ADK OCH SVALKO

# Föreläsning 6

## Undre gränser

- Med beslutsträd
  Exempel: sortering
- Med tuff motståndare
  Exempel: talgissning
  median

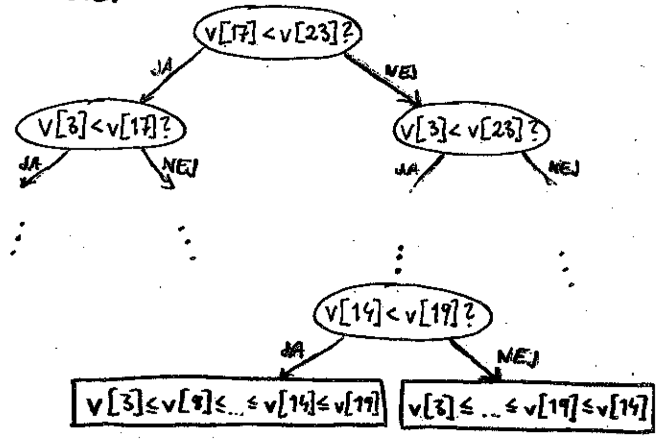
## Smart algoritm

- För att hitta i-te minsta talet

Hur många jämförelser krävs minst för att sortera $n$ element?

$$A: \{n \text{ element}\} \rightarrow \{n\text{-permutationer}\}$$

Anta att den enda operationen (förutom tilldelning) som kan göras på element är jämförelse mellan två element.
Då kan algoritmen $A$ beskrivas som ett beslutsträd:




Trädets höjd = tidskomplexiteten

Antalet löv ≥ antalet permutationer = $n!$ (om inga permutationer på förhand kan uteslutas)

Ett binärträd av höjd $h$ har högst $2^h$ löv

⇒ Tidskomplexiteten är $\Omega(n \log n)$

## Talgissning med tuff motståndare

Gissa ett tal mellan 1 och 10 — 1 ⊢ 10
**3**
Större! — 4 ⊢ 10
**8**
Mindre! — 4 ⊢ 7
**5**
Större! — 6 ⊢ 7
**6**
Större! — 7 ⊢ 7
Du klarade det inte på 4 gissningar. Rätt svar var 7.

```
GISSNING ← 0
MIN ← 1; MAX ← 10
WRITE("Gissa ett tal mellan " MIN " och " MAX)
WHILE MIN < MAX DO
   GISSNING ← GISSNING + 1
   X ← ReadInteger()
   M ← (MIN + MAX)/2
   IF X<M THEN WRITE("Större!") ELSE WRITE("Mindre!")
   IF X<M AND X≥MIN THEN MIN ← X+1
   IF X≥M AND X≤MAX THEN MAX ← X-1
WRITE("Du klarade det inte på " GISSNING " gissningar)
WRITE("Rätt svar var " MIN)
```

# Komplexitet för medianproblemet

**Problem:** Hitta medianen bland $n$ element

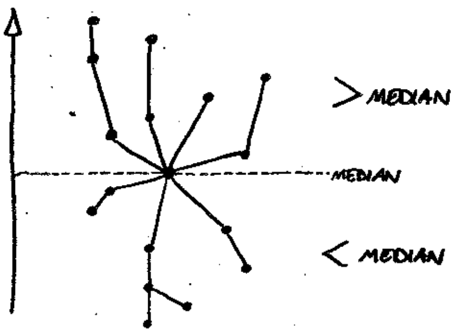
Hur många jämförelser krävs för att hitta medianelementet i värsta fallet?

## Trivial övre gräns:

Följande algoritm gör $O(n \log n)$ jämförelser
Mergesort($S[1..n]$); return $S[\frac{n+1}{2}]$

## Trivial undre gräns:

Varje algoritm som hittar medianen måste göra en kedja av jämförelser mellan varje annat tal och medianen:




Alltså krävs minst $n-1$ jämförelser.

# Undre gräns för median med tuff motståndare

Idé: Visa att en tuff motståndare kan tvinga algoritmen att göra onyttiga jämförelser

Jämförelsen $x > y$ är onyttig om $x >$ medianen och $y <$ medianen.

Taktik för tuffa motståndaren:
Välj $0$ som median
Tilldela element ett värde först då algoritmen vill jämföra det med något annat element

| Jämförelse mellan | Svara | Tilldela |
|---|---|---|
| nytt och nytt | $>$ | första positivt, andra negativt |
| positivt och nytt | $>$ | negativt tal |
| negativt och nytt | $<$ | positivt tal |
| känt och känt | sanningen | — |

När $\frac{n-1}{2}$ negativa eller $\frac{n-1}{2}$ positiva värden har tilldelats har inte motståndaren något val längre.

Varje algoritm kan på så sätt tvingas att göra $\frac{n-1}{2}$ onyttiga jämförelser utöver dom $n-1$ nyttiga.

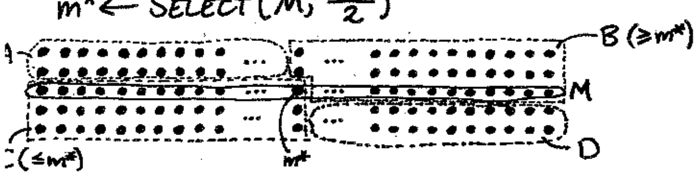
Vi får undre gränsen $n-1+\frac{n-1}{2} = \frac{3n-3}{2}$ jämförelser.

Bästa kända undre gränsen är $2{,}01n$ [Dor, Håstad, Ulfberg, Zwick 2001]

# Hitta i-te minsta talet i S

**Select($S[1..n]$, $i$) =**
If $n \le 5$ then Sortera($S[1..n]$); return $S[i]$

Dela upp $S$ i $\frac{n}{5}$ grupper med 5 tal i varje.
Hitta medianen i varje grupp och partitionera gruppen efteråt.
$M \leftarrow \{\text{medianer}\}$
$m^* \leftarrow$ Select$(M, \frac{n/5}{2})$




$S_1 \leftarrow C \cup \{x \in A \cup D : x \le m^*\}$
$S_2 \leftarrow B \cup \{x \in A \cup D : x > m^*\}$

If $i = |S_1|$ then return $m^*$
If $i < |S_1|$ then return Select$(S_1, i)$
else return Select$(S_2, i-(n-S_2))$

**Komplexitetsanalys:** (antal jämförelser)

$$T(n) \le \underbrace{6\cdot\frac{n}{5}}_{\text{hitta medianerna i grupperna}} + \underbrace{T\left(\frac{n}{5}\right)}_{\text{hitta } m^*} + \underbrace{2\cdot\frac{n}{5}}_{\text{fördela på } S_1 \text{ och } S_2} + \underbrace{T\left(\frac{n}{5}+\frac{n}{2}\right)}_{\text{rekursion på } S_1 \text{ eller } S_2} = 1.6n + T(0.2n) + T(0.7n)$$

Man kan visa att $T(n) \le 16n$
Detta ger övre gränsen $16n$ för medianproblemet.
Bästa kända övre gränsen är $2.95n$ [Dor, Zwick 1999]