# Complexiteit

## College 1

Docent: Lieuwe Vinkhuijzen

3 februari 2020

# Het 15-spel

**Spel met twee spelers**
Kies om beurten een getal uit $1, 2, 3, 4, 5, 6, 7, 8, 9$.
Wie het eerst met 3 van zijn eigen cijfers 15 kan maken, wint.

# Praktische zaken

- **Hoorcollege:** Maandag 11:15-13:00 in HL214
  - Zometeen ook hoorcollege 14:15-16:00 in Snellius
- **Werkcollege:** Maandag 14:15-16:00 in Snellius
- **Tentamen:** 8 juni 2020
- **Hertentamen:** 7 juli 2020
- **Assistenten:** Sebastiaan Brand, Daniëlle Gramsbergen
- **Vragen:** gewoon stellen
- **Vragen:** `l.t.vinkhuijzen@liacs.leidenuniv.nl`
- Website:
  `http://liacs.leidenuniv.nl/~vinkhuijzenlt`

# Praktische zaken

- **Huiswerk:** 4 opgaven
- **Eerste huiswerkopgave:** 2 maart
- **Eindcijfer** $=$ Tentamencijfer $+ \frac{1}{10}$Gemiddeld huiswerkcijfer

# Voorbeeld: CNF-Satisfiability

**Input:** $\psi(a, b, c) = (a \vee b) \wedge (a \vee c) \wedge (a \vee \neg b) \wedge (a \vee b \vee \neg c)$
**Vraag:** Is er $(a, b, c) \in \{0, 1\}^3$ met $\psi(a, b, c) = \text{True}$?

# Voorbeeld: CNF-Satisfiability

**Input:** $\psi(a, b, c) = (a \vee b) \wedge (a \vee c) \wedge (a \vee \neg b) \wedge (a \vee b \vee \neg c)$
**Vraag:** Is er $(a, b, c) \in \{0, 1\}^3$ met $\psi(a, b, c) = \text{True}$?

---

```
1: procedure CNF-SAT(ψ(x_1, ..., x_n))
2:     for x⃗ ∈ {0,1}^n do
3:         if ψ(x⃗) = True then
4:             return True
5:         end if
6:     end for
7:     return False
8: end procedure
```

**Aantal $\psi(\vec{x})$ evaluaties:** $\min(\text{Models}(\psi))$
**Worst case:** $2^n$ (Wanneer?)

# Voorbeeld: Kliek

**Input:** $G = (V, E), k \geq 0$
**Vraag:** Is er een $S \subseteq V$ van grootte $|S| \geq k$, zodanig dat $S$ een *kliek* van $G$ is?
**Definitie.** (*Kliek*) Een verzameling knopen $S \subseteq V$ is een kliek van $G$, als $\forall (u, v) \in S^2 : (u, v) \in E$.

# Voorbeeld: Kliek

**Input:** $G = (V, E), k \geq 0$
**Vraag:** Is er een $S \subseteq V$ van grootte $|S| \geq k$, zodanig dat $S$ een *kliek* van $G$ is?
**Definitie.** (*Kliek*) Een verzameling knopen $S \subseteq V$ is een kliek van $G$, als $\forall (u, v) \in S^2 : (u, v) \in E$.

---

```
1: procedure KLIEK(G = (V, E), k)
2:     for S ⊆ V met |S| = k do
3:         if ISKLIEK(S, G) then
4:             return True
5:         end if
6:     end for
7:     return False
8: end procedure
```

# Van CNF-Satisfiability naar Kliek

We maken van een formule, een graaf die een misschien een kliek heeft.

$$\begin{aligned} \psi = &(a \vee b \vee c) && \text{Clausule 1} \\ &\wedge(\neg a \vee c \vee d) && \text{Clausule 2} \\ &\wedge(\neg b \vee \neg c) && \text{Clausule 3} \end{aligned}$$

# Van CNF-Satisfiability naar Kliek

We maken van een formule, een graaf die een misschien een kliek heeft.

$$\begin{aligned} \psi = & (a \vee b \vee c) && \text{Clausule 1} \\ & \wedge (\neg a \vee c \vee d) && \text{Clausule 2} \\ & \wedge (\neg b \vee \neg c) && \text{Clausule 3} \end{aligned}$$

Maak graaf met $3 + 3 + 2 = 8$ knopen. Kies $k = 3$.
(**Waarschuwing:** De graaf is nogal onoverzichtelijk)

**Complexiteit:** Wat kunnen computers efficiënt berekenen, en wat niet?

# Voorbeeld: Fibonacchi

$$F(n) = F(n-1) + F(n-2) \qquad F(1) = F(2) = 1$$

$1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34, \ldots$

# Voorbeeld: Fibonacchi

$$F(n) = F(n-1) + F(n-2) \qquad F(1) = F(2) = 1$$

$1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34, \ldots$

---

```
1: procedure FIBONACCHI(n)
2:     if n ≤ 2 then
3:         return 1
4:     else
5:         return FIBONACCHI(n − 1) + FIBONACCHI(n − 2)
6:     end if
7: end procedure
```

**Aantal optellingen:** $F(n) \approx \phi^n \approx 1.6180^n$

# Voorbeeld: Fibonacchi

$$F(n) = F(n-1) + F(n-2) \qquad F(1) = F(2) = 1$$

$1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34, \ldots$

---

```
1: procedure FIBONACCHI-2(n)
2:     F[1] := 1                          ▷ F is een array
3:     F[2] := 1
4:     for i = 3 ... n do
5:         F[i] := F[i − 1] + F[i − 2]
6:     end for
7:     return F[n]
8: end procedure
```

**Aantal optellingen** $n - 2$

# Voorbeeld: Grootste gemene deler

$$ggd(n, m) = \max d : \quad d|n \text{ en } d|m$$

**Voorbeeld:** $ggd(12, 20) = 4$

# Voorbeeld: Grootste gemene deler

$$ggd(n, m) = \max d : \ \ d|n \text{ en } d|m$$

**Voorbeeld:** $ggd(12, 20) = 4$

---

```
1: procedure GGD(n, m)                          ▷ Stel n ≤ m
2:     ggd := 1
3:     for d = 2 . . . n do
4:         if n%d = 0 en m%d = 0 then
5:             ggd := d
6:         end if
7:     end for
8:     return ggd
9: end procedure
```

**Aantal modulo operaties:** $2ggd(n, m)$

**Worst case:** $n$

# Voorbeeld: Grootste gemene deler

$$ggd(n, m) = \max d : \ \ d|n \text{ en } d|m$$

**Voorbeeld:** $ggd(12, 20) = 4$

---

```
1: procedure EUCLIDES(n, m)                    ▷ Stel n ≤ m
2:     while n ≠ m do
3:         k := m%n
4:         n := m
5:         m := k
6:     end while
7:     return n
8: end procedure
```

**Aantal modulo operaties:** ???

**Worst case:** $\log_2(n)/\log_2(\phi) \approx 1.44 \log_2(n)$

**Vraag:** Hoe meet je de hoeveelheid tijd die een algoritme nodig heeft?

---

1. Identificeer operaties
2. Tel hoe vaak die worden uitgevoerd

# Voorbeeld: Tijd meten van Fibonacchi

```
1:  procedure FIBONACCHI-2(n)
2:      F[1] := 1                                   ▷ F is een array
3:      F[2] := 1
4:      i := 3
5:      while i ≤ n do
6:          j := i − 1
7:          k := i − 2
8:          F[i] := F[j] + F[k]
9:          i := i + 1
10:     end while
11:     return F[n]
12: end procedure
```

**Aantal operaties:** $3 + 4(n-2) + (n-1) = 5n - 6$

# Voorbeeld: Element zoeken in array

**Input:** Een array getallen: $A[1], \ldots, A[n]$, en een getal $t$
**Output:** De index waar $t$ zit, of 0

---

```
1:  procedure ZOEK(A[1], ..., A[n], t)
2:      i := 1
3:      while i ≤ n do
4:          if A[i] = t then
5:              return i
6:          end if
7:          i := i + 1
8:      end while
9:      return 0
10: end procedure
```

**Aantal operaties, worst case:** $1 + 3(n + 1) = 3n + 4$

# Voorbeeld: Binaïr zoeken in gesorteerd array

**Input:** Een oplopend gesorteerd array getallen:
$A[1] < A[2] < \ldots < A[n]$, en een getal $t$
**Output:** De index waar $t$ zit, of 0

---

```
1:  procedure ZOEK(A[1], ..., A[n], t)
2:      if n = 0 then
3:          return 0
4:      end if
5:      midden := A[⌈n/2⌉]
6:      if t = midden then
7:          return ⌈n/2⌉
8:      else
9:          if t < midden then
10:             return ZOEK(A[1], ..., A[⌈n/2⌉ − 1], t)
11:         else
12:             return ZOEK(A[⌈n/2⌉ + 1], ..., A[n], t)
13:         end if
14:     end if
```

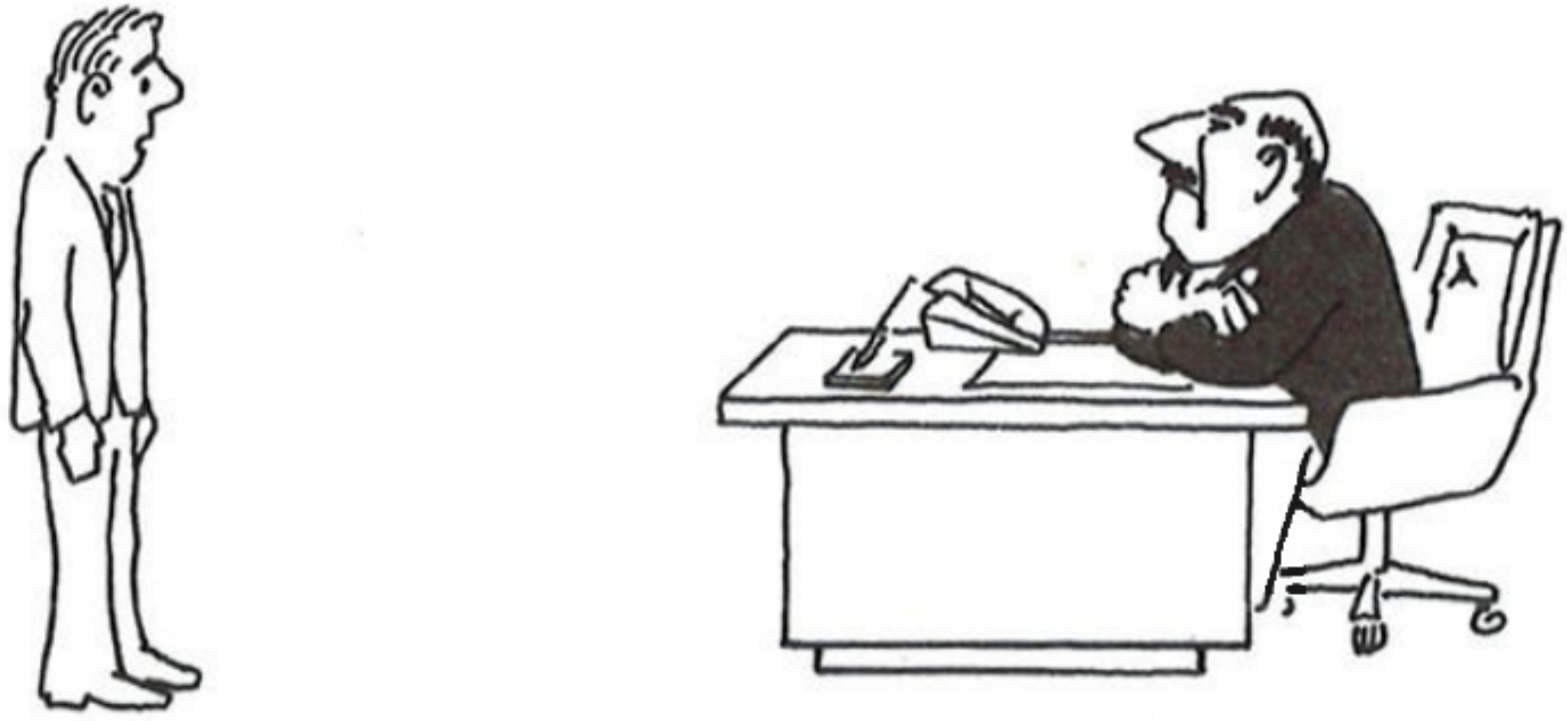
# Allerlei running times

| Puzzel | Poging 1 | Poging 2 |
|---|---|---|
| CNF-Satisfiability | $2^n$ | |
| Kliek | $\binom{n}{k}$ | |
| Fibonacchi | $1.6180^n$ | $5n - 6$ |
| ggd | $n$ | $1.44 \log_2(n)$ |
| Zoeken | $3n + 4$ | |
| Binaïr zoeken | $\log_2(n)$?? | |

Exponentieel

Polynomiaal

| Puzzel | Poging 1 | Poging 2 |
|---|---|---|
| CNF-Satisfiability | $2^n$ | |
| Kliek | $\binom{n}{k}$ | |
| Fibonacchi | $1.6180^n$ | $5n - 6$ |
| ggd | $n$ | $1.44 \log_2(n)$ |
| Zoeken | $3n + 4$ | |
| Binaïr zoeken | $\log_2(n)$?? | |



"I can't find an efficient algorithm, I guess I'm just too dumb."

"I can't find an efficient algorithm, because no such algorithm is possible!"

"I can't find an efficient algorithm, but neither can all these famous people."