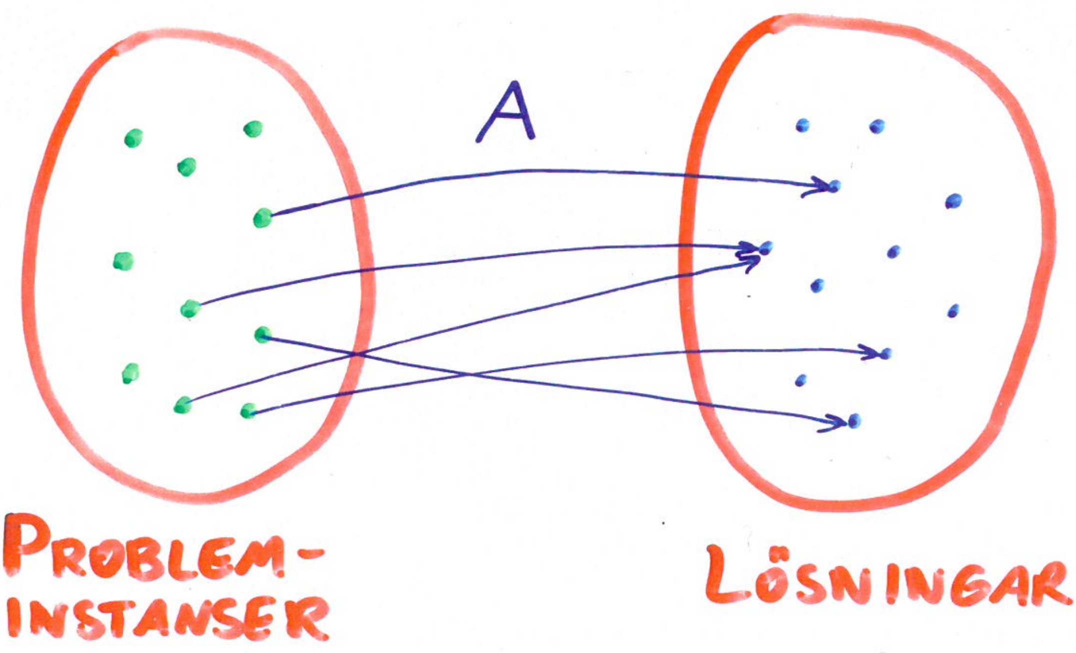
DEFINITION:

EN **ALGORITM** ÄR EN ÄNDLIG BESKRIVNING AV HUR MAN STEG FÖR STEG LÖSER ETT PROBLEM.

EN ALGORITM TAR OFTAST **INDATA** SOM BESKRIVER EN **PROBLEMINSTANS** OCH PRODUCERAR **UTDATA** SOM BESKRIVER PROBLEMINSTANSENS **LÖSNING**.

EN ALGORITM KAN SES SOM EN FUNKTION

A : PROBLEMINSTANSER → LÖSNINGAR



Analys av algoritmer

Tidskomplexitet

– Hur lång tid tar algoritmen i värsta fallet?

Som funktion av vad?

Vad är ett tidssteg?

Minneskomplexitet

– Hur stort minne behöver algoritmen i värsta fallet?

Som funktion av vad?

Mätt i vad?

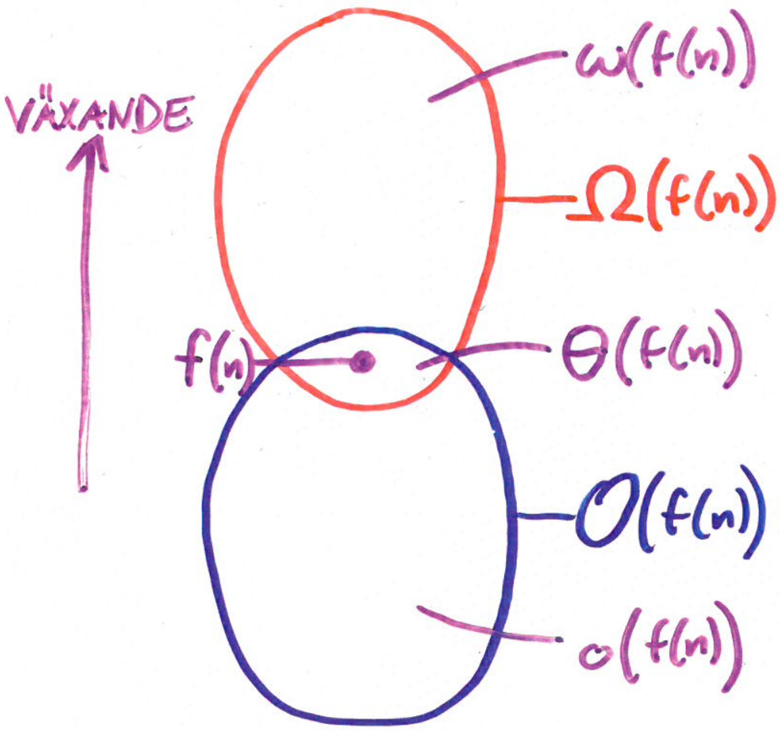
Tänk på att funktions- och proceduranrop också tar minne.

HUR KOMPLEXITET KAN ANGES

HUR ÄNDRAS KOMPLEXITETEN FÖR VÄXANDE STORLEK n PÅ INDATA?

ASYMPTOTISK KOMPLEXITET — VAD HÄNDER NÄR n VÄXER MOT OÄNDLIGHETEN?

MYCKET ENKLARE OM VI BORTSER FRÅN KONSTANTA FAKTORER.



$\omega(f(n))$ VÄXER SNABBARE ÄN $f(n)$

$\Omega(f(n))$ VÄXER MINST LIKA SNABBT SOM $f(n)$

$\Theta(f(n))$ VÄXER LIKA SNABBT SOM $f(n)$

$O(f(n))$ VÄXER HÖGST LIKA SNABBT SOM $f(n)$

$o(f(n))$ VÄXER LÅNGSAMMARE ÄN $f(n)$

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \begin{cases} 0 & \text{om } g(n) \in o(f(n)) \\ c>0 & \text{om } g(n) \in \Theta(f(n)) \\ \infty & \text{om } g(n) \in \omega(f(n)) \end{cases}$$

$O(f(n))$

$\Omega(f(n))$

EXEMPEL: BINÄRSÖKNING

BINSEARCH $(v[a..b], x) =$
IF $a<b$ THEN
  $m \longleftarrow \left\lfloor \frac{a+b}{2} \right\rfloor$
  IF $v[m]$.KEY $< x$ THEN
    RETURN BINSEARCH$(v[m+1..b], x)$
  ELSE RETURN BINSEARCH$(v[a..m], x)$
IF $v[a]$.KEY $= x$ THEN RETURN a
ELSE RETURN 'NOT FOUND'

ANALYS:

LÅT $T(n) = \overbrace{\text{TIDEN}}^{\text{I VÄRSTA FALL}}$ ATT SÖKA BLAND n TAL MED BINSEARCH.

$$T(n) = \begin{cases} \Theta(1) & \text{OM } n=1 \\ T\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(1) & \text{OM } n>1 \end{cases}$$

OM $n = 2^m$ FÅR VI $T(n) = \begin{cases} \Theta(1) & \text{OM } n=1 \\ T\left(\frac{n}{2}\right) + \Theta(1) & \text{OM } n>1 \end{cases}$

MÄSTARSATSEN (MASTER THEOREM) SÄGER DÅ

$$n^{\log_2 1} = n^0 \in \Theta(1)$$

DÅ ÄR $T(n) = \underline{\Theta(\log n)}$

LÖSNING TILL VANLIGA REKURSIONSEKVATIONER

SATS: ("MASTER THEOREM")

OM $a \geq 1, b > 1, d > 0$ SÅ HAR REKURSIONSEKVATIONEN

$$\begin{cases} T(n) = aT\left(\frac{n}{b}\right) + f(n) \\ T(1) = d \end{cases}$$

DEN ASYMPTOTISKA LÖSNINGEN

- $T(n) = \Theta(n^{\log_b a})$ OM $f(n) = \mathcal{O}(n^{\log_b a - \varepsilon})$ FÖR NÅGOT $\varepsilon > 0$
- $T(n) = \Theta(n^{\log_b a} \log n)$ OM $f(n) = \Theta(n^{\log_b a})$
- $T(n) = \Theta(f(n))$ OM $f(n) = \Omega(n^{\log_b a + \varepsilon})$ FÖR NÅGOT $\varepsilon > 0$

  OCH $af\left(\frac{n}{b}\right) \leq c \cdot f(n)$ FÖR NÅGON KONSTANT $c < 1$

  FÖR ALLA TILLRÄCKLIGT STORA n.

ANALYS AV PROBLEM

RINGA IN ETT PROBLEMS KOMPLEXITET!

ÖVRE GRÄNS:

GE EN ALGORITM SOM LÖSER PROBLEMET.

ALGORITMENS KOMPLEXITET ÄR EN ÖVRE GRÄNS FÖR PROBLEMETS KOMPLEXITET.

UNDRE GRÄNS:

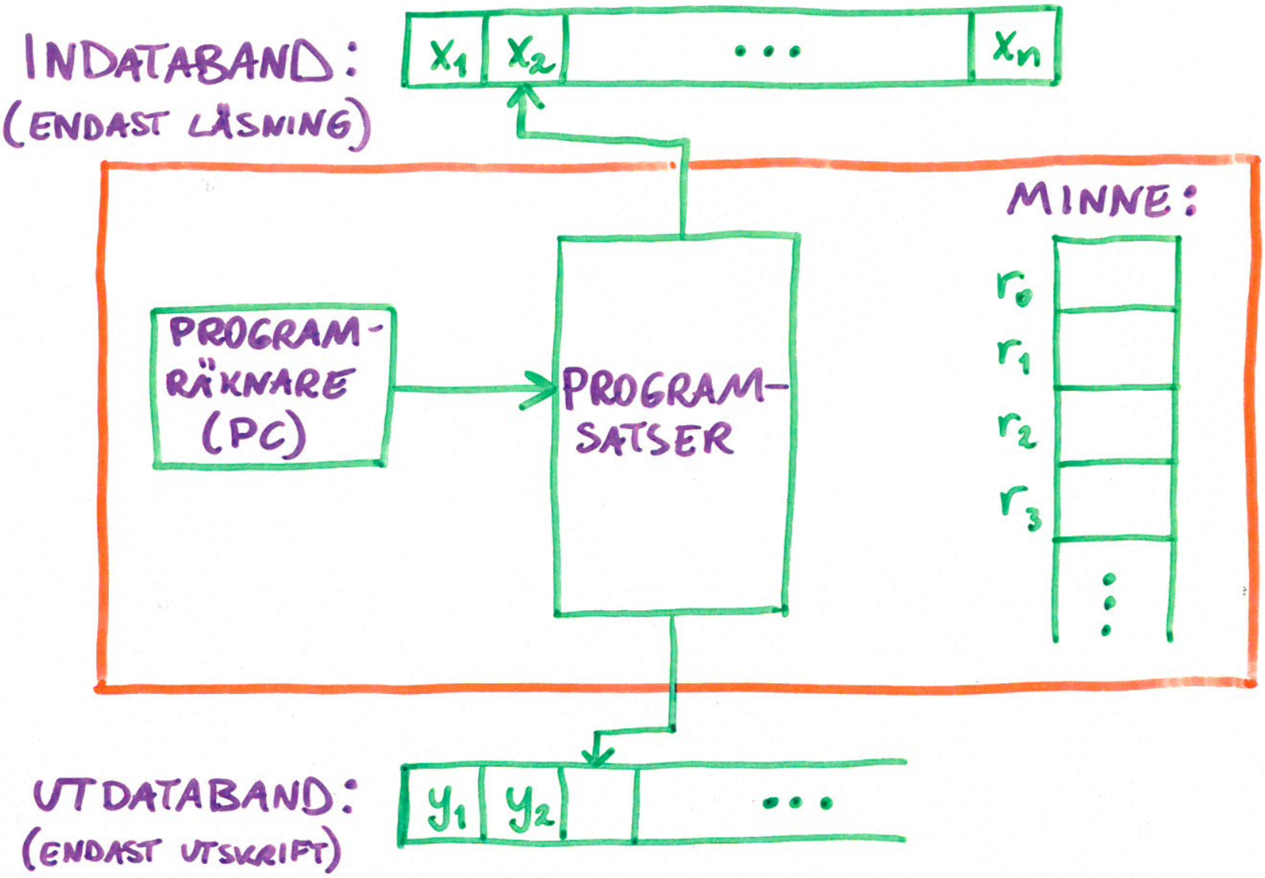
OFTA SVÅRT ATT ANGE.
EGENSKAPER HOS PROBLEMET MÅSTE ANVÄNDAS.

EXEMPEL:

- MÅSTE TITTA PÅ ALLA INDATA $\Rightarrow \Omega(n)$
- MÅSTE PRODUCERA HELA UTDATA
- BESLUTSTRÄD – ETT VISST ANTAL OLIKA FALL MÅSTE SÄRSKILJAS

BERÄKNINGSMODELL 1: RAM

(RANDOM ACCESS MACHINE)



PROGRAMMET BESTÅR AV VANLIGA SATSER SOM UTFÖRS SEKVENSIELLT (INTE PARALLELLT).

VARJE SATS KAN BARA LÄSA OCH PÅVERKA ETT KONSTANT ANTAL MINNESPLATSER. BARA EN SYMBOL KAN LÄSAS/SKRIVAS I TAGET. VARJE SATS TAR KONSTANT TID.

PÅ GRUND AV $O(\ )$-NOTATIONENS ROBUSTHET KAN VI STRUNTA I VILKA VÄRDEN KONSTANTERNA HAR

KOSTNADSMÅTT

ENHETSKOSTNAD

- VARJE OPERATION TAR EN TIDSENHET
- VARJE VARIABEL TAR EN MINNESENHET

BERÄKNINGSMODELL: RAM

BITKOSTNAD

- VARJE BITOPERATION TAR EN TIDSENHET
- VARJE BIT TAR UPP EN MINNESENHET

BERÄKNINGSMODELLER $\begin{cases} \text{RAM MED BEGRÄNSAD ORDLÄNGD} \\ \text{TURINGMASKIN} \end{cases}$

ANVÄNDS NÄR ALGORITMEN RÄKNAR MED TAL AV GODTYCKLIG STORLEK.

EXEMPEL: ADDITION AV TVÅ n-BITSHELTAL

TID $O(1)$ MED ENHETSKOSTNAD

TID $O(n)$ MED BITKOSTNAD