Iteratieve Algoritmen

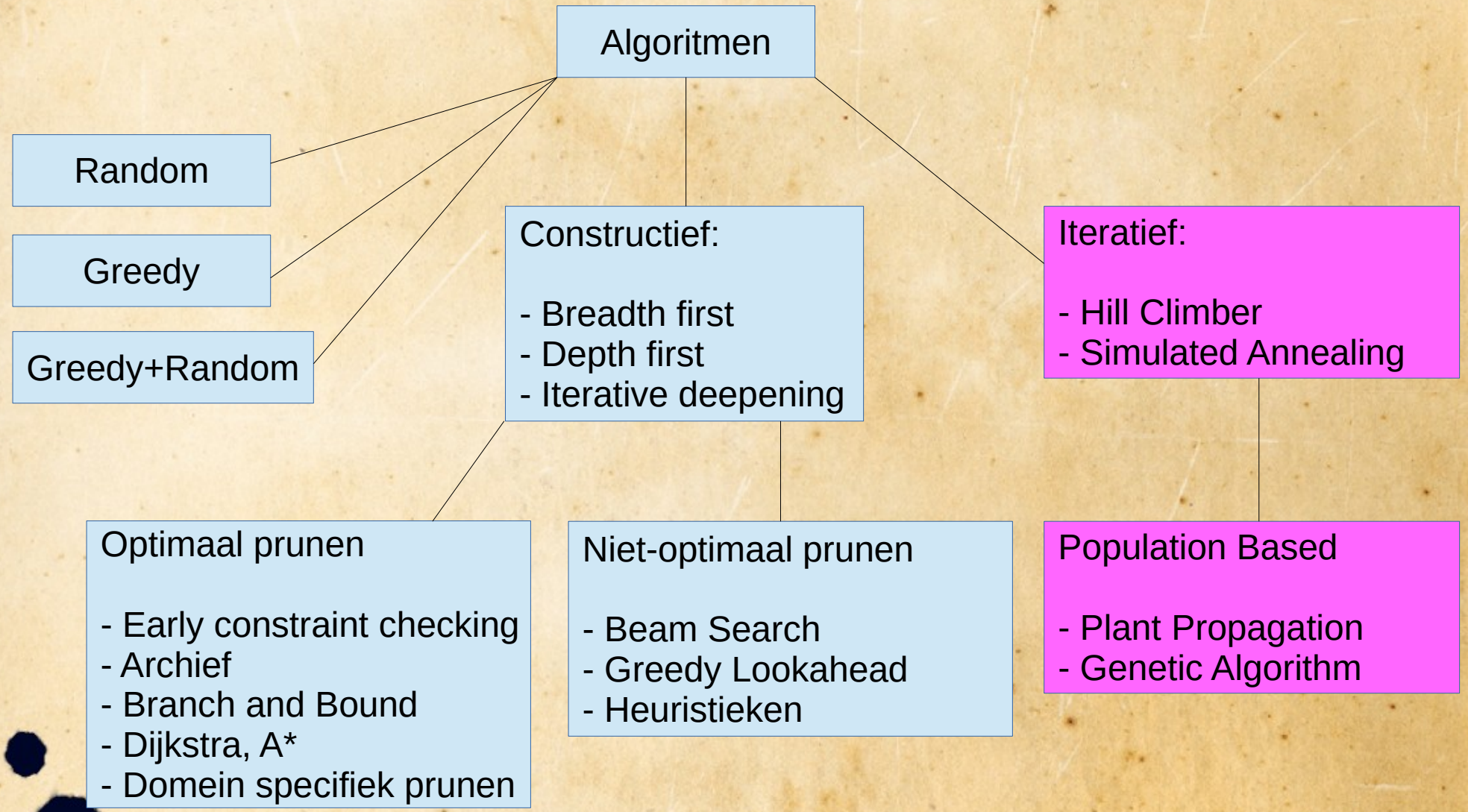
Bas Terwijn <b.terwijn@uva.nl>

Algoritmen en Heuristieken

Minor Programmeren

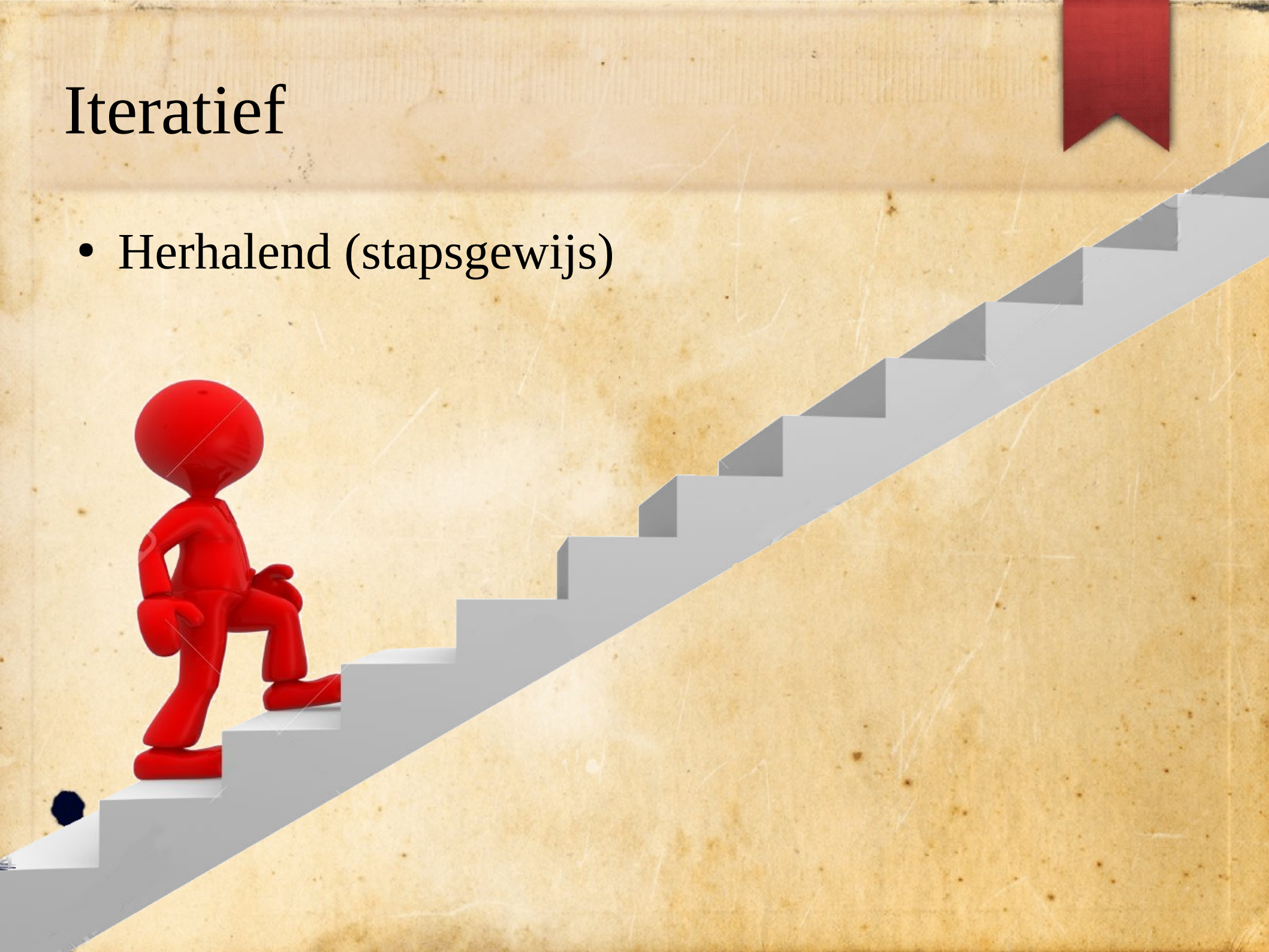
Universiteit van Amsterdam

Algoritmen


Algoritmen
Random
Greedy
Greedy+Random
Constructief:
- Breadth first
- Depth first
- Iterative deepening
Iteratief:
- Hill Climber
- Simulated Annealing
Optimaal prunen
- Early constraint checking
- Archief
- Branch and Bound
- Dijkstra, A*
- Domein specifiek prunen
Niet-optimaal prunen
- Beam Search
- Greedy Lookahead
- Heuristieken
Population Based
- Plant Propagation
- Genetic Algorithm

Iteratief

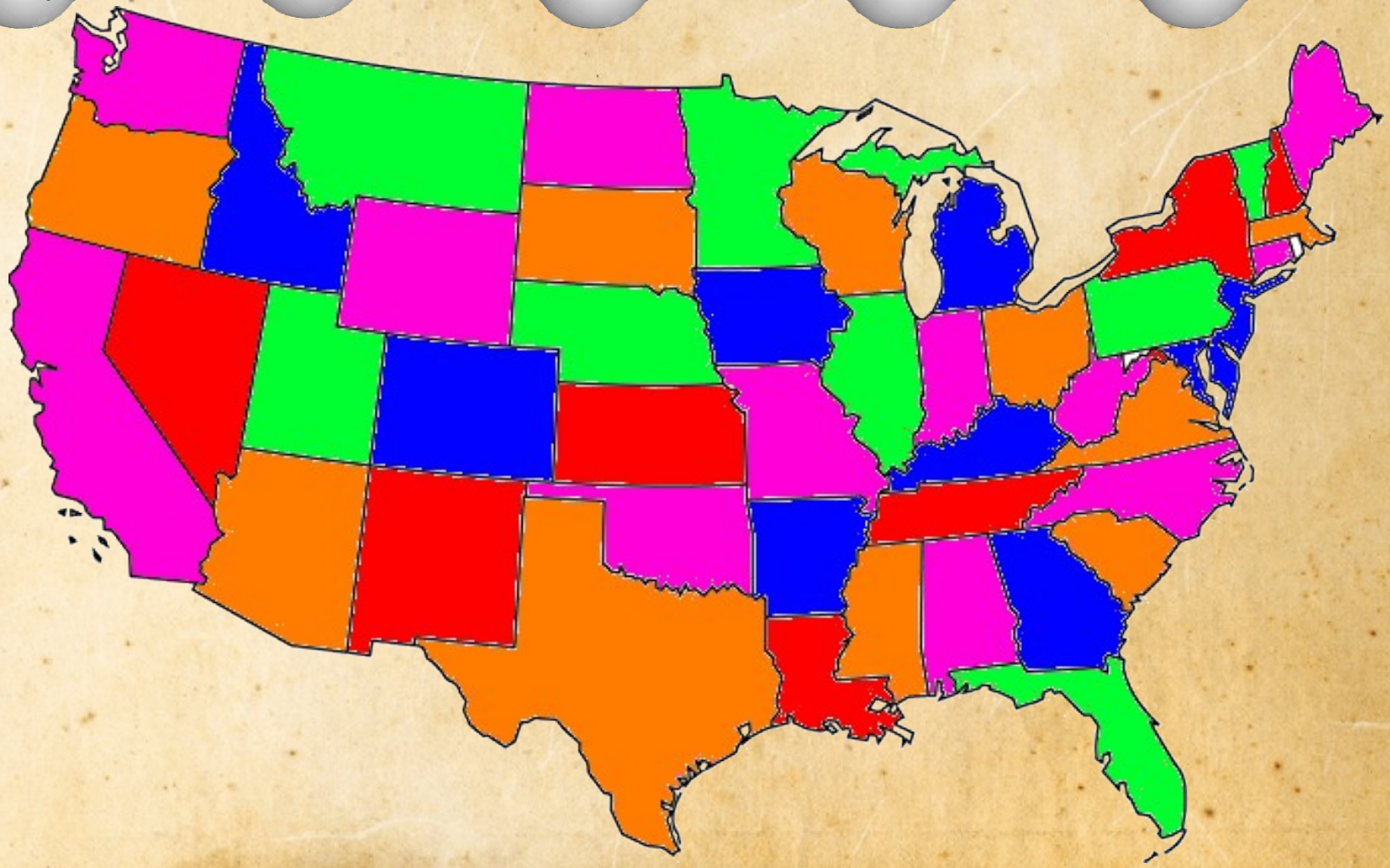
- Herhalend (stapsgewijs)



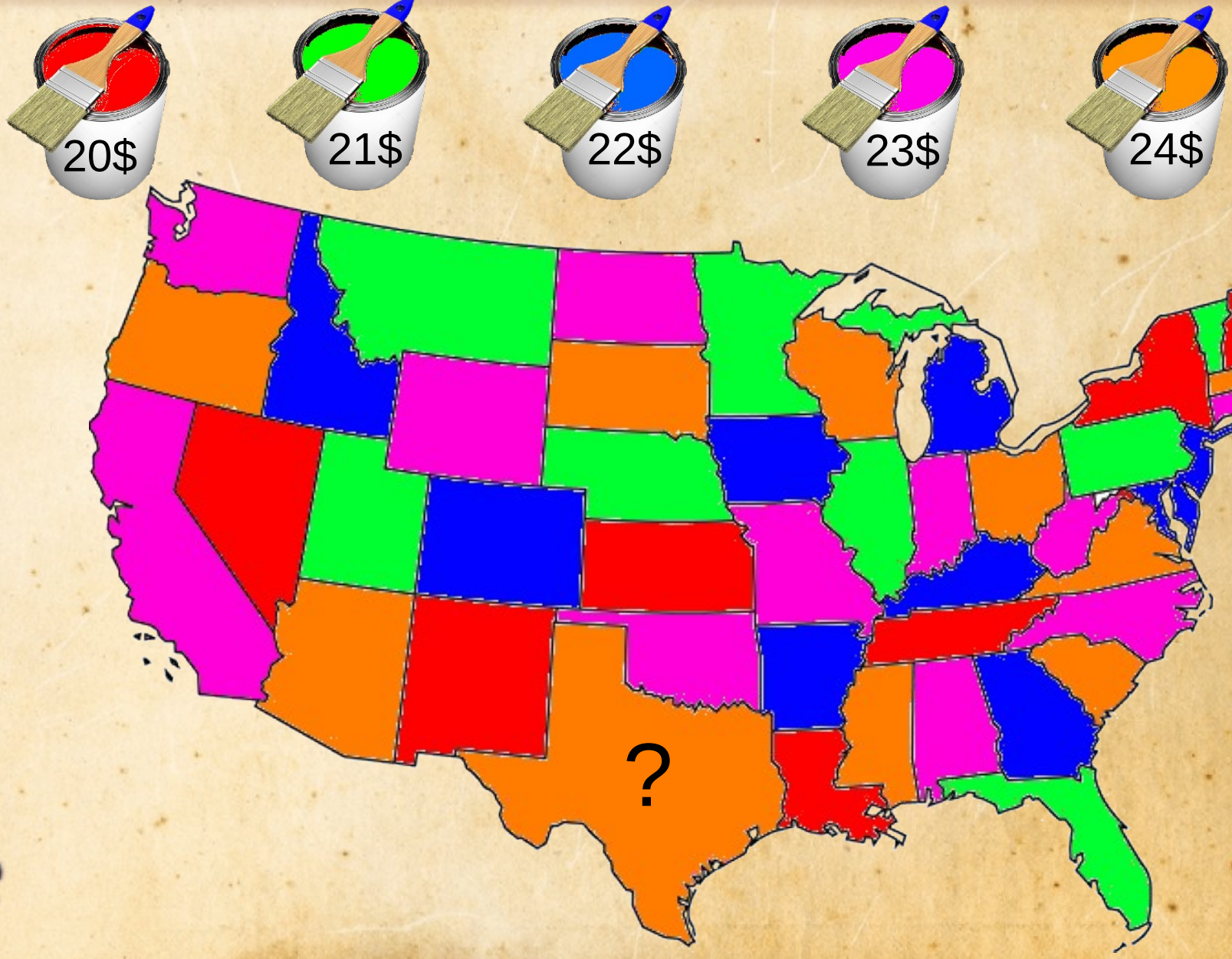
Hill Climber, pseudo code

```
Kies een random start state
Herhaal:
    Doe een kleine random aanpassing
    Als de state is verslechterd:
        Maak de aanpassing ongedaan
```

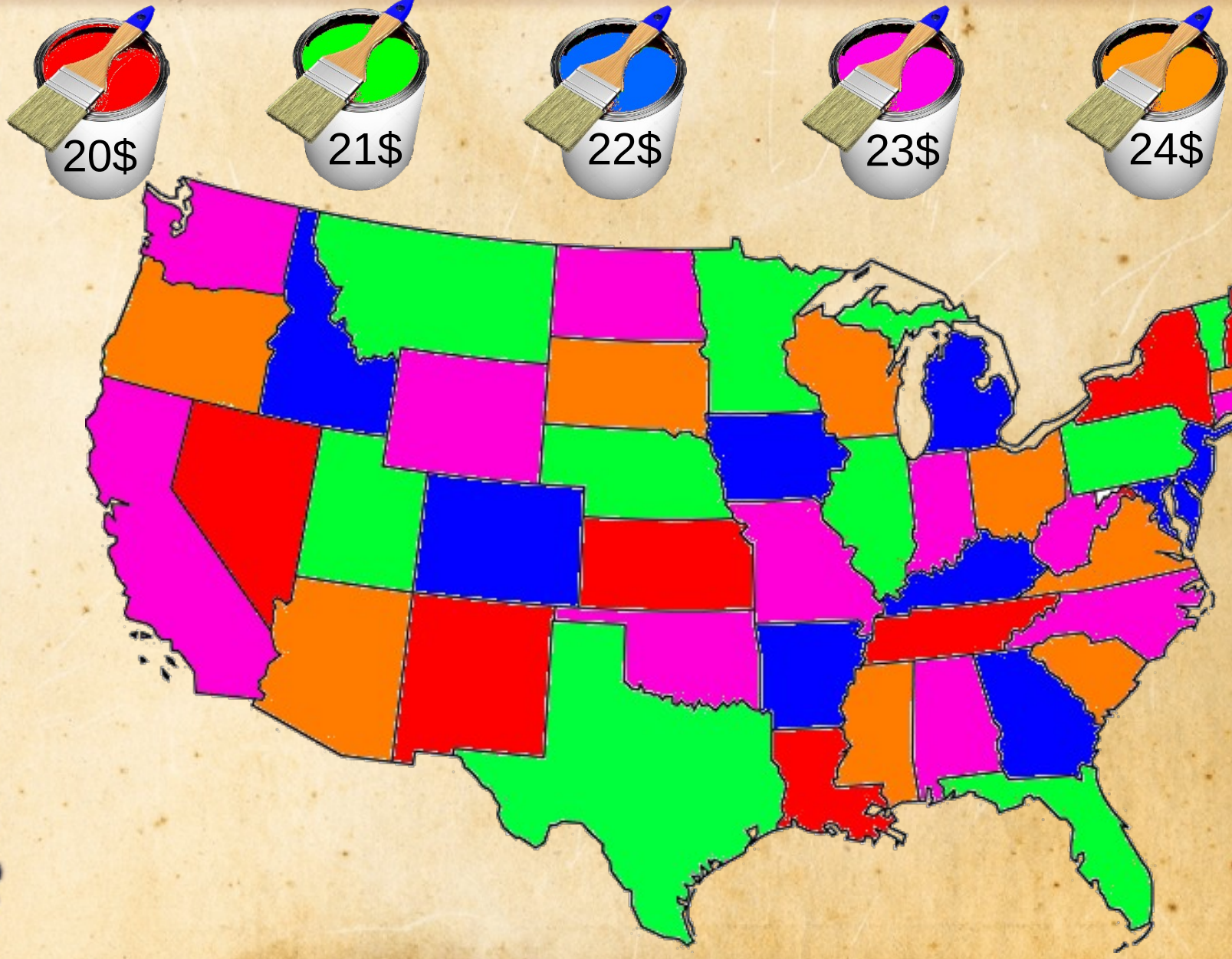
Hill Climber



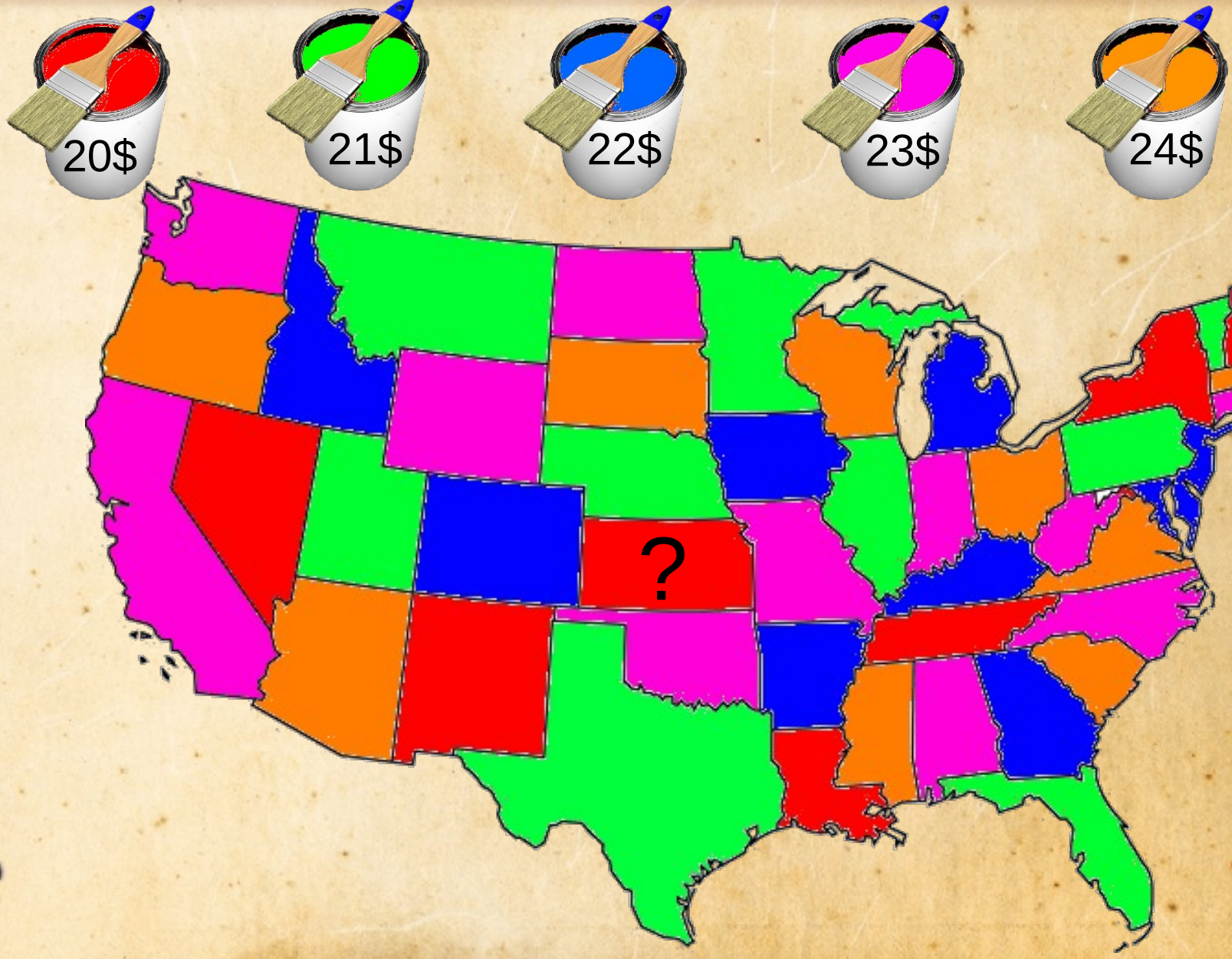
Hill Climber



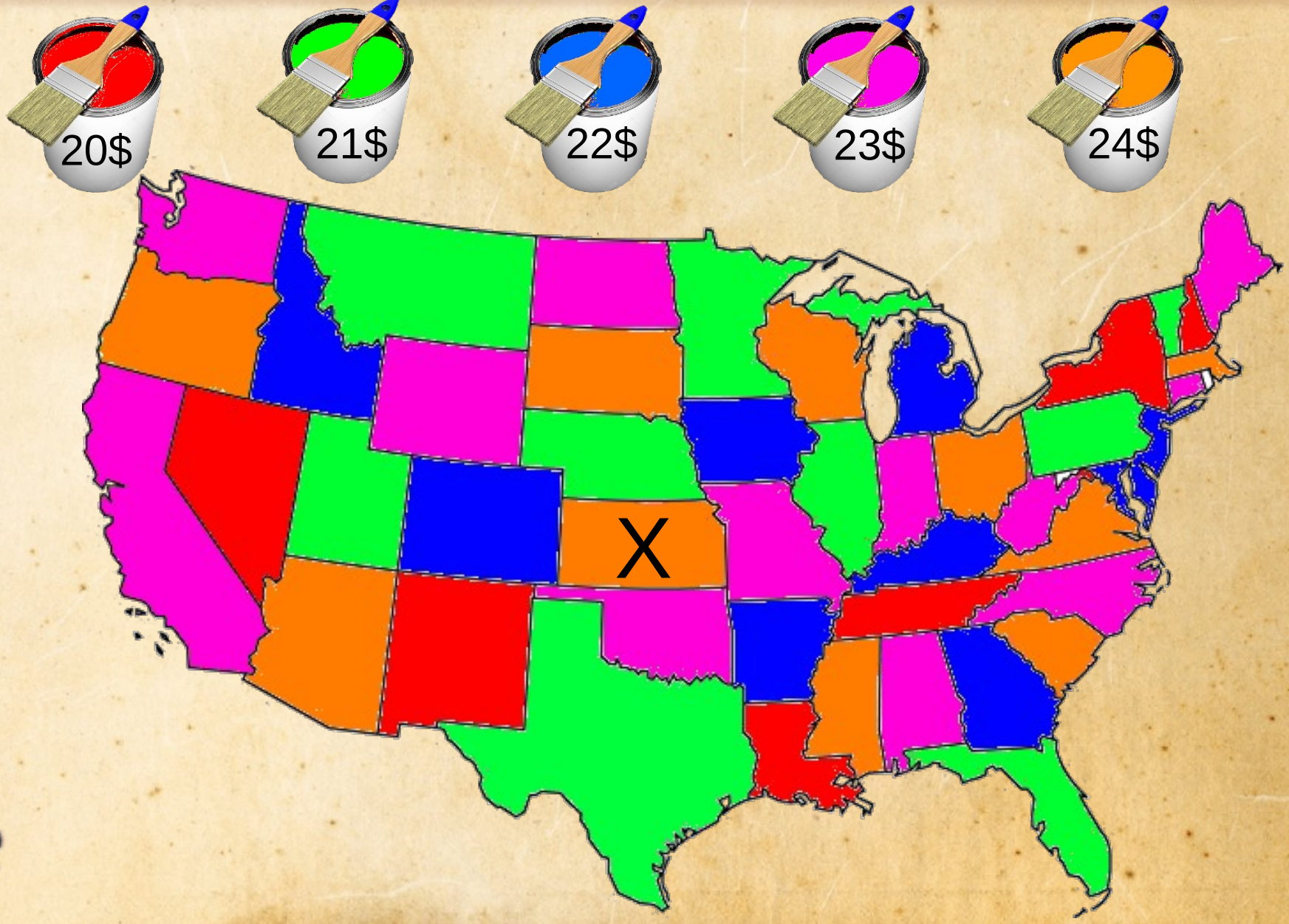
Hill Climber



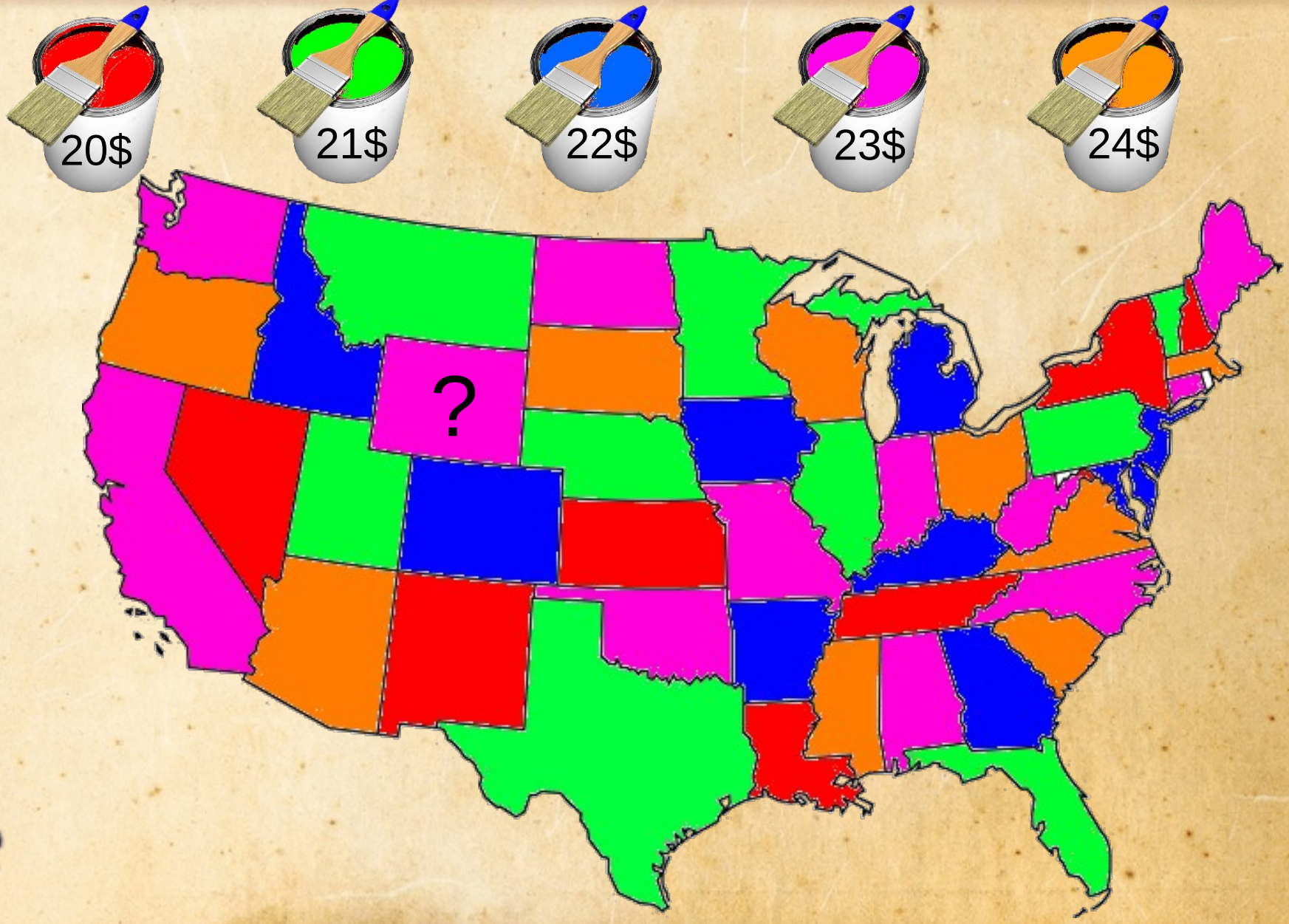
Hill Climber



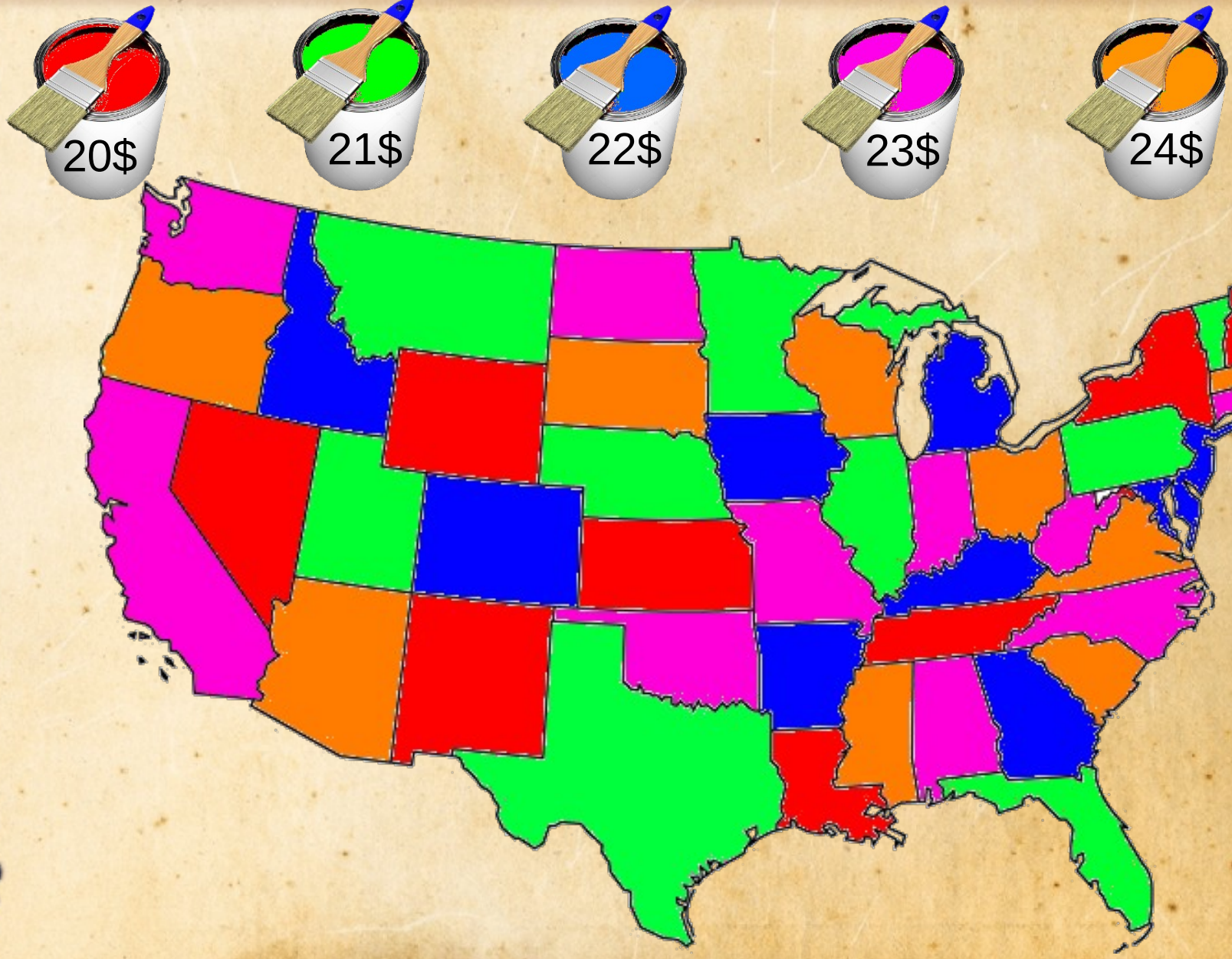
Hill Climber



Hill Climber



Hill Climber

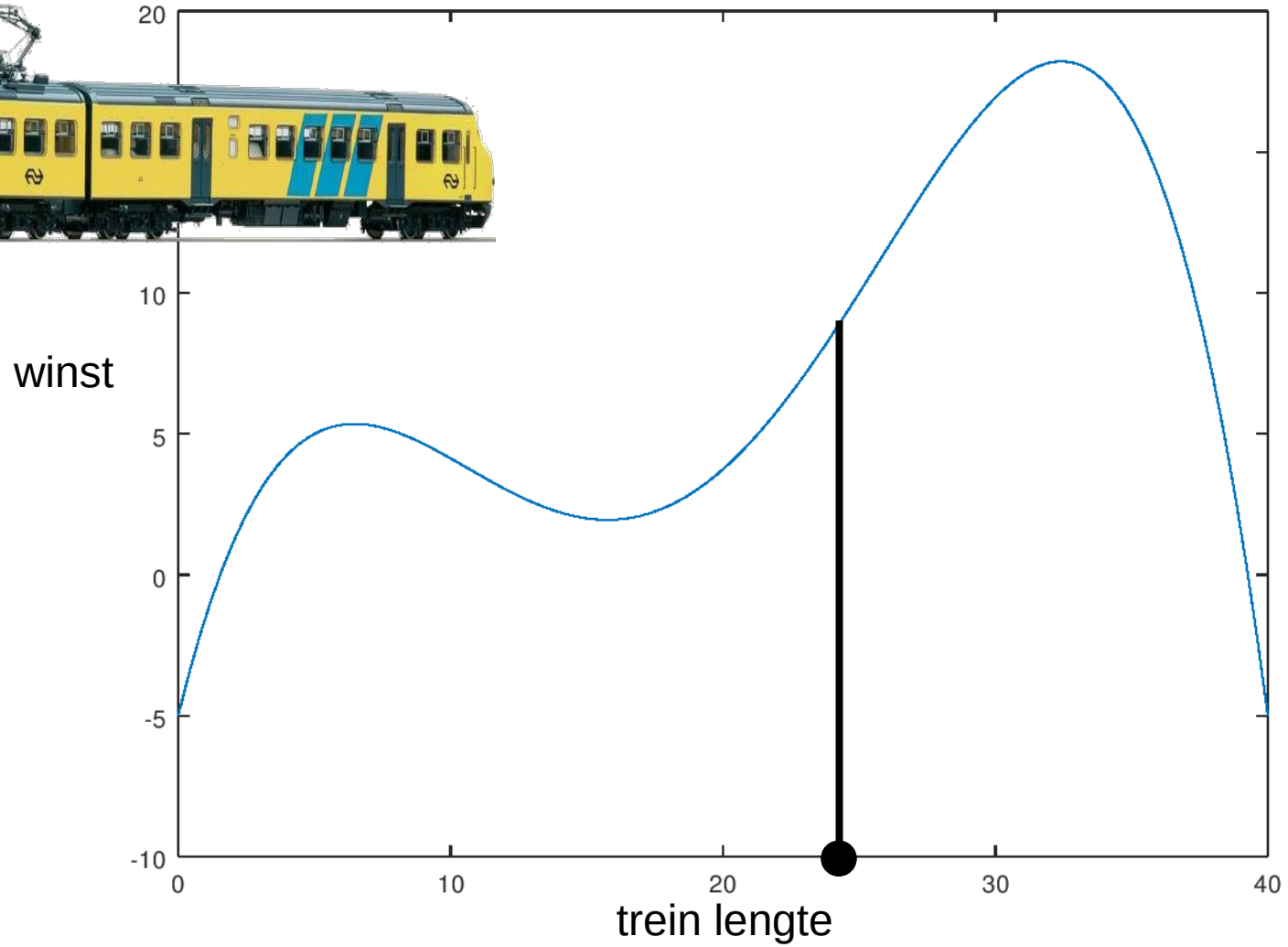


Hill Climber

- 1 dimensionaal trein probleem

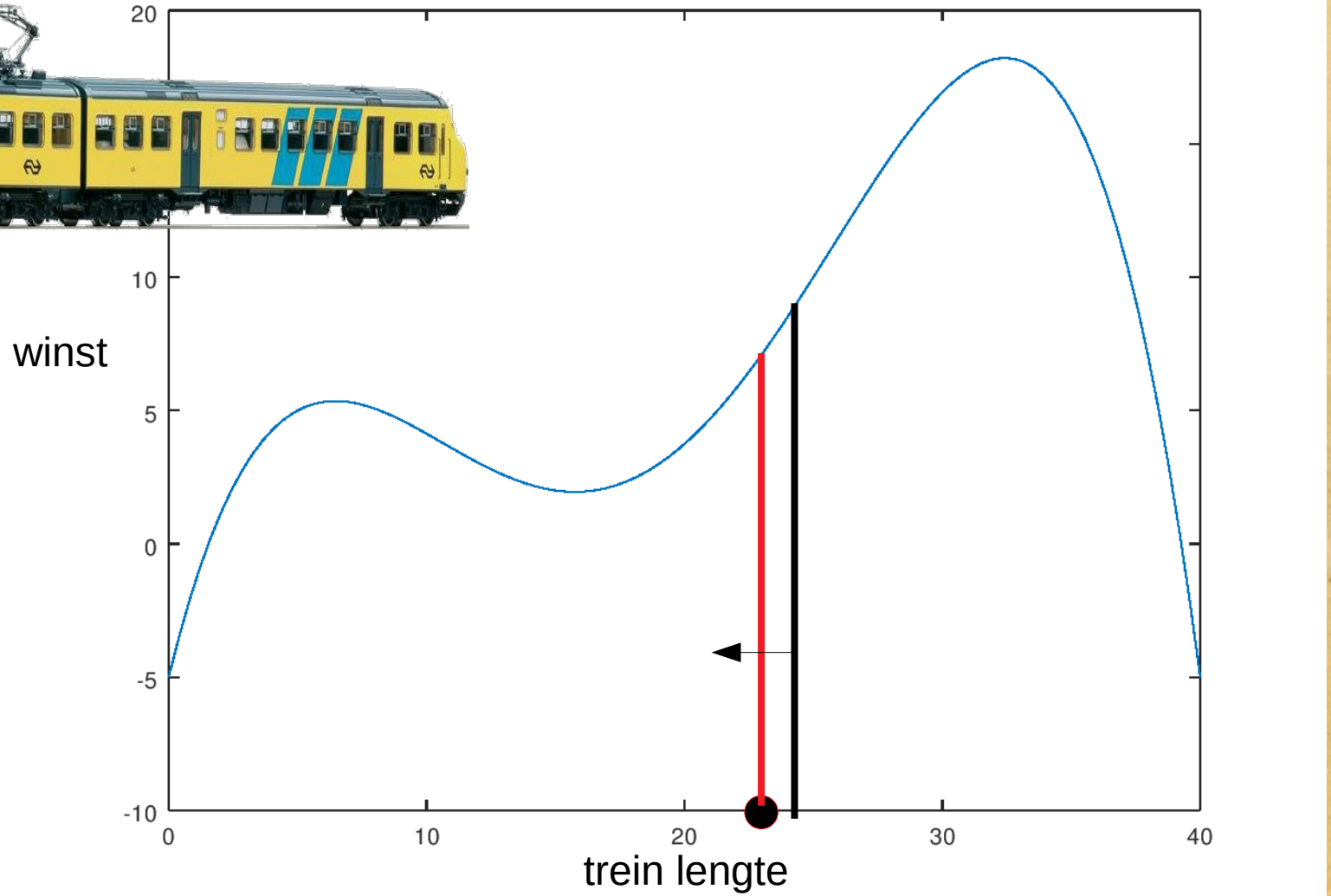
Hill Climber

- 1 dimensionaal trein probleem



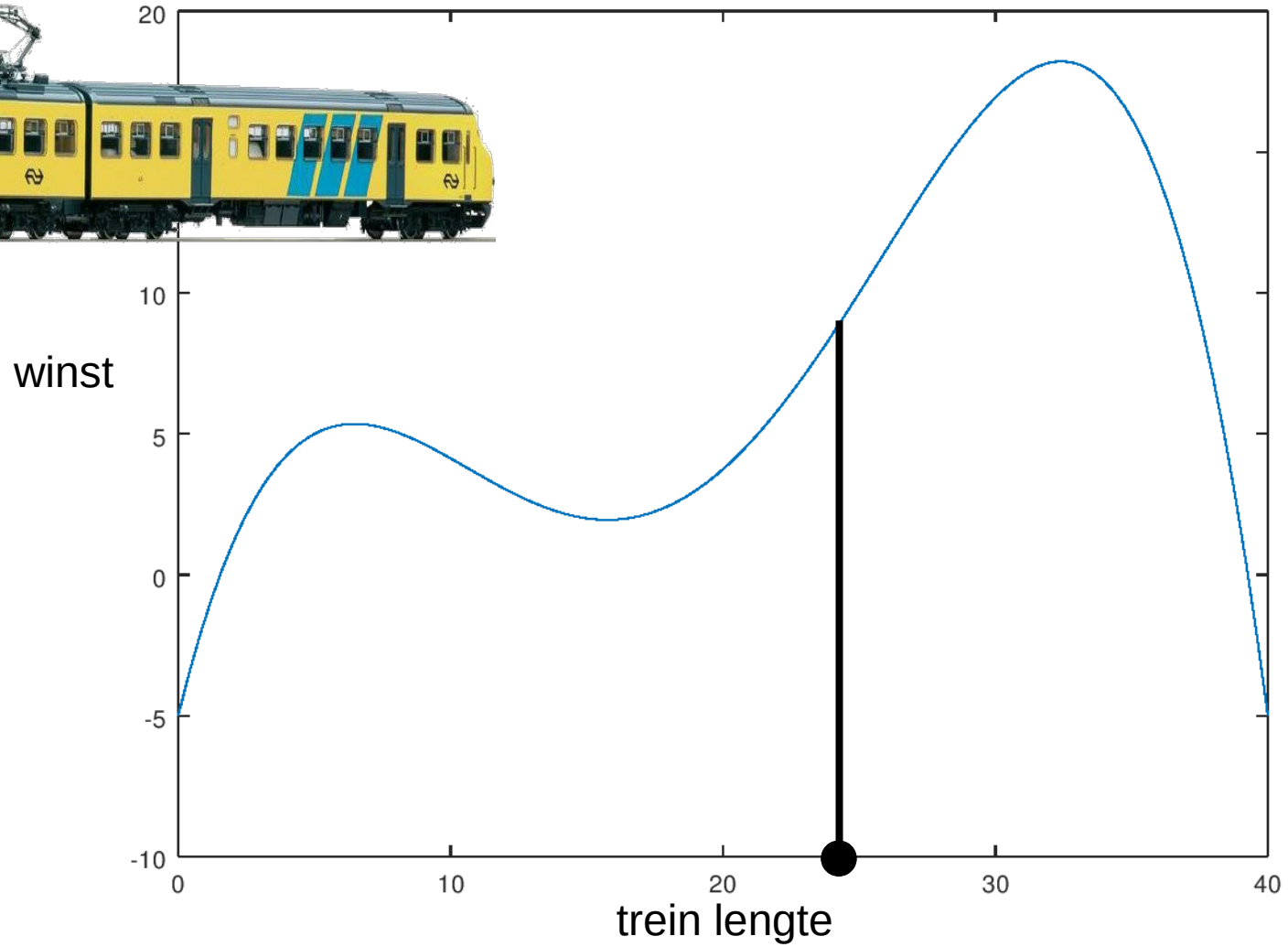
Hill Climber

- 1 dimensionaal trein probleem



Hill Climber

- 1 dimensionaal trein probleem

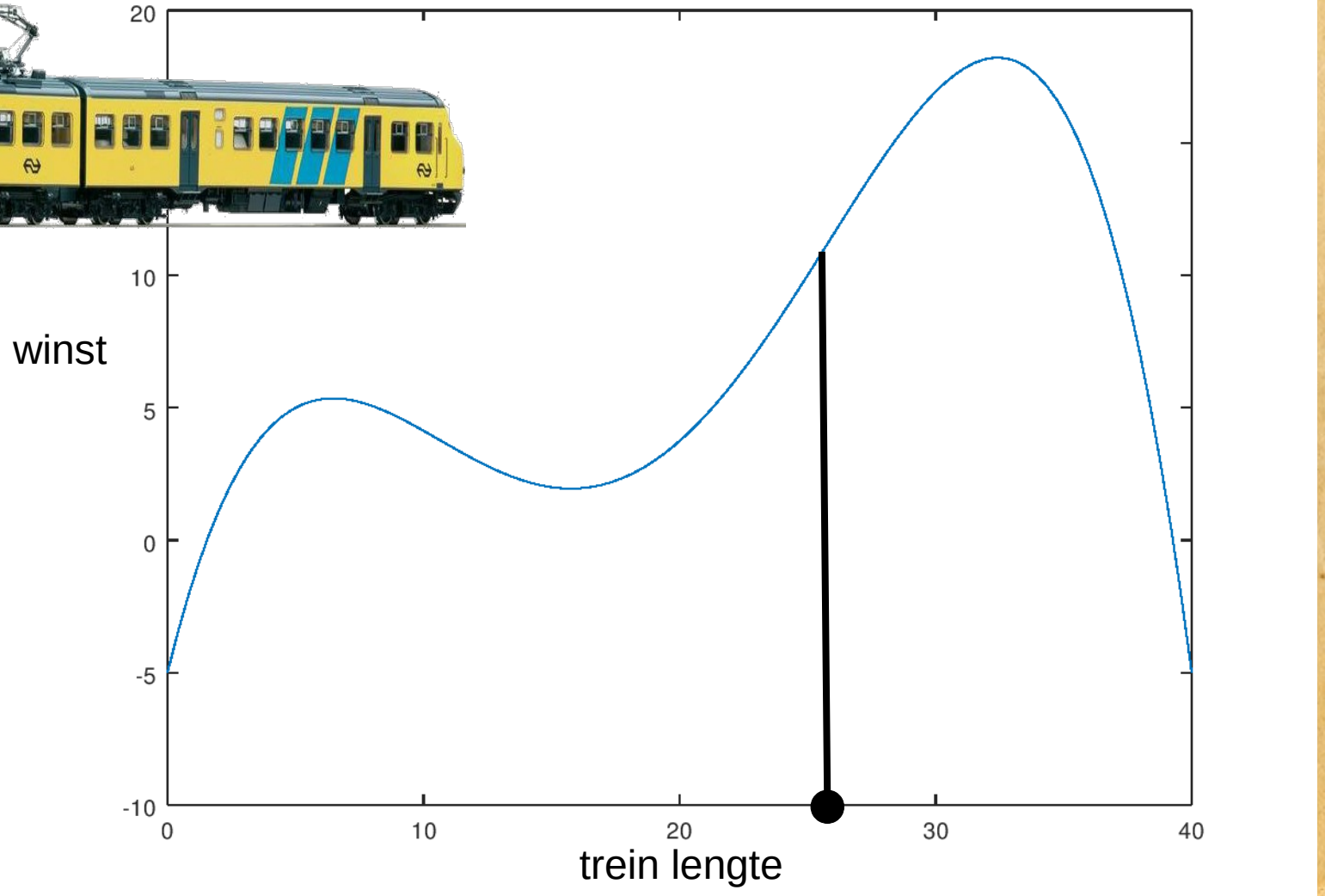


Hill Climber

- 1 dimensionaal trein probleem

Hill Climber

- 1 dimensionaal trein probleem



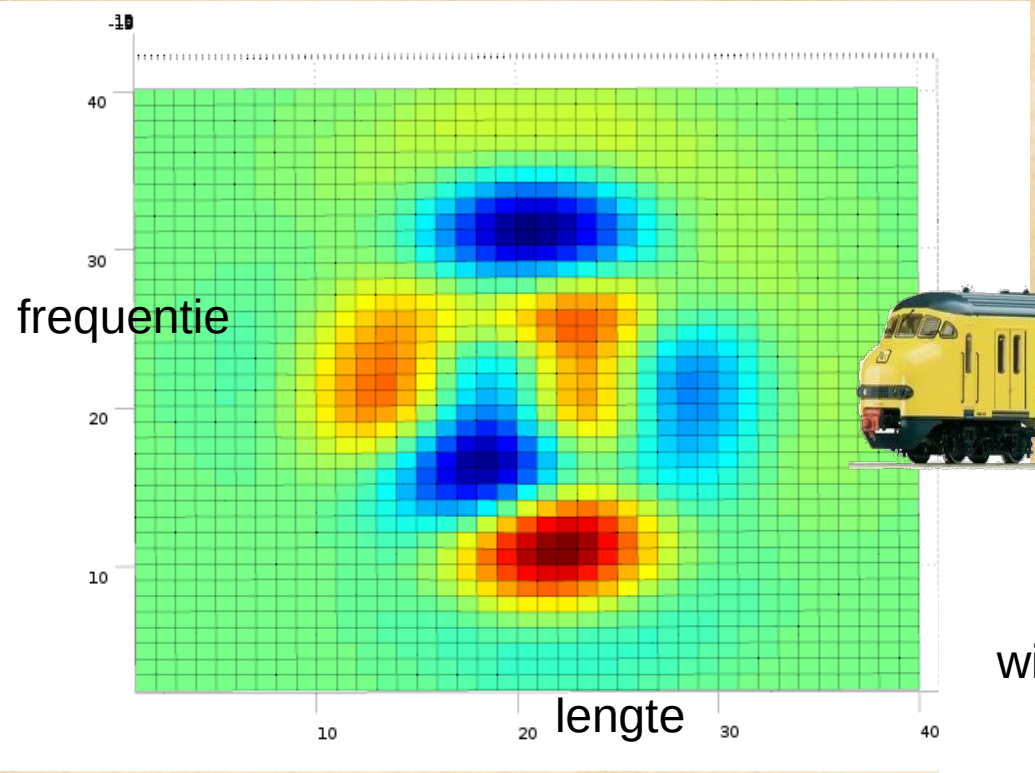
Hill Climber

- 1 dimensionaal trein probleem

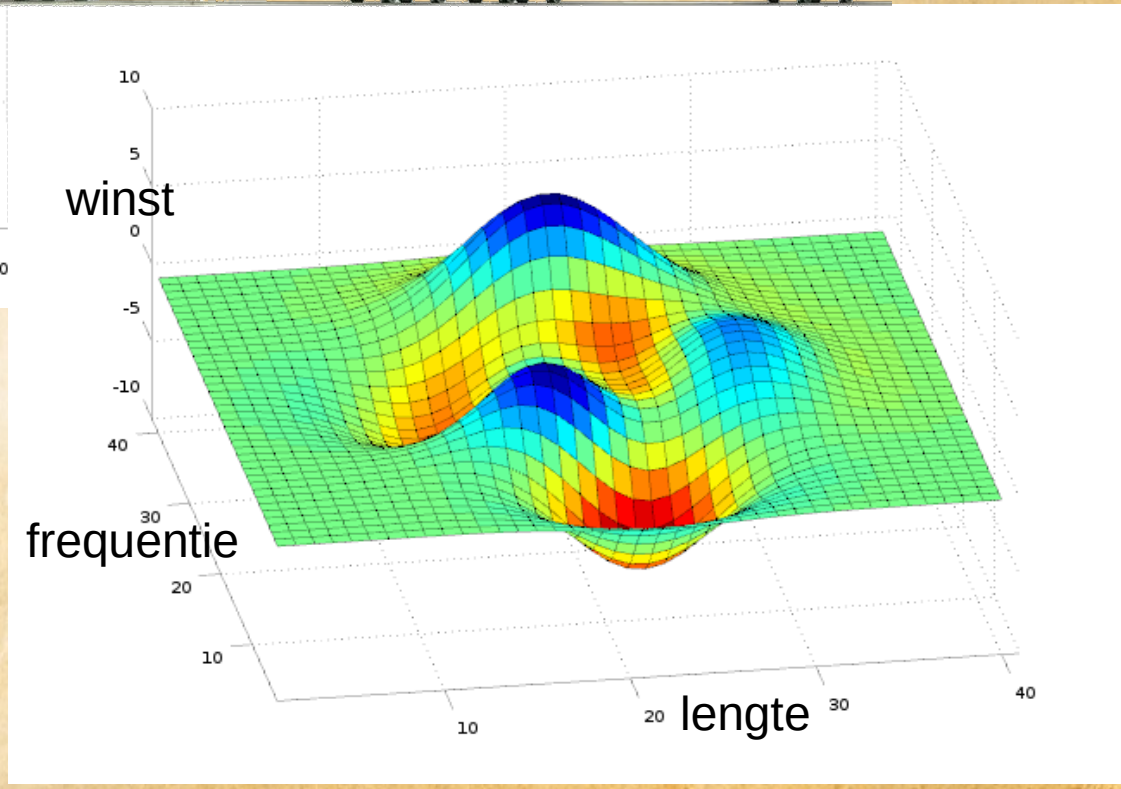
Hill Climber

- 1 dimensionaal trein probleem

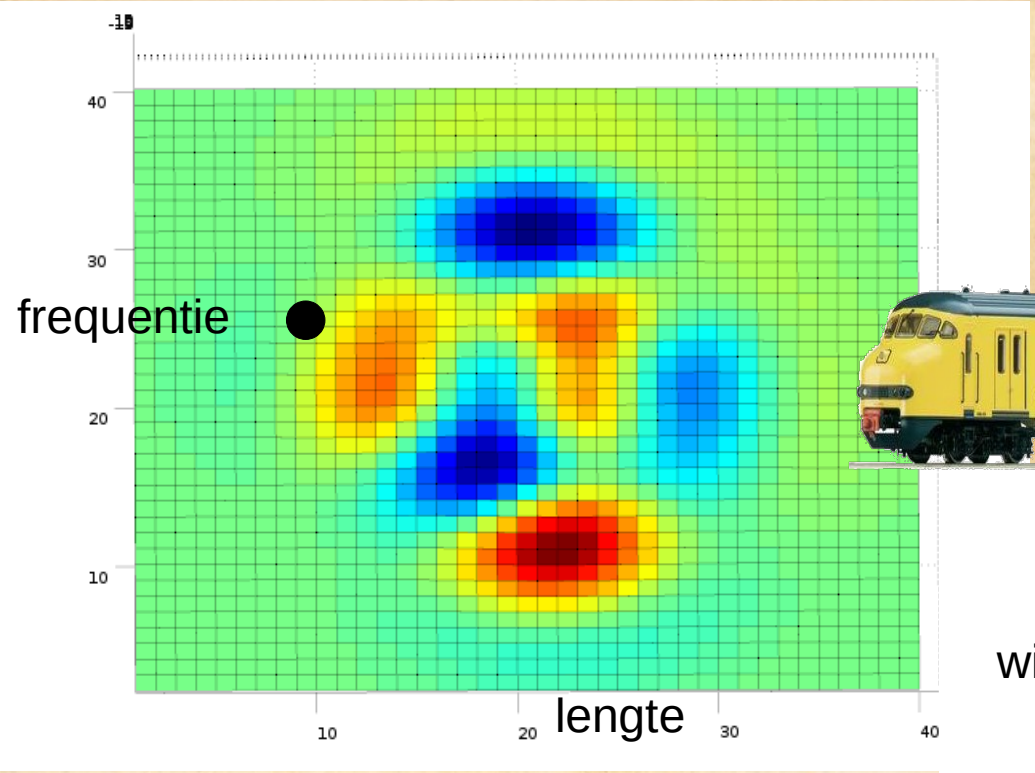
Hill Climber



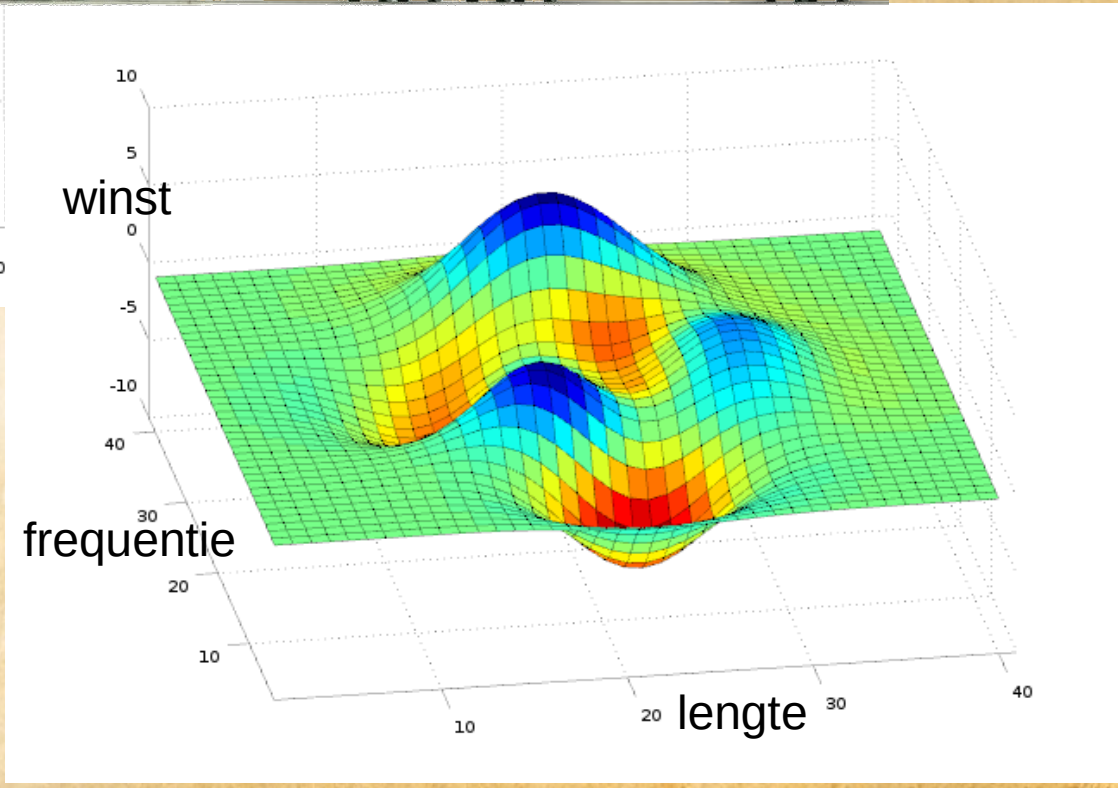
- 2 dimensionaal trein probleem



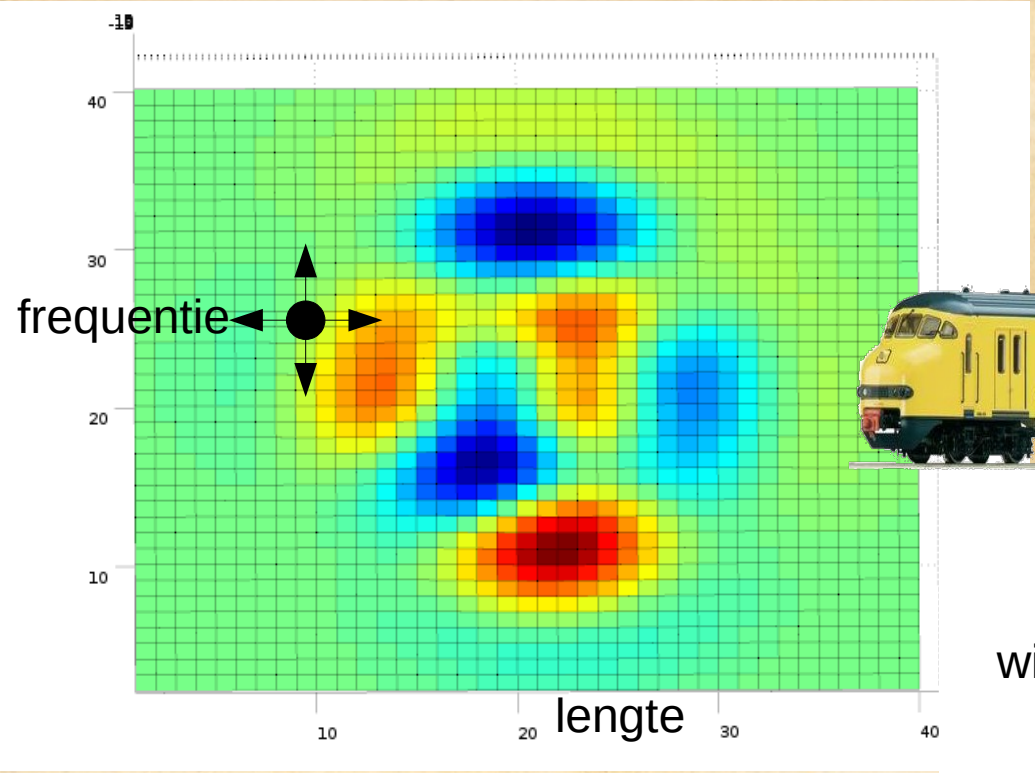
Hill Climber



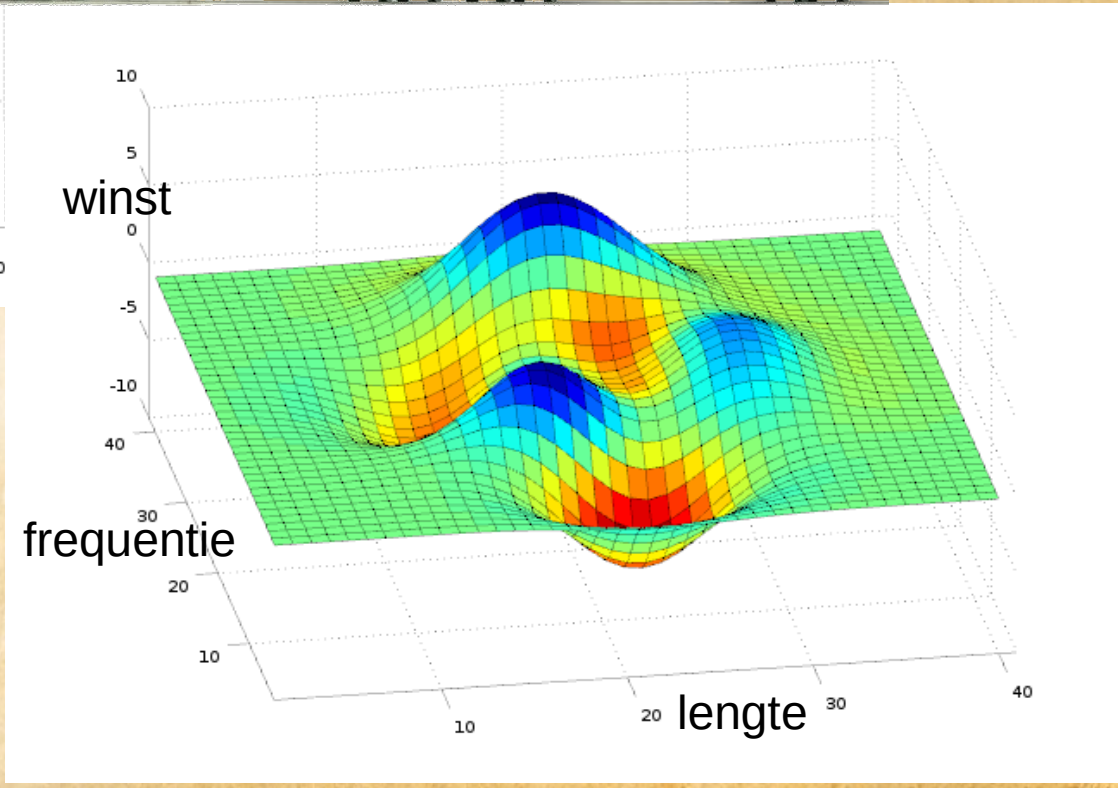
- 2 dimensionaal trein probleem



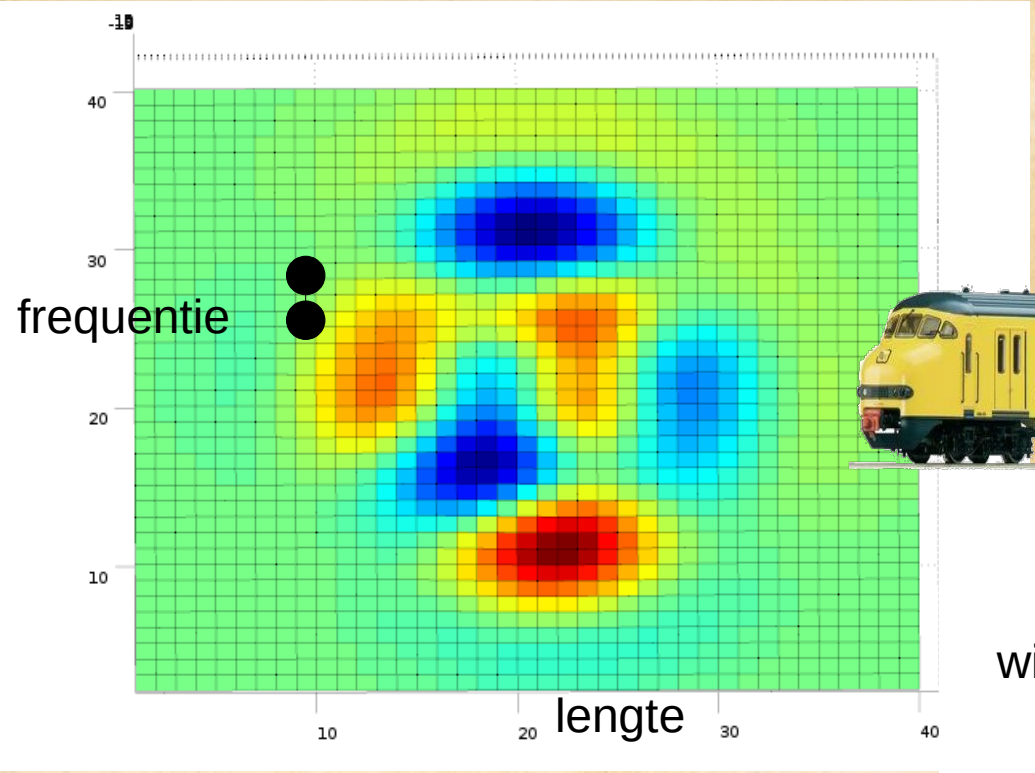
Hill Climber



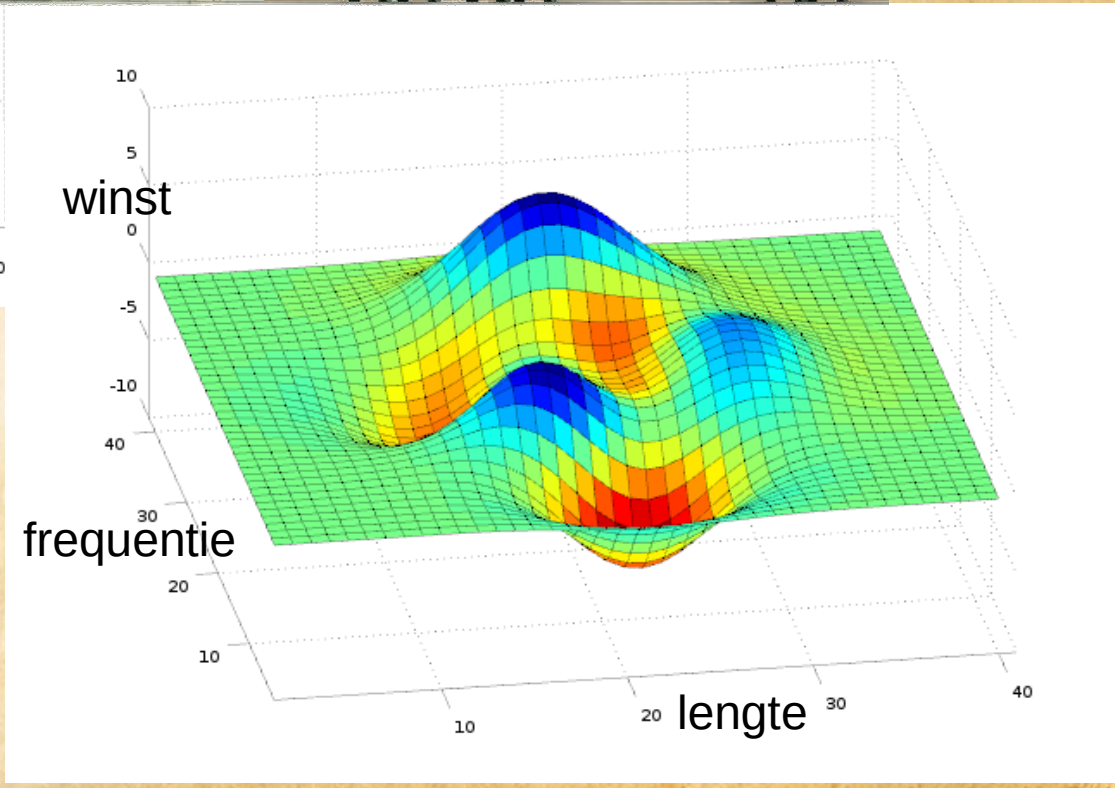
- 2 dimensionaal trein probleem



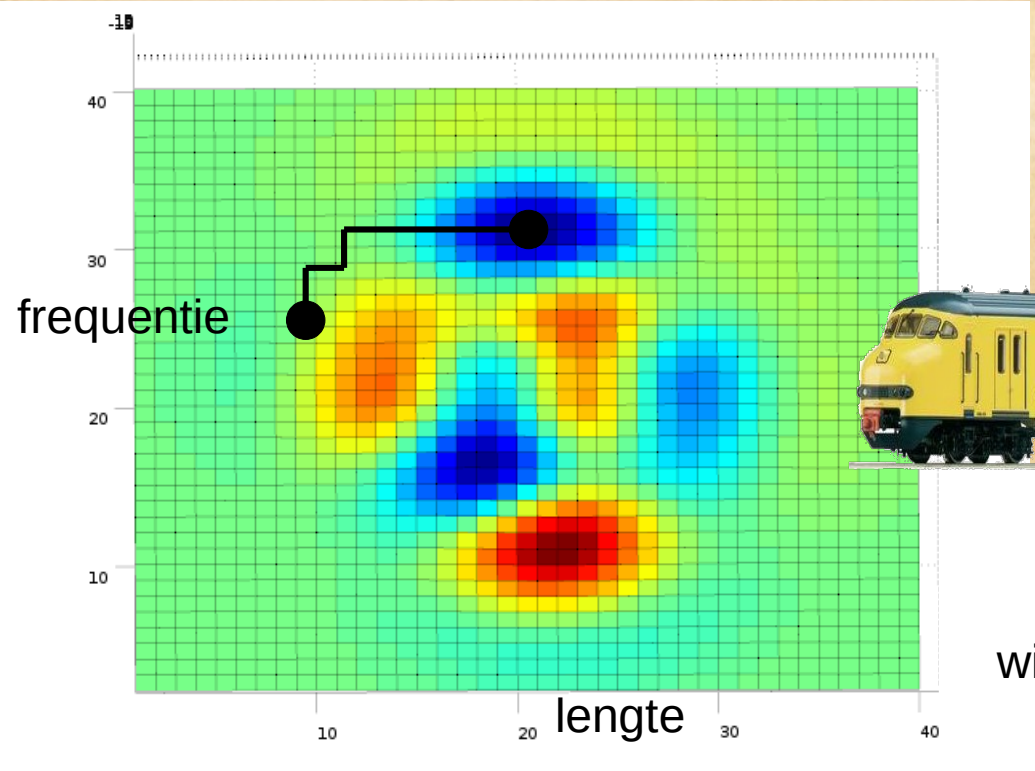
Hill Climber



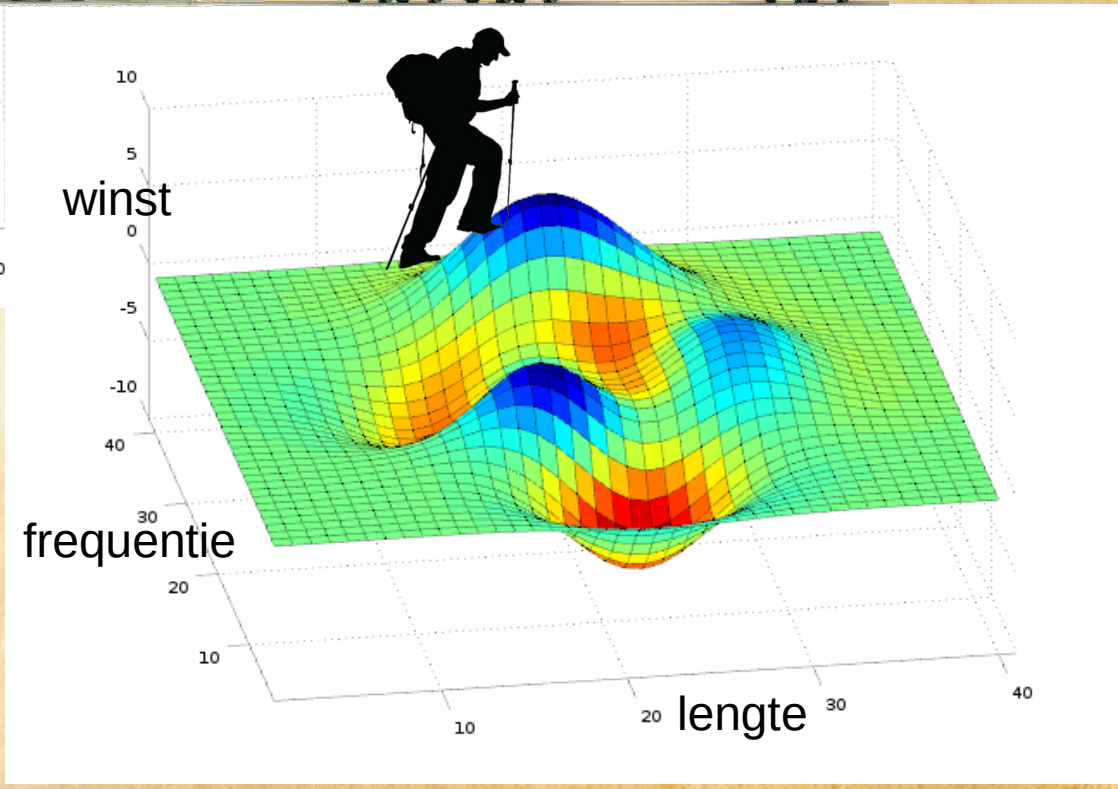
- 2 dimensionaal trein probleem



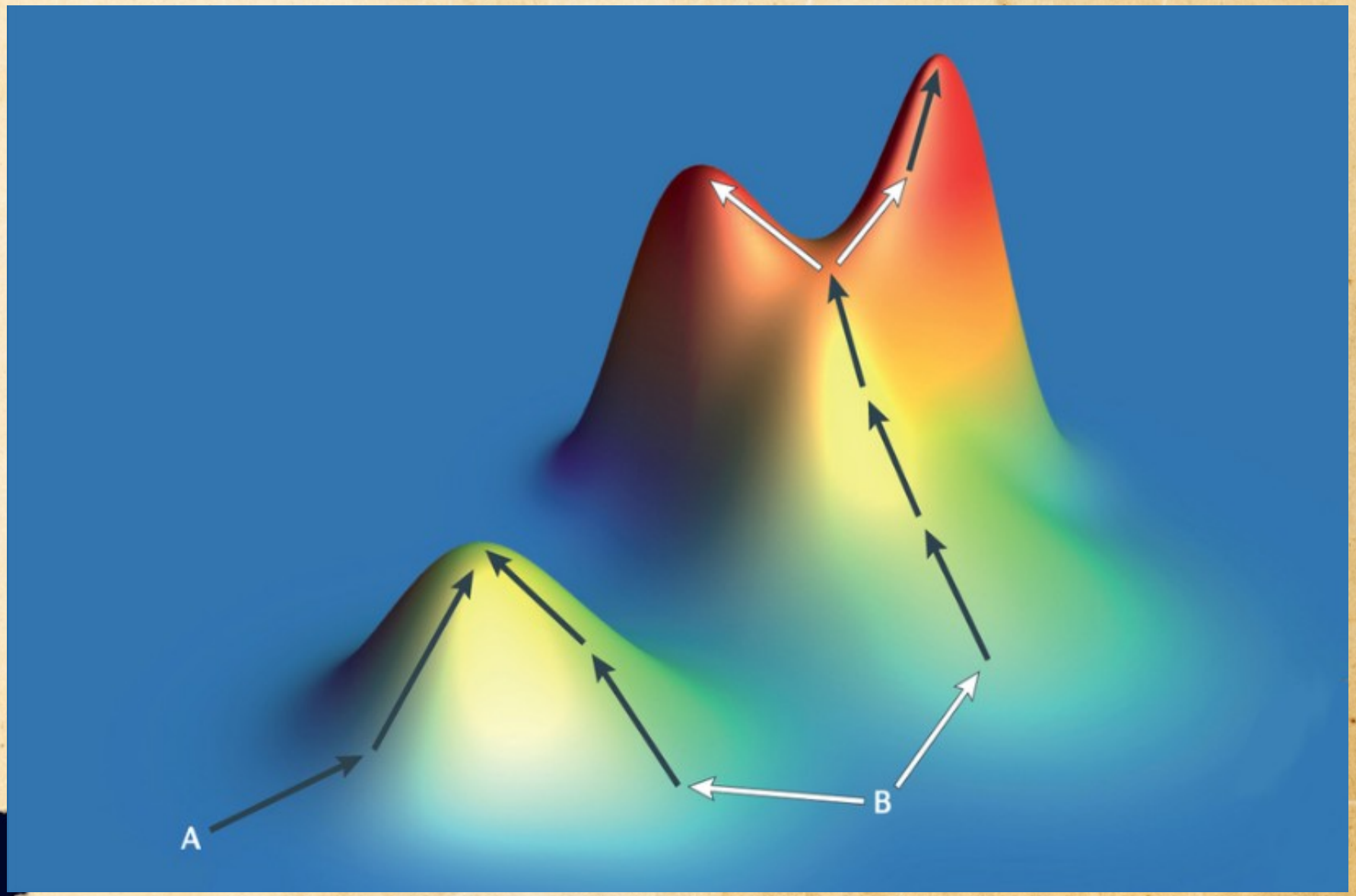
Hill Climber



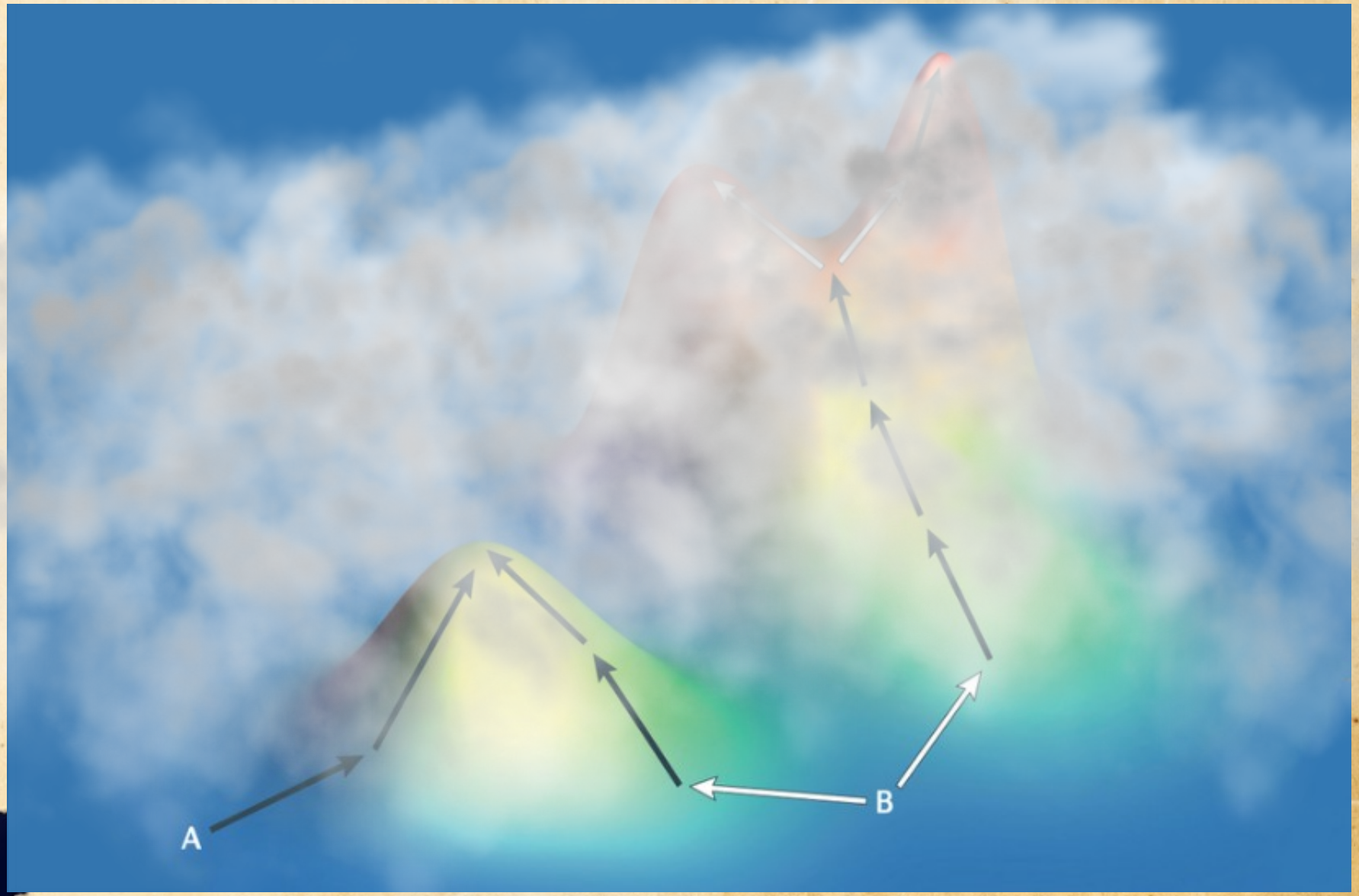
- 2 dimensionaal trein probleem



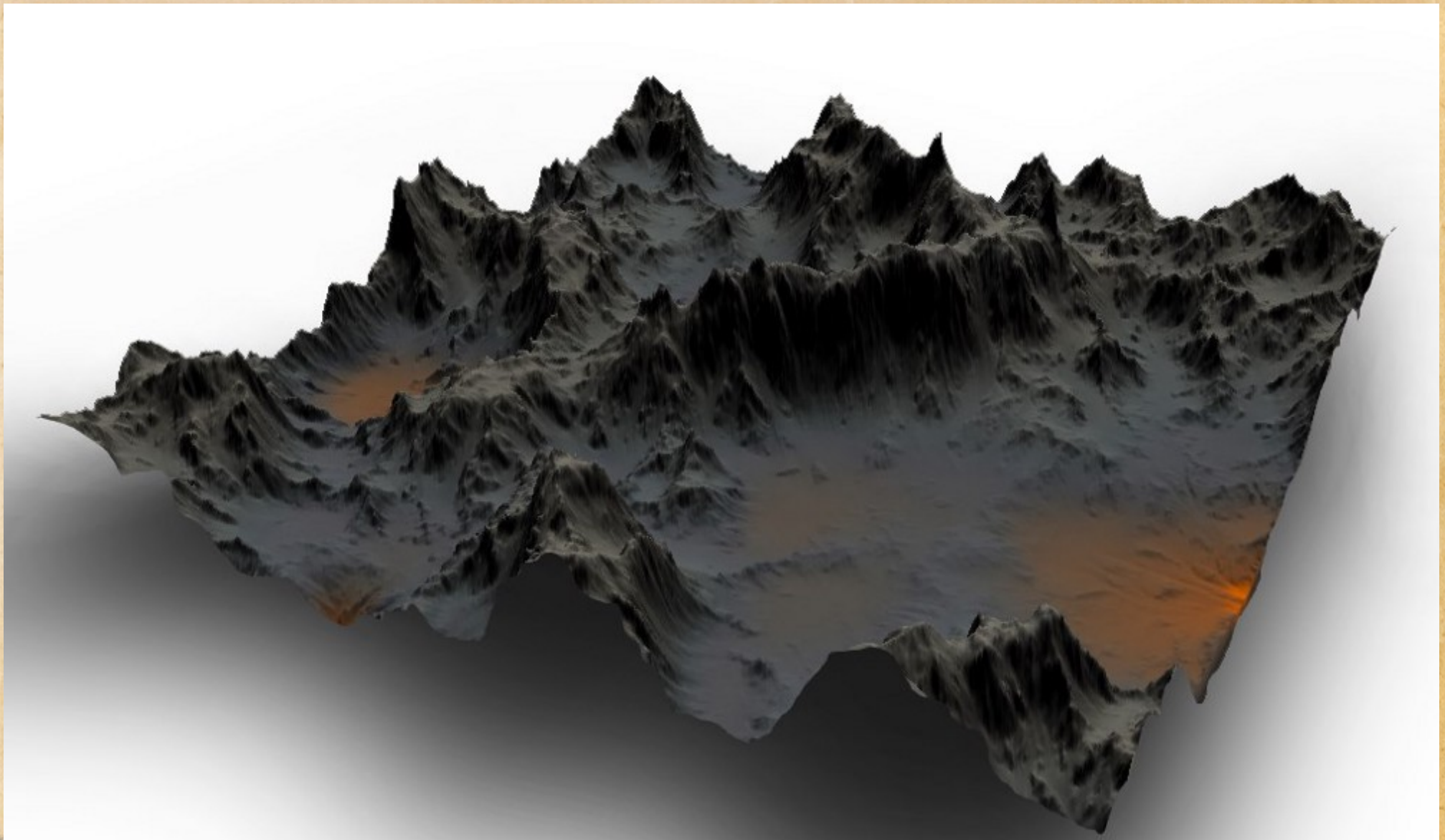
Hill Climber



Hill Climber

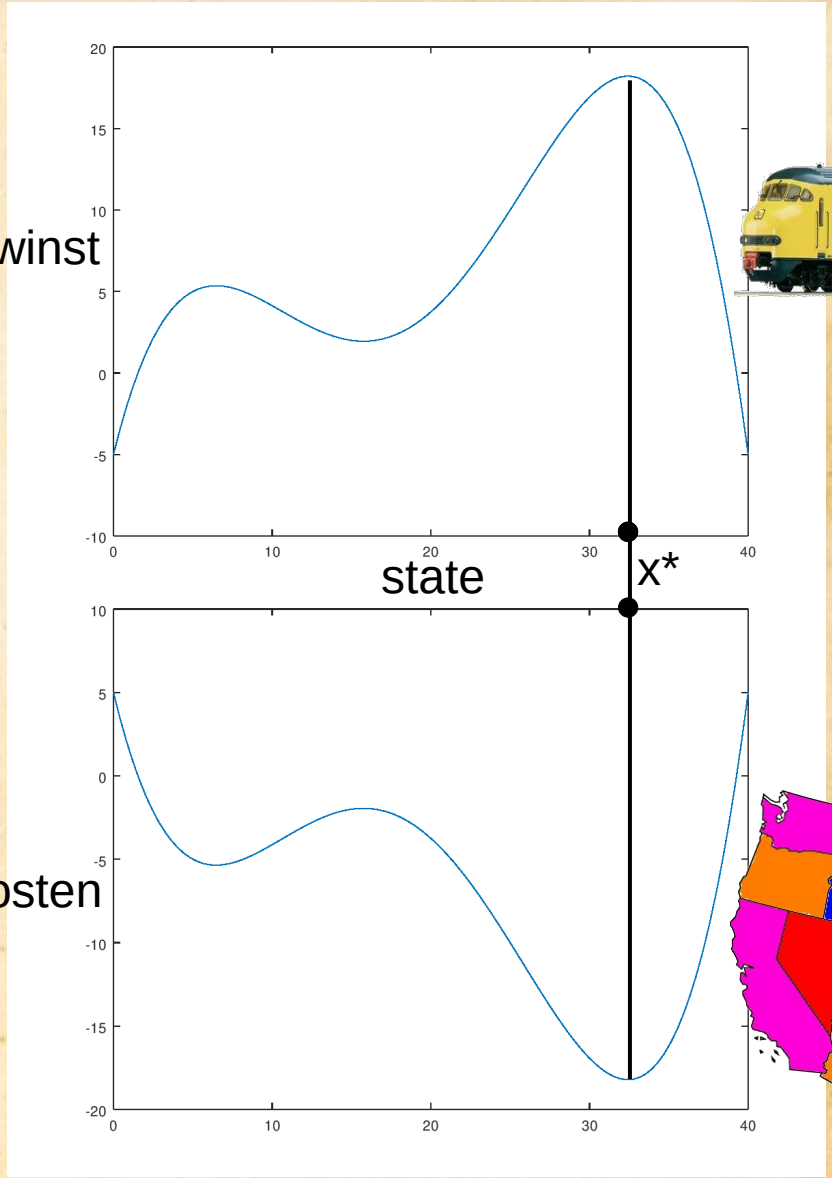


Hill Climber



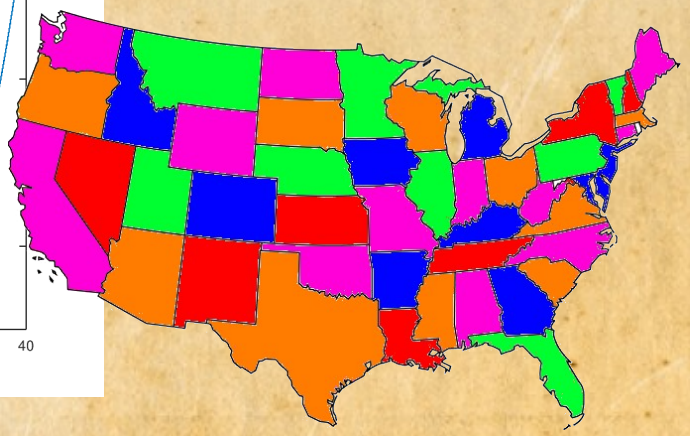
Groot, complex en hoog-dimensionaal landschap

Maximaliseren of Minimaliseren

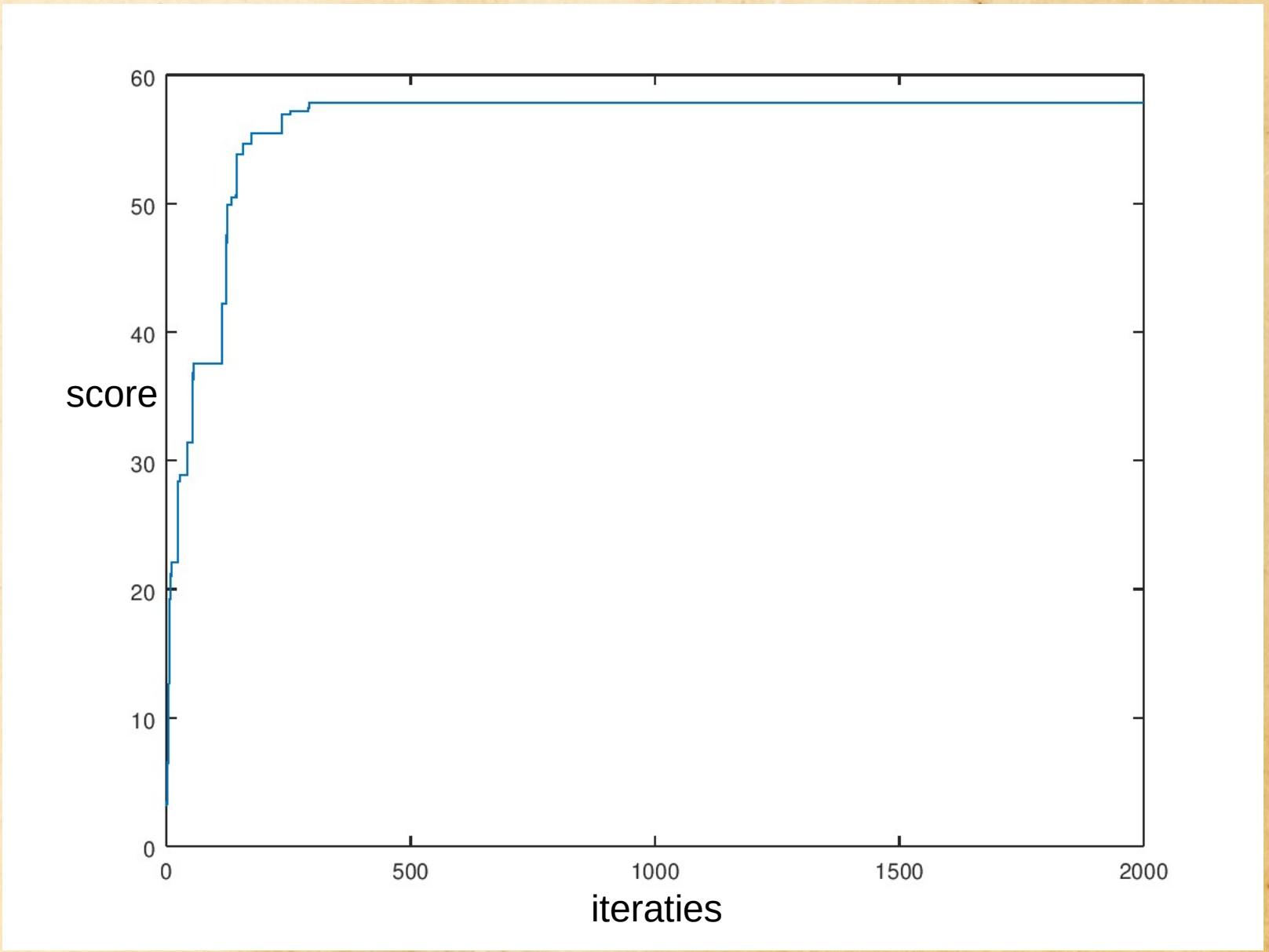


$$x^* = \arg\max_x f(x)$$

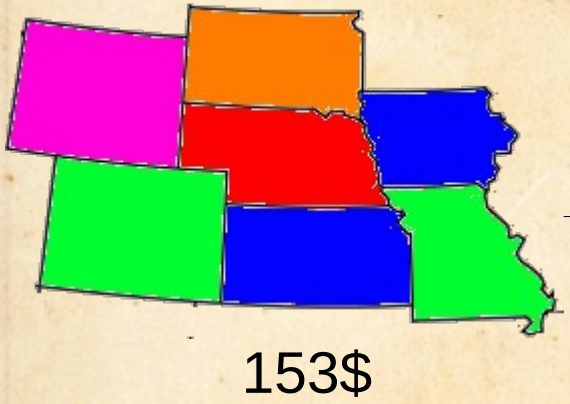
$$x^* = \arg\min_x -f(x)$$



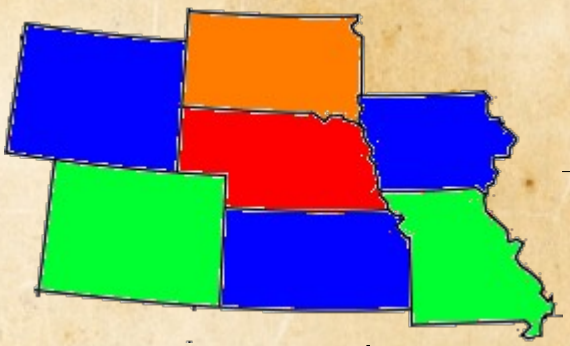
Convergentie



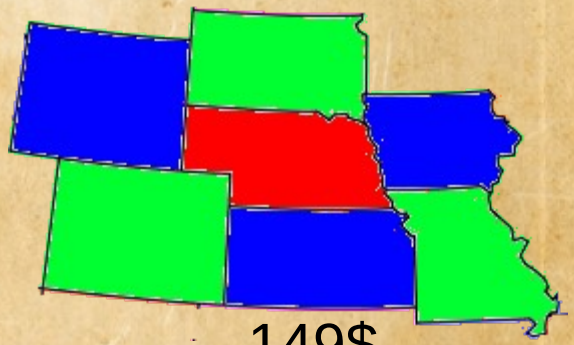
Convergentie



153$



152$

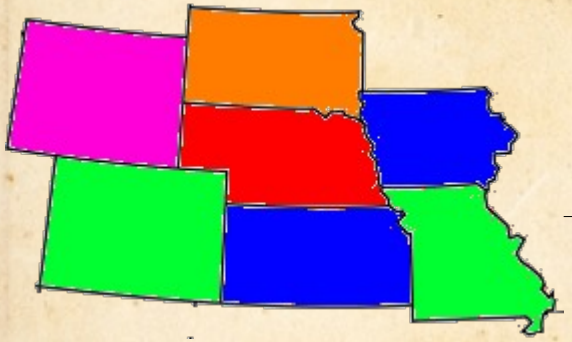


149$

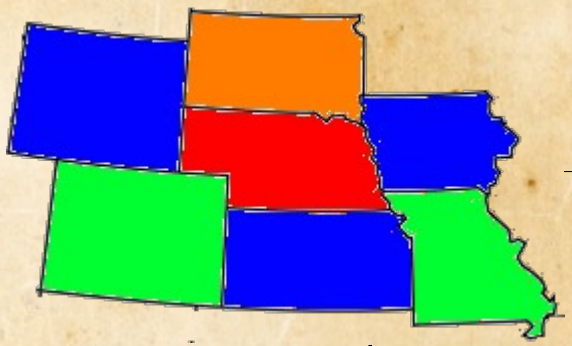
Lokaal optimum?

Globaal optimum?

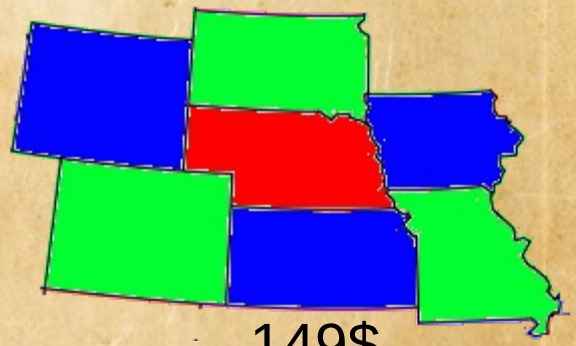
Convergentie



153$



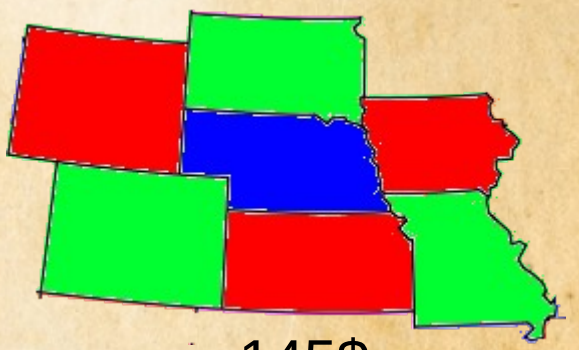
152$



149$

Lokaal optimum? ✓

Globaal optimum? ✗



145$

Restart Hill Climber, pseudo code

Kies een random start state

Herhaal **tot na N-keer niet meer verbetert**:

    Doe een kleine random aanpassing

    Als de state is verslechterd:

        Maak de aanpassing ongedaan

Restart Hill Climber, pseudo code

Herhaal:

Kies een random start state

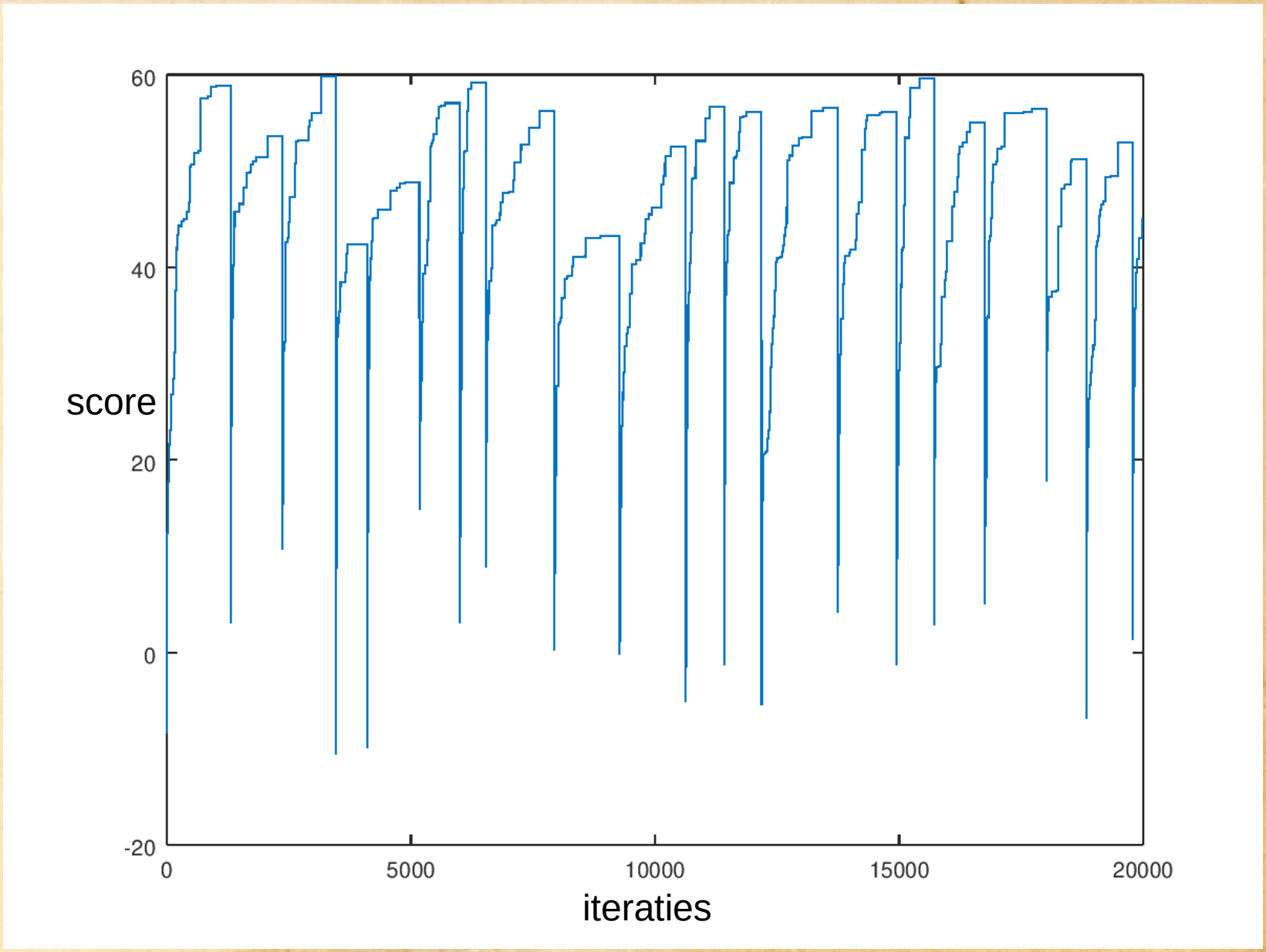
Herhaal **tot na N-keer niet meer verbetert**:

Doe een kleine random aanpassing

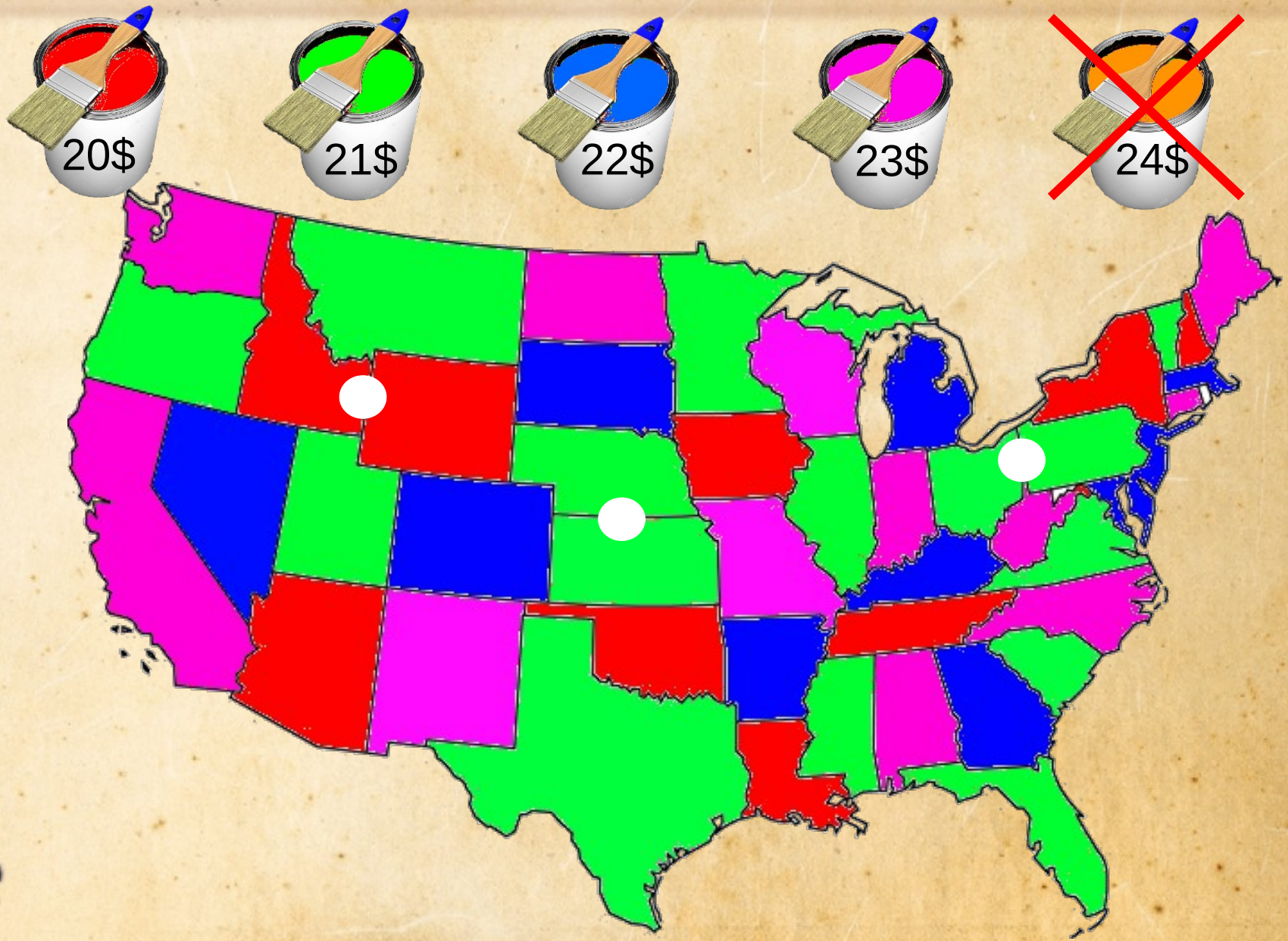
Als de state is verslechterd:

Maak de aanpassing ongedaan

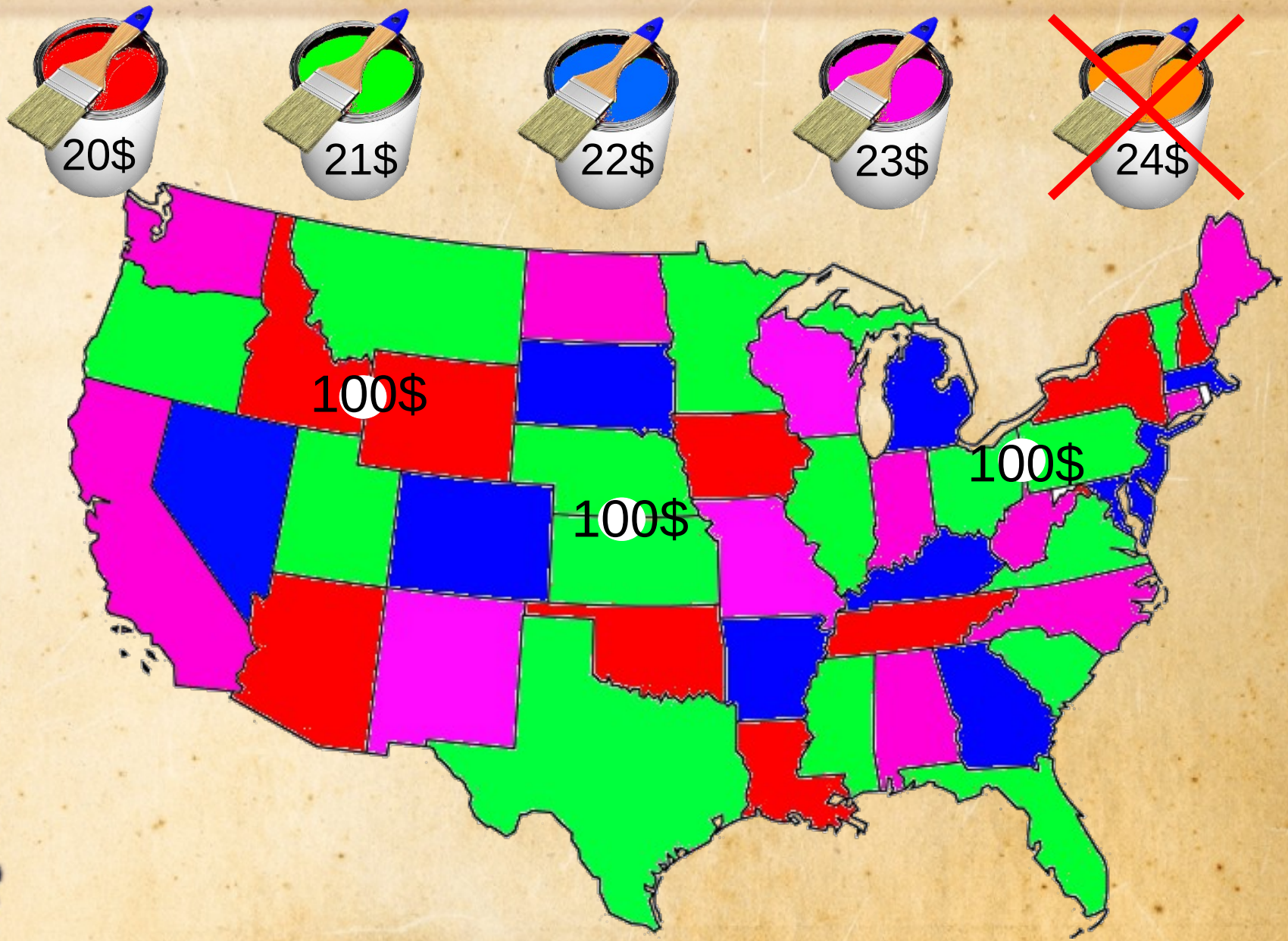
Restart



Constraint relaxation



Constraint relaxation



Constraint relaxation, pseudo code

Herhaal:

    Kies een random start state **zonder constraint checks**

    Herhaal totdat N-keer niet meer verbetert:

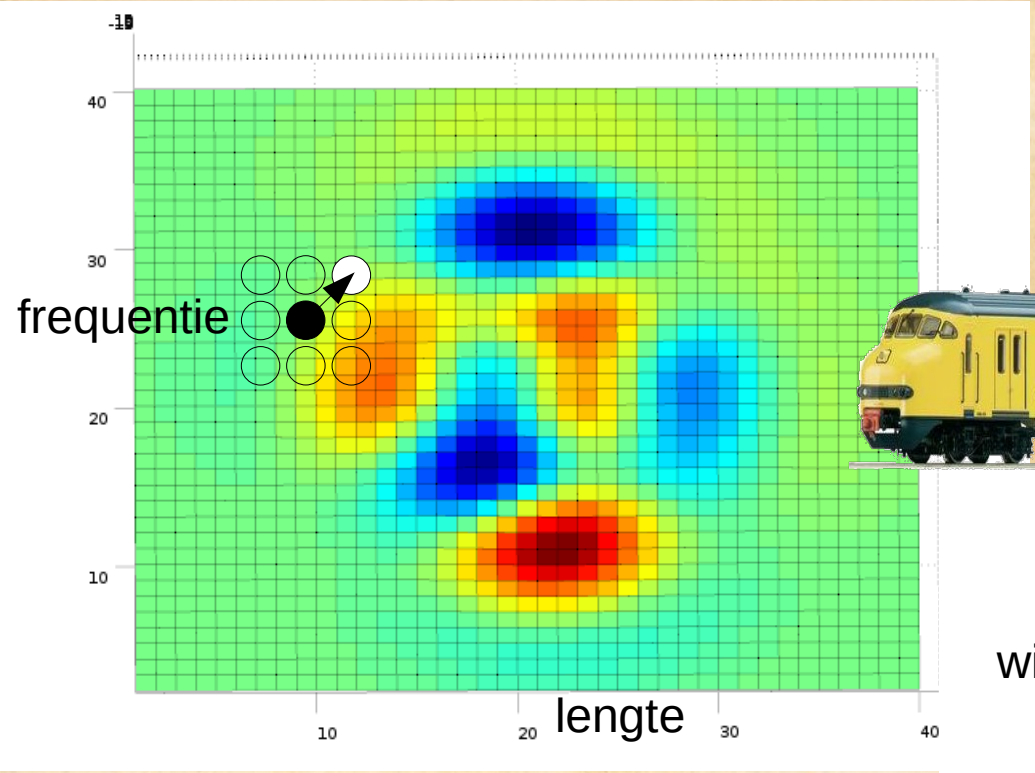
        Doe een kleine random aanpassing

        Geef strafpunten voor constraint-schendingen

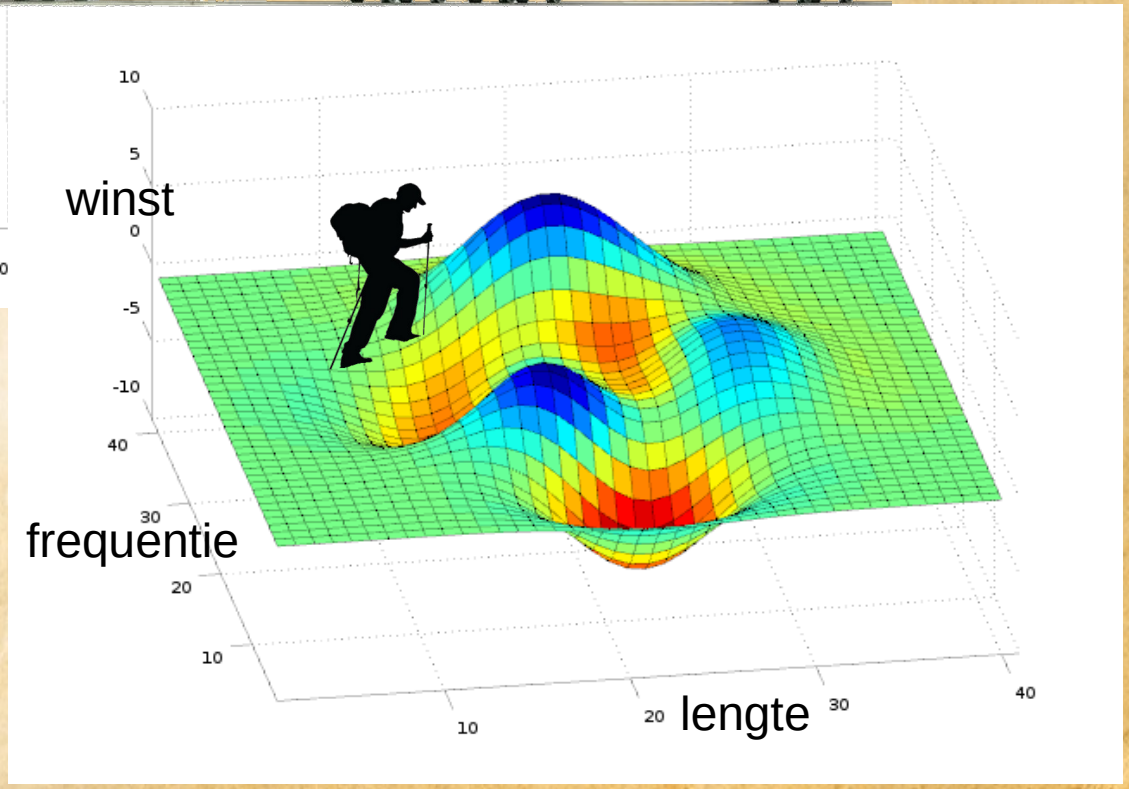
        Als de state is verslechterd:

            Maak de aanpassing ongedaan

Steepest Ascent Hill Climber



Random step: **Stochastic** Hill Climber
Best of all steps: **Steepest Ascent** Hill Climber



Steepest Ascent Hill Climber

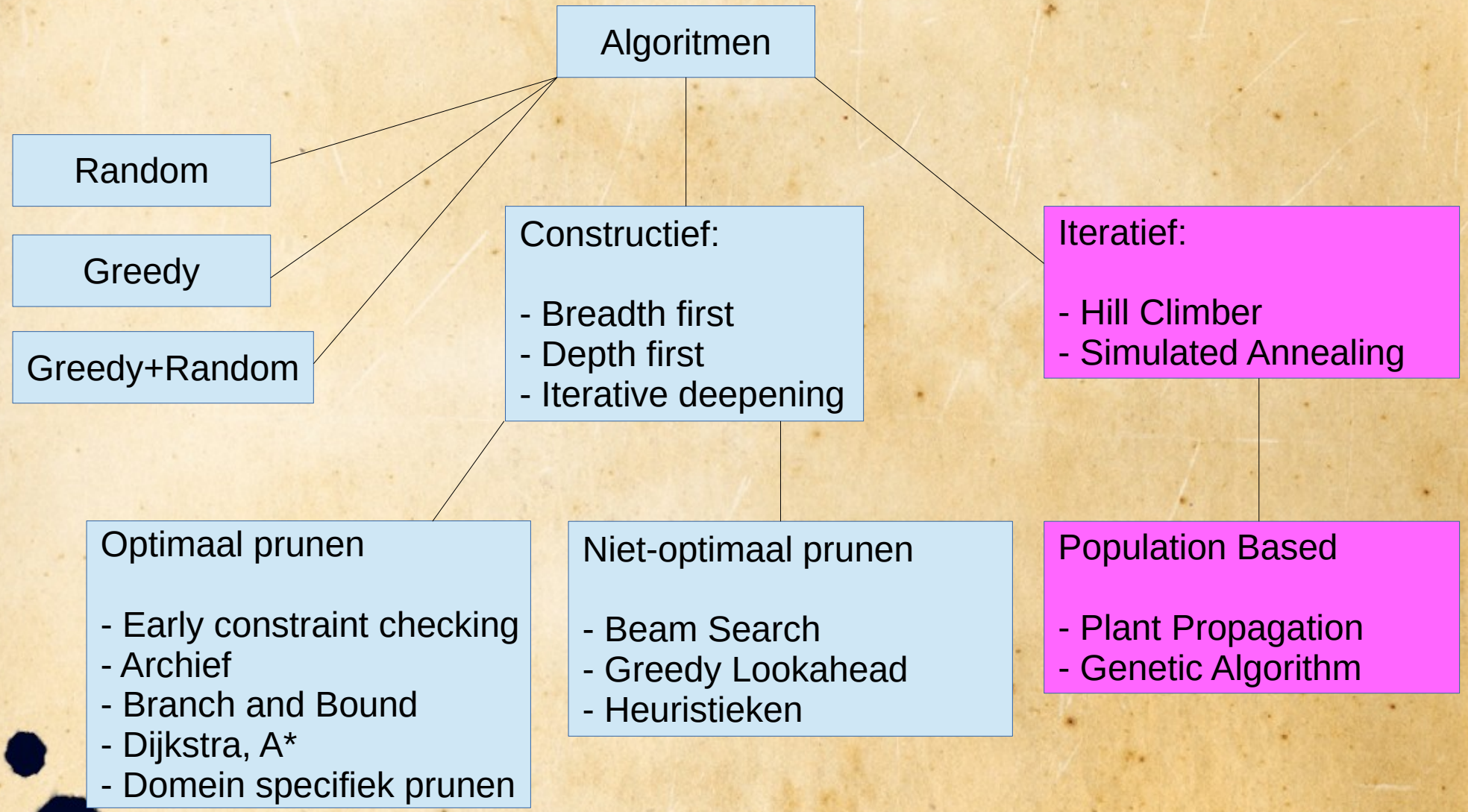
Herhaal:

Kies een random start state

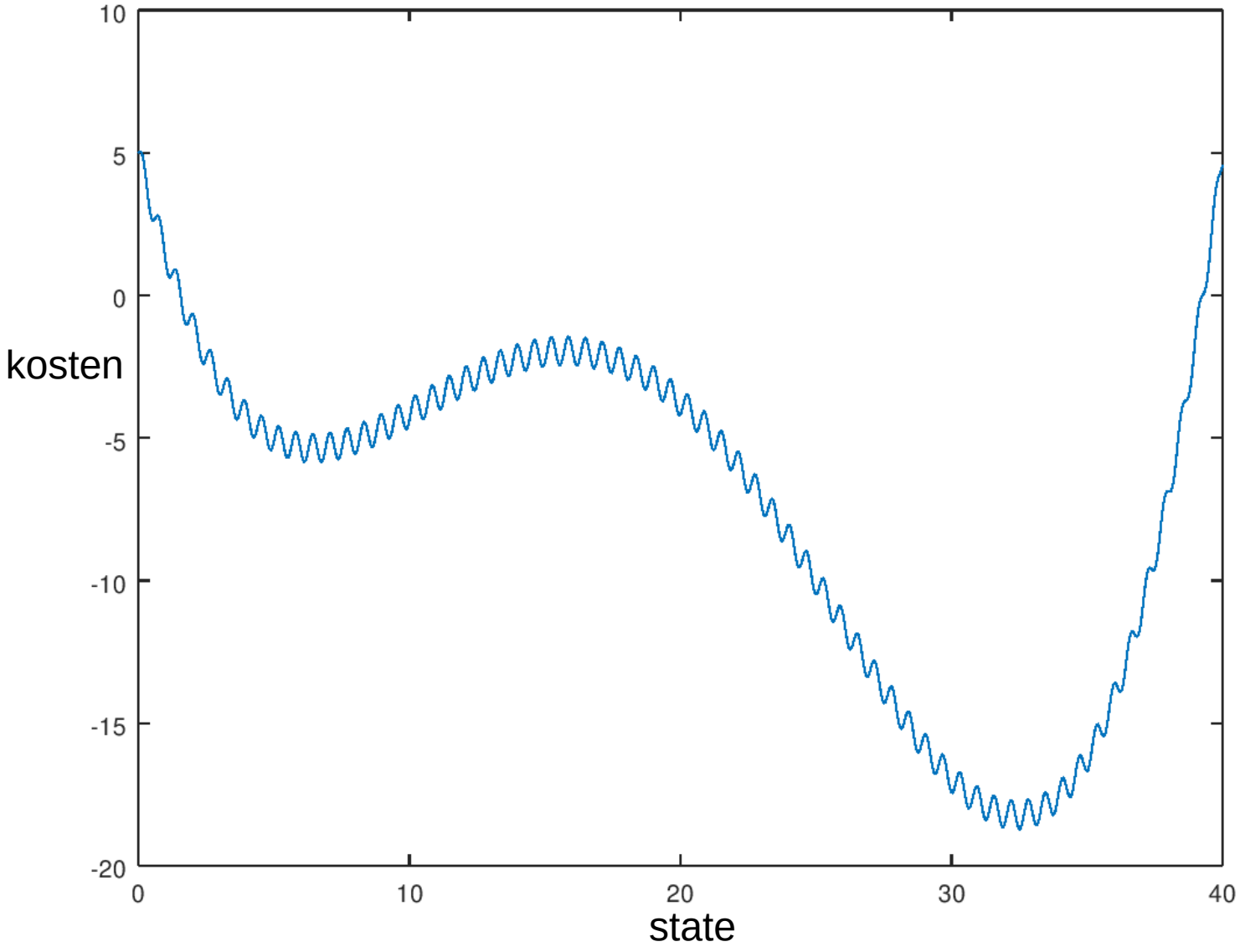
Herhaal **totdat niet meer** verbetert:

Doe **beste van alle mogelijke kleine aanpassingen**

Algoritmen

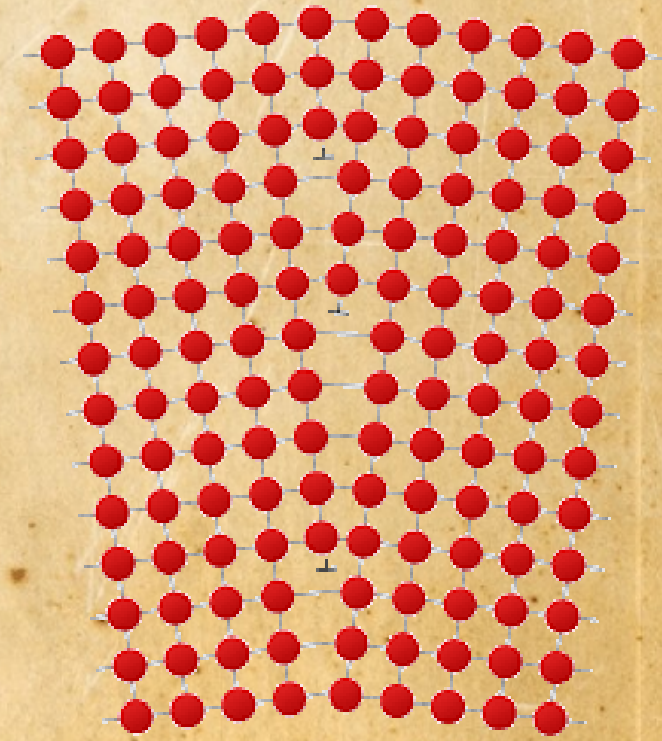

Algoritmen
Random
Greedy
Greedy+Random
Constructief:
- Breadth first
- Depth first
- Iterative deepening
Iteratief:
- Hill Climber
- Simulated Annealing
Optimaal prunen
- Early constraint checking
- Archief
- Branch and Bound
- Dijkstra, A*
- Domein specifiek prunen
Niet-optimaal prunen
- Beam Search
- Greedy Lookahead
- Heuristieken
Population Based
- Plant Propagation
- Genetic Algorithm

Simulated Annealing

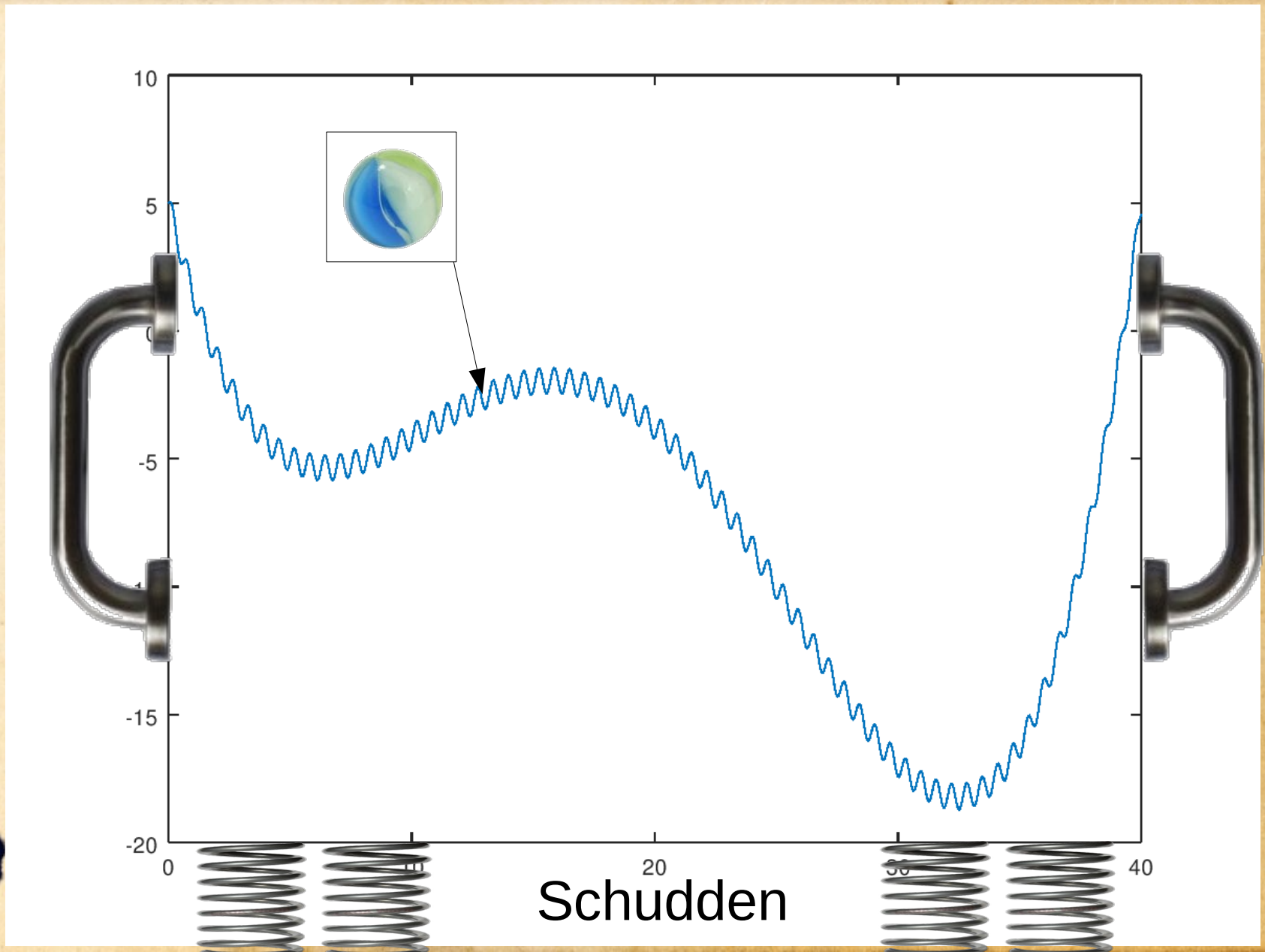


Metaal Annealing

- Kristalstructuur van metaal versterken

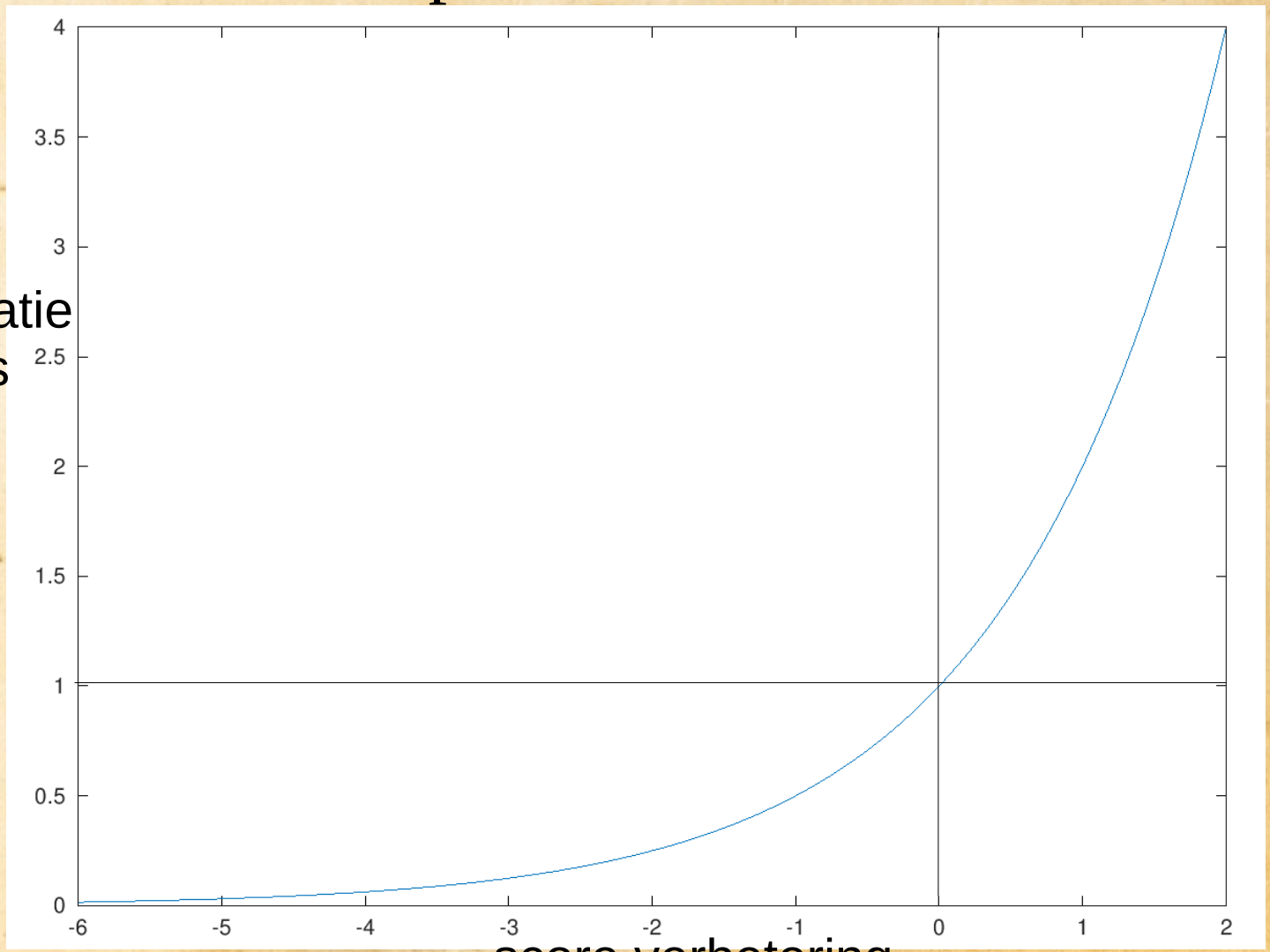


Simulated Annealing



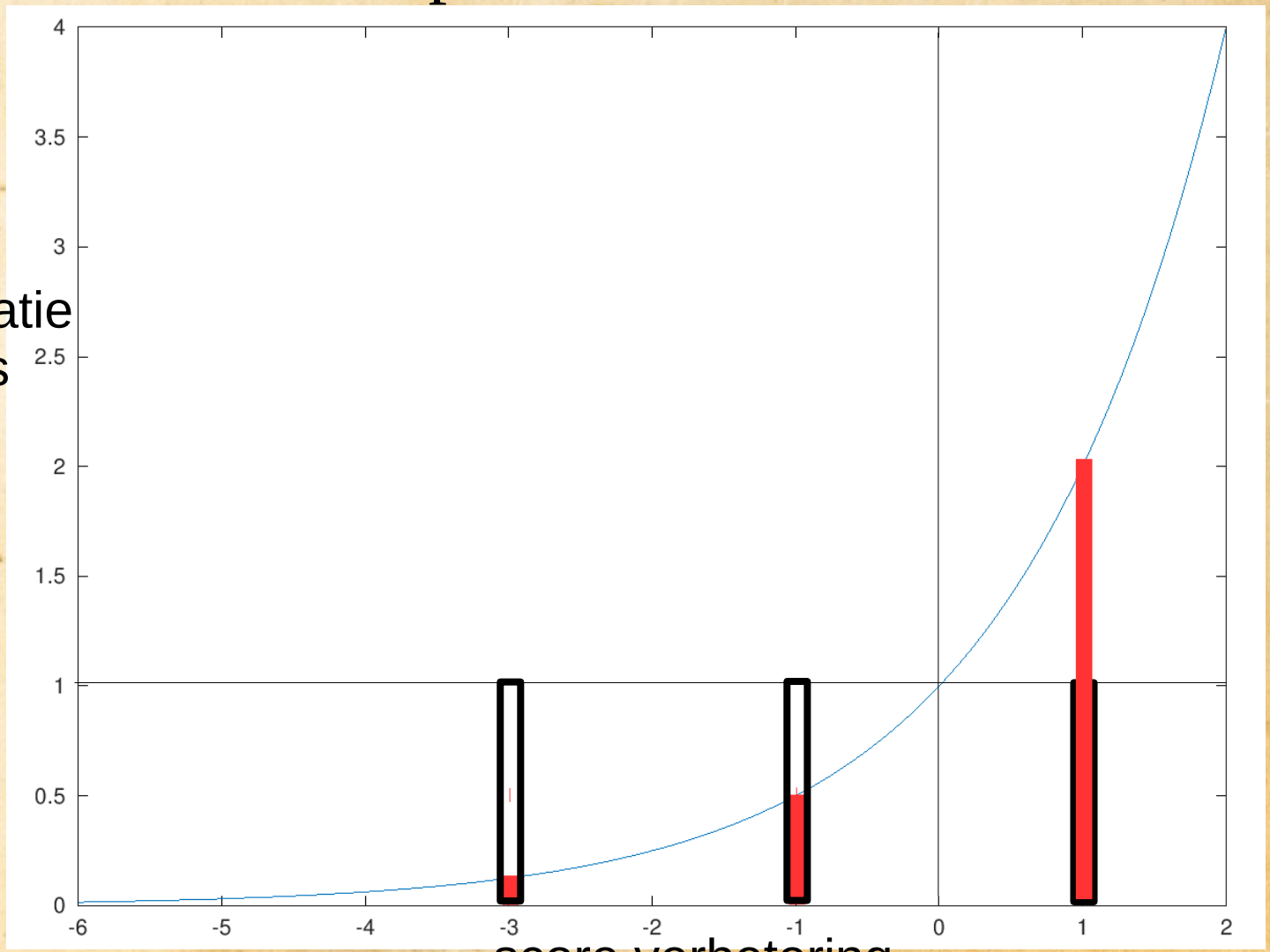
Simulated Annealing

- Minimaliseren: acceptatiekans = $2^{(\text{score\_old} - \text{score\_new})}$



Simulated Annealing

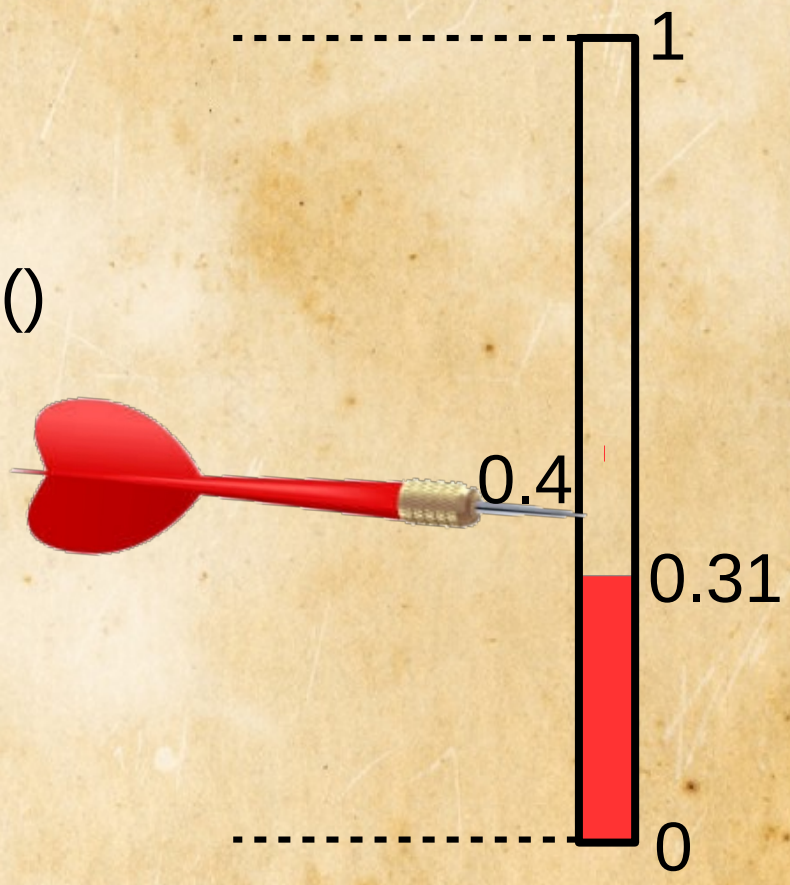
- Minimaliseren: acceptatiekans = $2^{(\text{score\_old} - \text{score\_new})}$



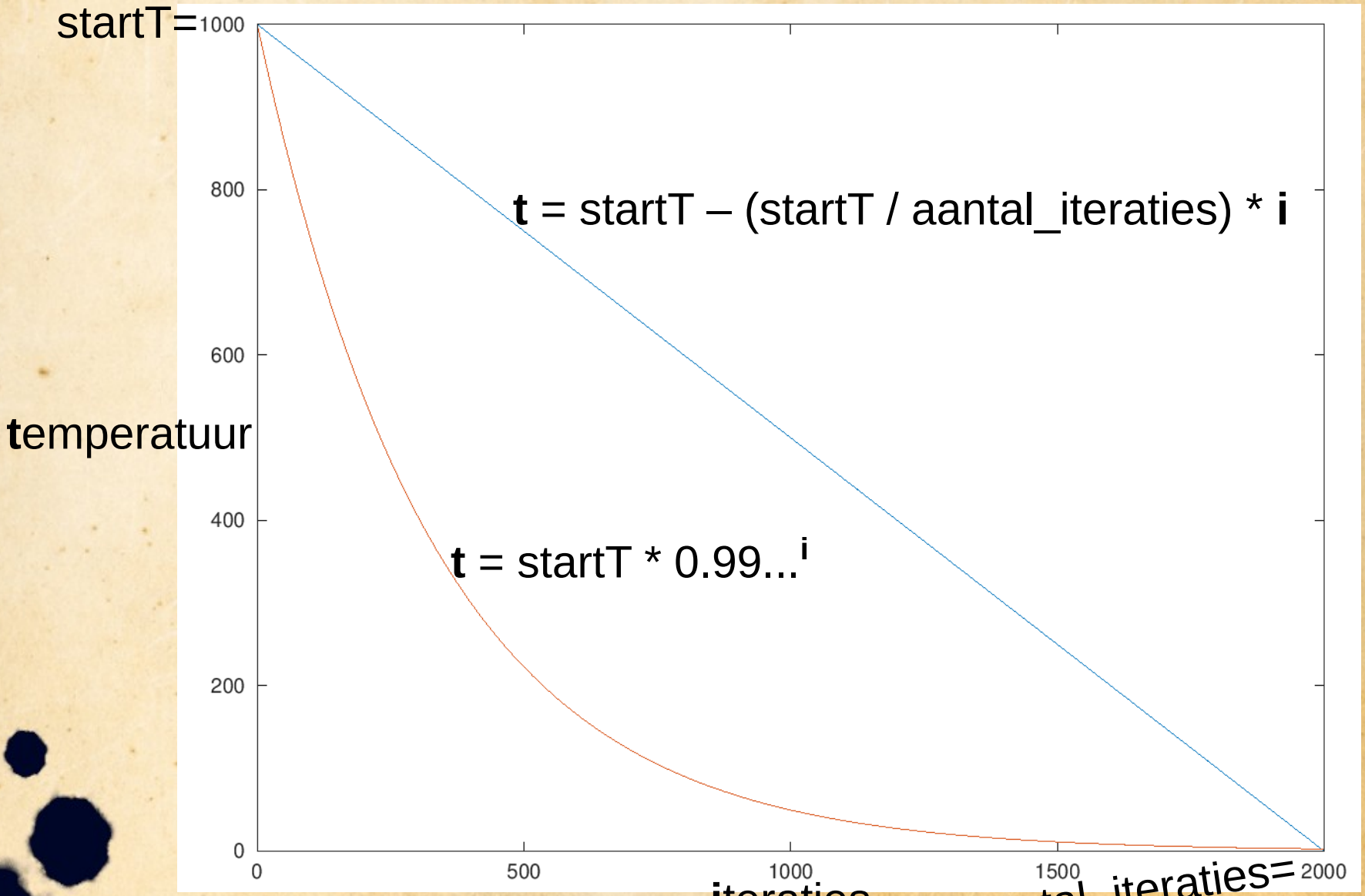
Simulated Annealing

- Wel of niet accepteren?

```
r = random.random()
if r < kans:
    …
```

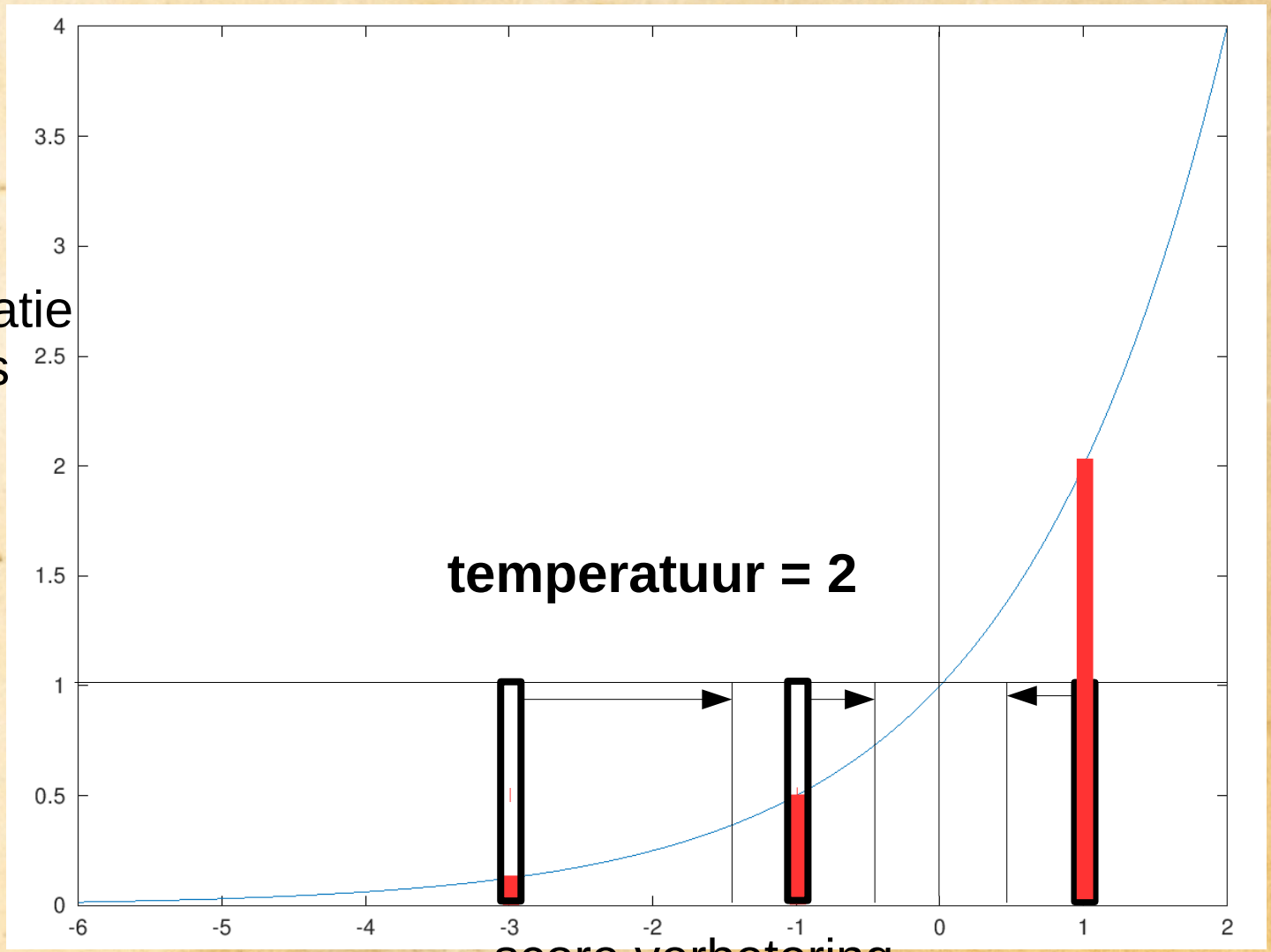


Simulated Annealing



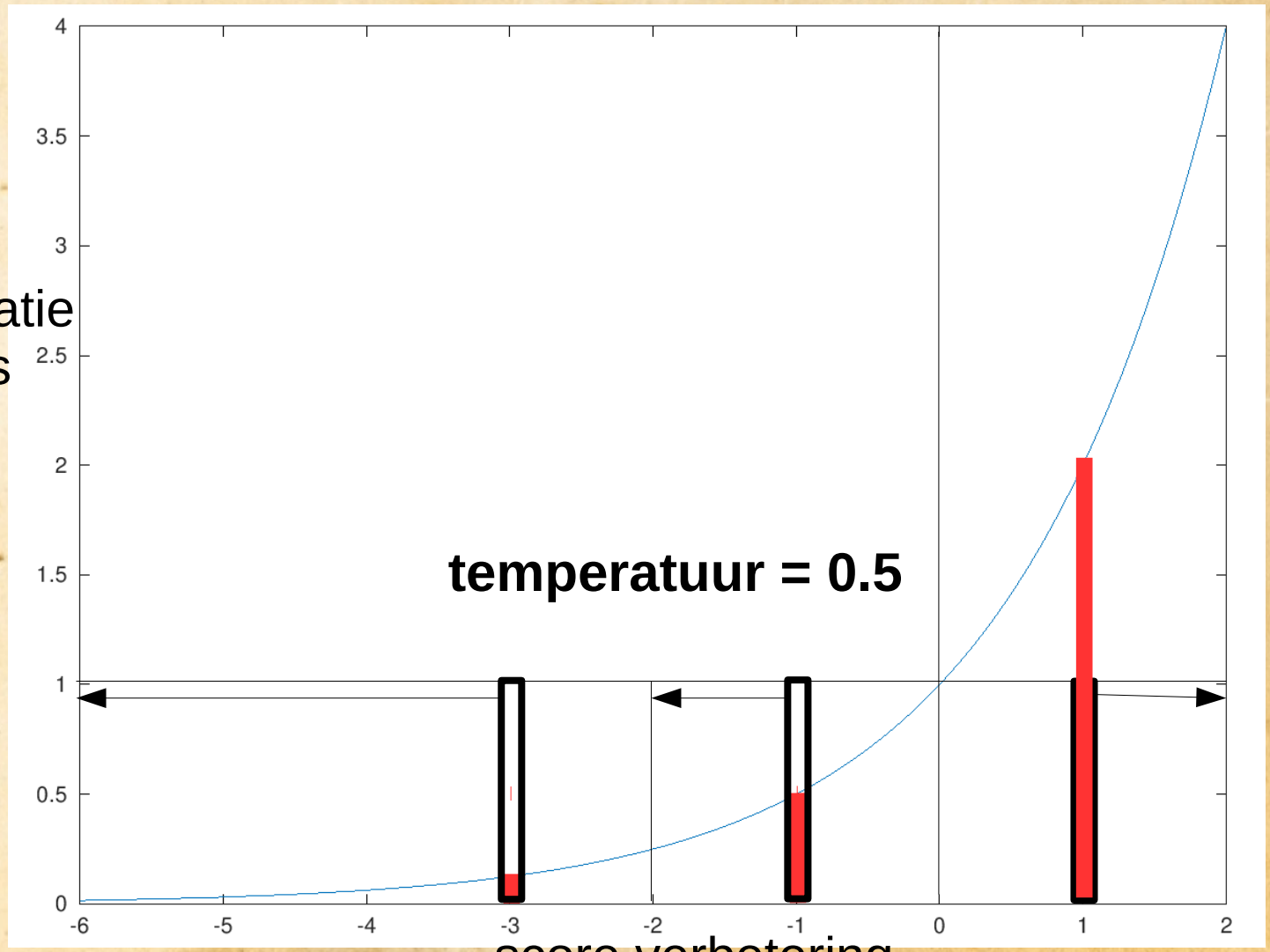
Simulated Annealing

- acceptatiekans = $2^{(\text{ score\_old} - \text{score\_new }) / \textbf{temperatuur}}$



Simulated Annealing

- acceptatiekans = $2^{(\text{ score\_old} - \text{score\_new }) / \textbf{temperatuur}}$



Simulated Annealing, pseudo code

Herhaal:

    Kies een random start state

    Zet de start-temperatuur

    Herhaal **N iteraties**:

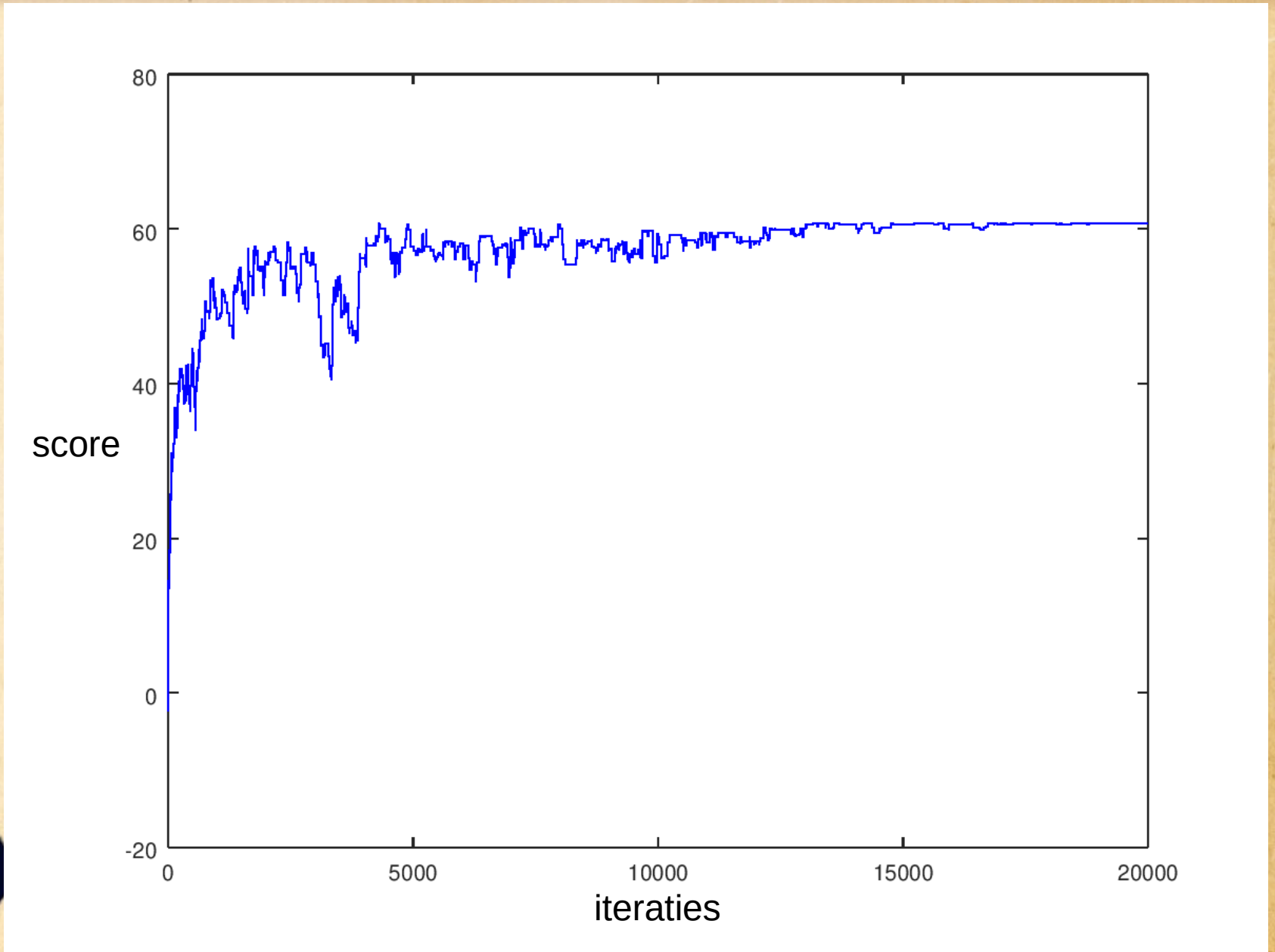
        Doe een kleine random aanpassing

        Als **random( ) > kans(oud, nieuw, temperatuur)**:

            Maak de aanpassing ongedaan

        Verlaag temperatuur

Simulated Annealing



Simulated Annealing, **T**emperatuur

- Welke **V**erslechtering wil ik bij de start 50% acceptatiekans geven? bv **V** = -20

$$2^{V/T} = \tfrac{1}{2} \qquad (50\%)$$

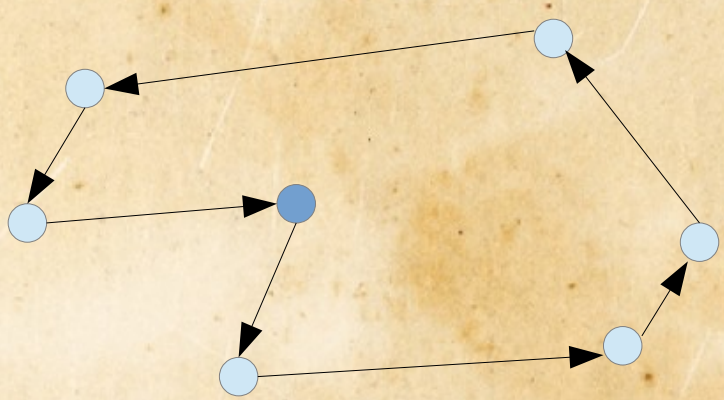
$$T = -V = 20$$

$$2^{-20/20} = 2^{-1} = \tfrac{1}{2}$$

probeer startT = [.., 5, 10, **20**, 40, 80, ...]

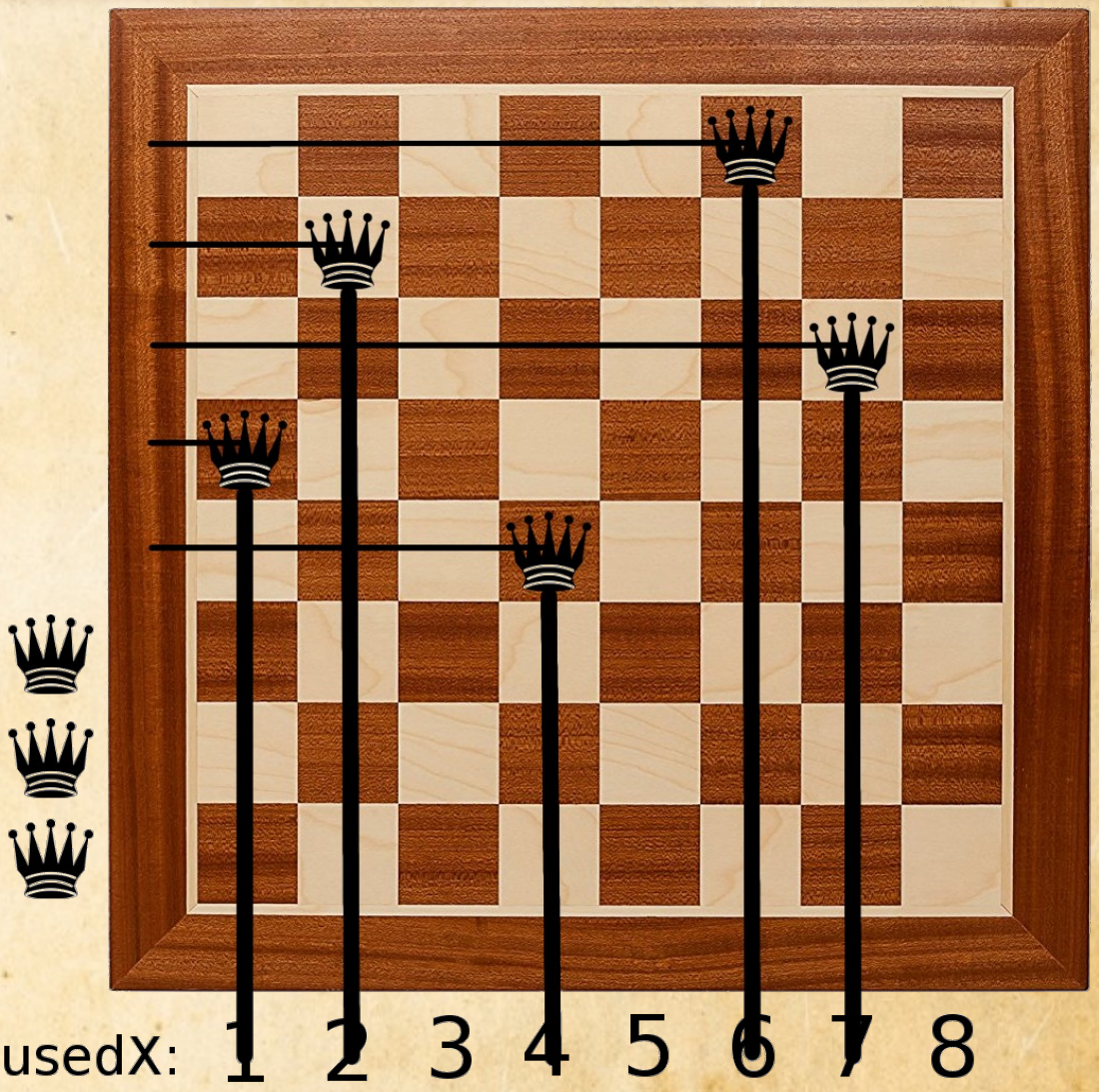
Traveling Salesman, Demo

- Een verkoper zoekt de kortste route langs N steden en terug naar huis:



- Youtube Video:
  - https://www.youtube.com/watch?v=SC5CX8drAtU
    - Greedy
    - Random
    - Hill Climber (Local Search)
    - Simulated Annealing

N-Queens



N-Queens

```
=== N:10
_queensX: [1, 3, 5, 7, 9, 0, 2, 4, 6, 8]
. Q . . . . . . . .
. . . Q . . . . . .
. . . . . Q . . . .
. . . . . . . Q . .
. . . . . . . . . Q
Q . . . . . . . . .
. . Q . . . . . . .
. . . . Q . . . . .
. . . . . . Q . . .
. . . . . . . . Q .
```

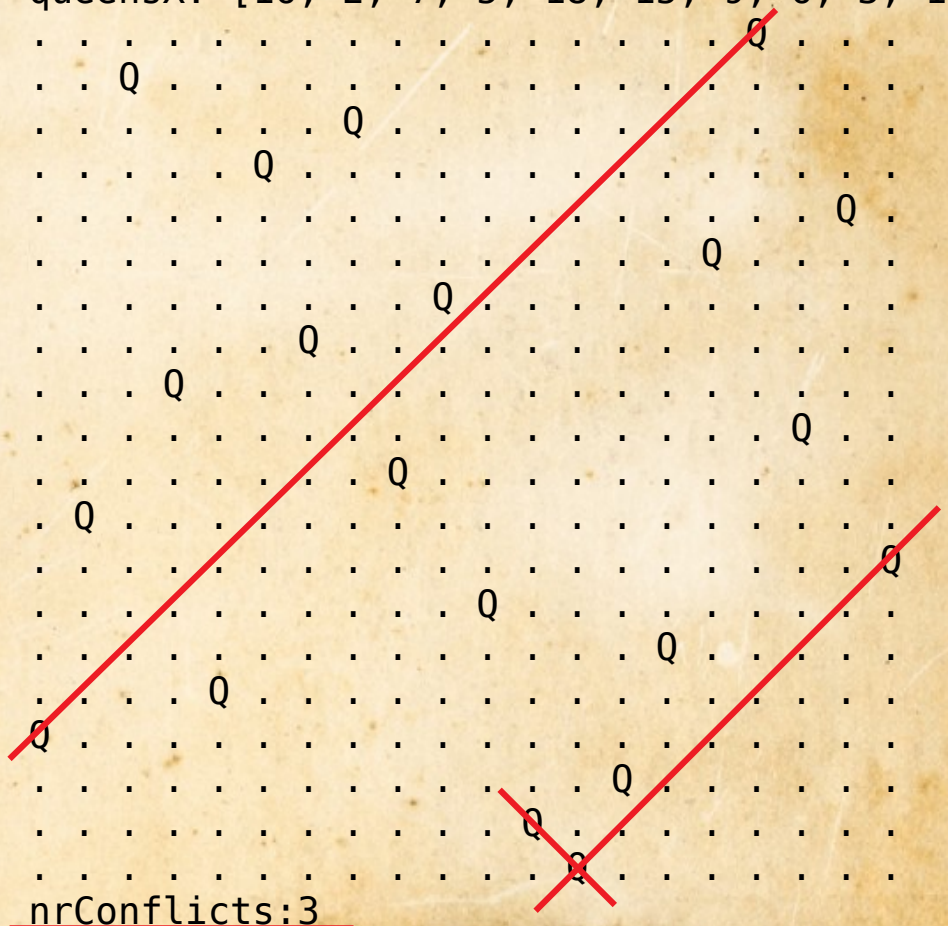
Pattern works for even N that are not 6K + 2

N	not working
4	
6	
8	*
10	
12	
14	*
16	
18	
20	*
22	
24	
26	*
28	
..	..

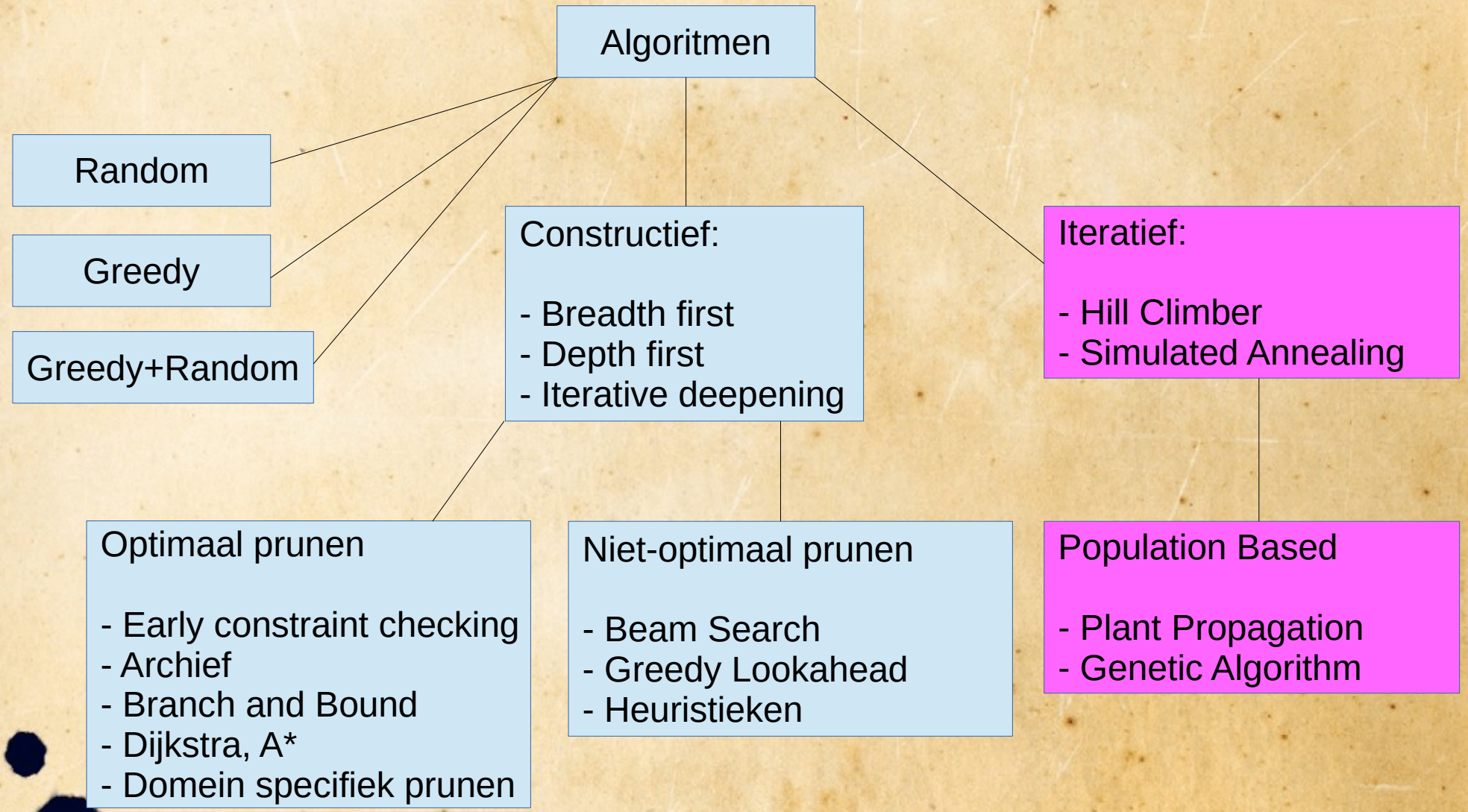
N-Queens, Demo

GitHub: https://github.com/bterwijn/NQueens

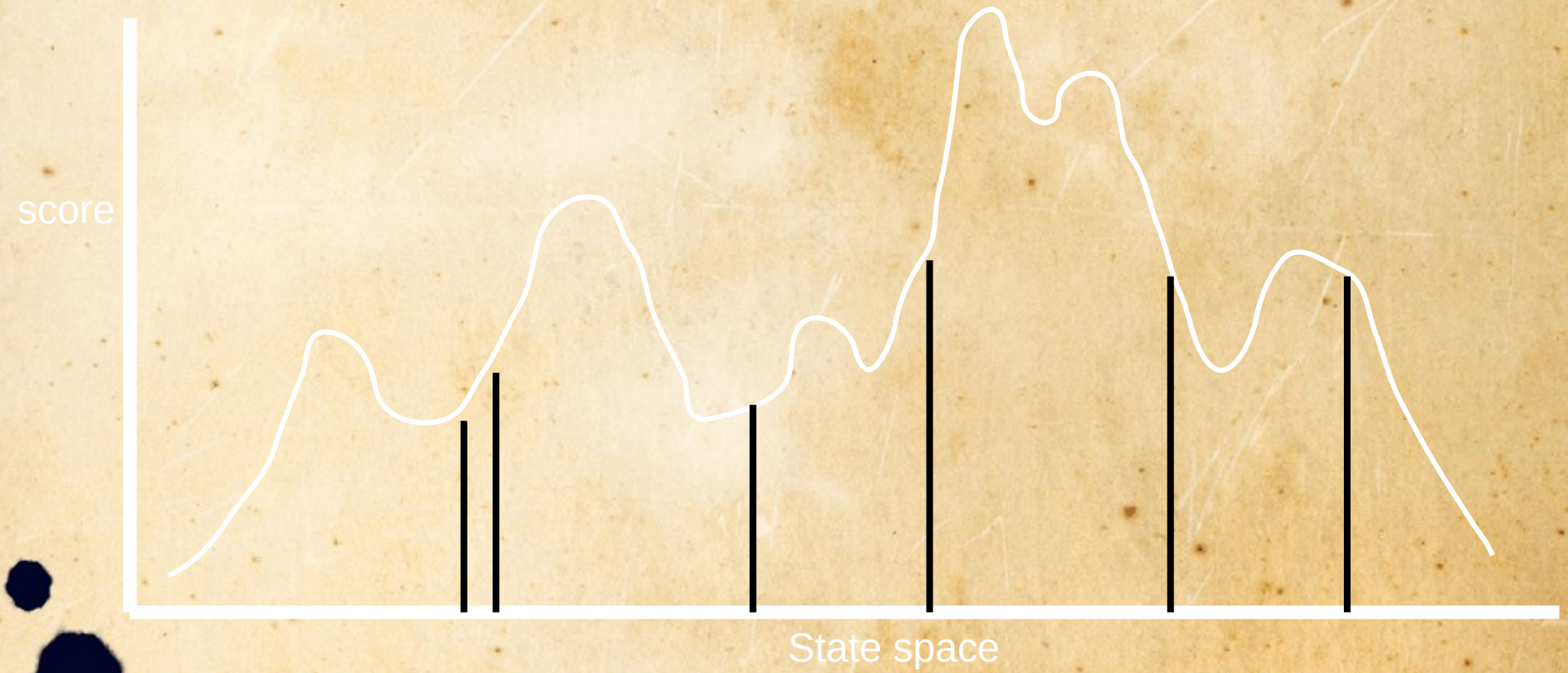
```
python3 -m NQueens.ConstructiveAlgorithms.Random 20 p
queensX: [16, 2, 7, 5, 18, 15, 9, 6, 3, 17, 8, 1, 19, 10, 14, 4, 0, 13, 11, 12]
```



Algoritmen


Algoritmen
Random
Greedy
Greedy+Random
Constructief:
- Breadth first
- Depth first
- Iterative deepening
Iteratief:
- Hill Climber
- Simulated Annealing
Optimaal prunen
- Early constraint checking
- Archief
- Branch and Bound
- Dijkstra, A*
- Domein specifiek prunen
Niet-optimaal prunen
- Beam Search
- Greedy Lookahead
- Heuristieken
Population Based
- Plant Propagation
- Genetic Algorithm

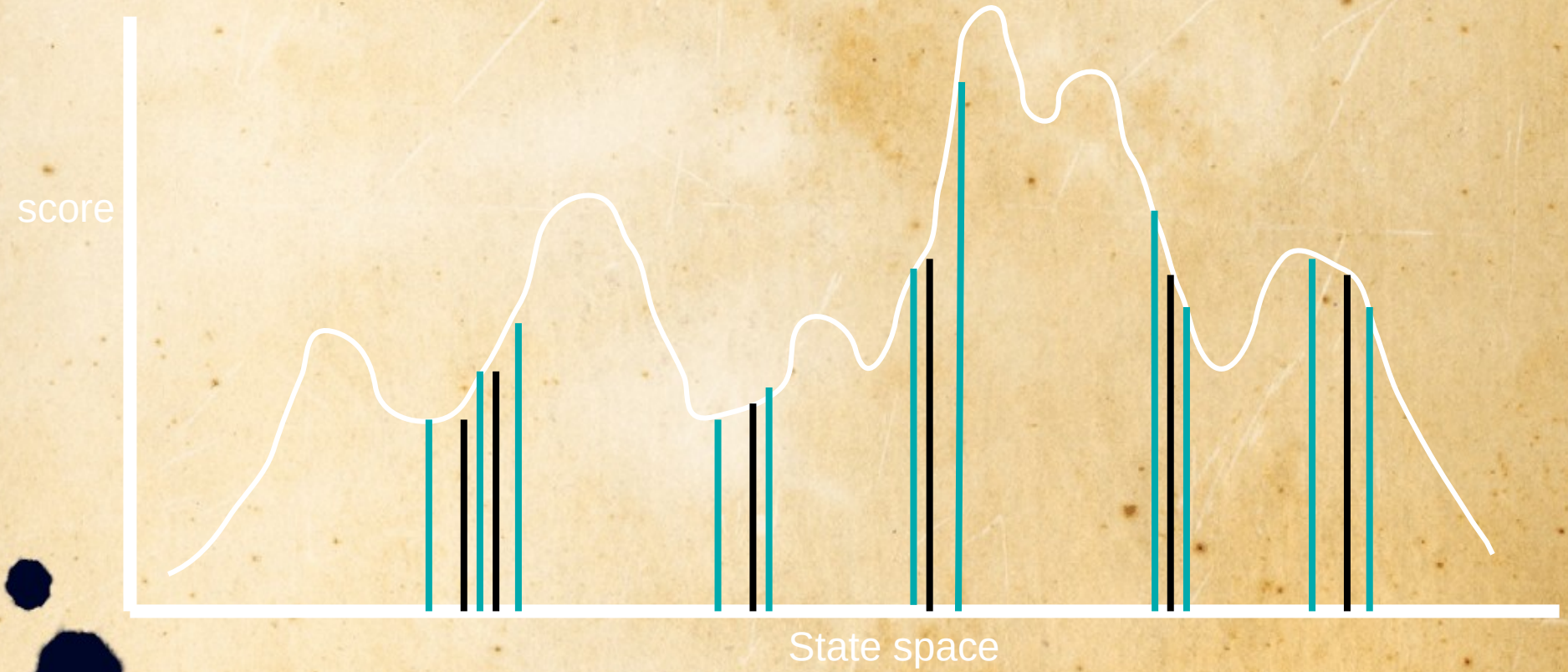
Population based algorithms



Populatie van Hill Climbers: 6

Population based algorithms

Populatie krijgt nieuwe kinderen



Populatie van Hill Climbers: 6

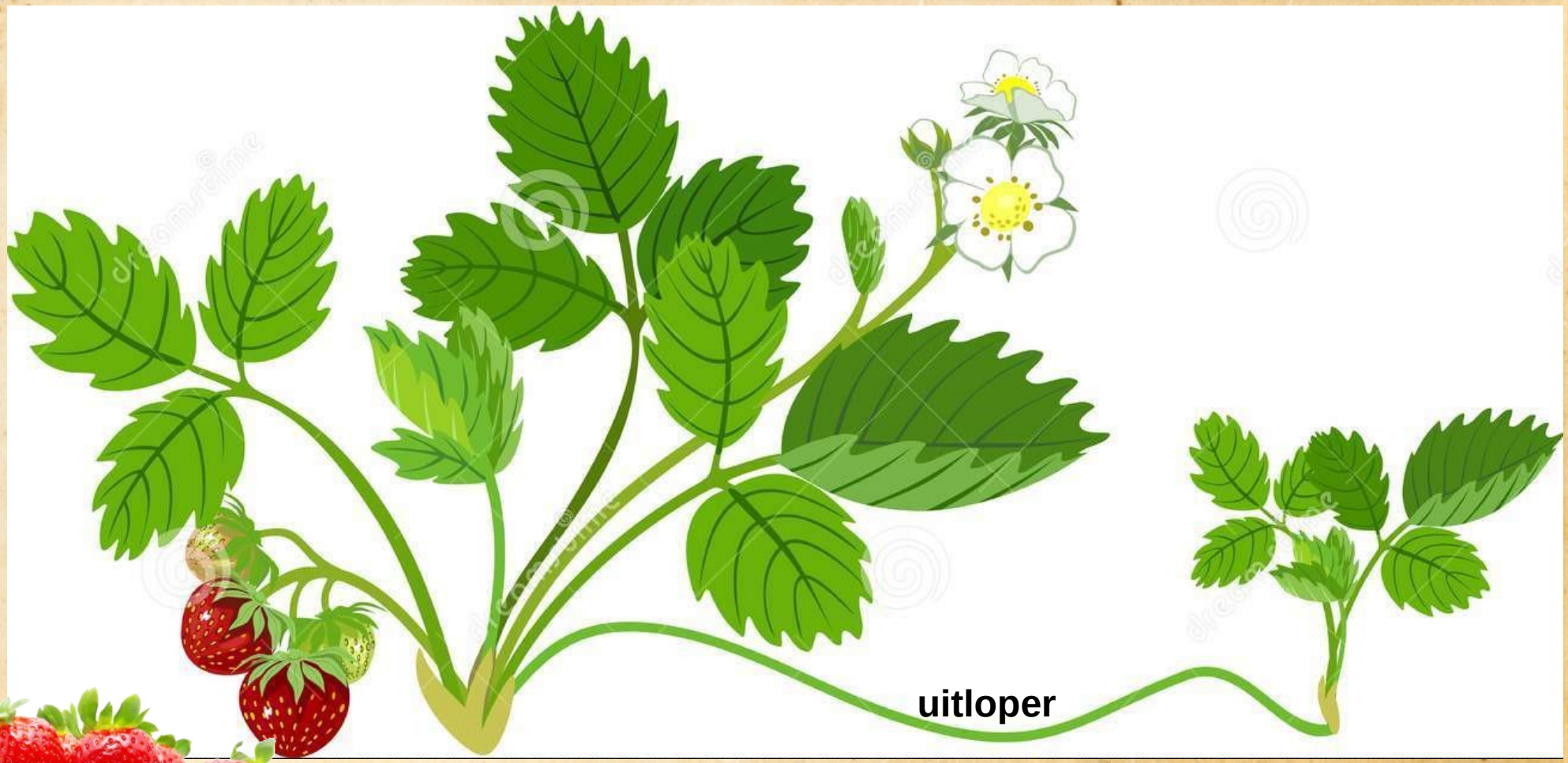
Population based algorithms

Populatie van Hill Climbers: 6

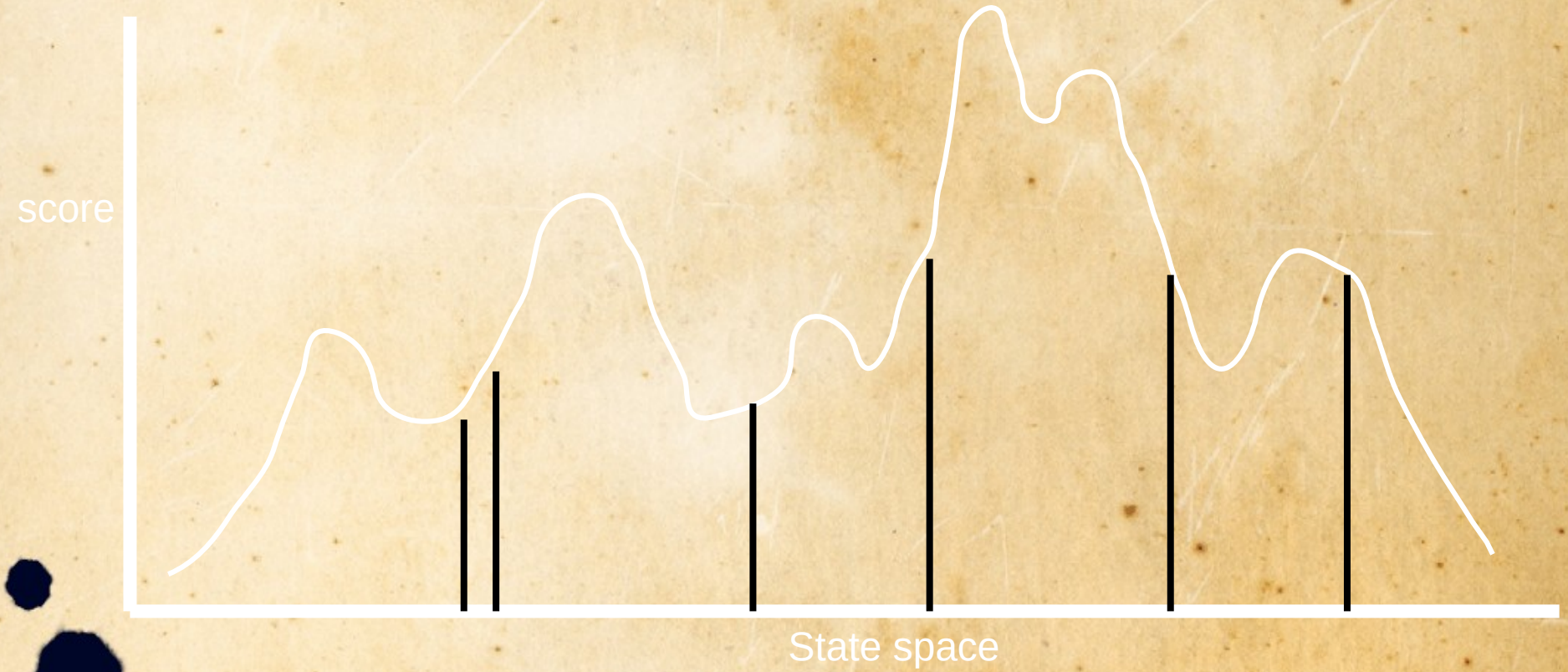
Plant Propagation Algorithm

- Paper:
  - **Nature-Inspired Optimisation Approaches and the New Plant Propagation Algorithm**
    - Abdellah Salhi, Eric S Fraga

Plant Propagation Algorithm

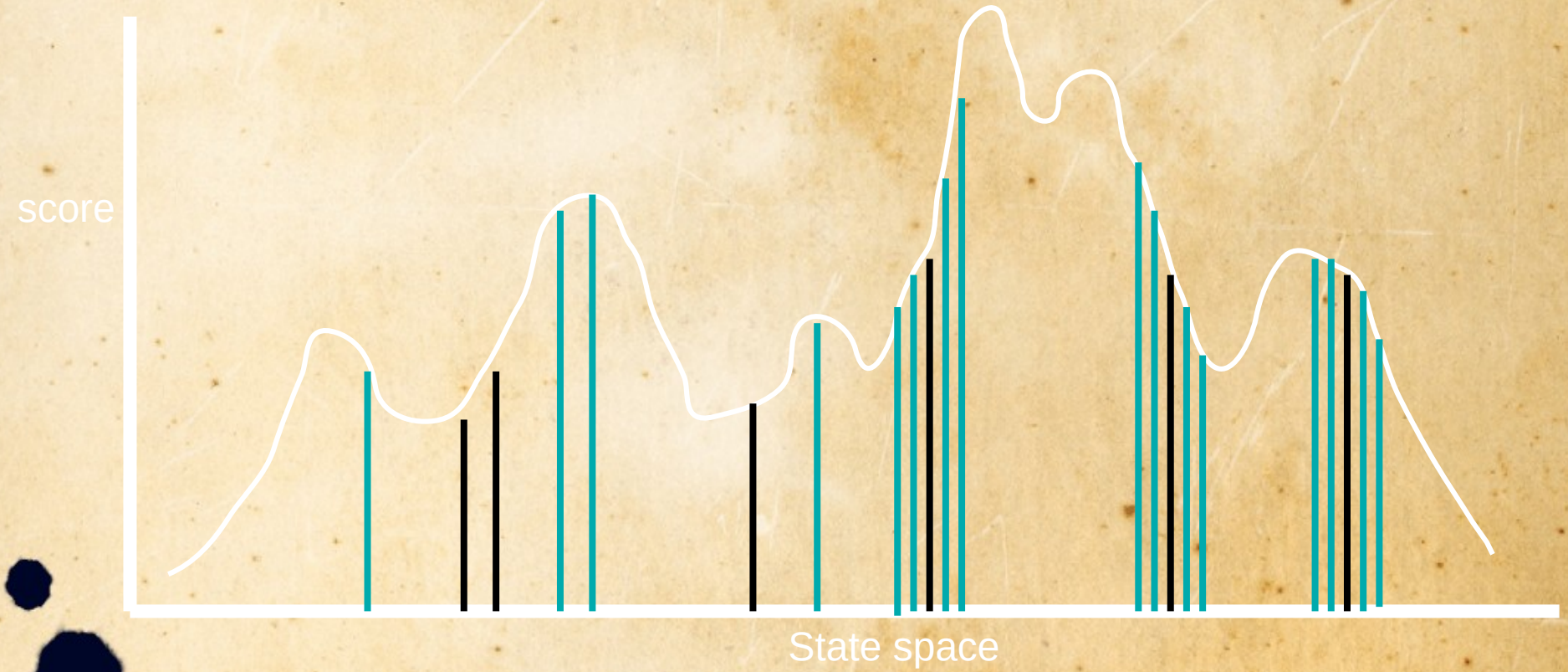


Plant Propagation Algorithm



Populatie van Hill Climbers: 6

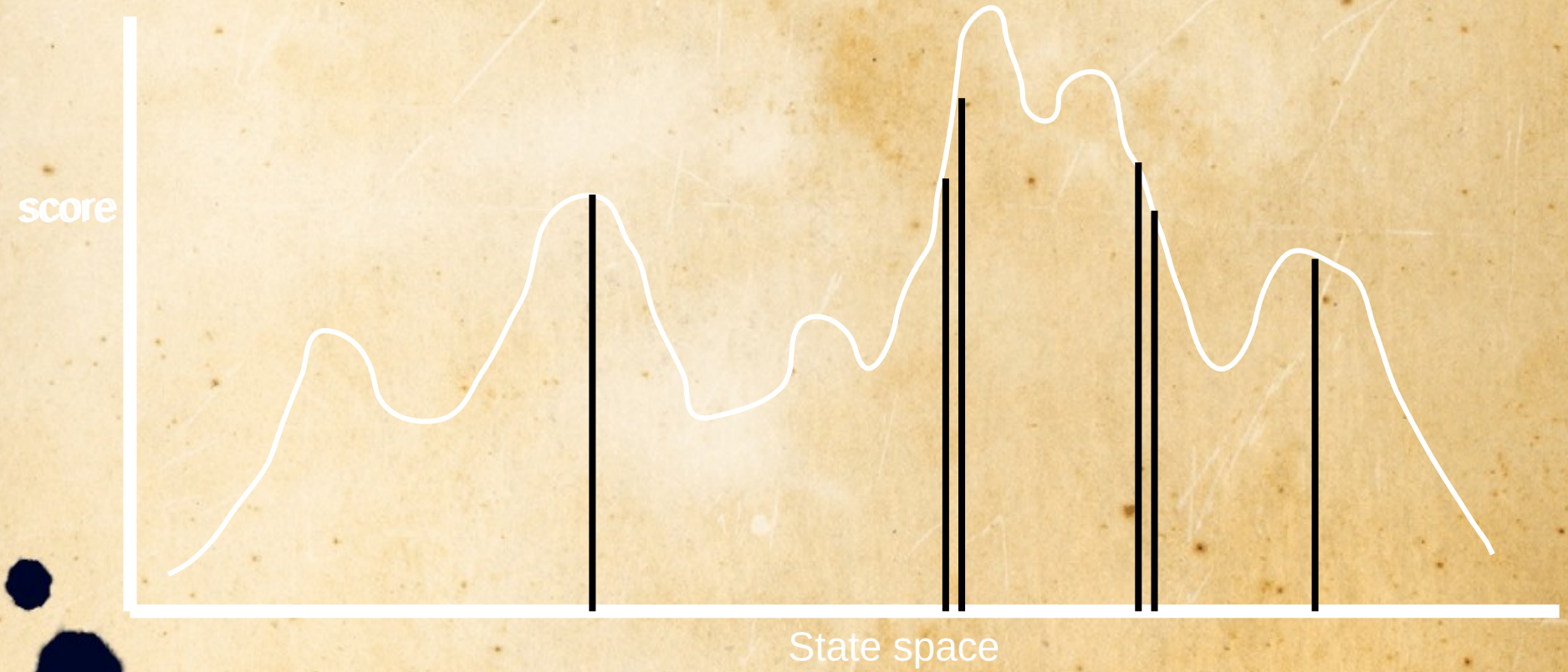
Plant Propagation Algorithm



Populatie van Hill Climbers: 6

Plant Propagation Algorithm

Selecteer beste 6



Populatie van Hill Climbers: 6

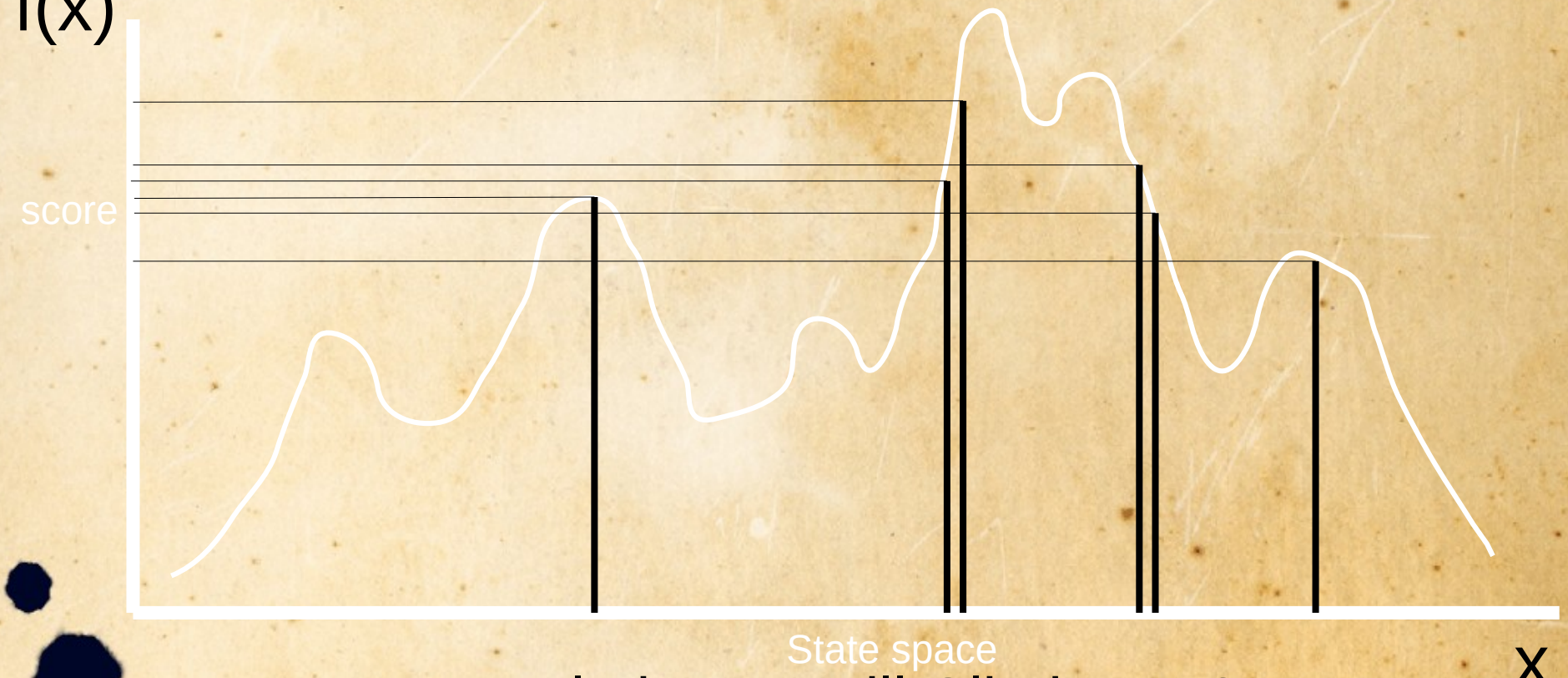
Plant Propagation Algorithm

Require: objective $f(x)$, $x \in \mathcal{R}^n$
  Generate a population $P = \{p_i, i = 1, \ldots, m\}$
  $g \leftarrow 1$
  for $g \leftarrow 1$ **to** $g_{\max}$ **do**
    compute $N_i = f(p_i), \forall p_i \in P$
    sort P in descending order of N
    create new population ϕ
    for each $p_i, i = 1, \ldots, m$ **do** {best m only}
      $r_i \leftarrow$ set of runners where both the size of the set and the distance for each runner (individually) is proportional to the fitness N_i
      $\phi \leftarrow \phi \cup r_i$ {append to population; death occurs by omission above}
    end for
    $P \leftarrow \phi$ {new population}
  end for
  return P, the population of solutions

Plant Propagation Algorithm

$$N(x) = \frac{1}{2}\left(\tanh\left(4f(x) - 2\right) + 1\right)$$

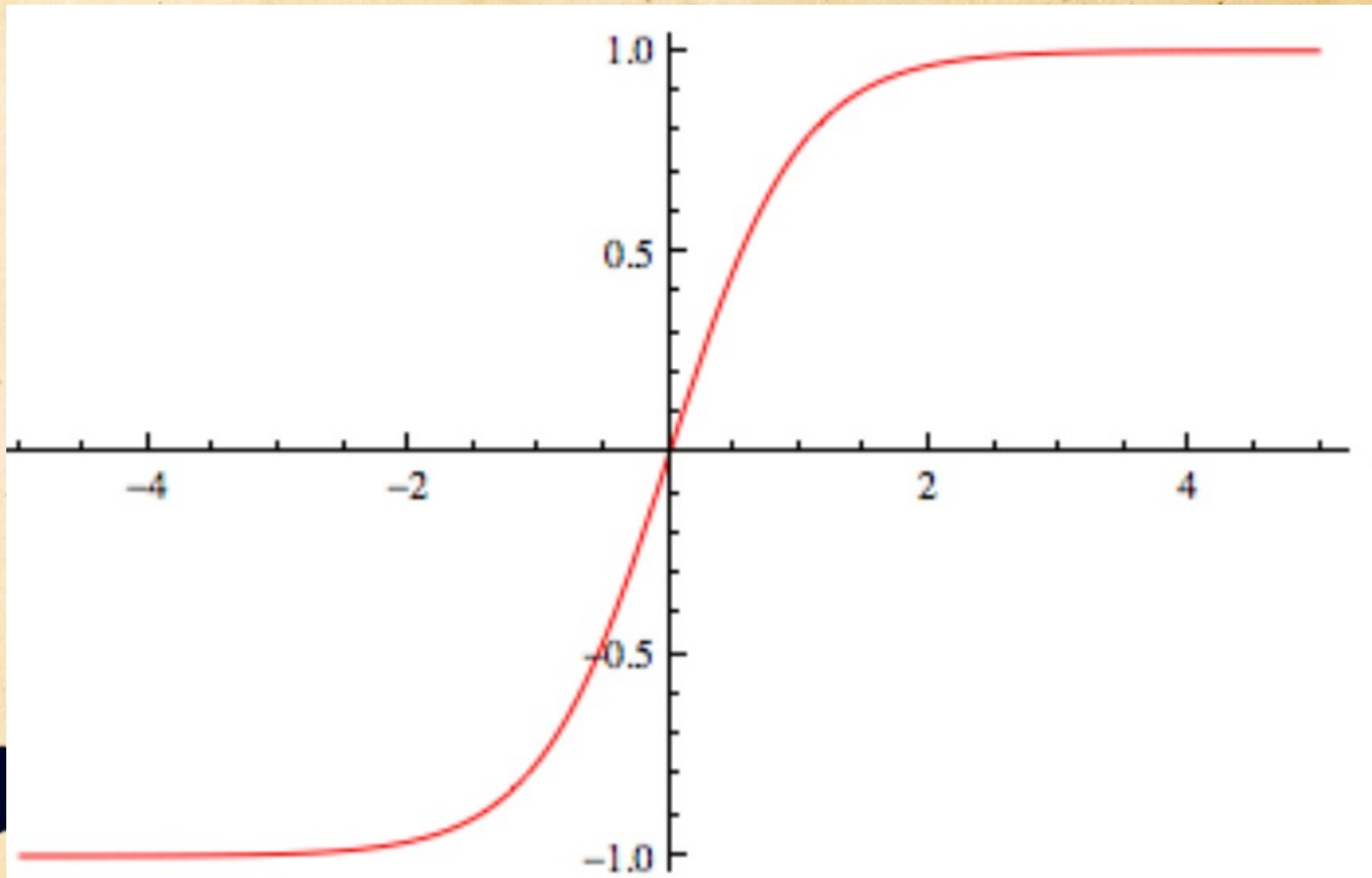
values lie strictly in $(0, 1)$.



Populatie van Hill Climbers: 6

Plant Propagation Algorithm

tanh(f(x))



Plant Propagation Algorithm

Number of runners : $n_r = \lceil n_{\max} N_i r \rceil$

$n_{\max}$ is the maximum number of runners to generate

in this paper, $n_{\max} = 5$.

$r \in [0, 1]$ is a randomly chosen number

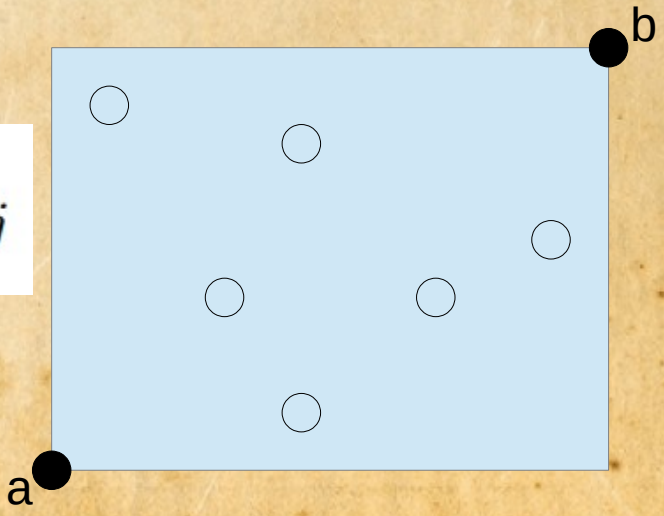
Plant Propagation Algorithm

Distance of runner: $$d_{r,j} = 2(1 - N_i)(r - 0.5)$$

Each $d_{r,j}$ will be in (-1,1).

for $j = 1, \ldots, n$, where n is the dimension of the search space.

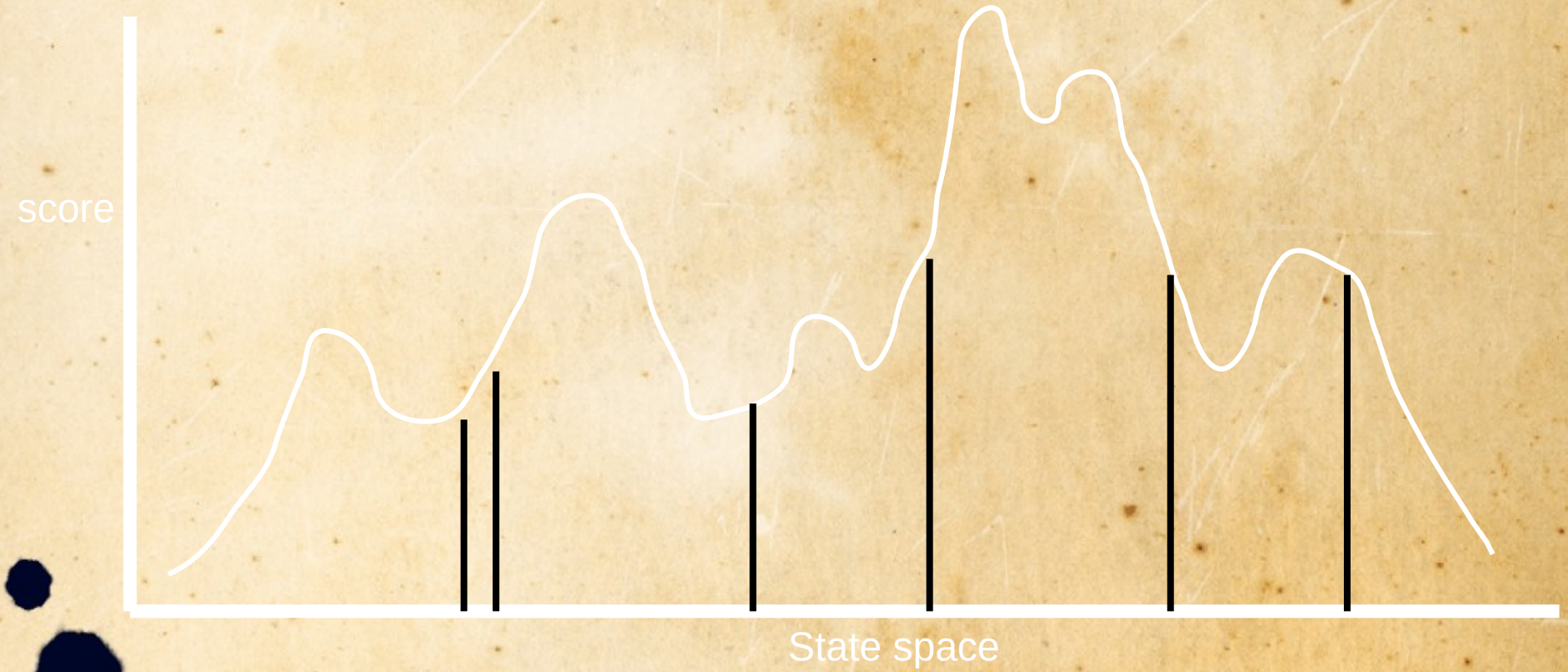
$$x_j^* = x_j + (b_j - a_j)d_{r,j}$$



Genetic Algorithm

- Website:
  - **https://www.tutorialspoint.com/genetic_algorithms**

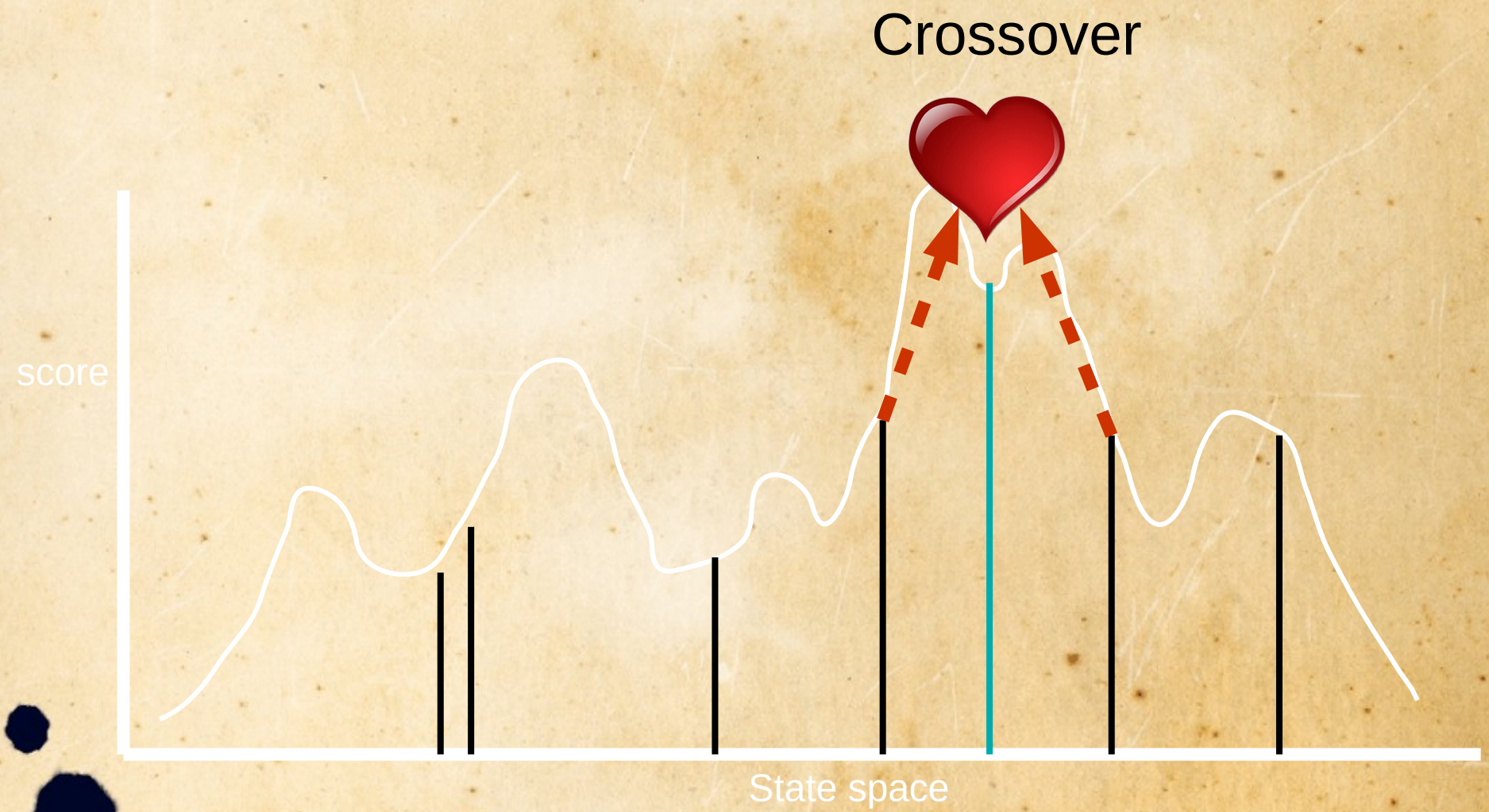
Genetic Algorithm



Populatie van Hill Climber: 6

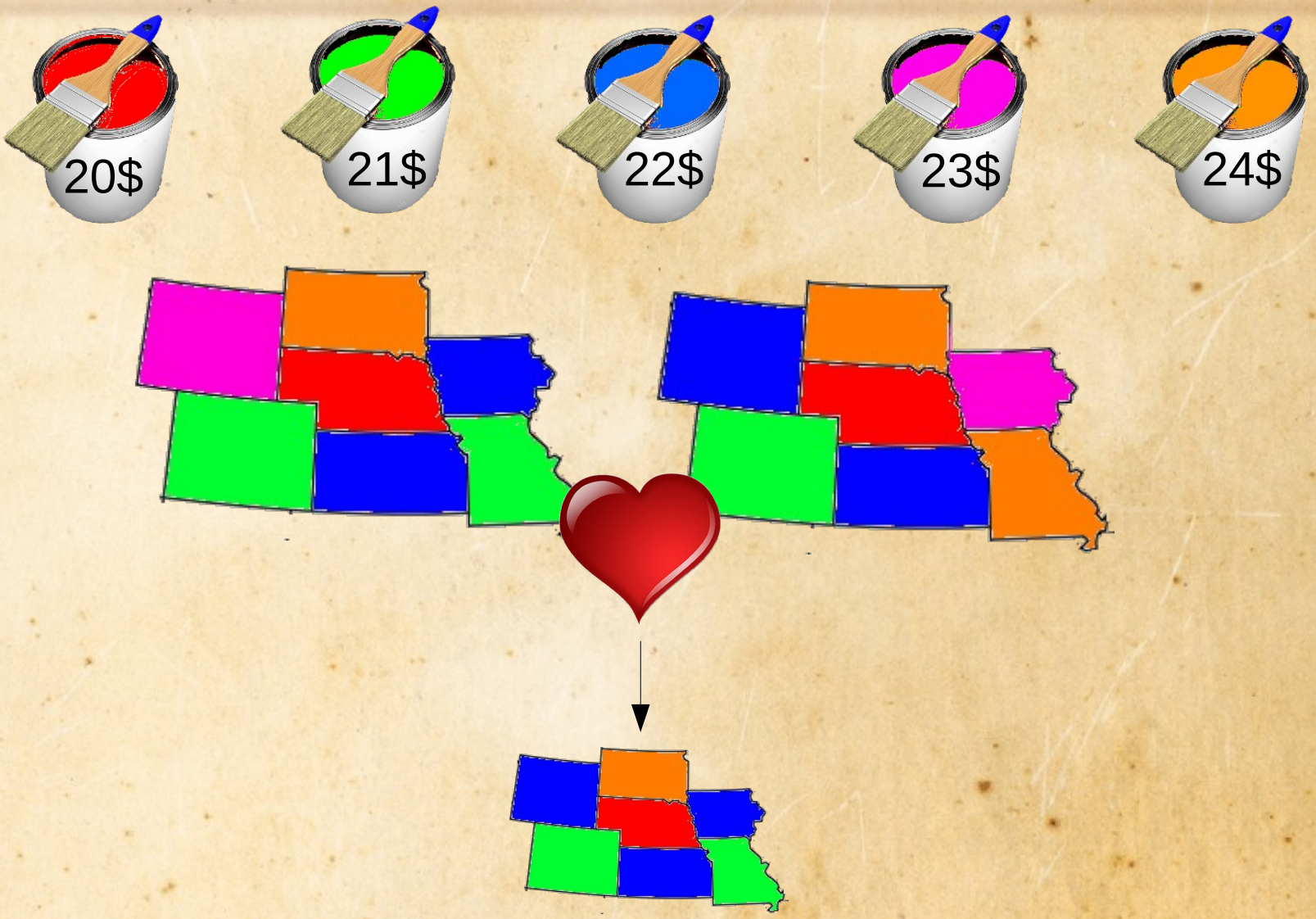
Genetic Algorithm

Genetic Algorithm

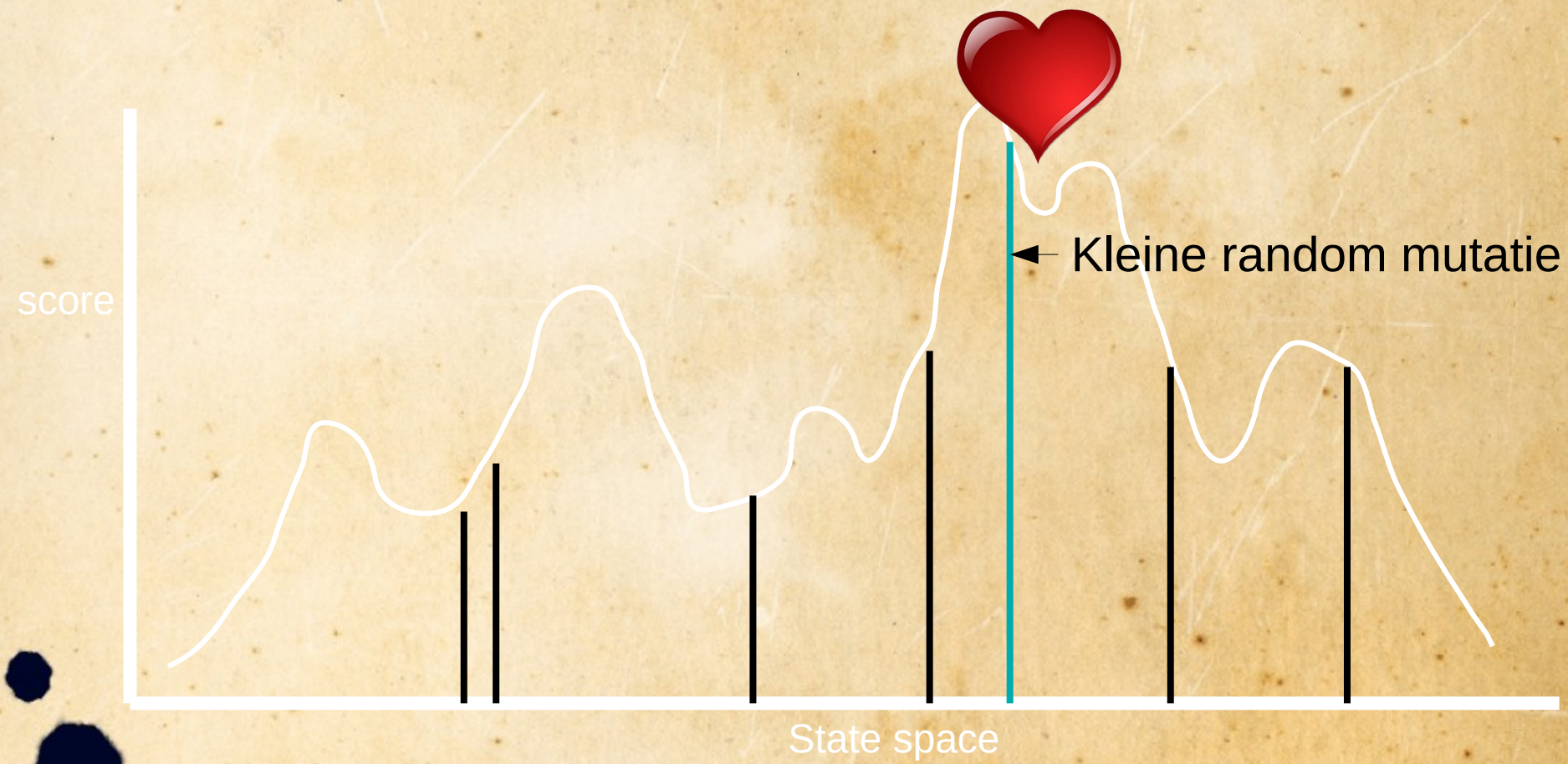


Populatie van Hill Climber: 6

Genetic Algorithm

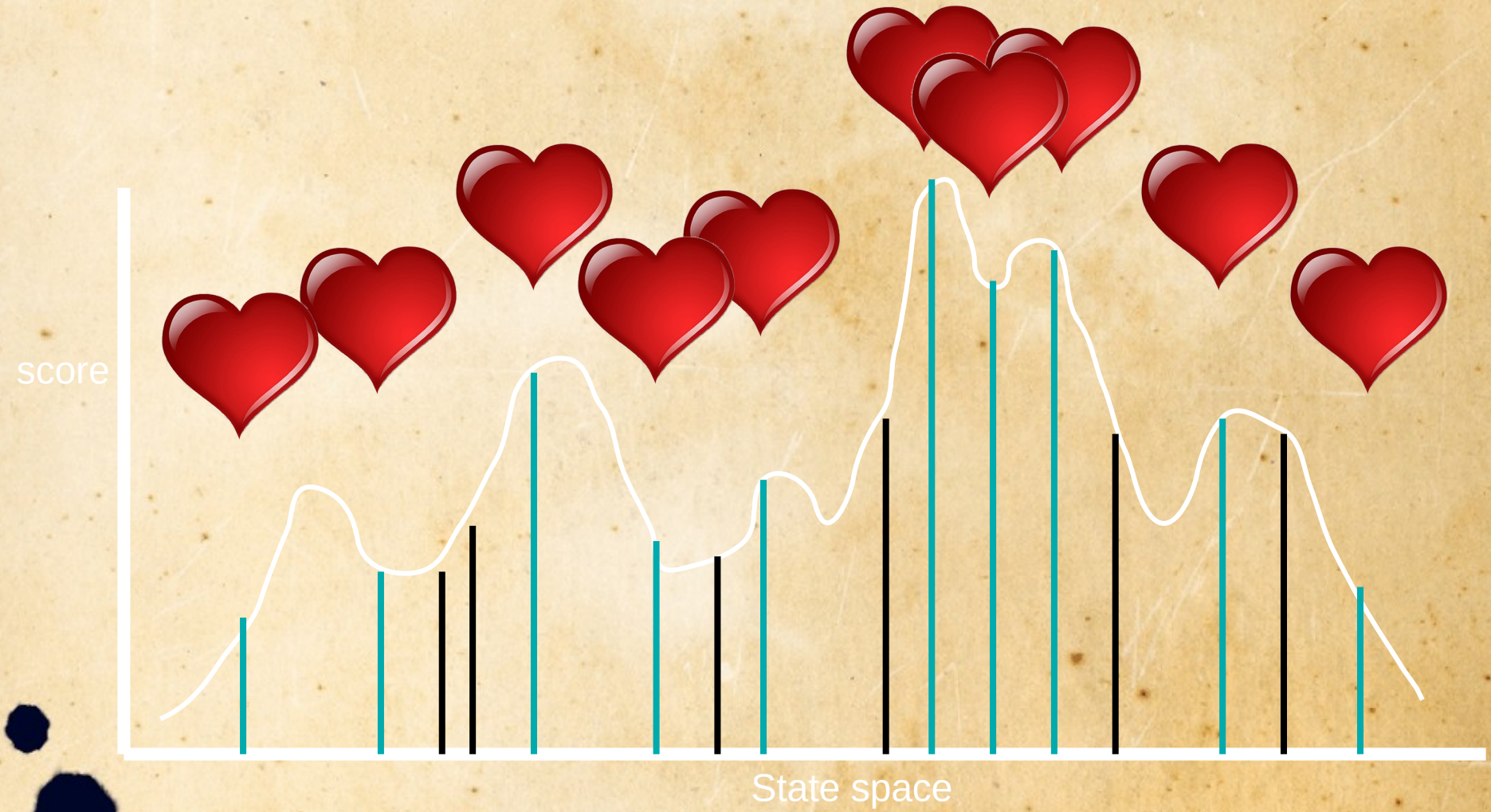


Genetic Algorithm



Populatie van Hill Climber

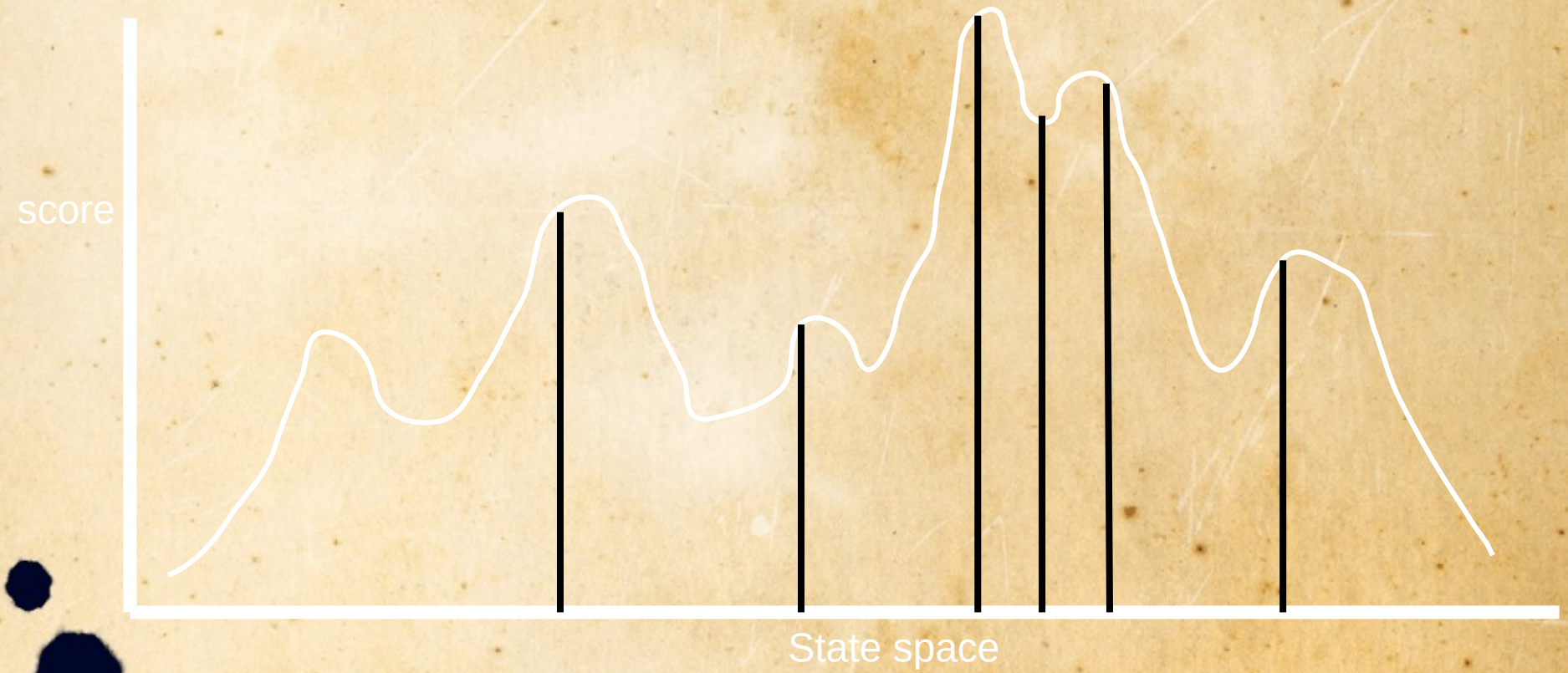
Genetic Algorithm



Populatie van Hill Climber

Genetic Algorithm

Select the best 6

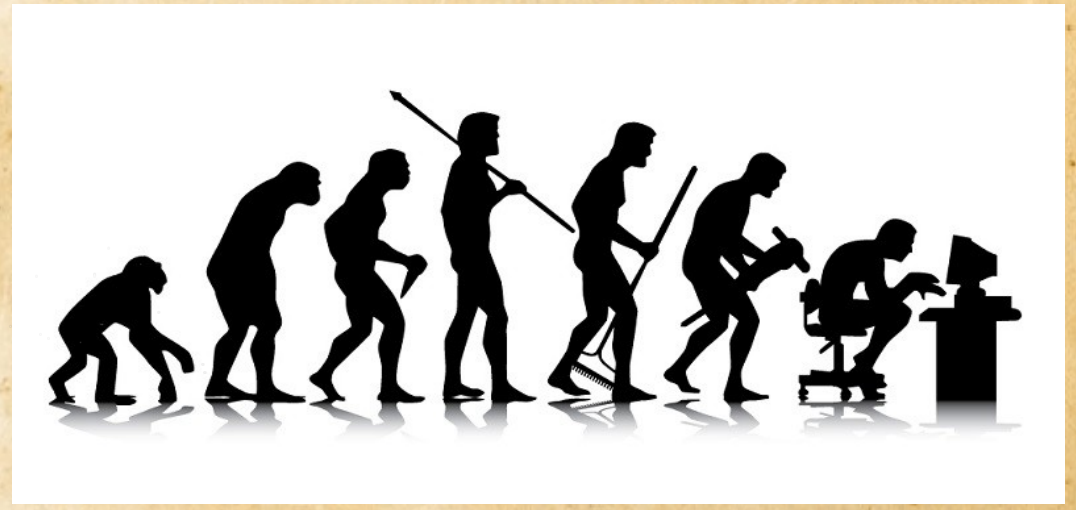


Populatie van Hill Climber: 6

Genetic Algorithm

https://www.wherewithalfilms.com

Genetic Algorithm



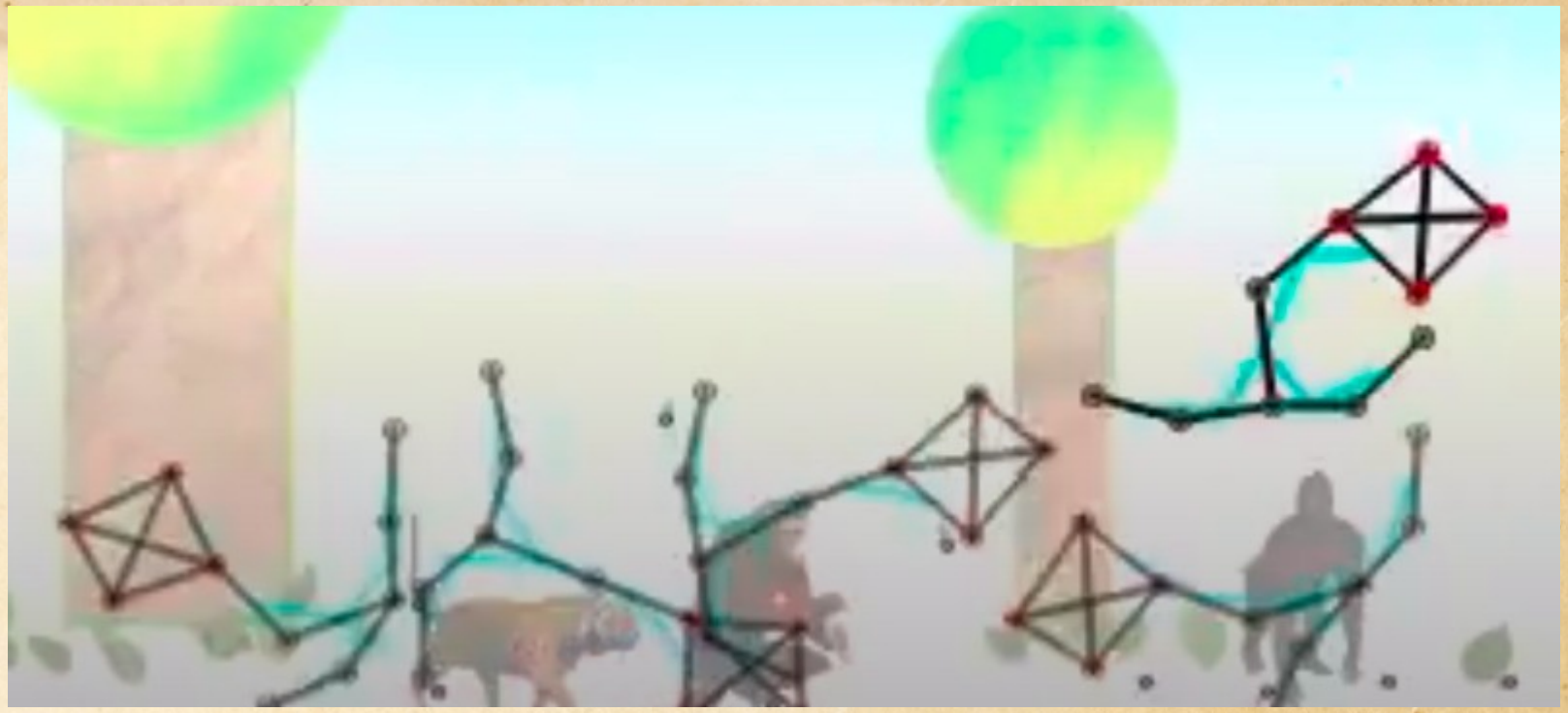
Algoritmen en Heuristieken

Simulation Hypothesis

Andere population based algoritmen

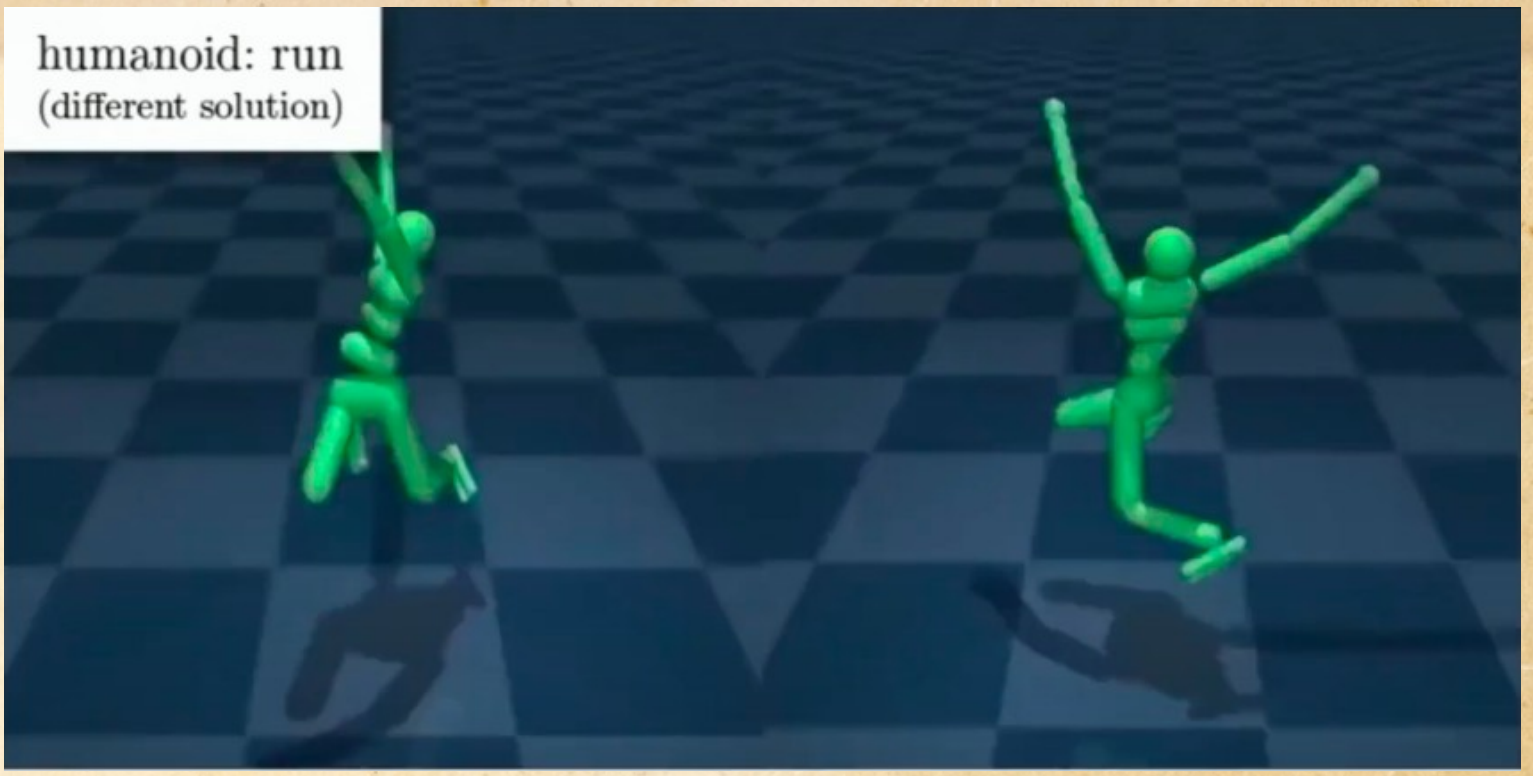
- Ant colony optimization
- Artificial immune system
- Bee colony optimization
- Brain storm optimization
- Fireworks algorithm
- Particle swarm optimization
- ...

Population based, Demo



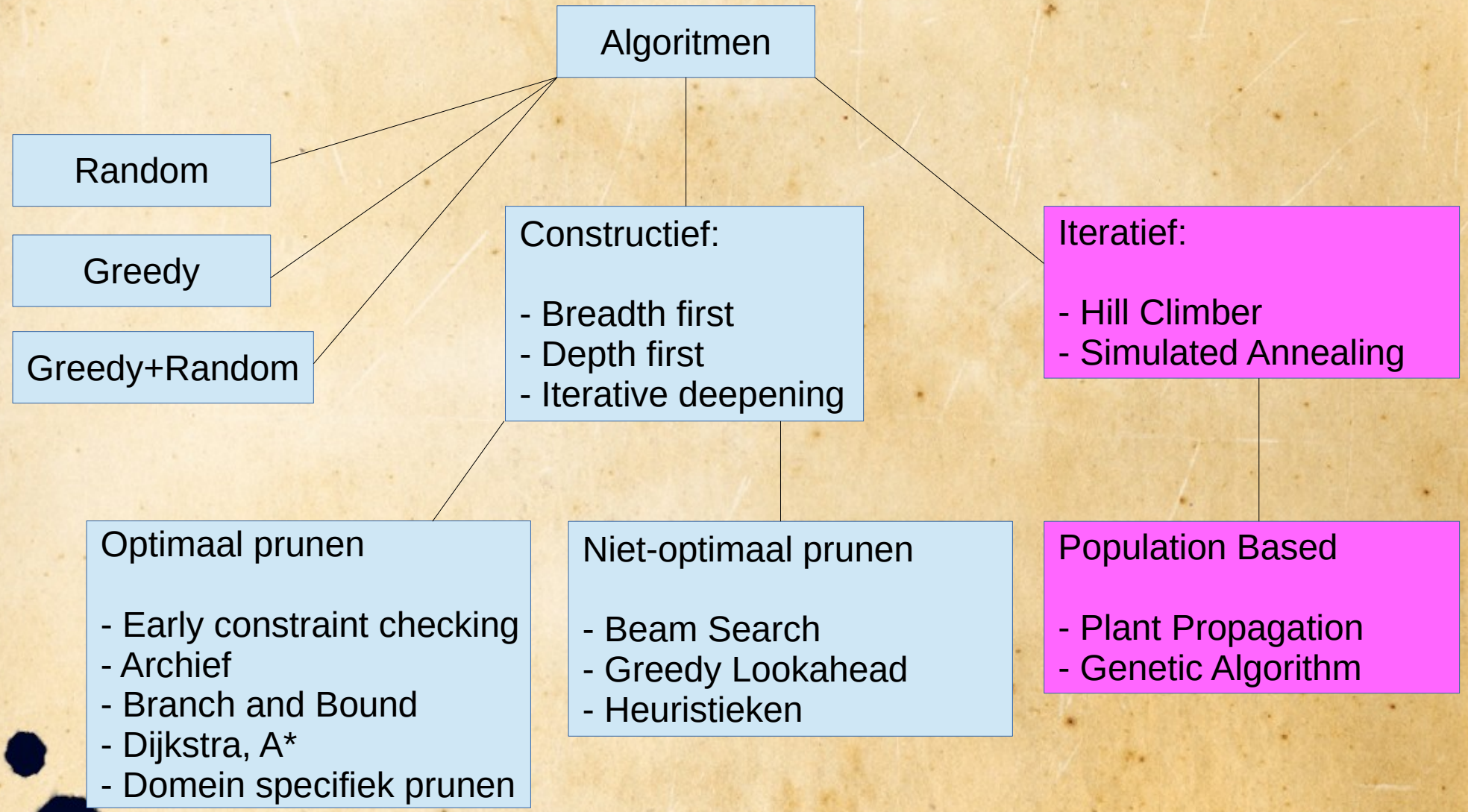
- Learn to walk, Roberto Mior (miorsoft.itch.io)
  - Youtube video:
    - https://youtu.be/qtmG8mMGbpo

Population based, solution?



- Deepmind Run, different solution:
  - https://youtu.be/eYAX4NwV428
    - Population restart
    - Reward shaping

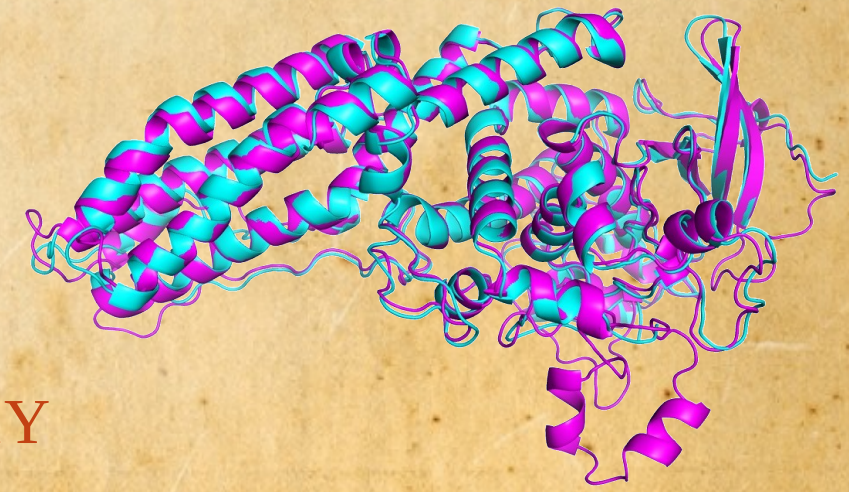
Algoritmen


Algoritmen
Random
Greedy
Greedy+Random
Constructief:
- Breadth first
- Depth first
- Iterative deepening
Iteratief:
- Hill Climber
- Simulated Annealing
Optimaal prunen
- Early constraint checking
- Archief
- Branch and Bound
- Dijkstra, A*
- Domein specifiek prunen
Niet-optimaal prunen
- Beam Search
- Greedy Lookahead
- Heuristieken
Population Based
- Plant Propagation
- Genetic Algorithm

State of the art: zelf-lerende algoritmen

- Bijvoorbeeld: Deepmind

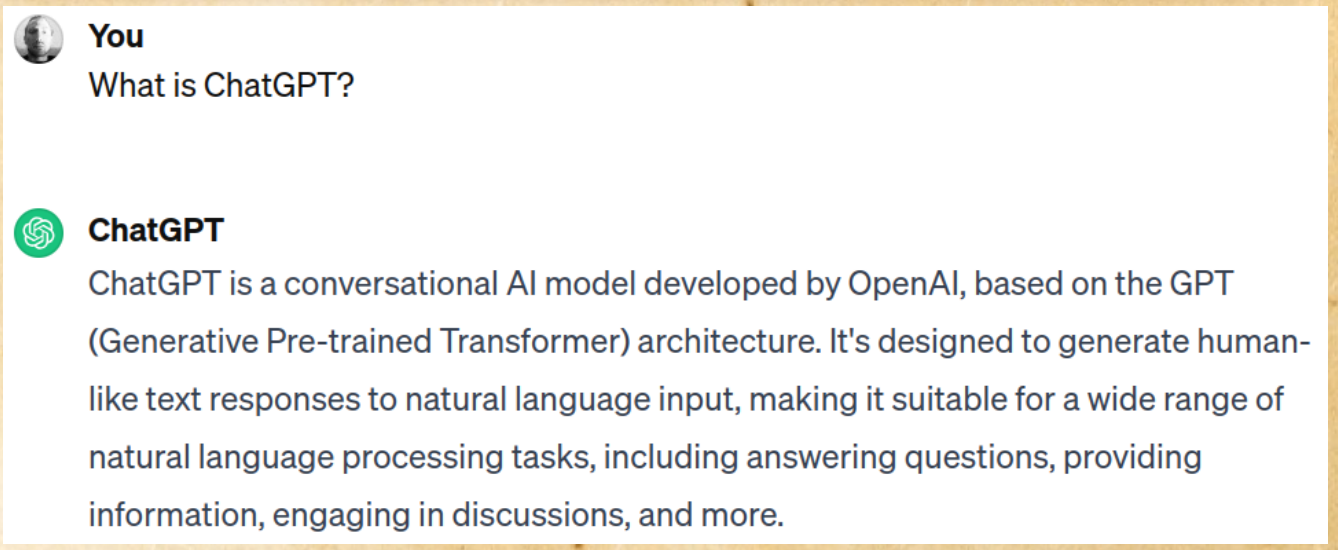
- AlphaGo - The Movie (1:30uur):
  https://youtu.be/WXuK6gekU1Y



- AlphaFold (0:07uur):
  https://youtu.be/WXuK6gekU1Y

State of the art: zelf-lerende algoritmen

- ChatGTP
https://chat.openai.com/



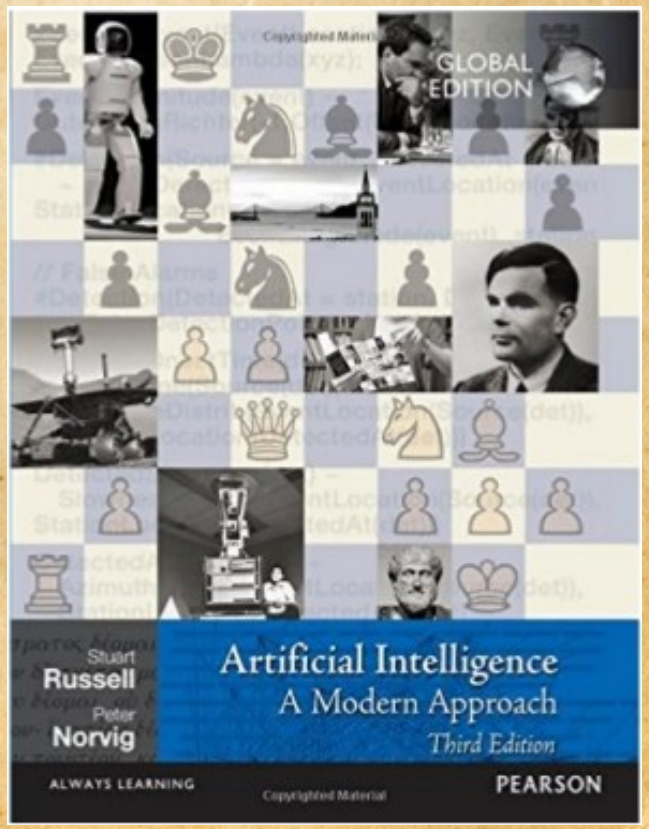
- Waymo
Self Driving Taxis (San Francisco)
https://youtu.be/Y8qfHlpe31k

Literatuur

Artificial Intelligence: A Modern Approach

Third Edition

Stuart J. Russell and Peter Norvig



Hfdstk 3.4 Breadth first, Depth first
Hfdstk 3.5 A*
Hfdstk 4.1 **Hill Climber, Simulated Annealing**
Hfdstk 6.3 Most Constraint Variable heuristic

Lecture video 2020, terugkijken

- Youtube playlist, Iteratieve algoritmen:
  - https://www.youtube.com/playlist?list=PLJBtJTYGPSzJaxroYW-6OH1NRuUFqpGER

Tips

- Eerst de basis algoritmen, daarna pas creatieve uitbreiding
- Houd je code zo simpel mogelijk
  - Johan Cruijff: “Voetbal is simpel. Wat moeilijk is, is simpel voetballen.”
- Maak een klein probleempje om mee te testen
- Los je case met de hand op voor goede heurstieken ideeën
- Doe leuke onderzoekjes, experimenteer
  - welke heuristiek/parameter-waarde werkt het best?
- Future work, wat zou ik gaan doen in vervolgonderzoek?