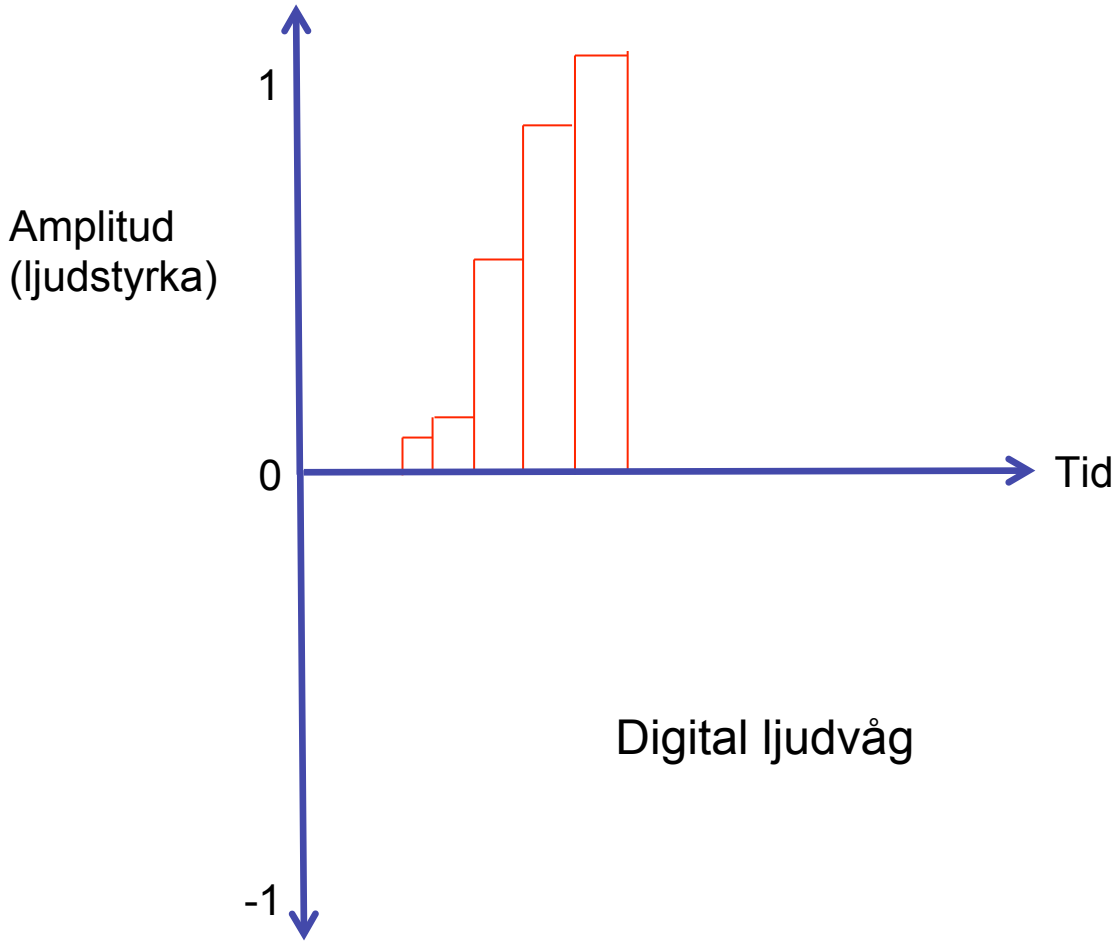
- Sampling går ut på att man med jämna tidsintervall läser av amplituden (ljudstyrkan) på ljudvågen
- Hur ofta detta görs avgörs av kodningens *samplingsfrekvens*
- Samplingsfrekvensen är hur många avläsningar per sekund som görs
- Högre samplingsfrekvens ger bättre ljudkvalitet men kräver större lagring

Sampling



Analog ljudvåg

Sampling



Amplitud

- Amplituden är ett kontinuerligt värde mellan -1 och 1
- Denna måste också översättas till ett digitalt värde
- Ju fler bitar som används, ju bättre ljudkvalitet
- CD ljud har 16 bitar
- Musik som streamas i högre kvalitet har ofta 24 bitar

Ljudkvalitet

- Ljudkvaliteten förbättras med högre samplingsfrekvens och bitar
- Det finns dock en gräns för hur små nyanser det mänskliga örat kan höra
- CD ljud har en samplingsfrekvens på 44,1 kHz och 16 bitar
- Varför har en samplingsfrekvens på drygt 40 kHz valts?

Ljudkvalitet

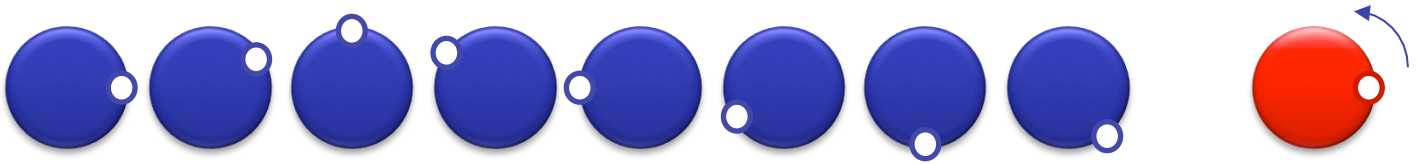
- Det visar sig att det finns en lägre gräns för vilken samplingsfrekvens vi kan välja
- Den lägre gränsen anges i Nyquists lag och innebär att samplingsfrekvensen måste vara minst dubbelt så hög som den högsta frekvensen
- Eftersom människor hör upp till cirka 20 kHz, måste samplingsfrekvensen för musik vara 40 kHz eller högre

Aliasing

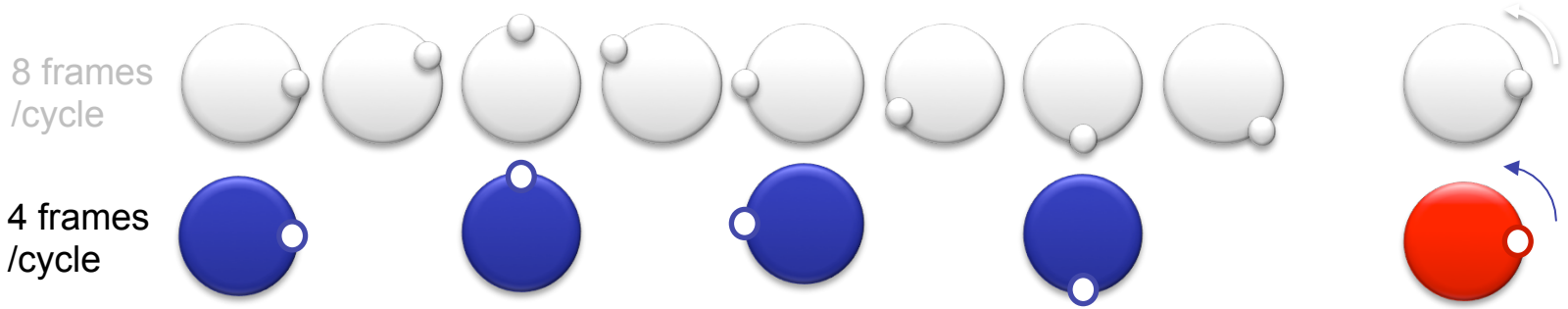
- Har vi en lägre samplingsfrekvens kan ett problem kallat *aliasing* uppstå
- När detta händer är samplingsfrekvensen så låg att flera olika analoga kurvor kan passa in på den digitala kurvan
- Detta kan man ibland se på film där propellrar på flygplan plötsligt börjar rotera baklänges
- De höga frekvenserna återges som låga frekvenser och i motsatt fas

Aliasing

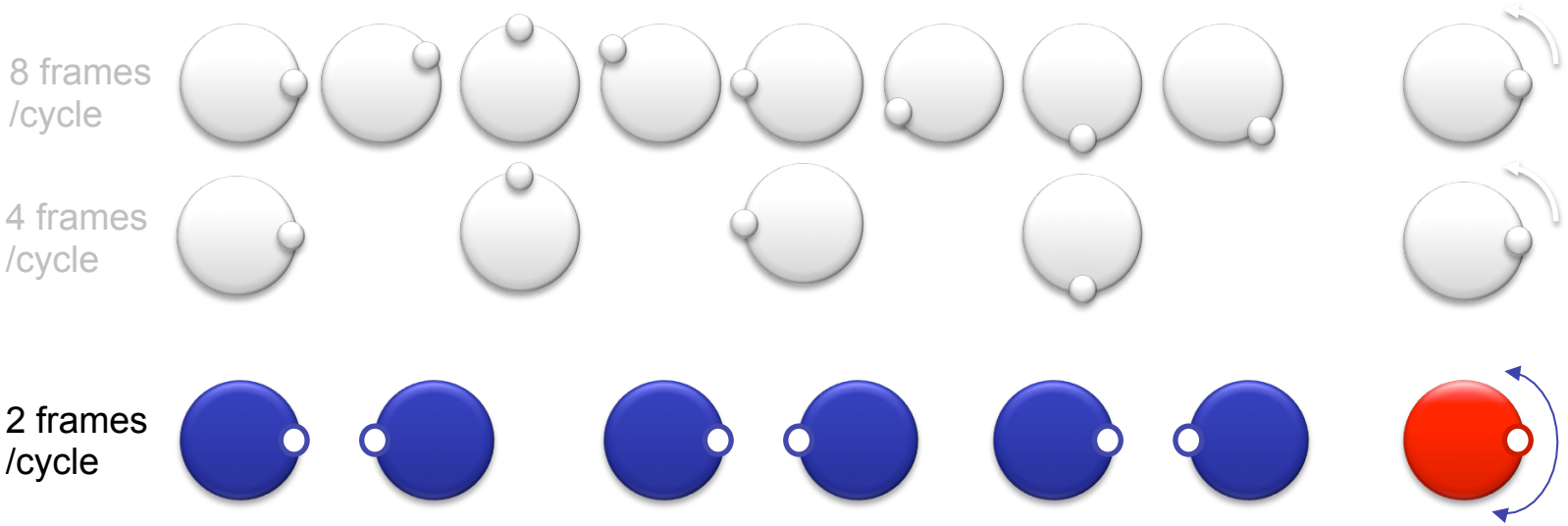
8 frames
/cycle



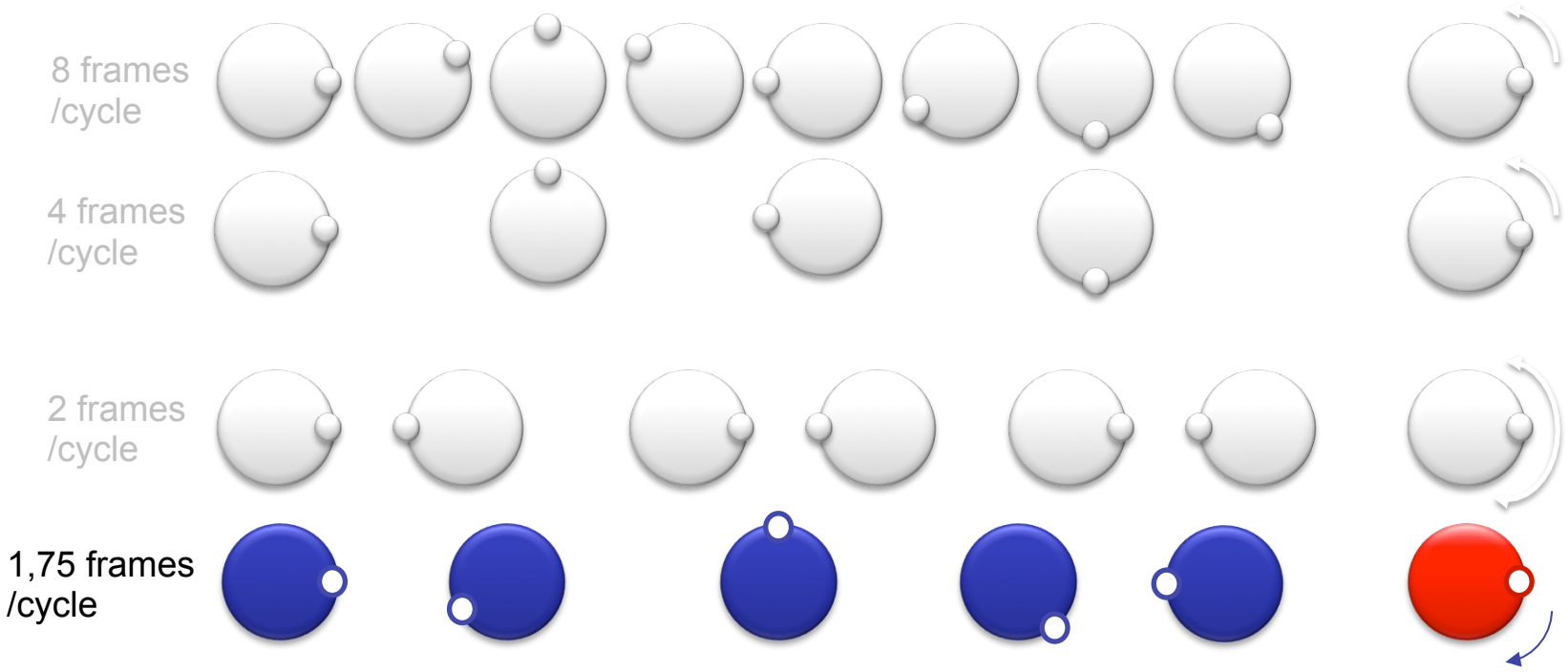
Aliasing



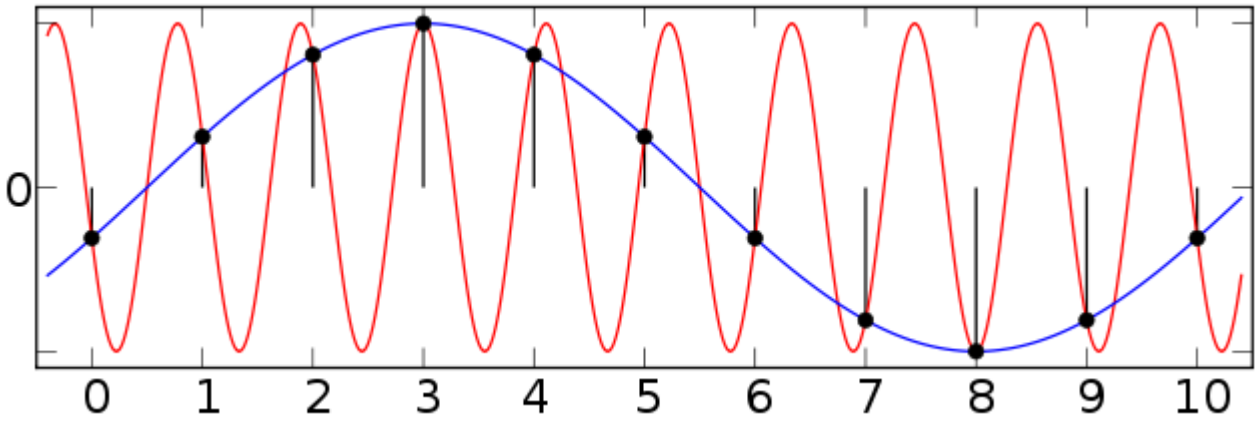
Aliasing



Aliasing


8 frames /cycle
4 frames /cycle
2 frames /cycle
1,75 frames /cycle

Aliasing



Två olika sinuskurvor, blå och röd, passar in på de samplingar vi har (svart).

Storlek

- Antag att vi har ett fem minuter långt ljud som är samplat i 48 kHz och 8 bitar
- Totalt behövs 48000*5*60 = 14,4 miljoner samplingar
- Varje sampling behöver 8 bitar (alltså en byte), så hela ljudfilen tar upp cirka 14 MB lagringsutrymme
- Om vi i stället använder 16 bitar behövs dubbelt så mycket utrymme, cirka 28 MB

Film

Film

- Full HD film har en upplösning på 1920x1080 pixels = 2073000 pixels
- Varje pixel tar som sagt upp 24 bitar, så varje bildruta kräver 2073000*24 bitar = 49766400 bitar
- Räknat i byte blir det 2073000*3 ≈ 6 MB
- Varje sekund visas 25 bilder vilket blir ungefär 150 Mbit/s
- Förutom bild ska även flerkanalsljud streamas
- Det blir en hel del data!

Kompression

Kompression

- Ljud och bild kräver mycket lagringsutrymme
- För att minska storleken på filer används kompression
- Kompression minskar antalet bitar som krävs för att lagra en fil och därmed minskar filstorleken
- En okomprimerad bild av hög kvalitet kan ta upp 10 MB lagringsutrymme
- Samma bild komprimerad, beroende på teknik och kvalitet, kan i stället ta upp 1 MB
- DVD video komprimeras till mellan 3 Mbit/s och 9,5 Mbit/s

Kompression

- Att komprimera en fil innan den lagras kräver en del beräkningsarbete för processorn
- För att visa en komprimerad fil krävs att vi dekomprimerar den vilket också kräver beräkningsarbete
- Det är dock ofta värt att komprimera ljud och bild

Kompression

- Syftet med komprimering är att eliminera onödiga bitar
- Vilka bitar som är onödiga är dock inte trivialt att identifiera, och beror till stor del på vilket typ av information som ska komprimeras (text, ljud, bild, …)

Enkel bildkomprimering

- Antag att vi har en svartvit bild
- Datan består då endast av ettor (vit) och nollor (svart)
- Beroende på hur bilden ser ut kan vi ha väldigt långa sekvenser av antingen ettor eller nollor:
  - 11111111110001111111111111111111
- I stället för att spara varje bit kan vi i stället koda om så att vi anger antalet vita pixlar, sen antalet svarta pixlar, sen vita pixlar igen, …

Enkel bildkomprimering

Bit-sträng	Kodning
11111111110001111111111111111111	10, 3, 19
11111111110000011111111111100000	10, 5, 14, 5

- Beroende på hur många bitar vi använder för att lagra varje sekvens kan vi spara en del utrymme
- Första raden kräver okomprimerad 32 bitar
- Använder vi 5 bitar per sekvens krävs 3*5 = 15 bitar vid kompression
- Andra raden kräver dock 4*5 = 20 bitar

Komprimeringsalgoritmer

- Det finns ett stort antal komprimeringsalgoritmer vi kan använda för olika typer av filer
- För bilder finns till exempel jpeg eller png
- För ljud finns mp3 eller flac
- Vi skiljer mellan algoritmer som är *lossless* eller *lossy*

Komprimeringsalgoritmer

- En lossless algoritm garanterar att den komprimerade datan är en exakt representation av den okomprimerade datan
- Ingen information förloras

- En lossy algoritm försöker identifiera bitar av liten betydelse och eliminera dessa
- Information förloras vid komprimeringen

Bildkomprimering

Lossless

Lossy

Bildkomprimering

- Bilder med skarpa konturer som text eller figurer ser sämre ut i lossy komprimering än foton
- Foton komprimeras därför ofta i jpeg (lossy)
- Skärmdumpar från program bör i stället komprimeras med png (lossless)

Ljudkomprimering

- Algoritmer för att komprimera ljud försöker eliminera mindre viktiga ljudvågor, som till exempel ljudvågor med hög frekvens
- Det troligen vanligaste formatet för ljudkomprimering är mp3 (lossy)
- Vill vi i stället ha en lossless algoritm kan vi använda flac

Variabler och Listor

Variabler och Instruktioner

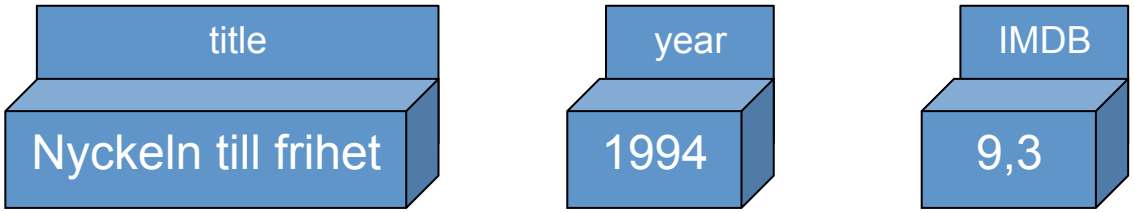
- Två viktiga delar i programkod är variabler och instruktioner
- Instruktioner är när vi utför något, till exempel en addition
- Variabler är behållare för data, och används när vi ska lagra data i minnet

Variabler

- En variabel kan ses som en namngiven plats i datorns minne
- Det är en behållare där vi kan lagra data av någon typ
  - Heltal, decimaltal, boolean, text, …
- När en variabel skapas allokeras den mängd minne som variabeln kräver
  - Vilket som vi pratat om beror på datatypen

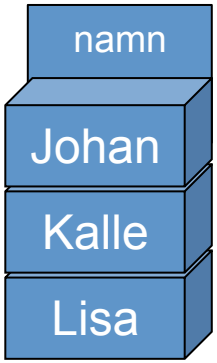
Variabler

- Variabler kan sedan användas i vårt program
- Vi skiljer mellan variabler (värdet kan ändras) och konstanter (värdet anges en gång och kan sedan inte ändras)



Lista

- En variabel kan vara innehålla ett värde
- Ibland behöver vi lagra flera värden av samma typ, till exempel titeln på ett antal filmer eller pixlar i en bild
- En lista är en namngiven behållare för flera värden:



Lista

- Varje plats i en lista är en behållare för ett värde
- Alla värden i en lista är av samma typ
- När vi ska använda ett värde i en lista behöver vi ange vilken plats i listan värdet finns på
- Detta kallas för index
- Index börjar på 0 och slutar på listans längd - 1
  - Det kan förekomma programspråk där index börjar på 1 men det är ovanligt

Namngivning

Namngivning

- Alla variabler och listor måste namnges
  - Och metoder, klasser, datastrukturer, … (mer om dessa senare)
- Det finns en del krav vi måste följa vid namngivning:
  - Namnet måste vara unikt (skiftlägeskänsligt)
  - Det får inte börja på en siffra
  - Det får endast innehålla bokstäver, siffror och underscore _
- Om vi behöver en lista med namn kan vi till exempel välja att kalla den:
  - xzfgr
  - a_5
  - names

Bra och mindre bra namn

- *xzfgr* eller *a_5* är mindre bra än *names*
- Bra namn ska vara korta och deskriptiva
- De ska beskriva en funktion, som att listan innehåller namn
- I stället för mellanslag (de är som sagt inte tillåtna) använder vi stor bokstav eller underscore för sammansatta namn:
  - listOfNames
  - list_of_names
- Namn med endast en bokstav ska i regel undvikas, förutom:
  - Variabler i matematiska funktioner: y = 2a+5b
  - Koordinater (x, y och z)
  - Räknare i upprepningar (mer om detta senare)