



Iteratieve Algoritmen

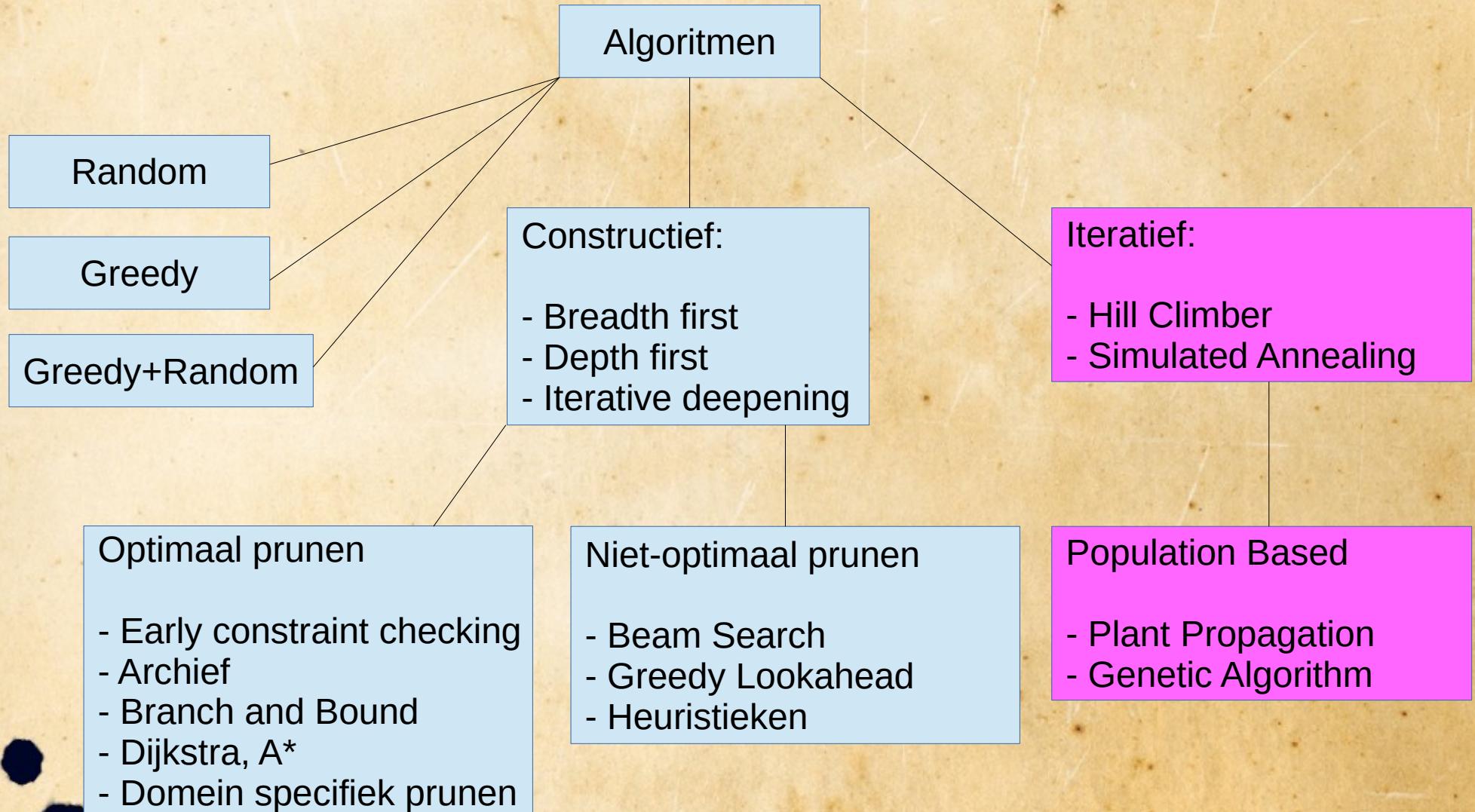
Bas Terwijn <b.terwijn@uva.nl>

Algoritmen en Heuristieken

Minor Programmeren

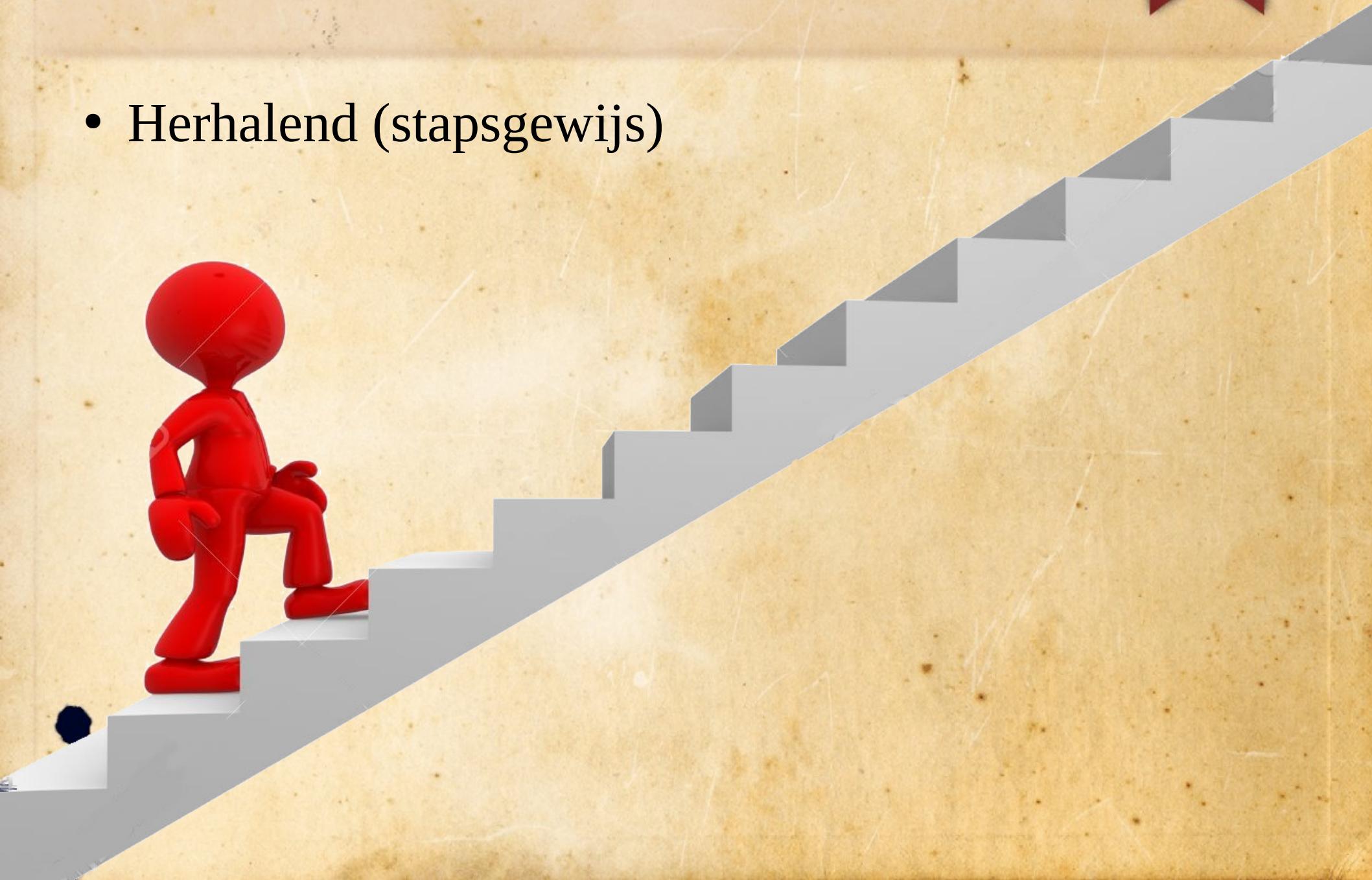
Universiteit van Amsterdam

Algoritmen



Iteratief

- Herhalend (stapsgewijs)



Hill Climber, pseudo code

Kies een random start state

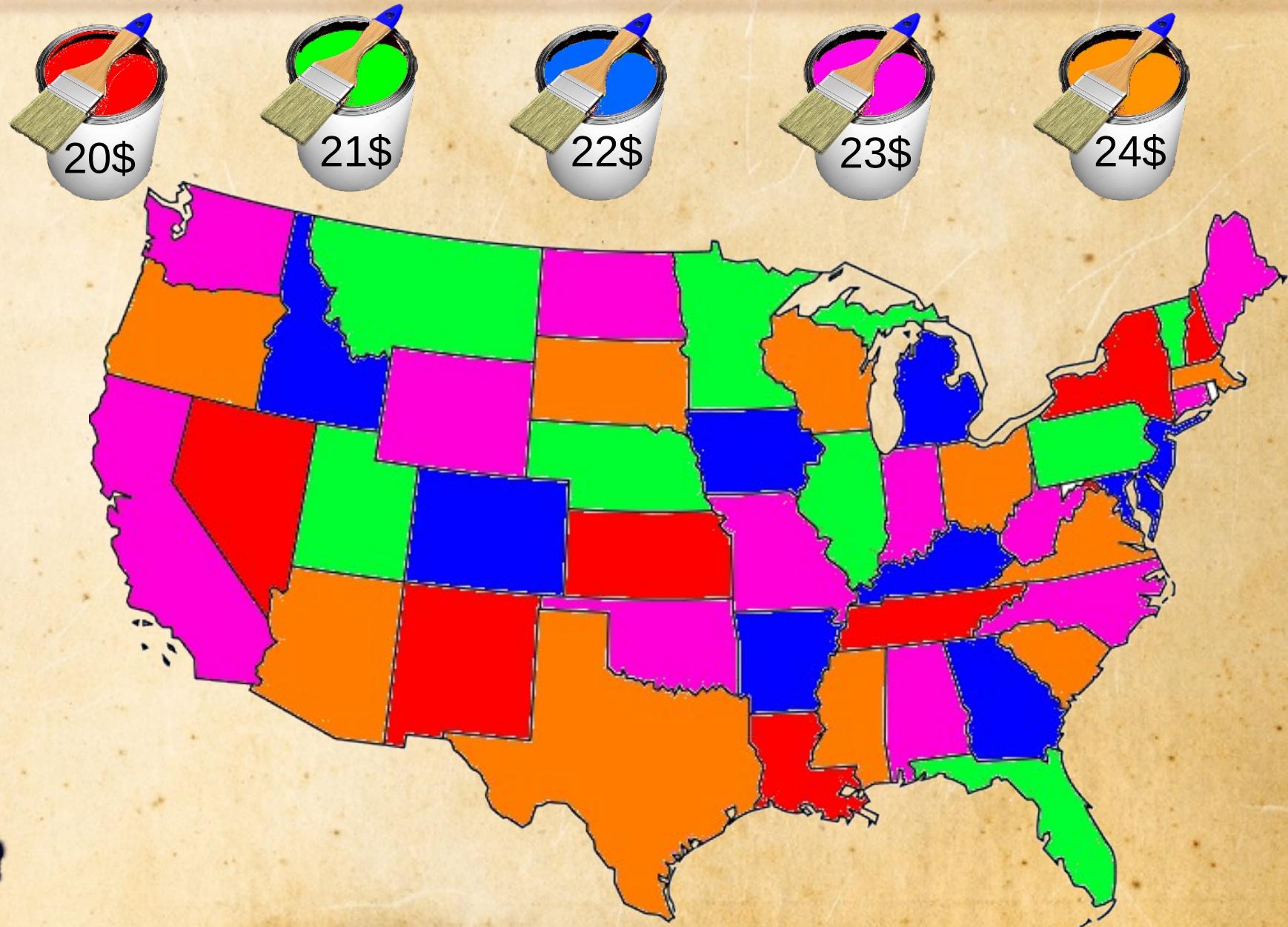
Herhaal:

Doe een kleine random aanpassing

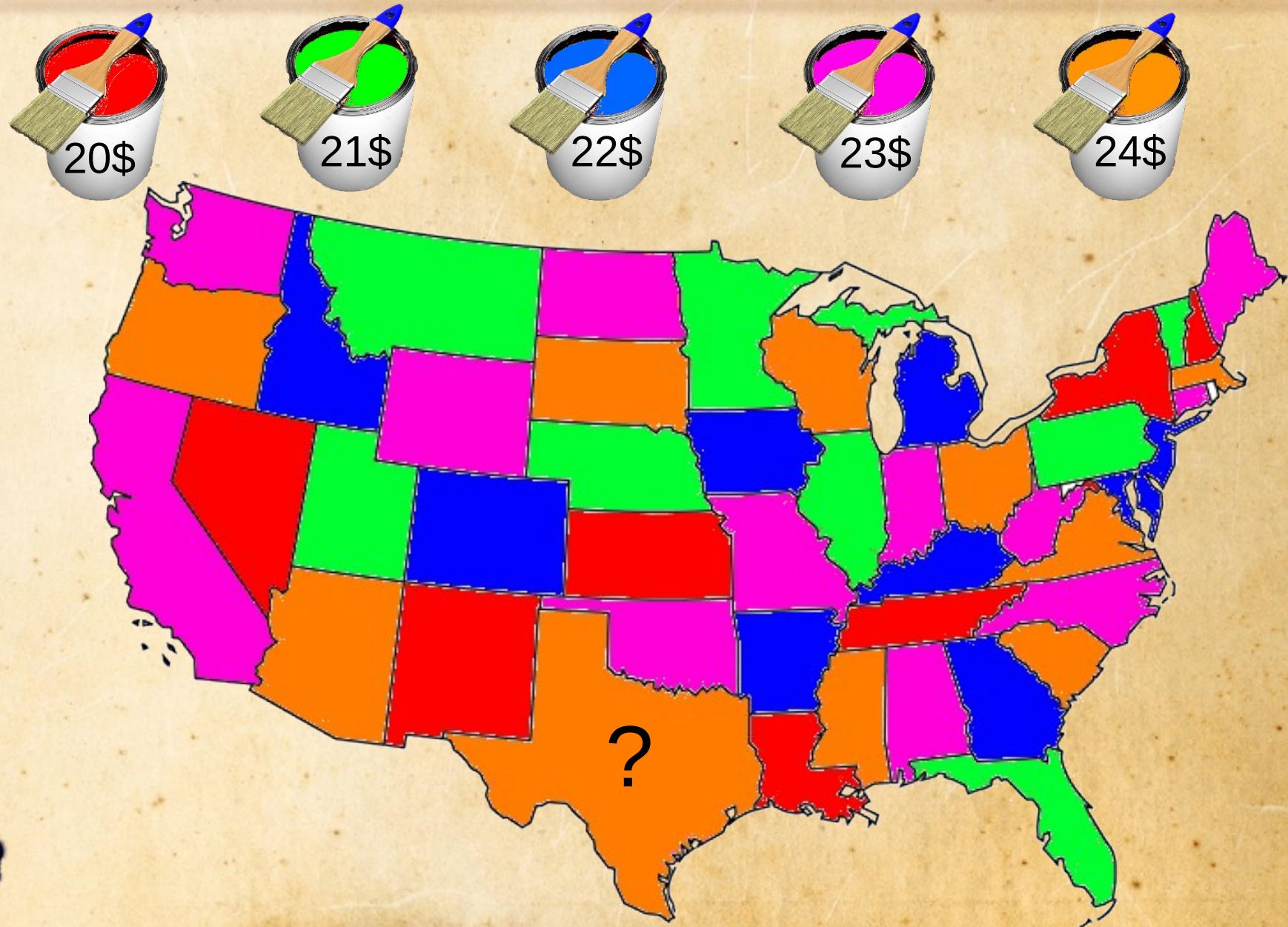
Als de state is verslechterd:

Maak de aanpassing ongedaan

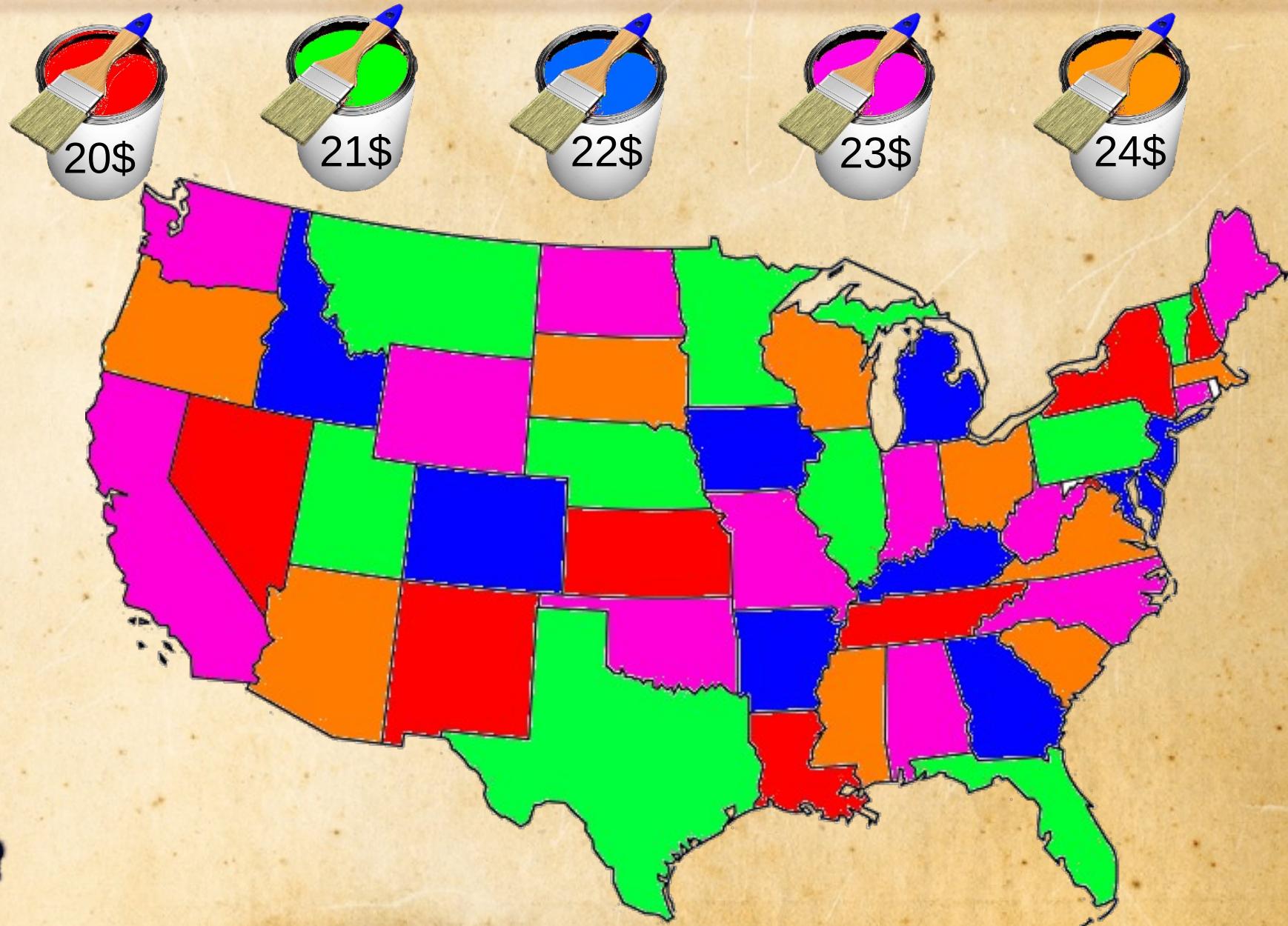
Hill Climber



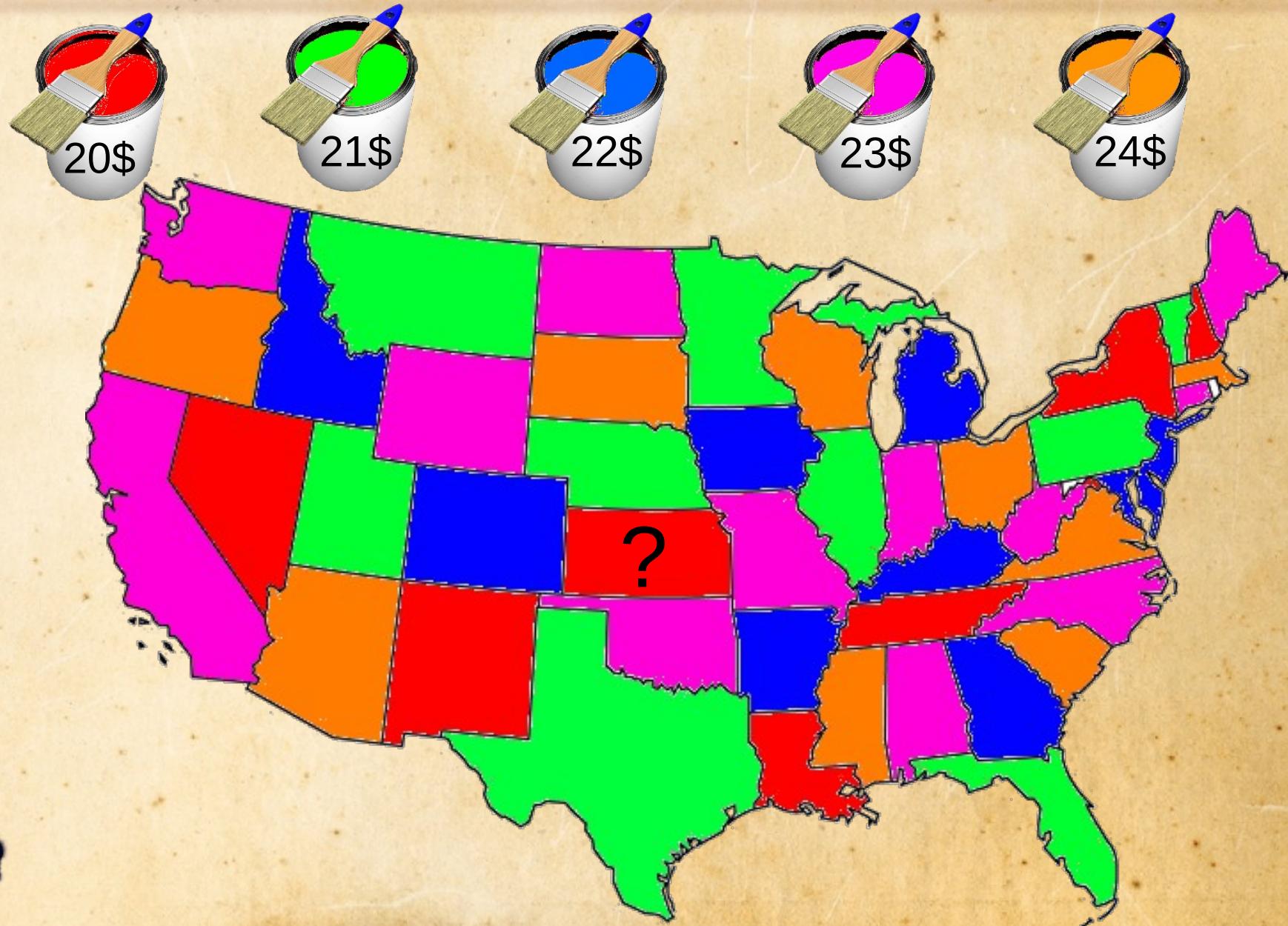
Hill Climber



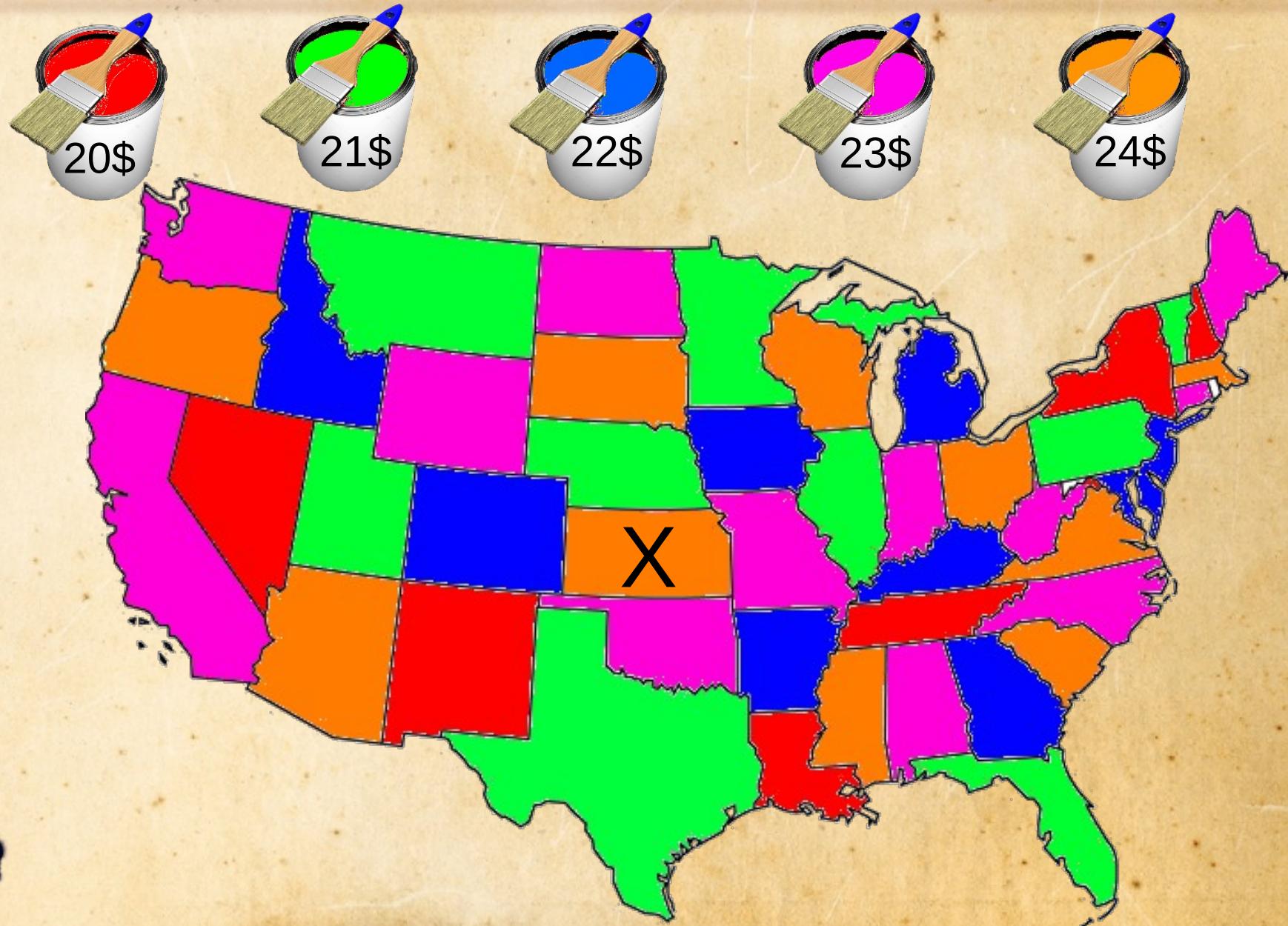
Hill Climber



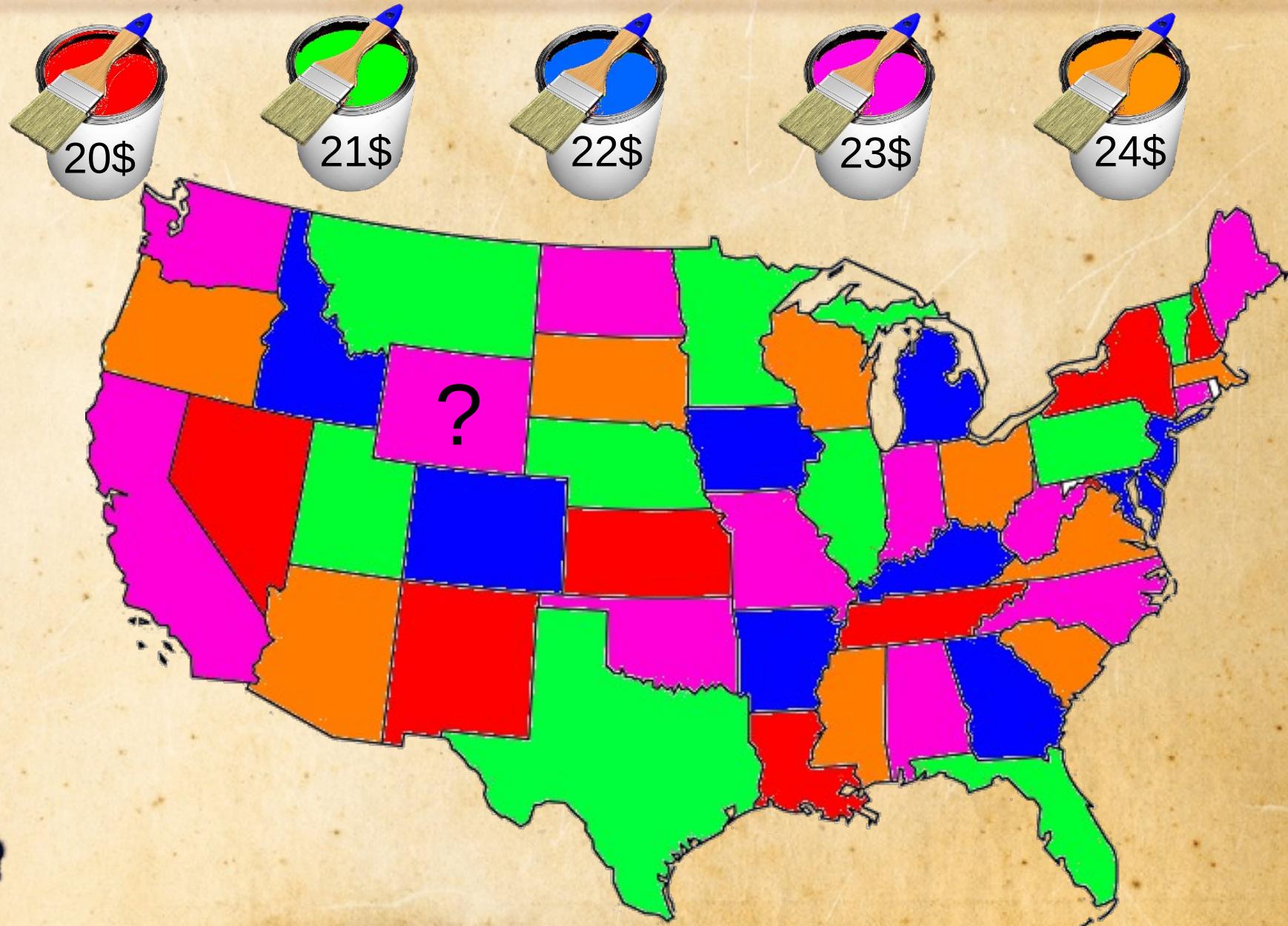
Hill Climber



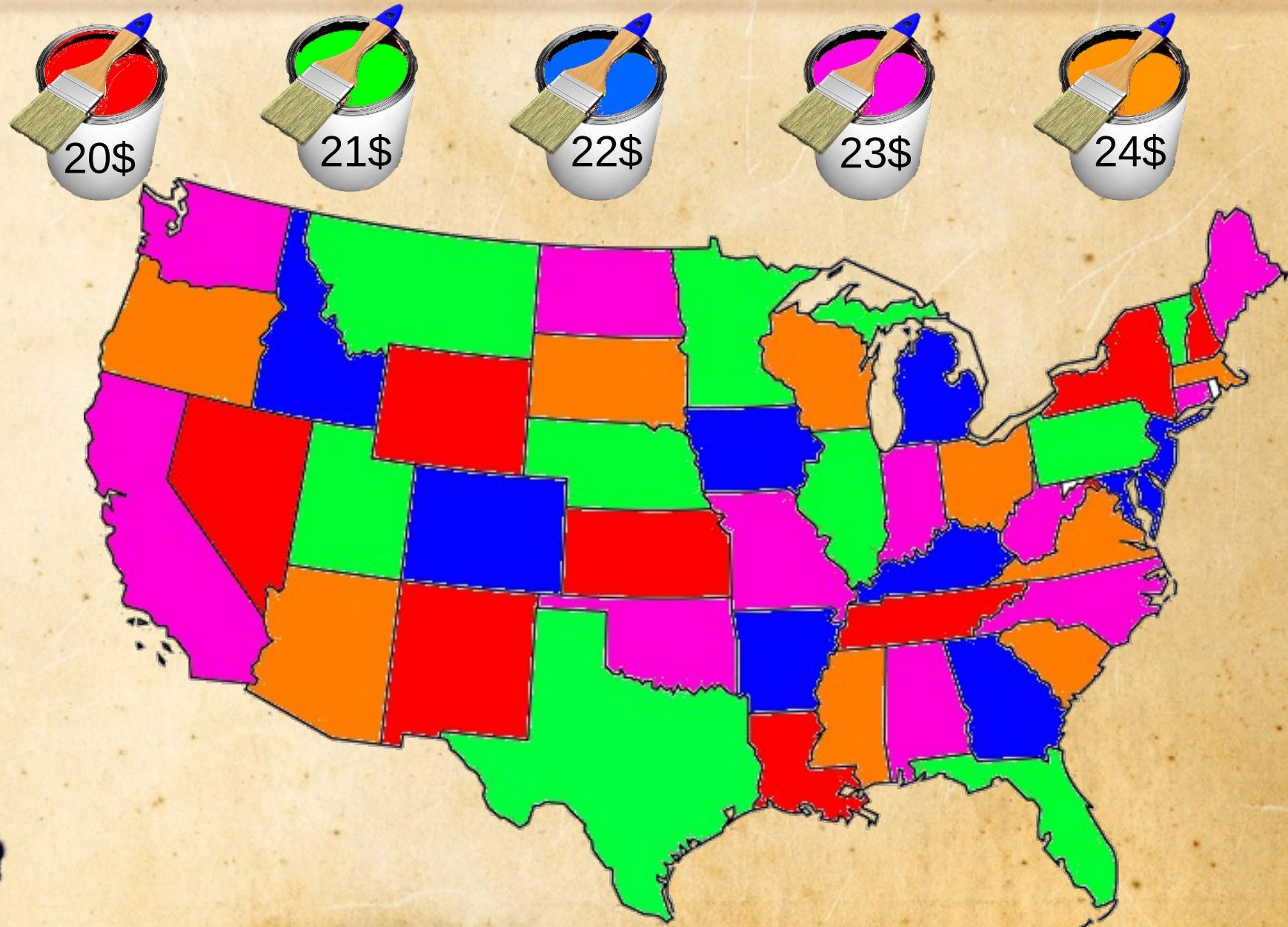
Hill Climber



Hill Climber

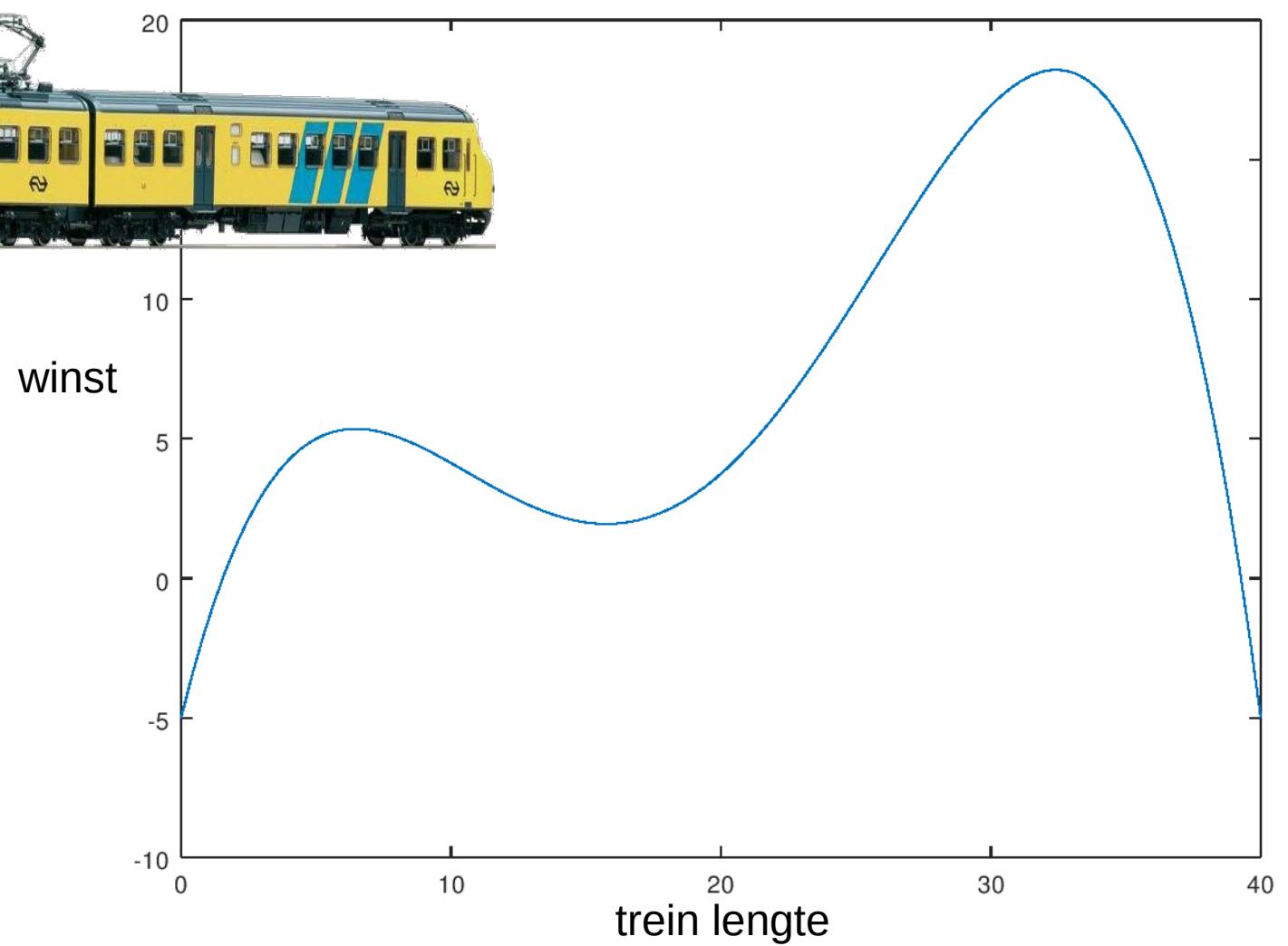


Hill Climber



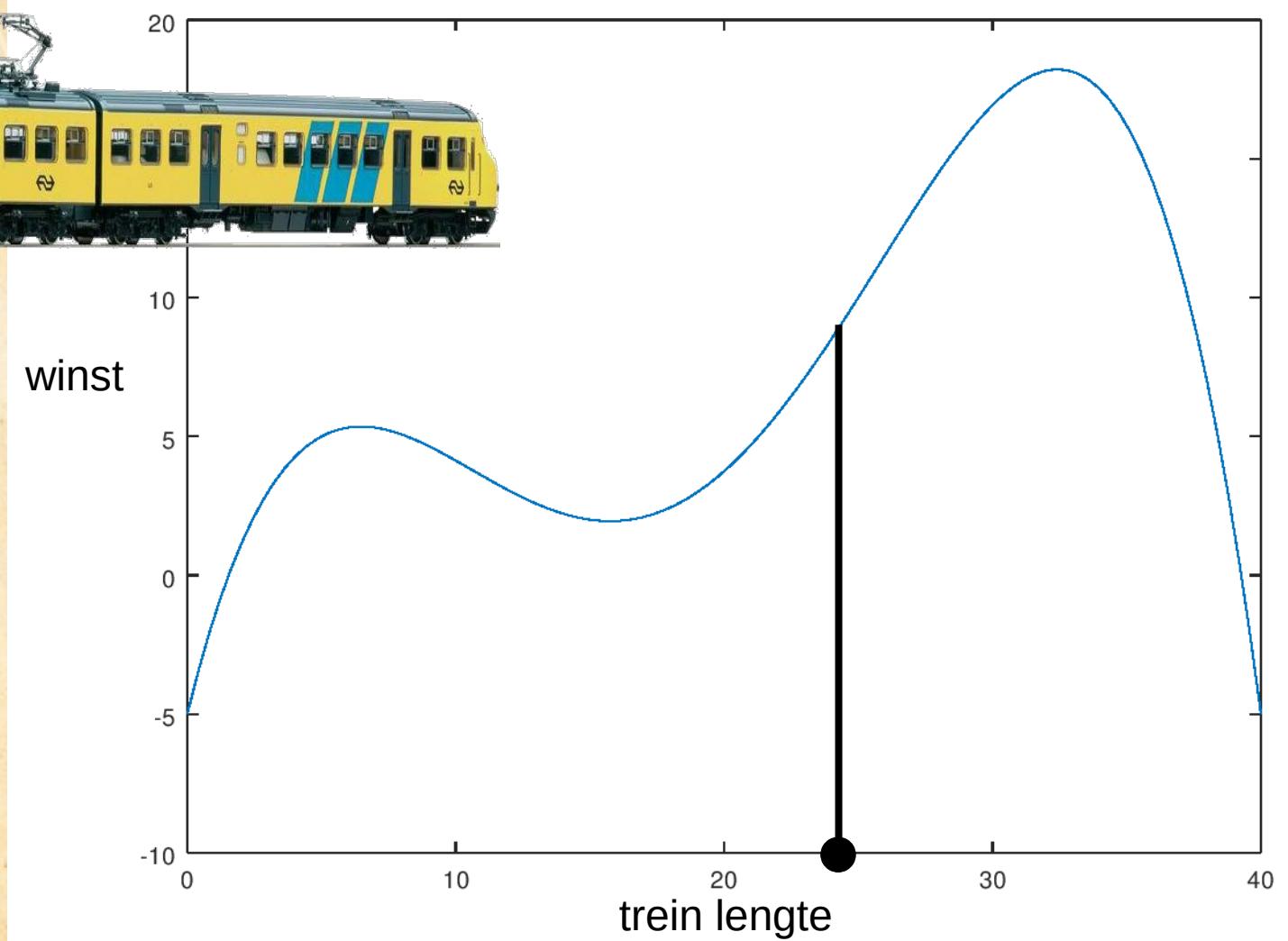
Hill Climber

- 1 dimensionaal trein probleem



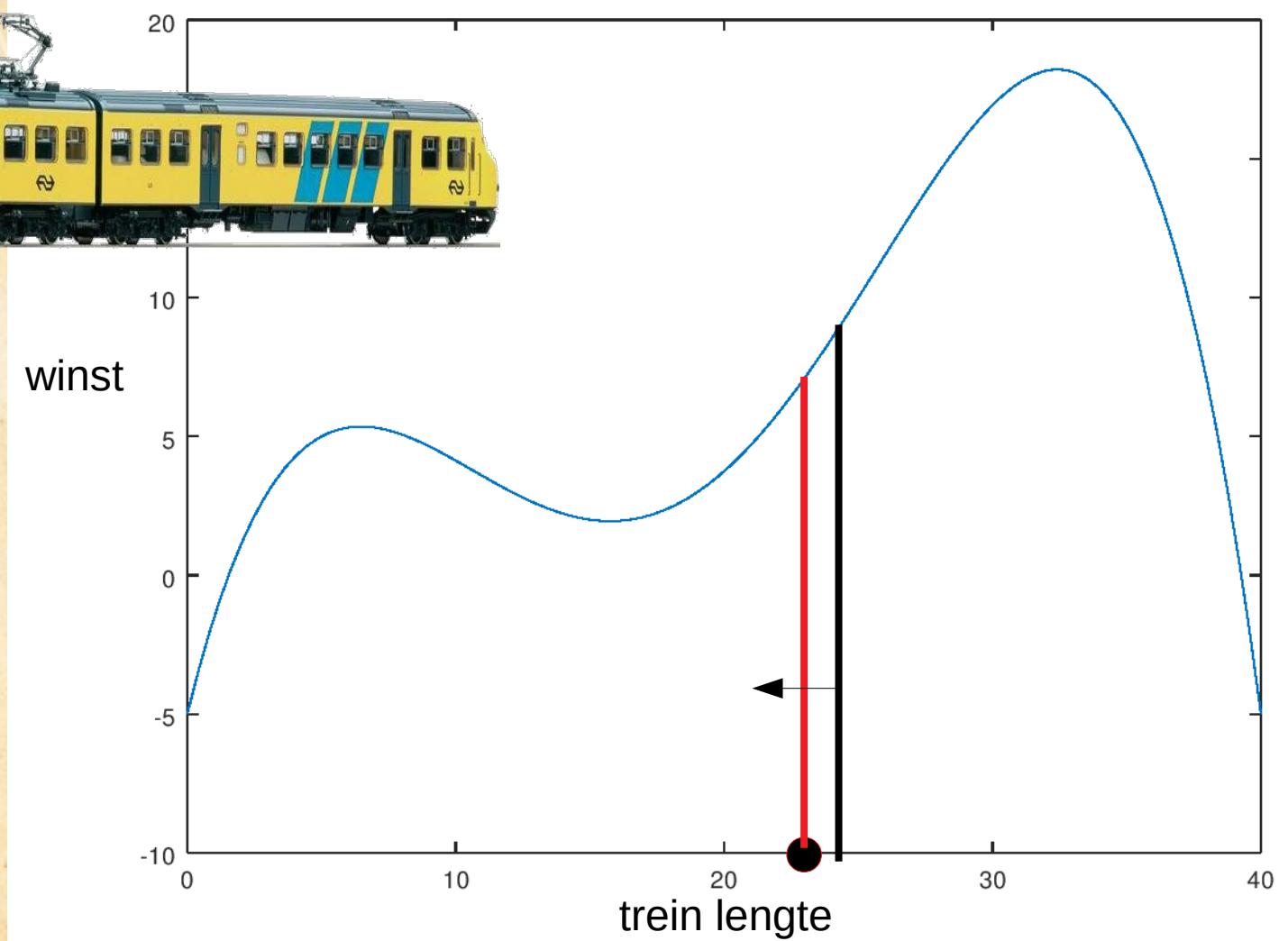
Hill Climber

- 1 dimensionaal trein probleem



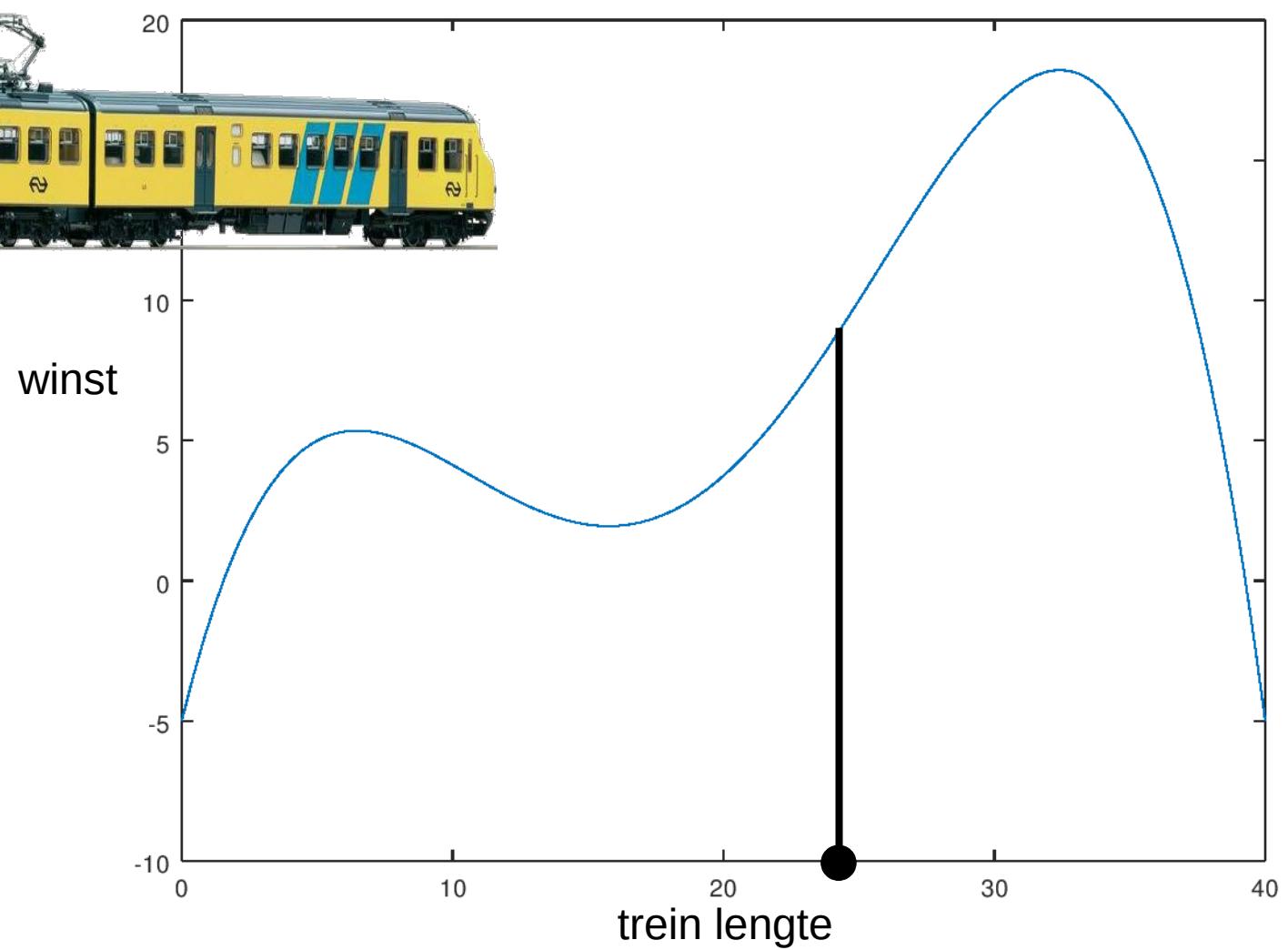
Hill Climber

- 1 dimensionaal trein probleem



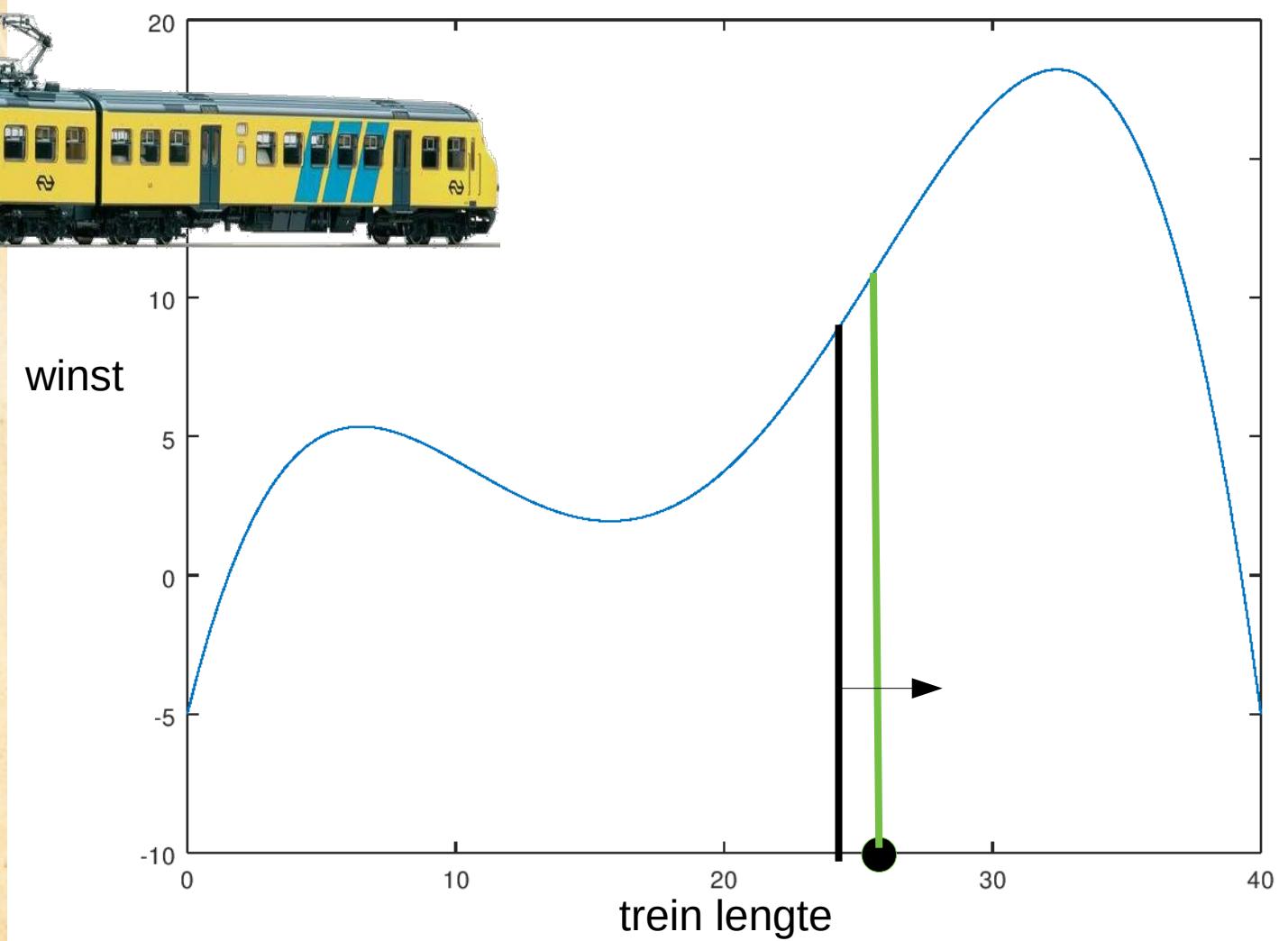
Hill Climber

- 1 dimensionaal trein probleem



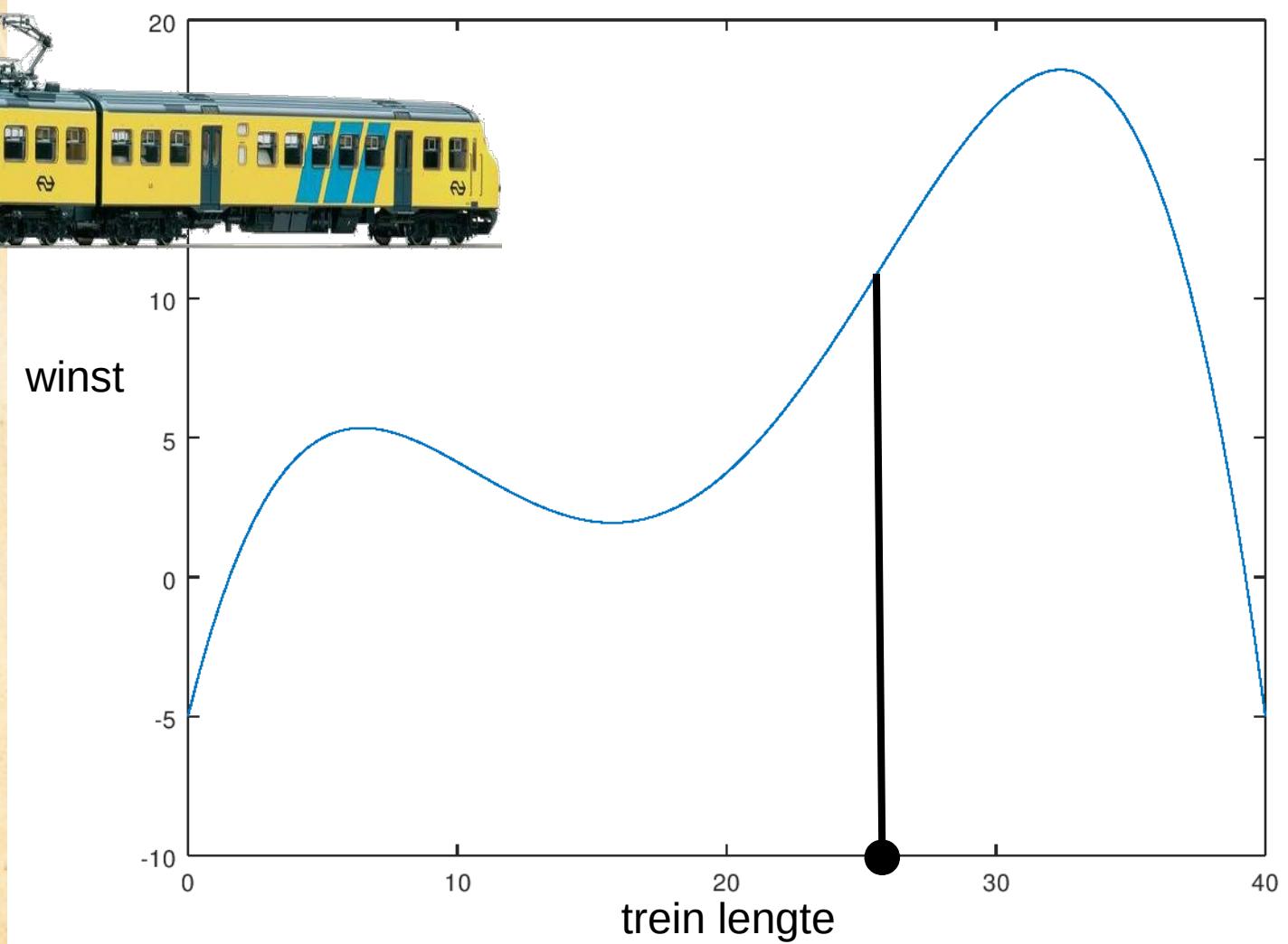
Hill Climber

- 1 dimensionaal trein probleem



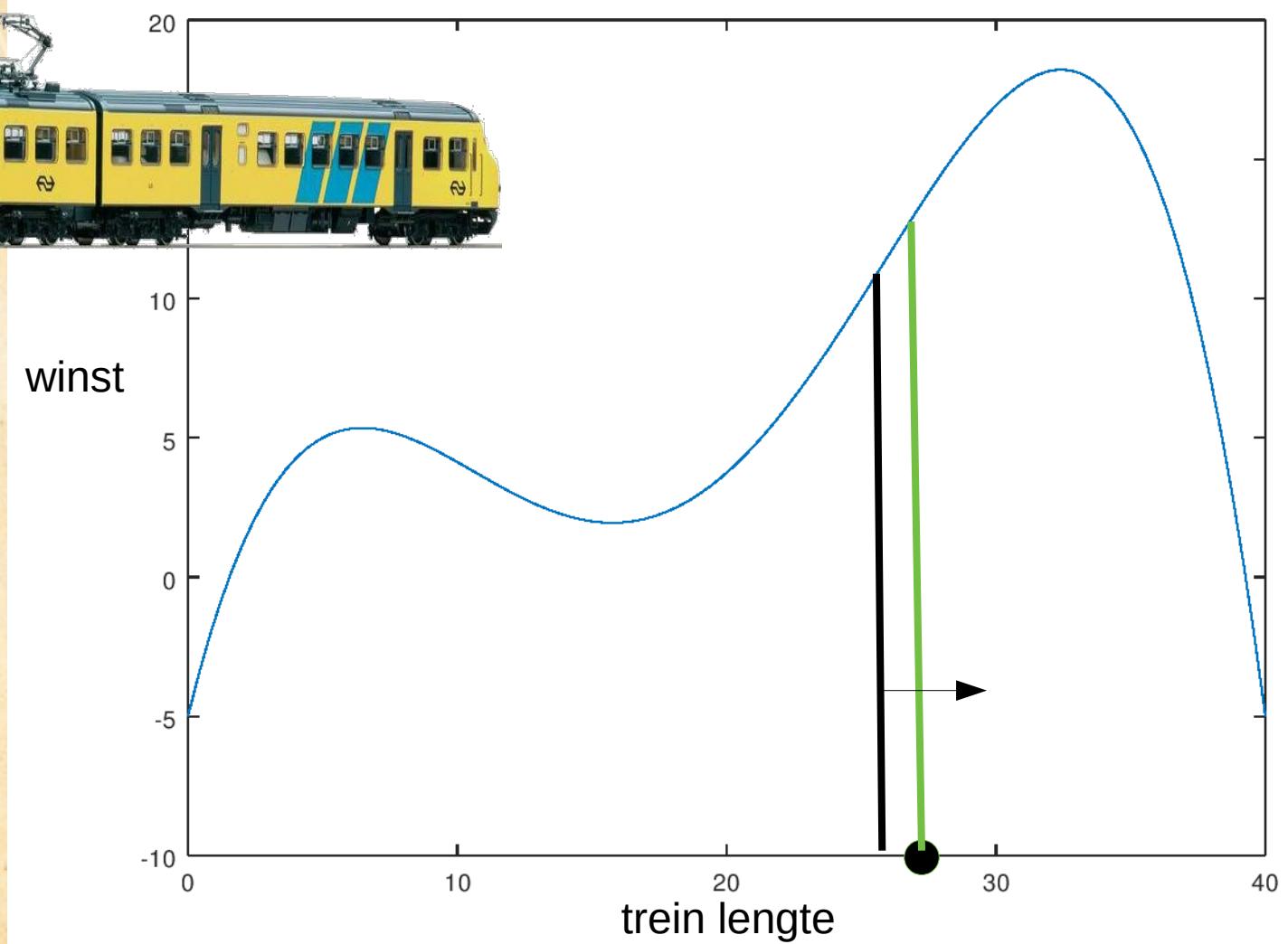
Hill Climber

- 1 dimensionaal trein probleem



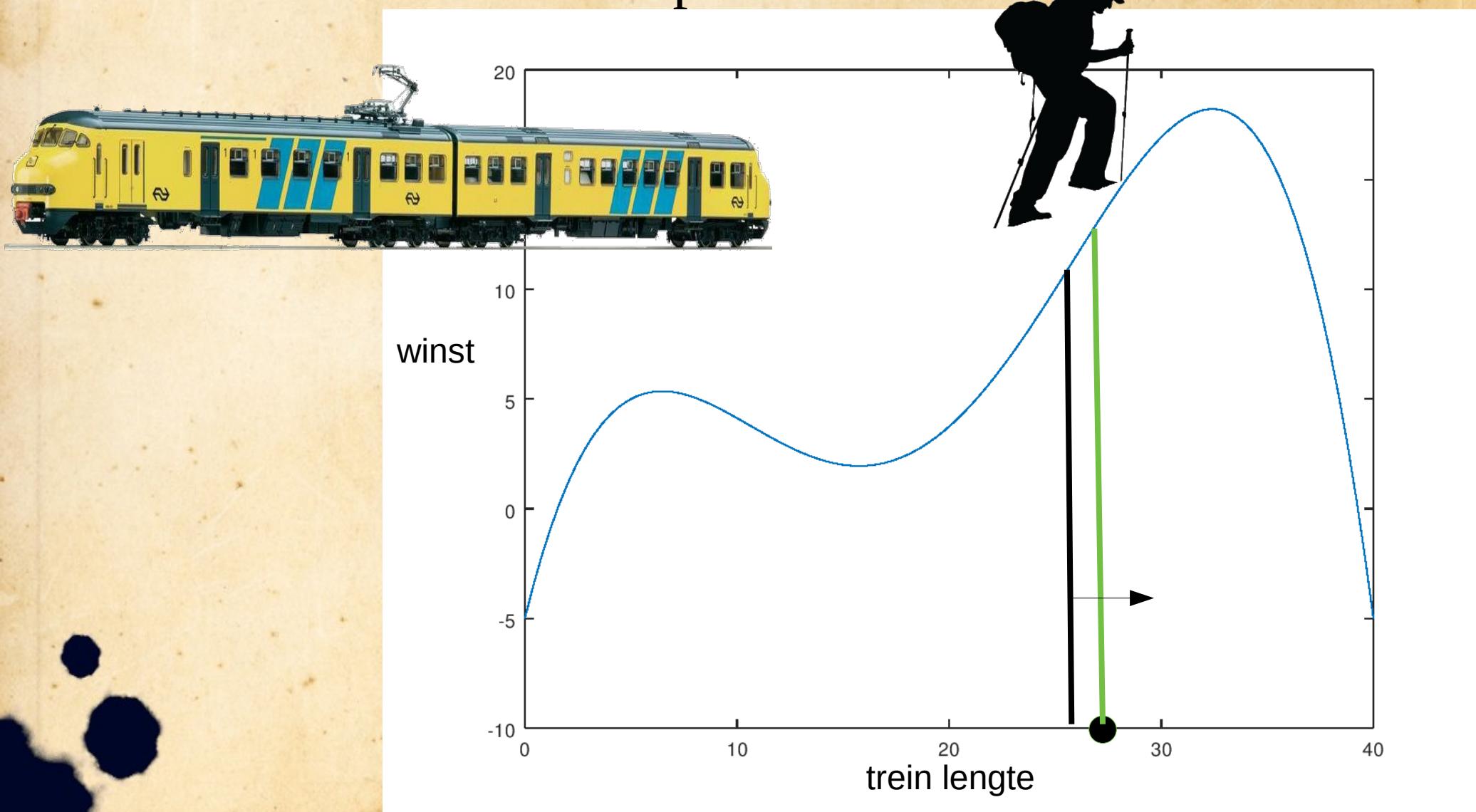
Hill Climber

- 1 dimensionaal trein probleem

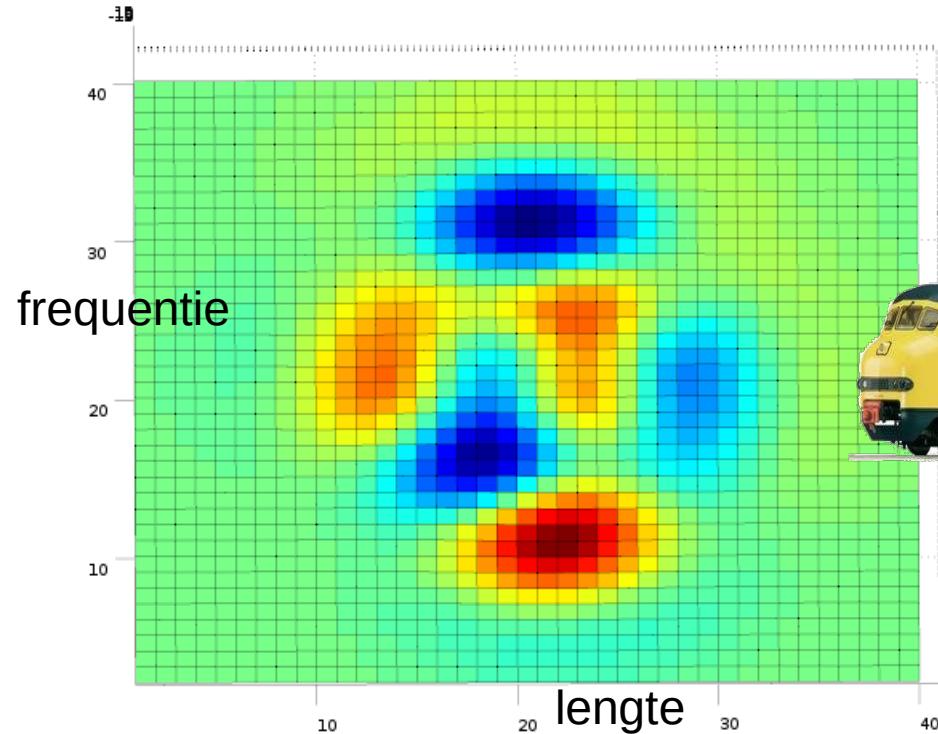


Hill Climber

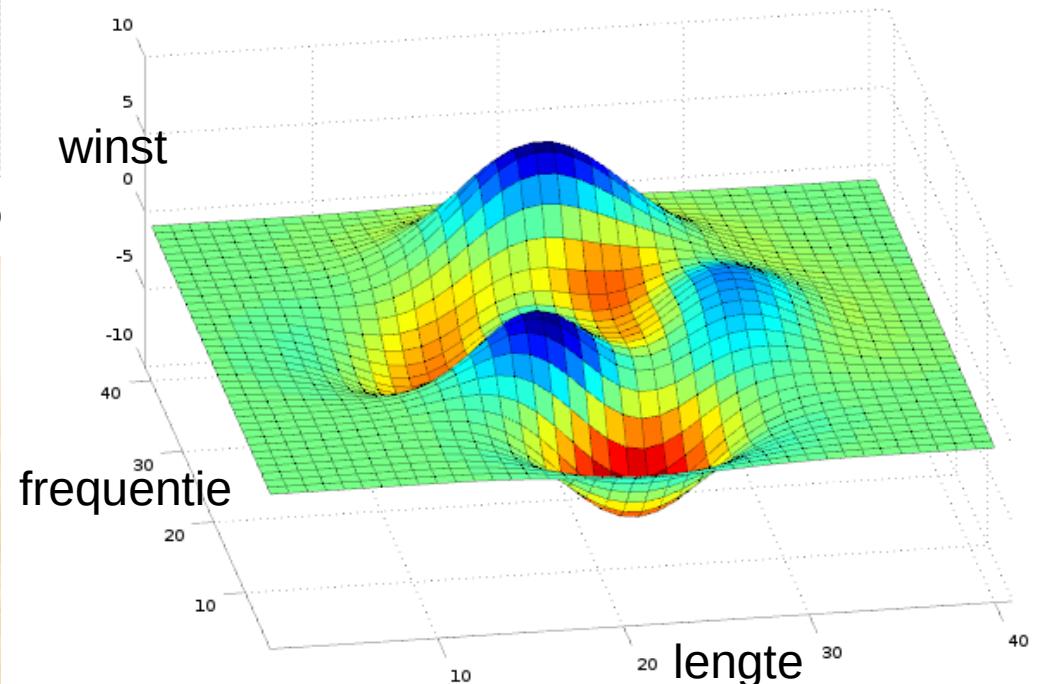
- 1 dimensionaal trein probleem



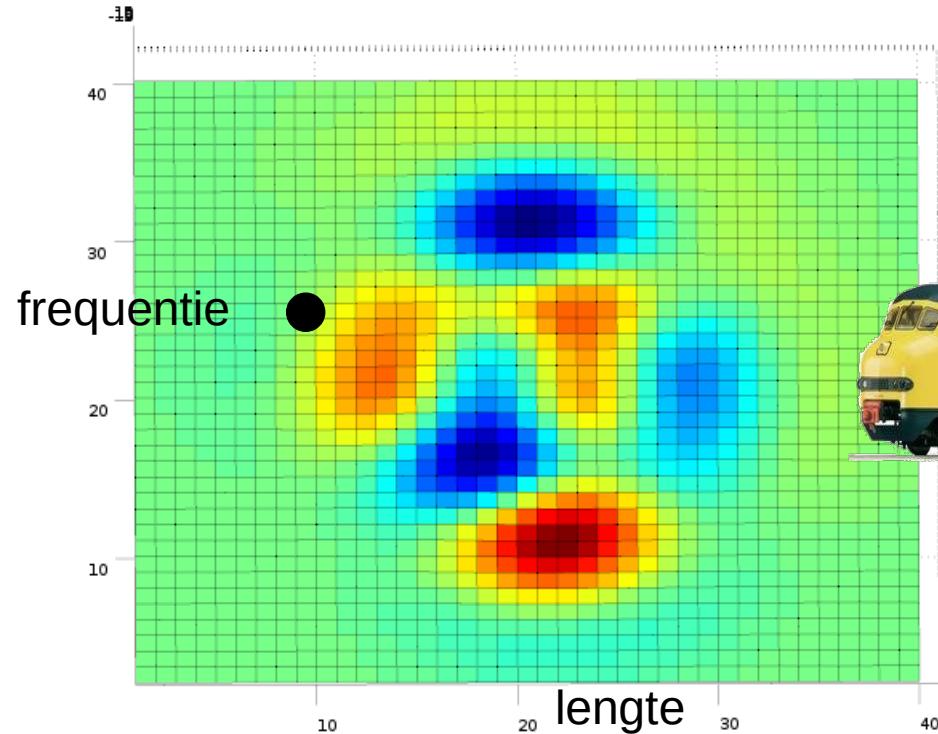
Hill Climber



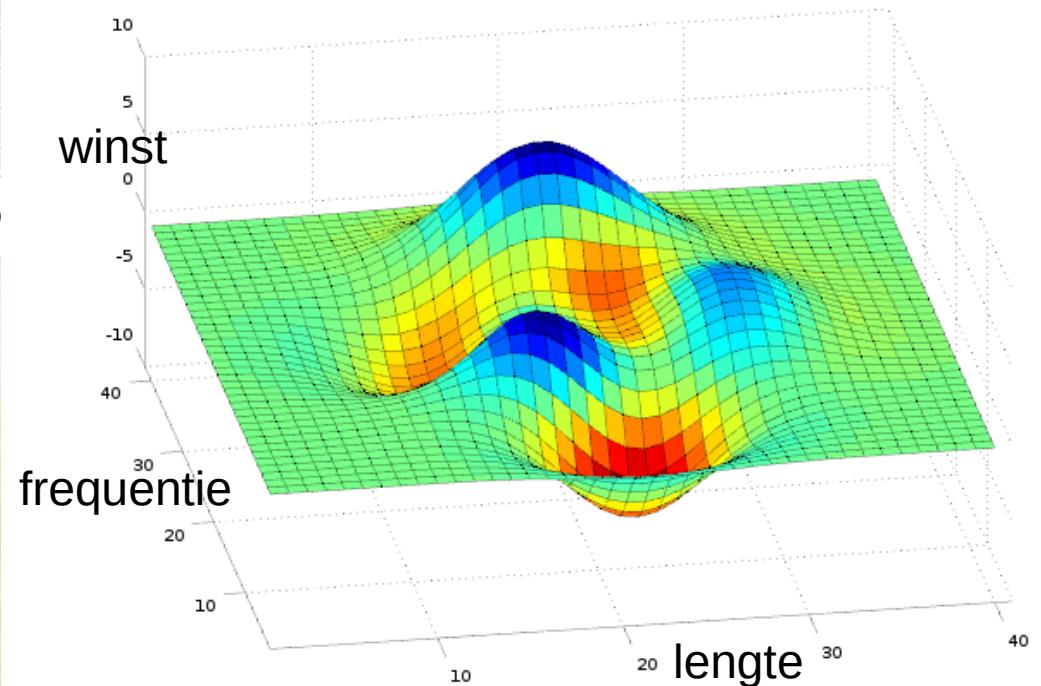
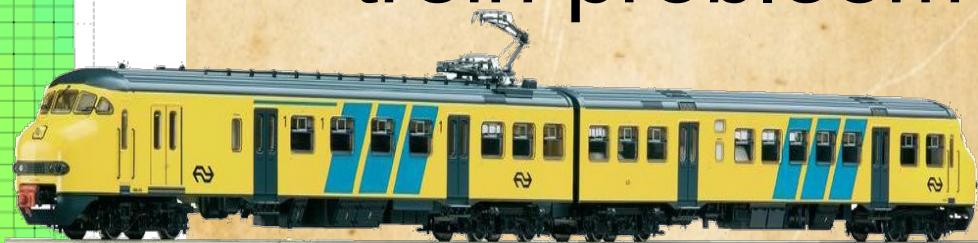
- 2 dimensionaal trein probleem



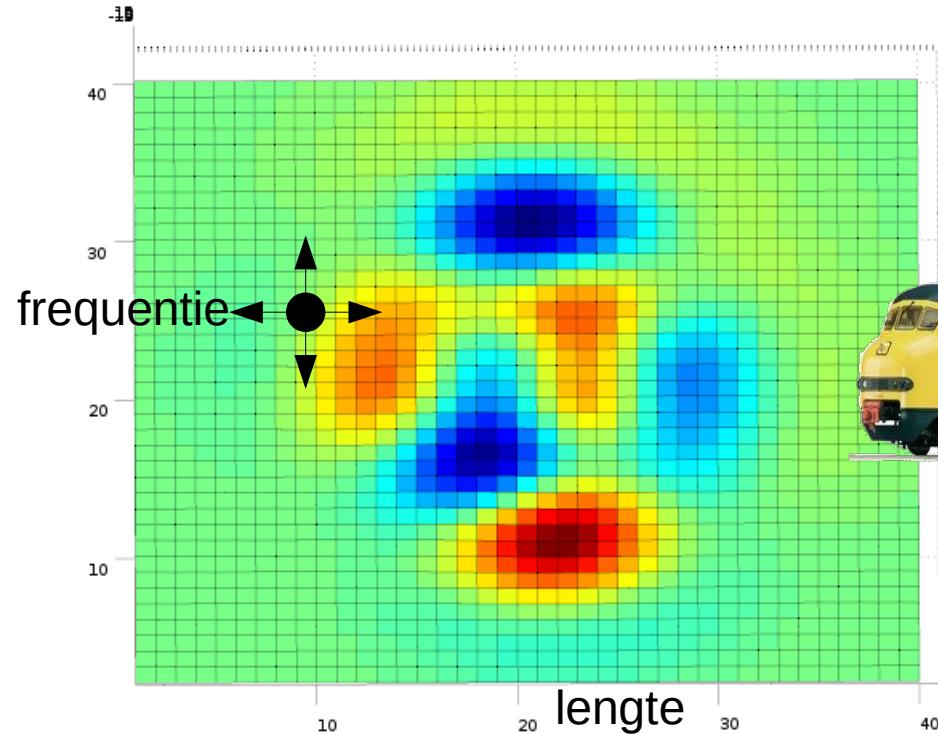
Hill Climber



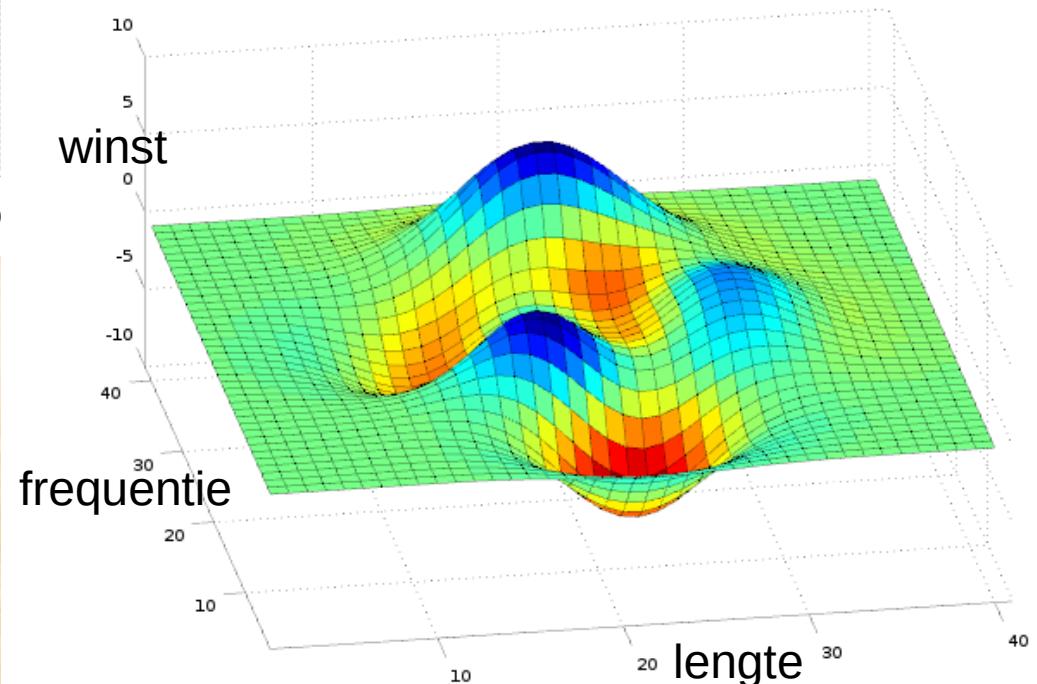
- 2 dimensionaal trein probleem



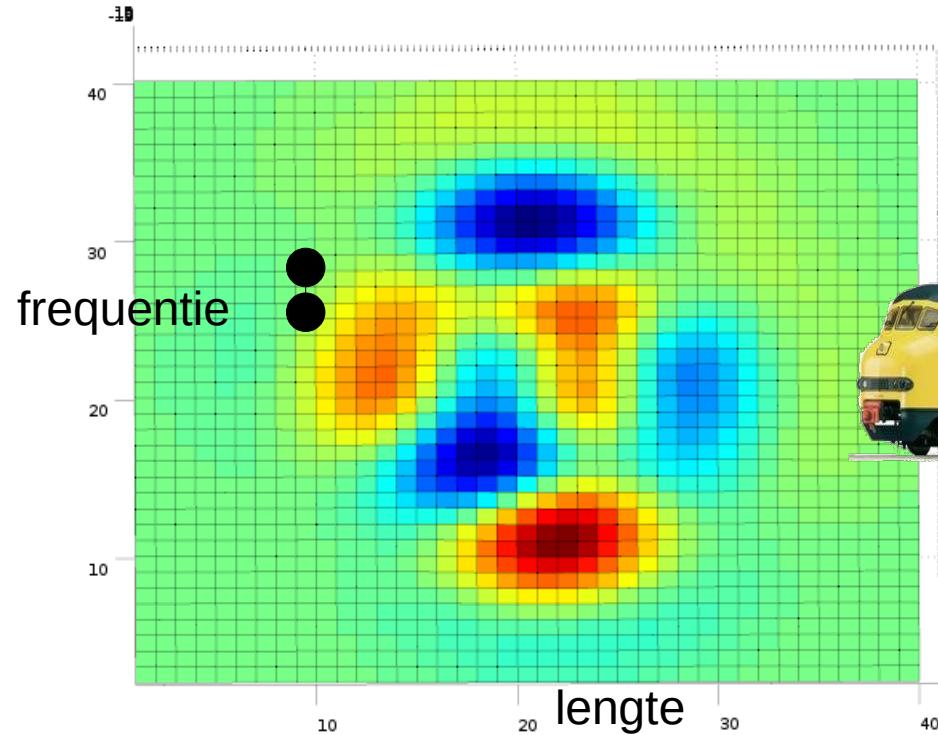
Hill Climber



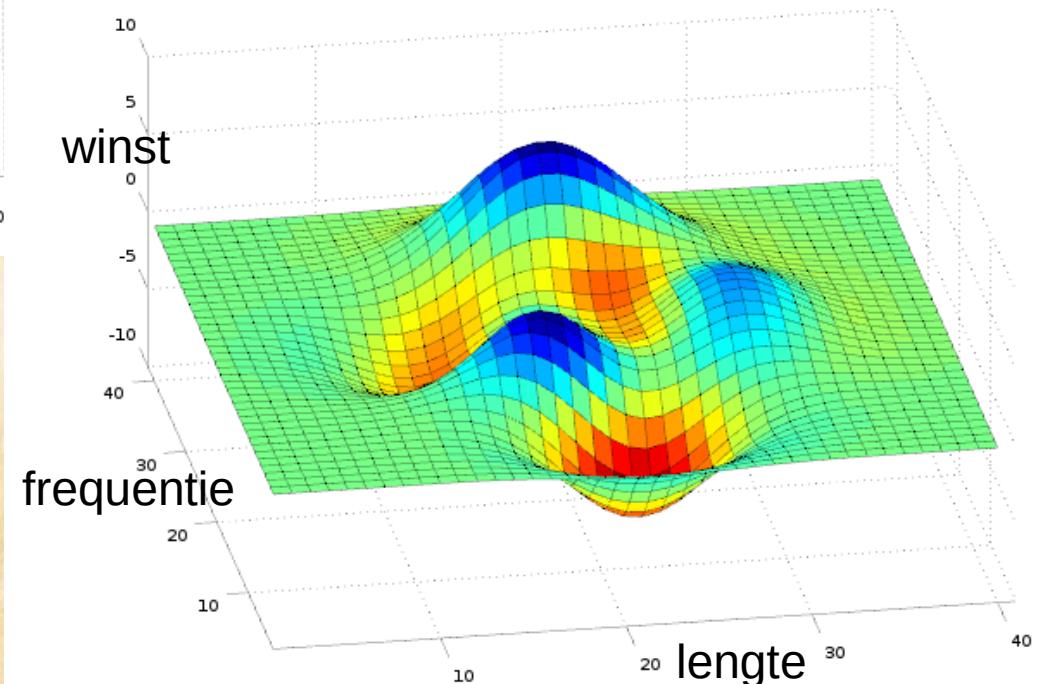
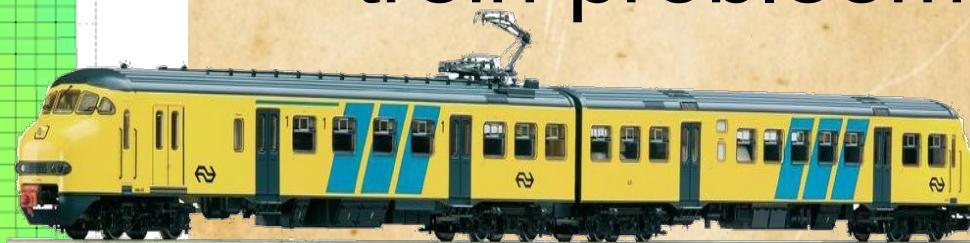
- 2 dimensionaal trein probleem



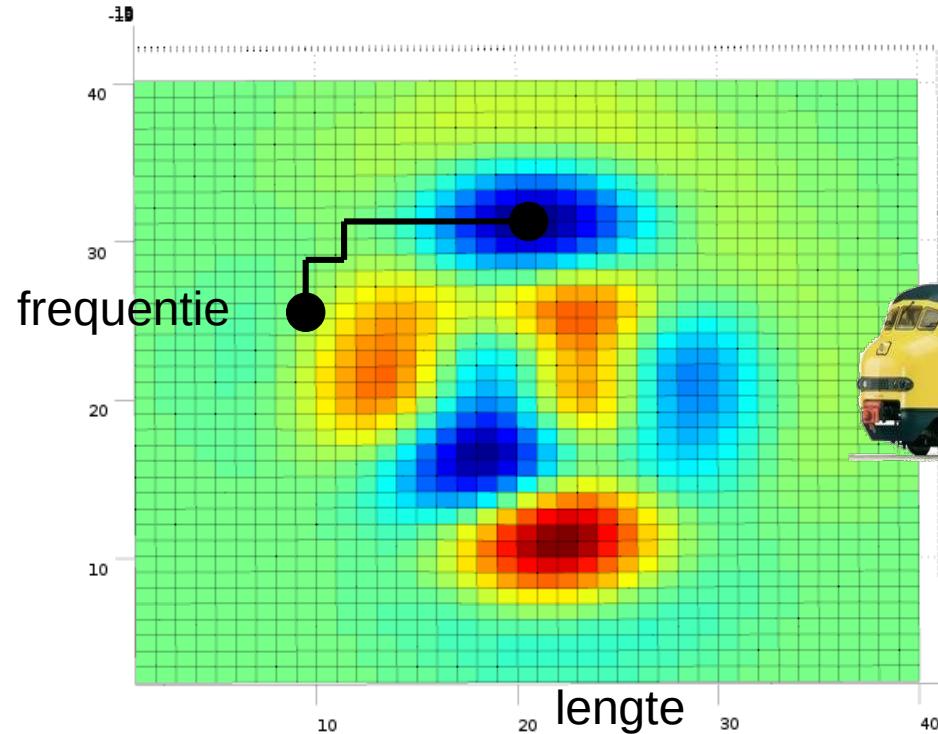
Hill Climber



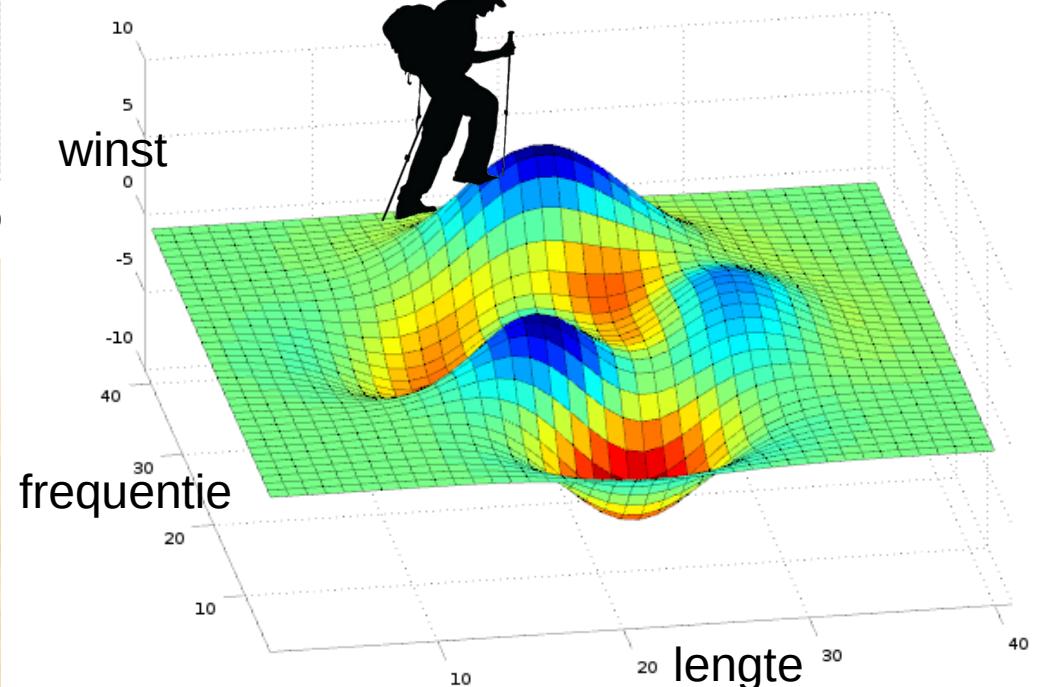
- 2 dimensionaal trein probleem



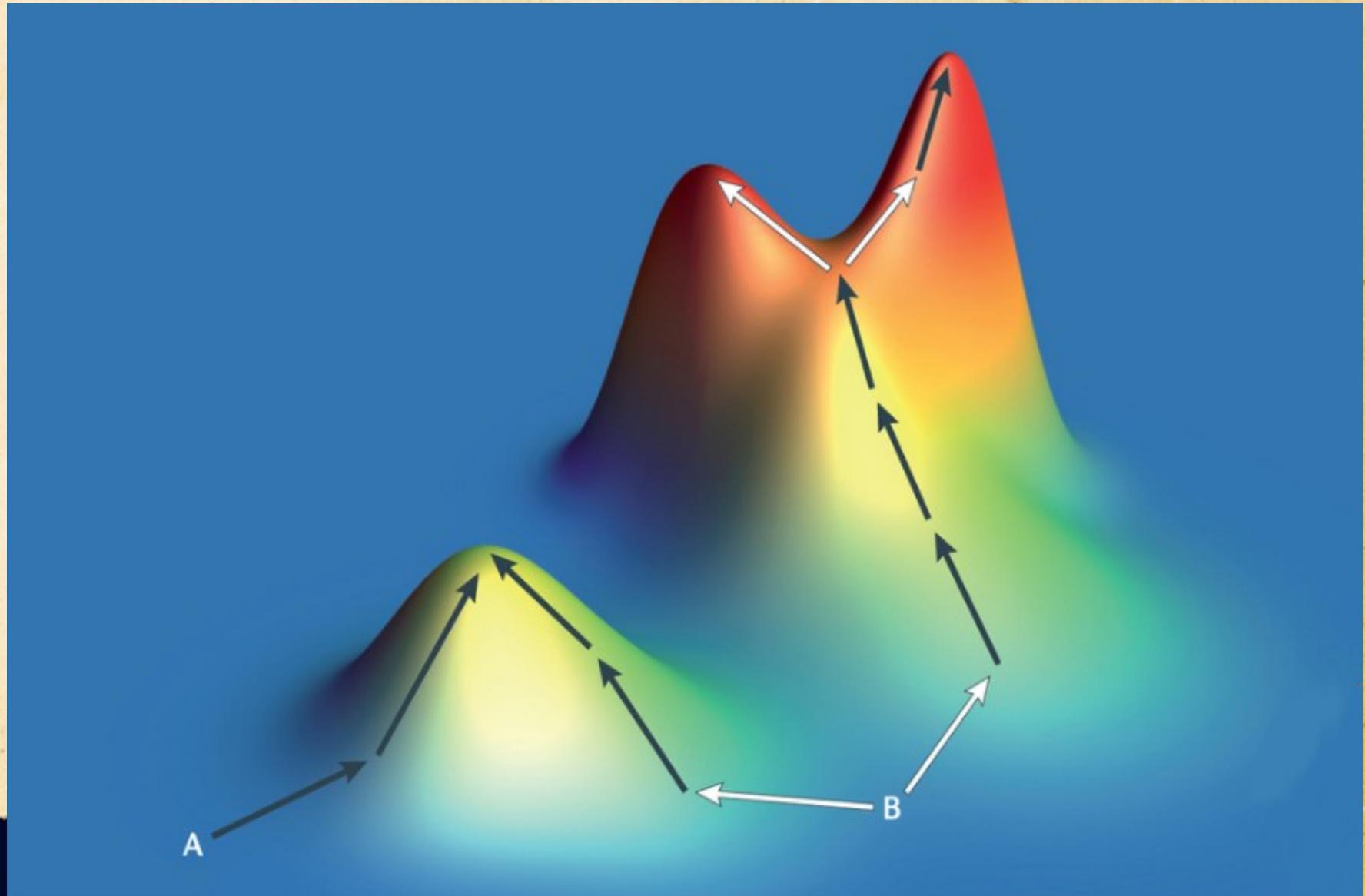
Hill Climber



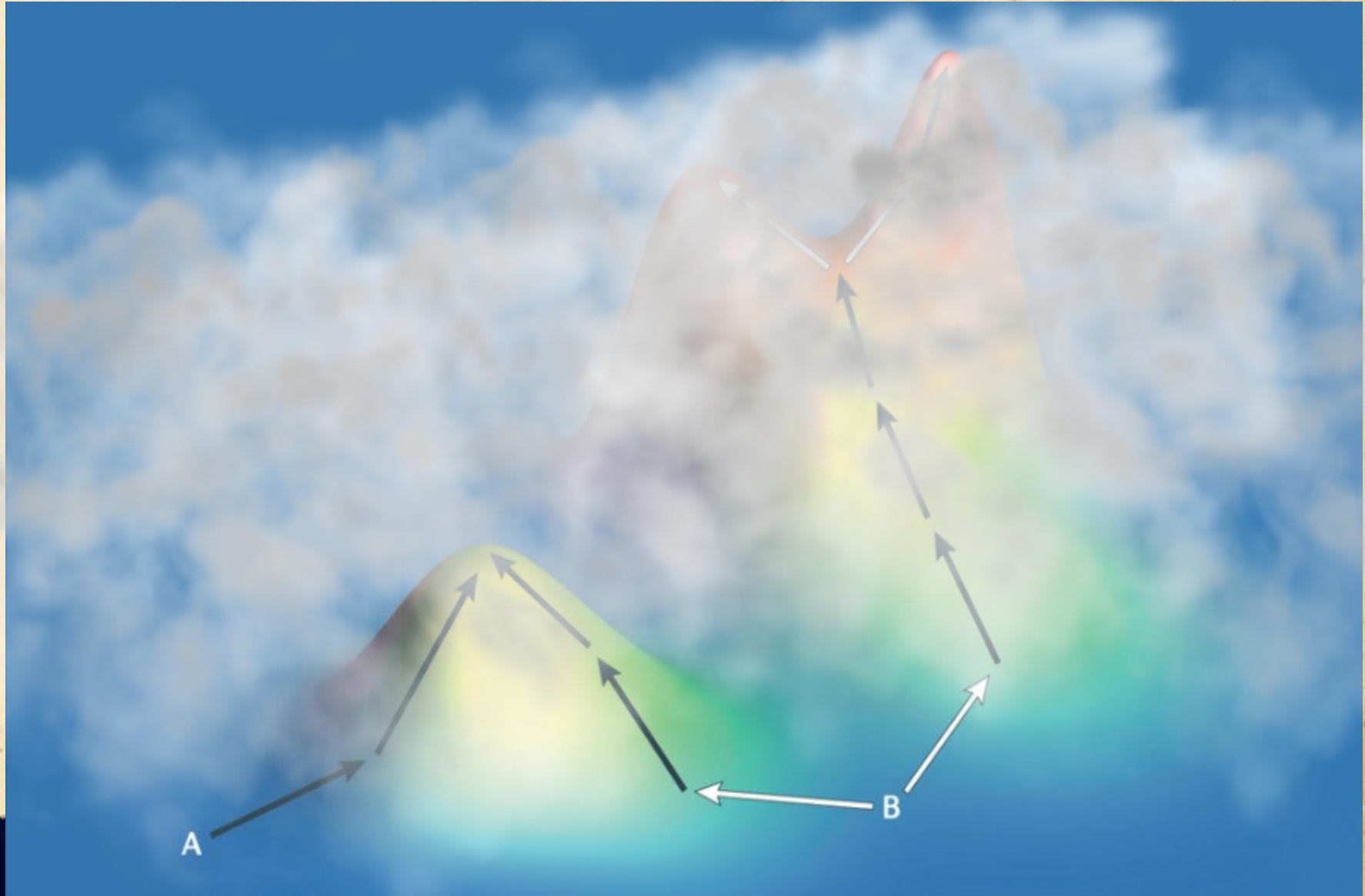
- 2 dimensionaal trein probleem



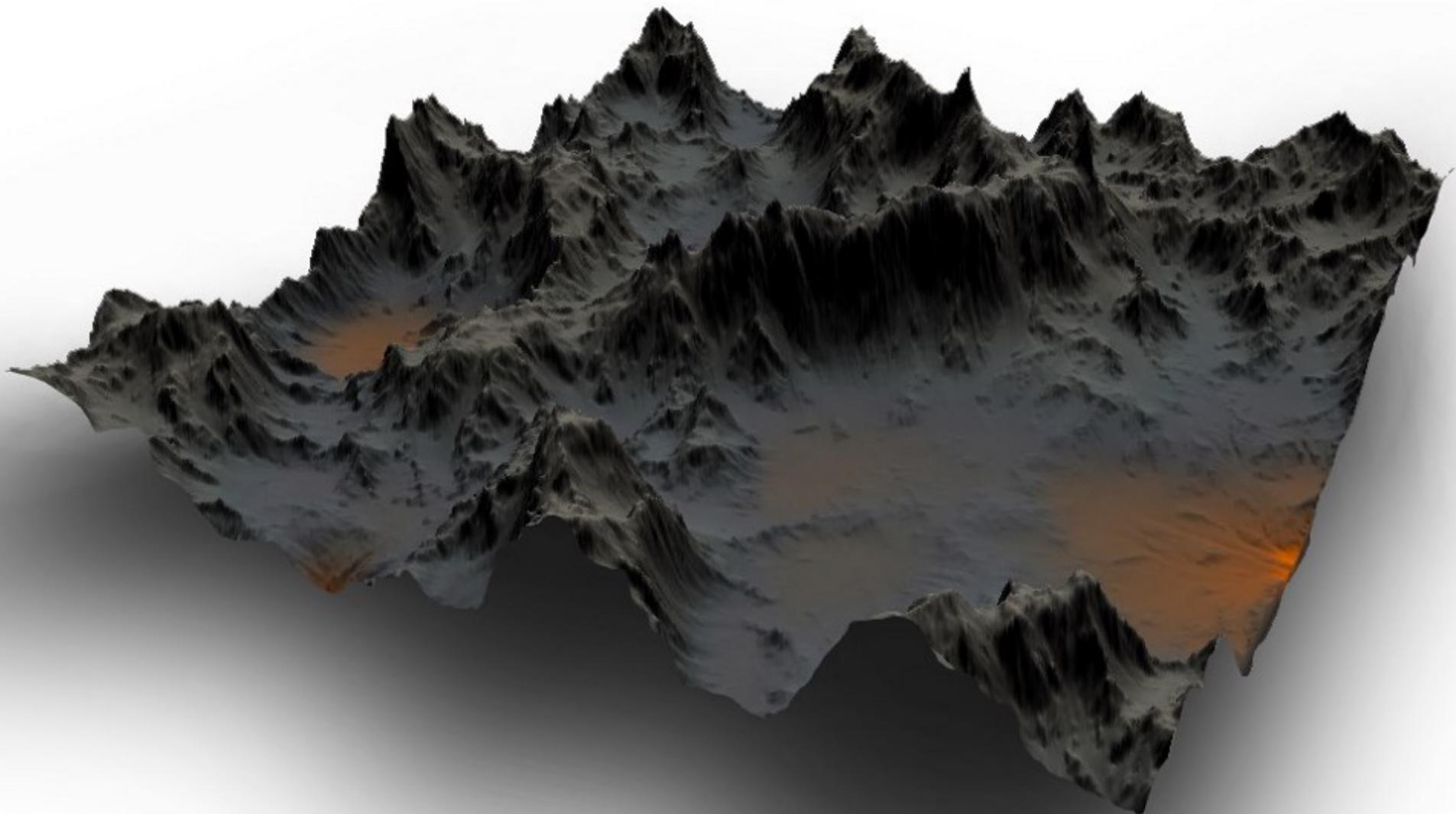
Hill Climber



Hill Climber

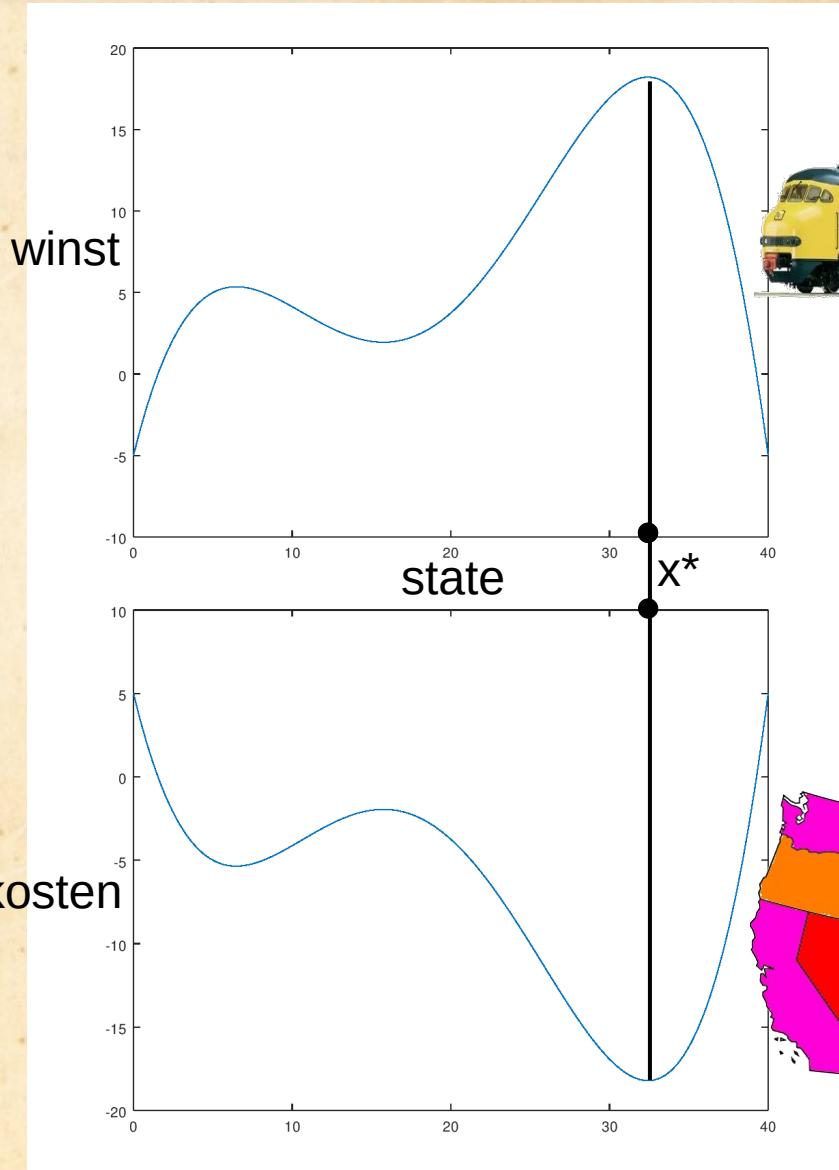


Hill Climber



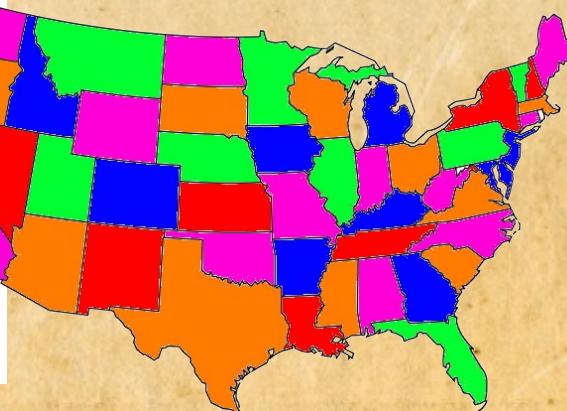
Groot, complex en hoog-dimensionaal landschap

Maximaliseren of Minimaliseren

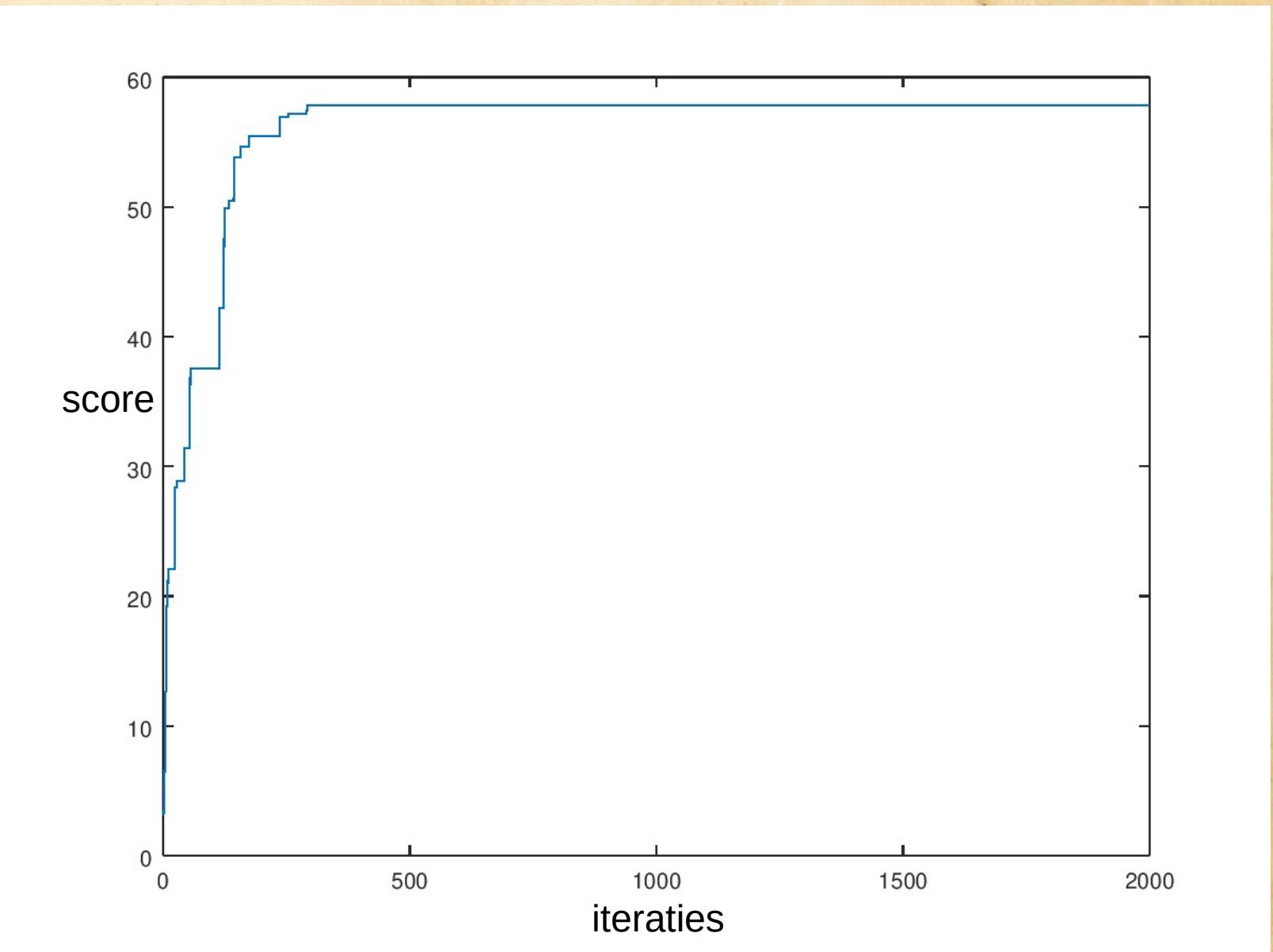


$$x^* = \arg \max_x f(x)$$

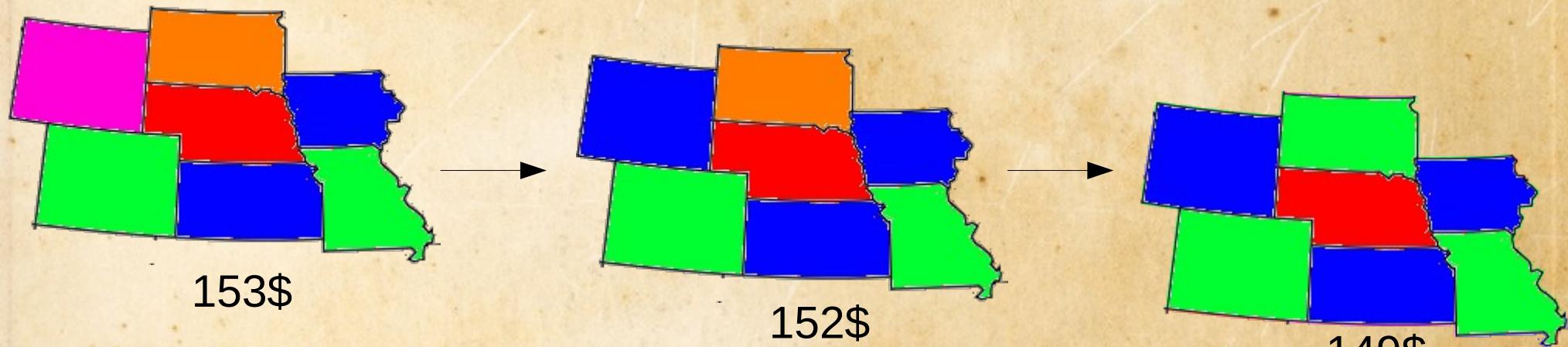
$$x^* = \arg \min_x -f(x)$$



Convergentie

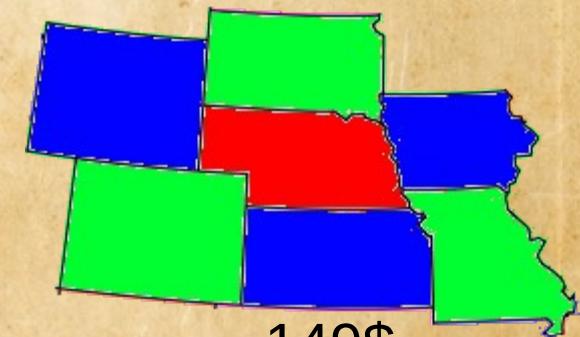
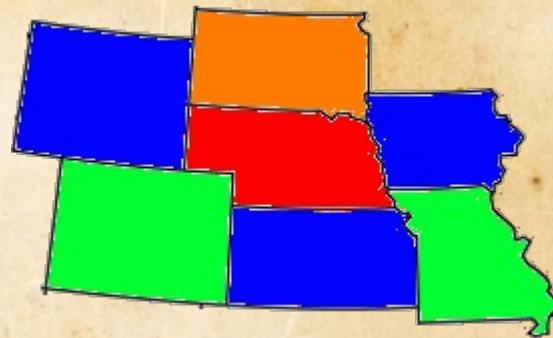


Convergentie

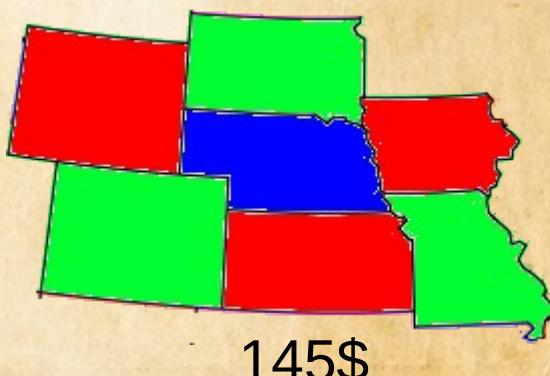


Lokaal optimum?
Globaal optimum?

Convergentie



Lokaal optimum? ✓
Globaal optimum? ✗



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

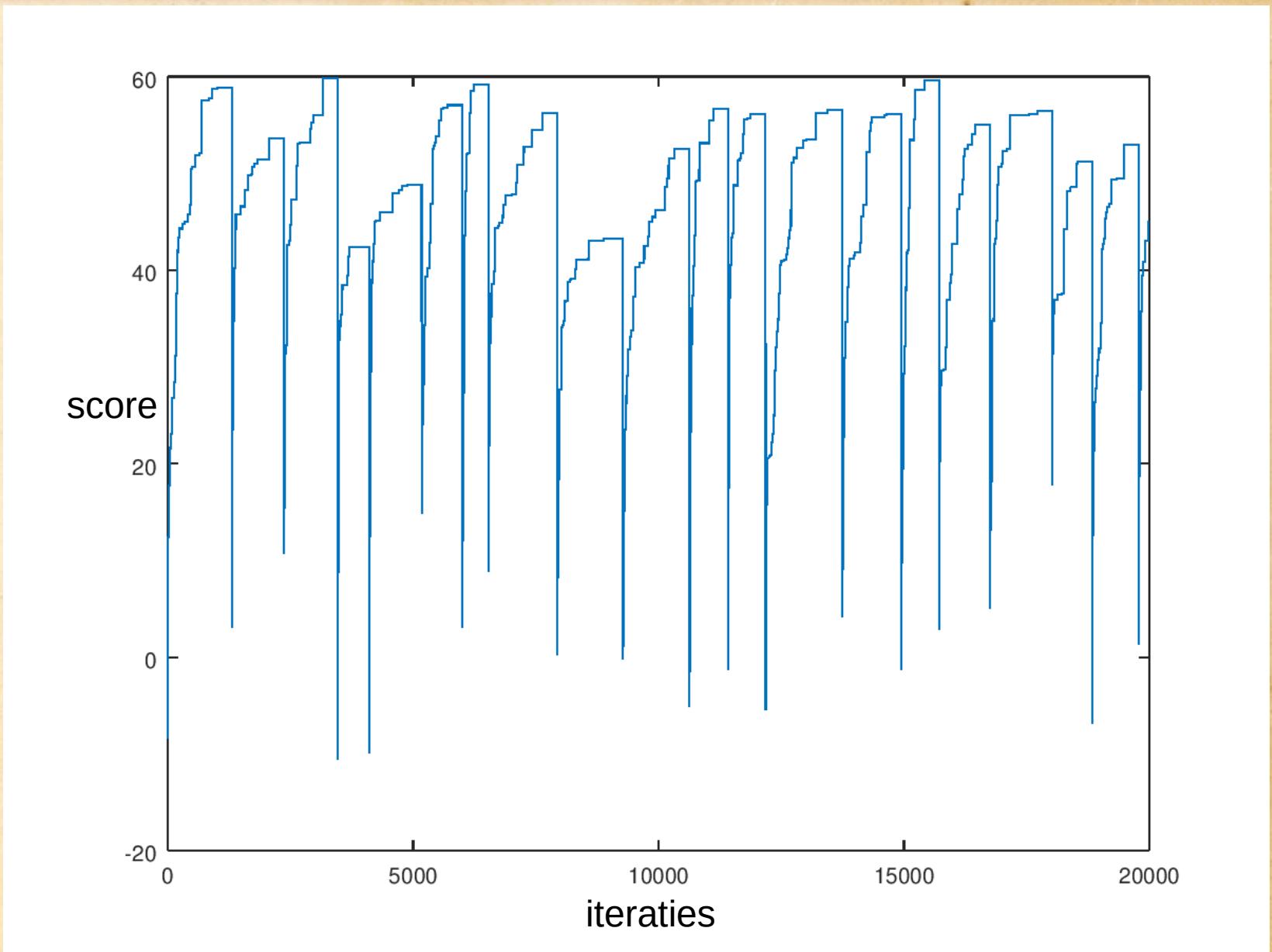
Herhaal tot na N-keer niet meer verbetert:

Doe een kleine random aanpassing

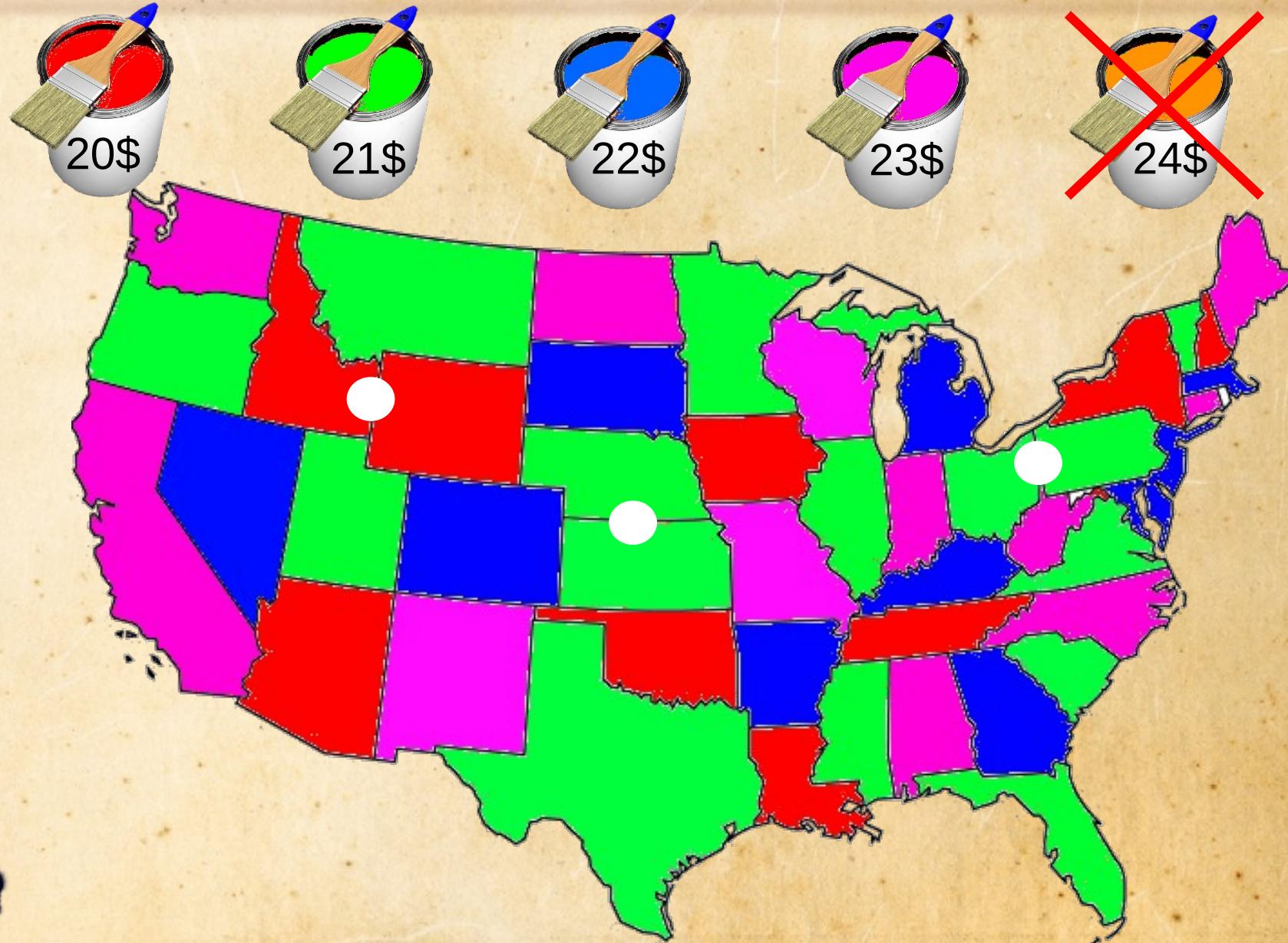
Als de state is verslechterd:

Maak de aanpassing ongedaan

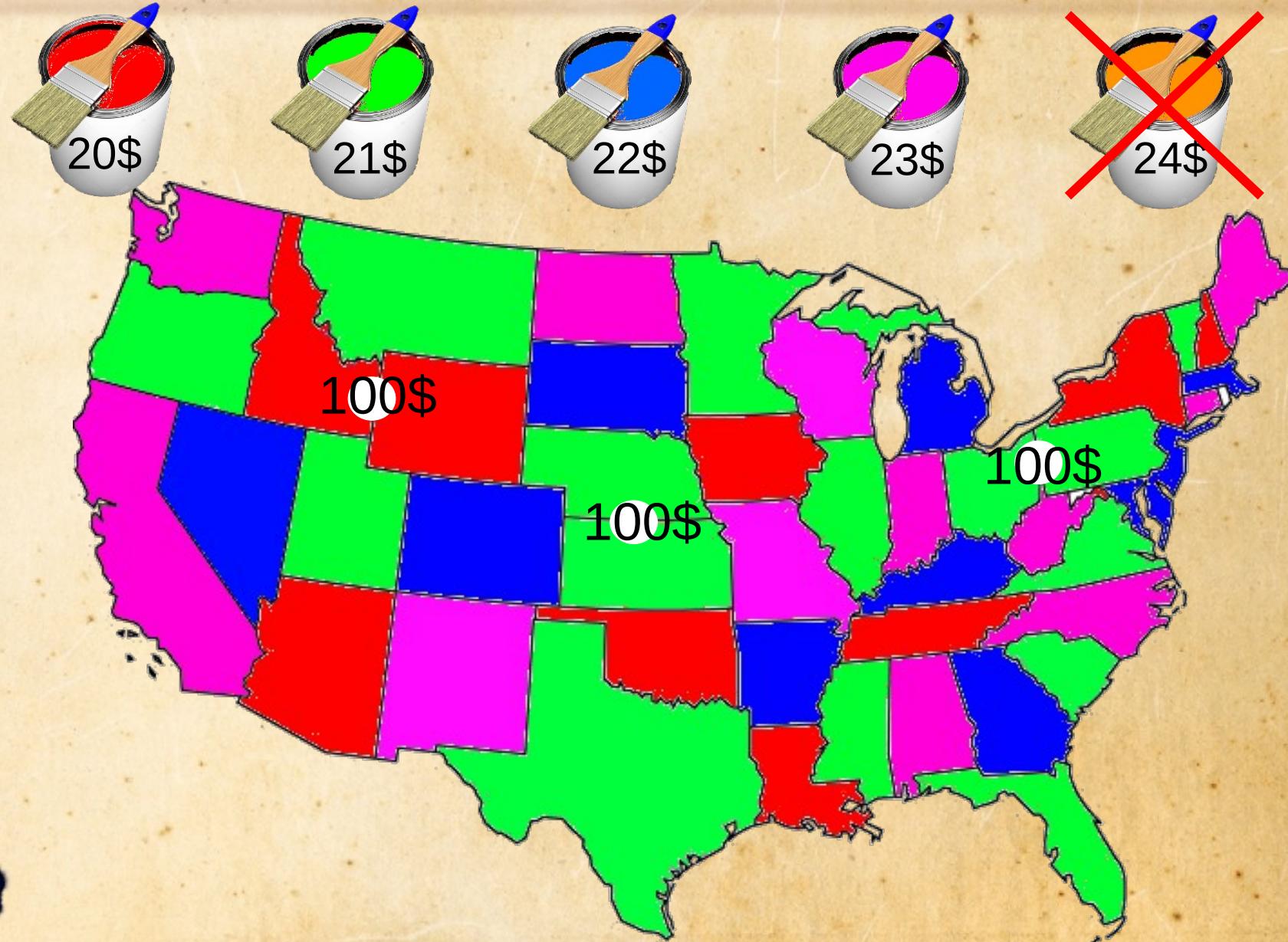
Restart



Constraint relaxation



Constraint relaxation



Constraint relaxation, pseudo code

Herhaal:

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

Herhaal totdat N-keer niet meer verbetert:

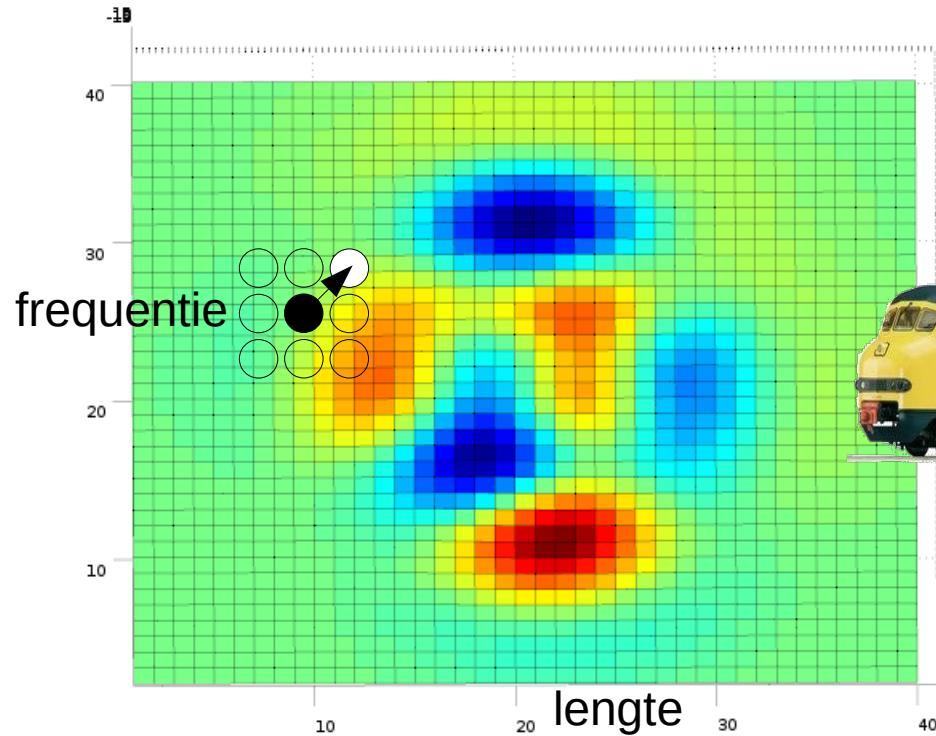
Doe een kleine random aanpassing

Geef strafpunten voor constraint-schendingen

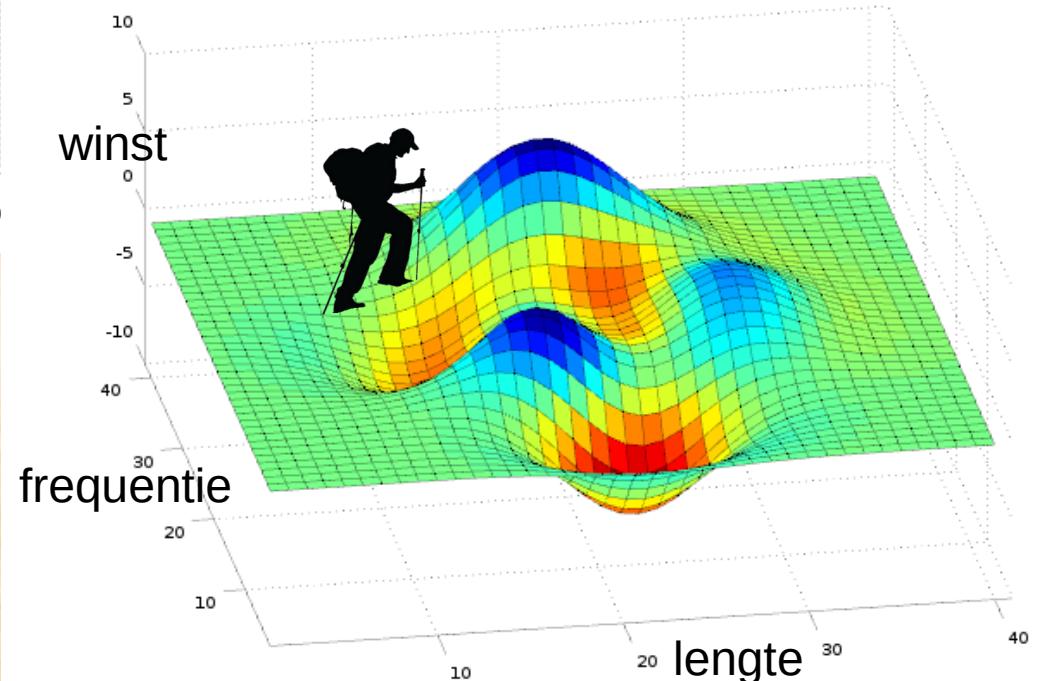
Als de state is verslechterd:

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



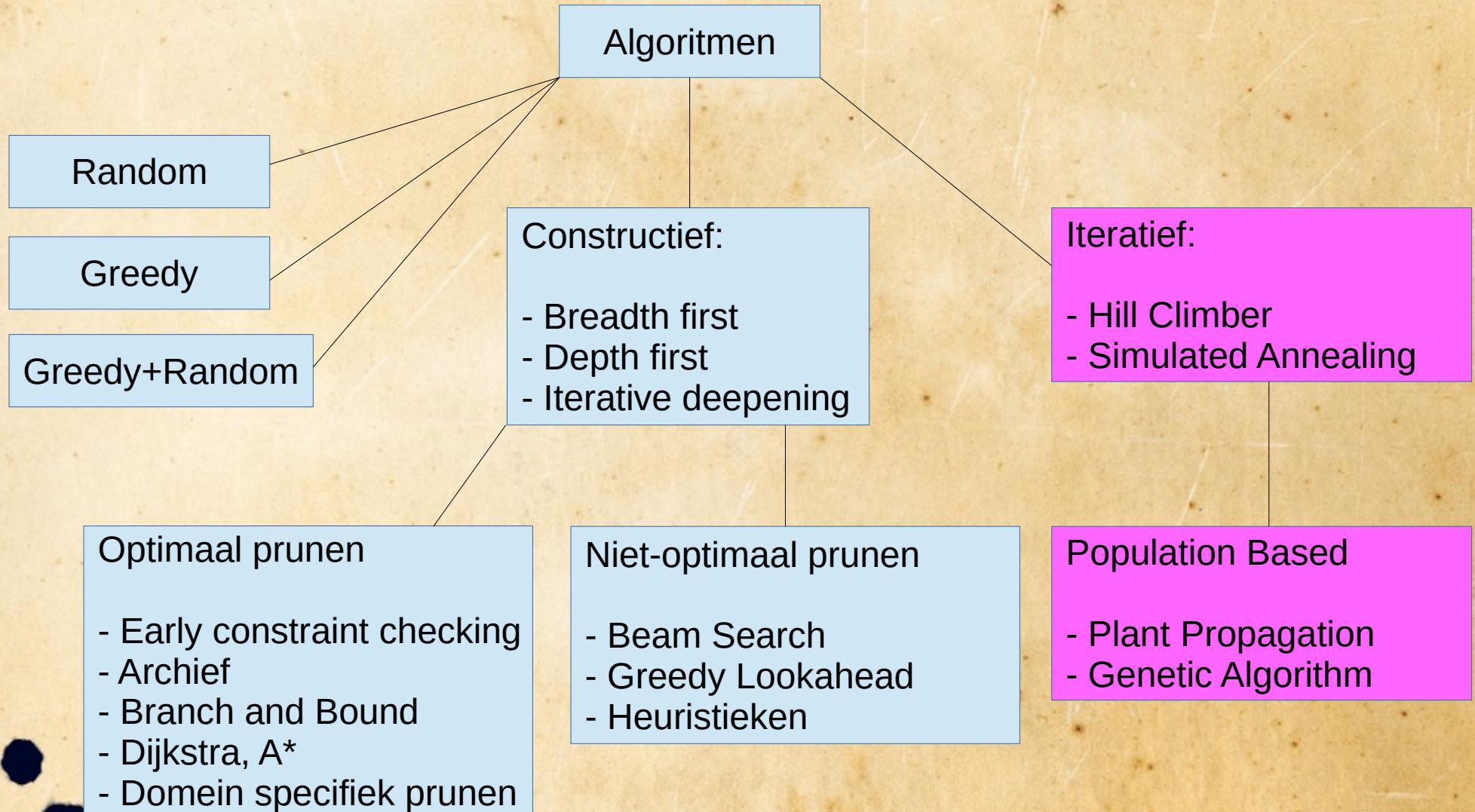
Herhaal:

Kies een random start state

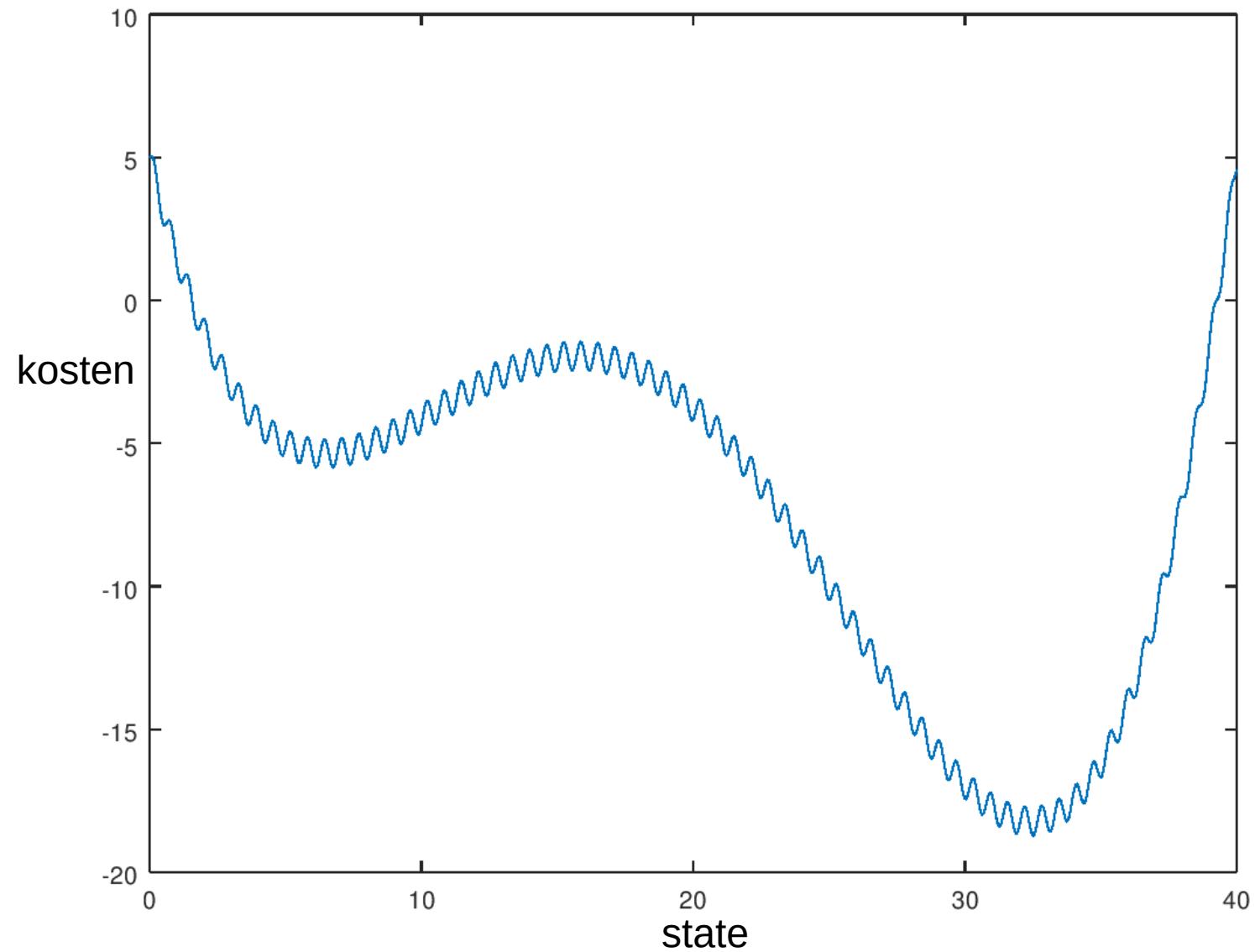
Herhaal **totdat niet meer** verbetert:

Doe beste van alle mogelijke kleine aanpassingen

Algoritmen

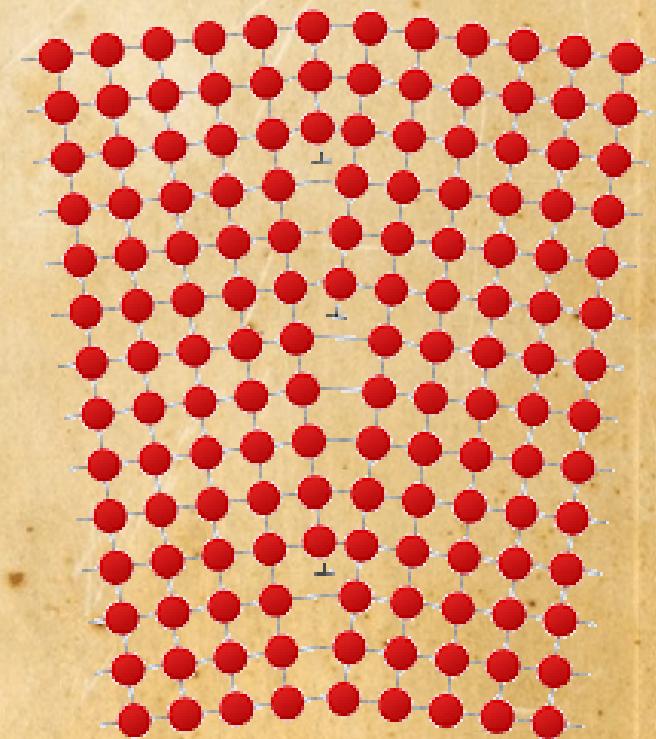


Simulated Annealing

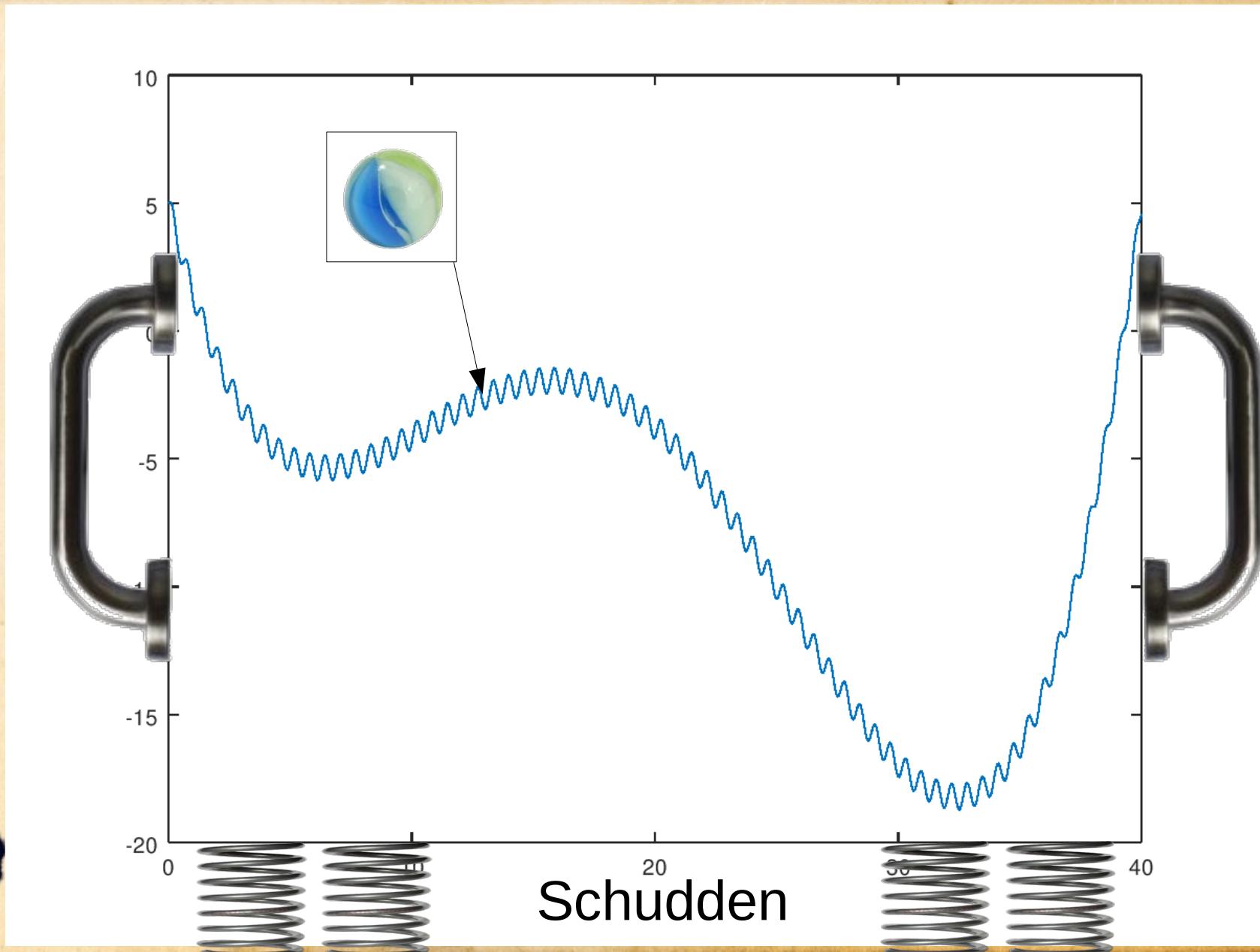


Metaal Annealing

- Kristalstructuur van metaal versterken



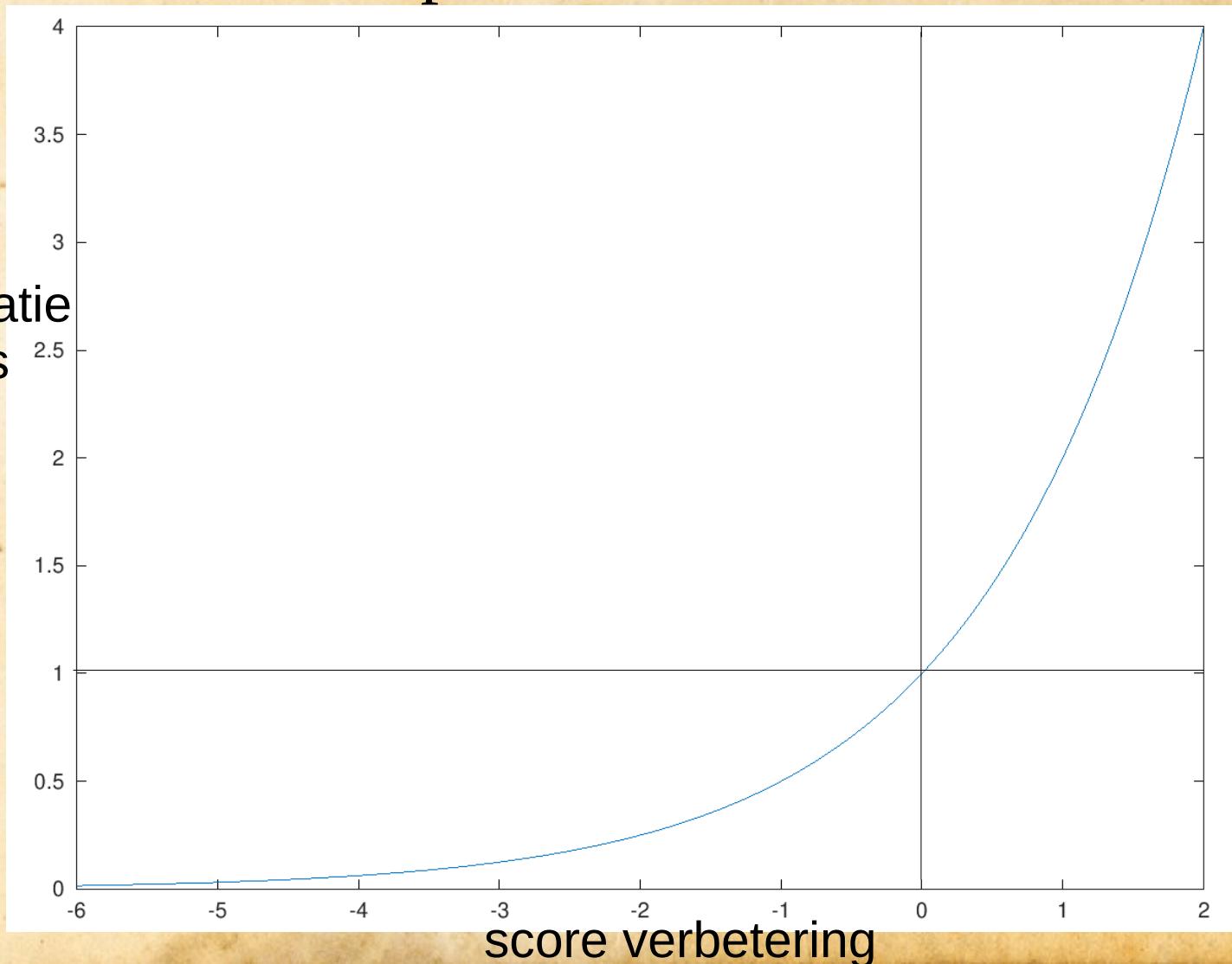
Simulated Annealing



Simulated Annealing

- Minimaliseren: acceptatiekans = $2^{(score_{old} - score_{new})}$

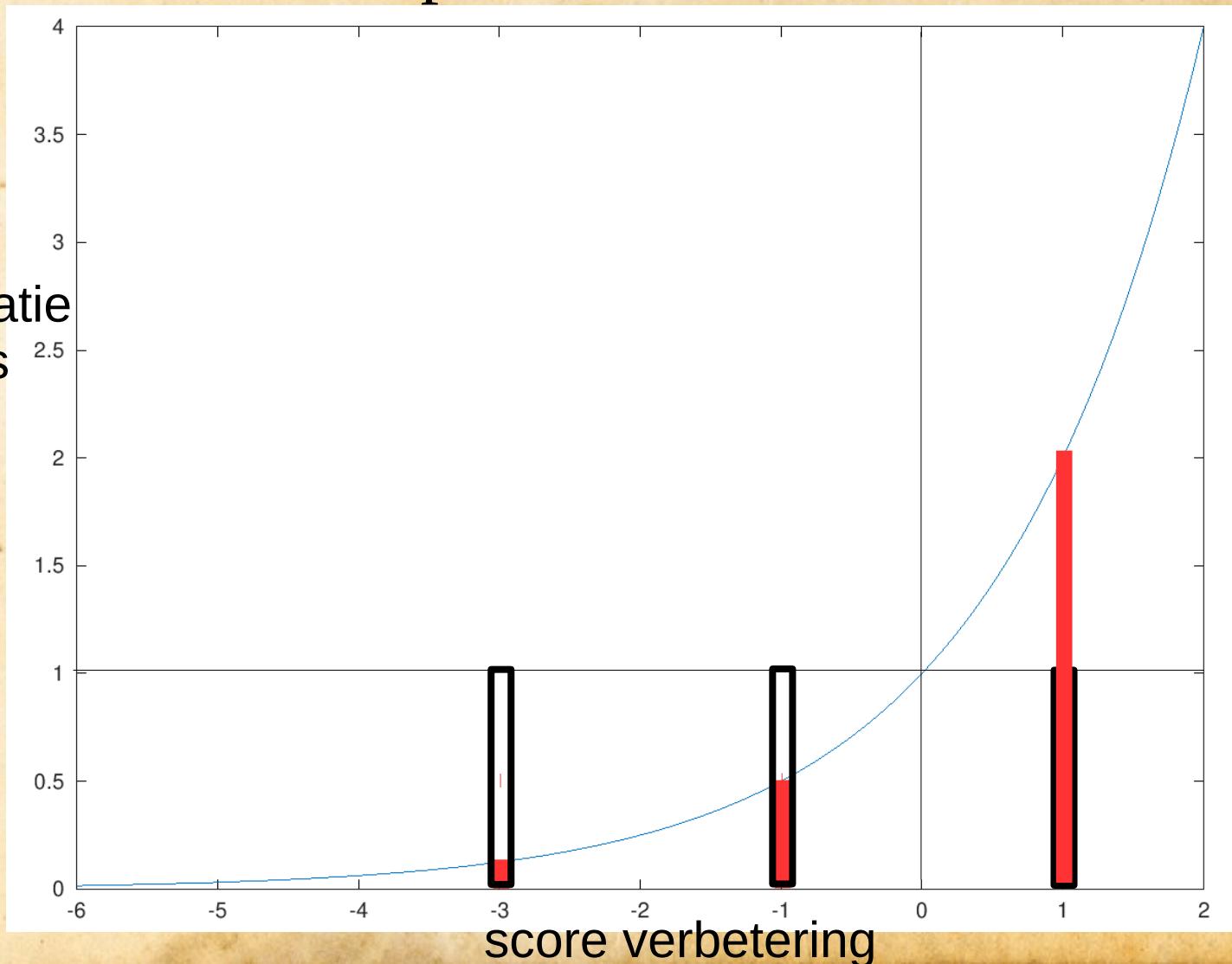
acceptatie
kans



Simulated Annealing

- Minimaliseren: acceptatiekans = $2^{(score_{old} - score_{new})}$

acceptatie
kans

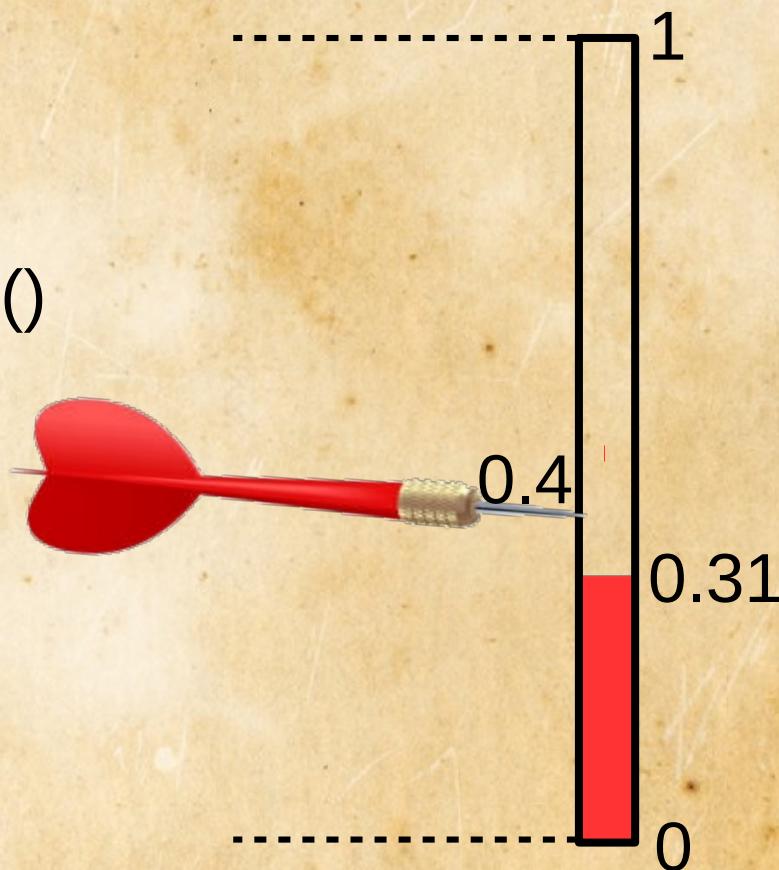


Simulated Annealing

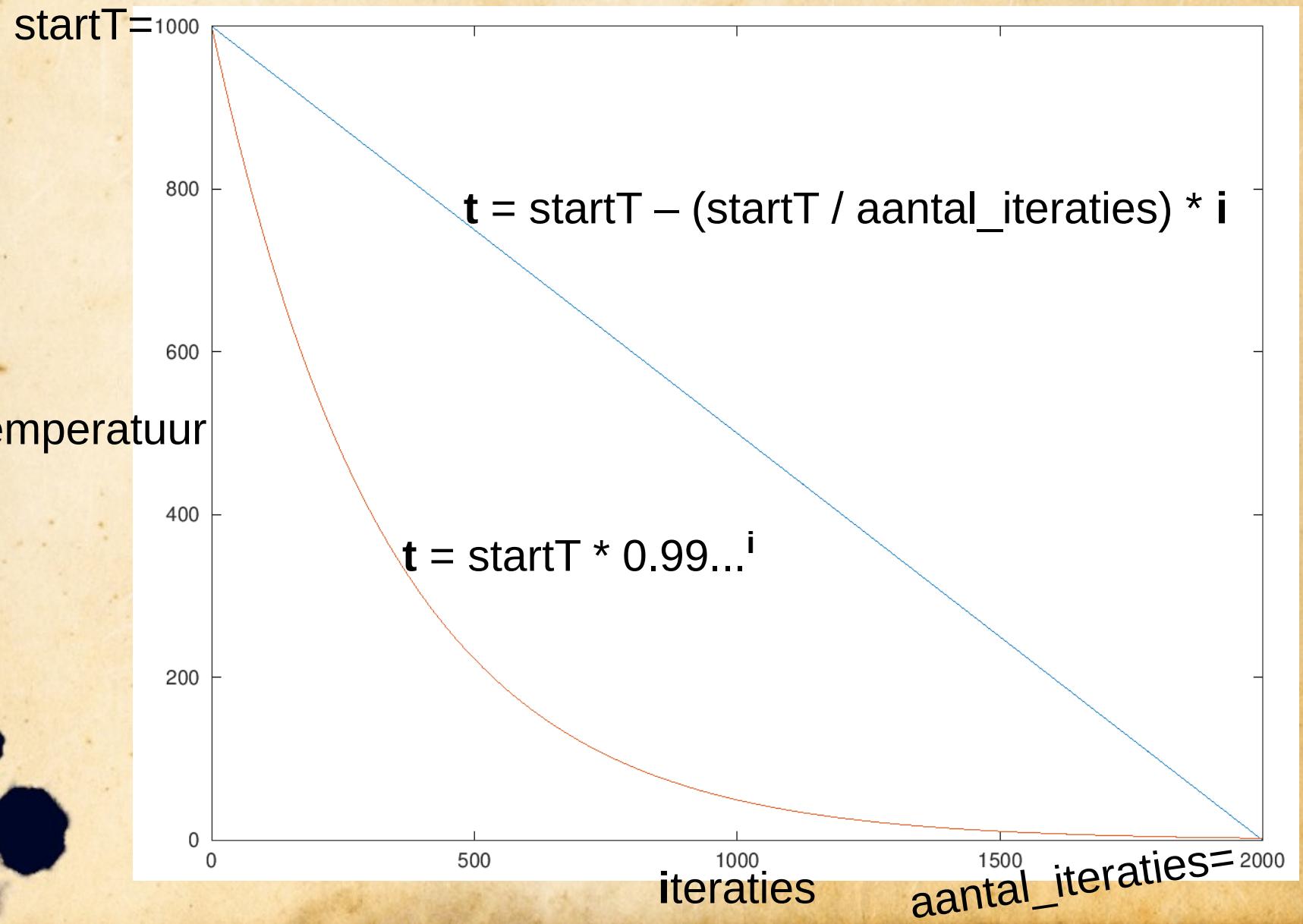
- Wel of niet accepteren?

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

```
...
```



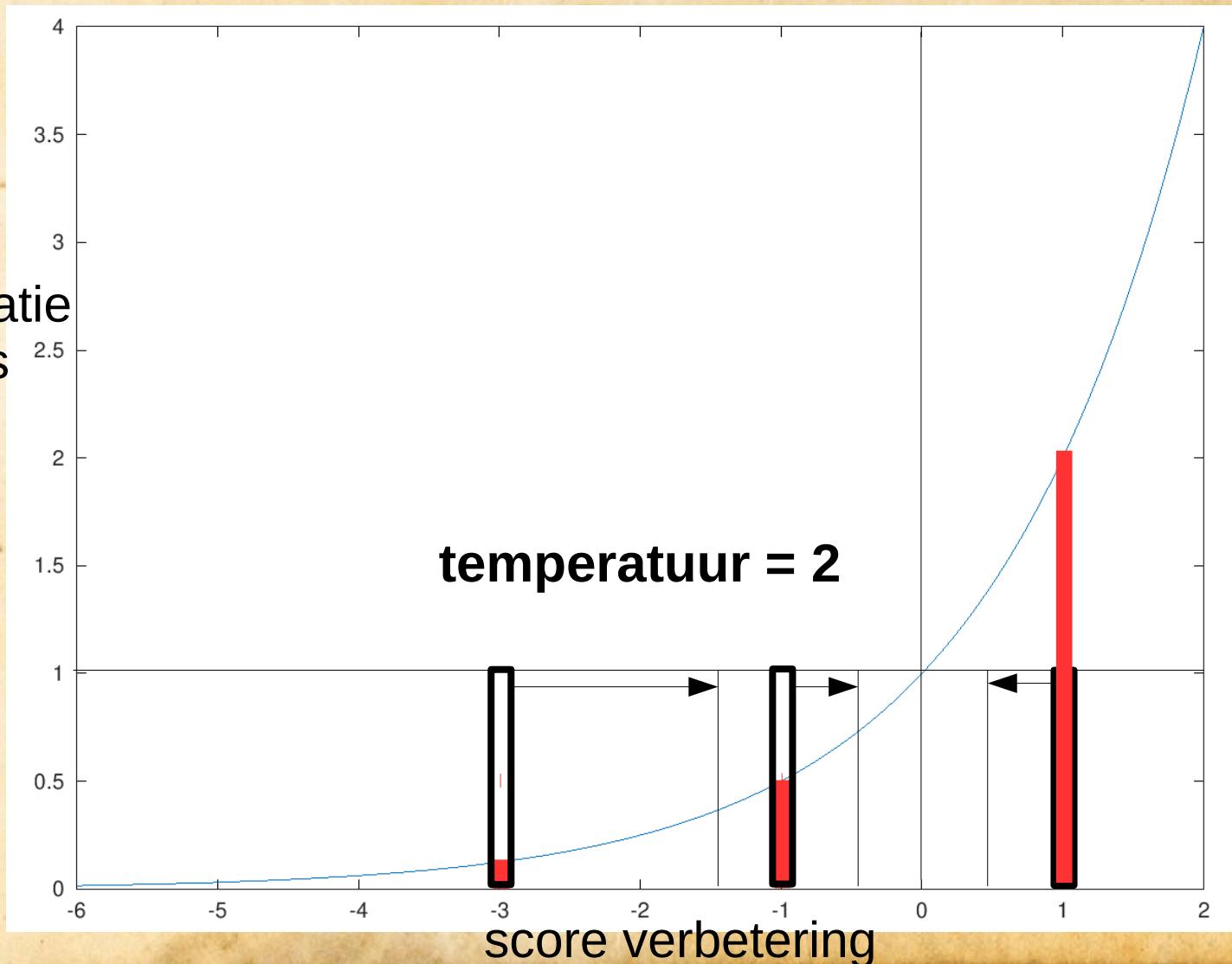
Simulated Annealing



Simulated Annealing

- acceptatiekans = $2^{(\text{score_old} - \text{score_new}) / \text{temperatuur}}$

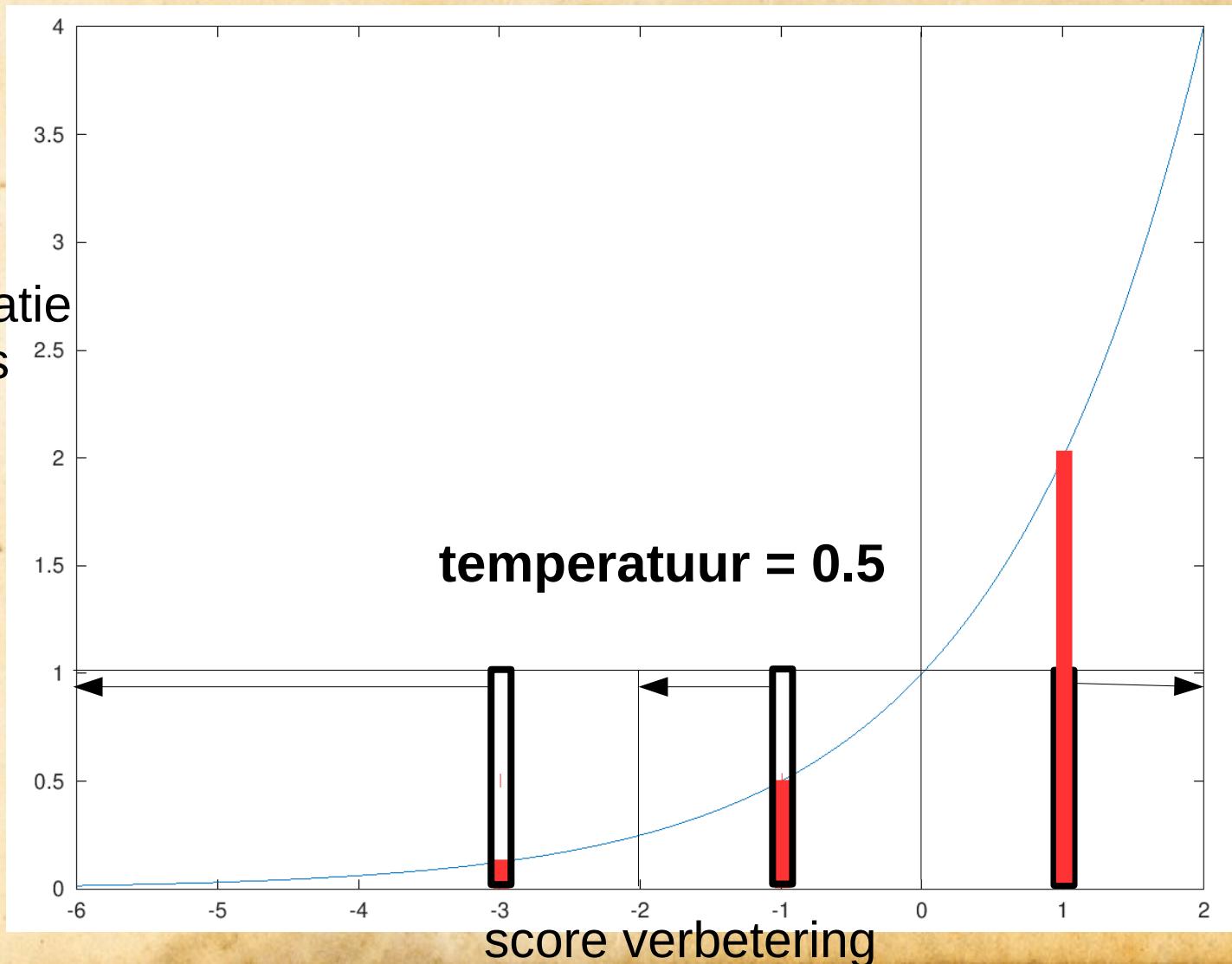
acceptatie
kans



Simulated Annealing

- acceptatiekans = $2^{(\text{score_old} - \text{score_new}) / \text{temperatuur}}$

acceptatie
kans



Simulated Annealing, pseudo code

Herhaal:

Kies een random start state

Zet de start-temperatuur

Herhaal **N iteraties**:

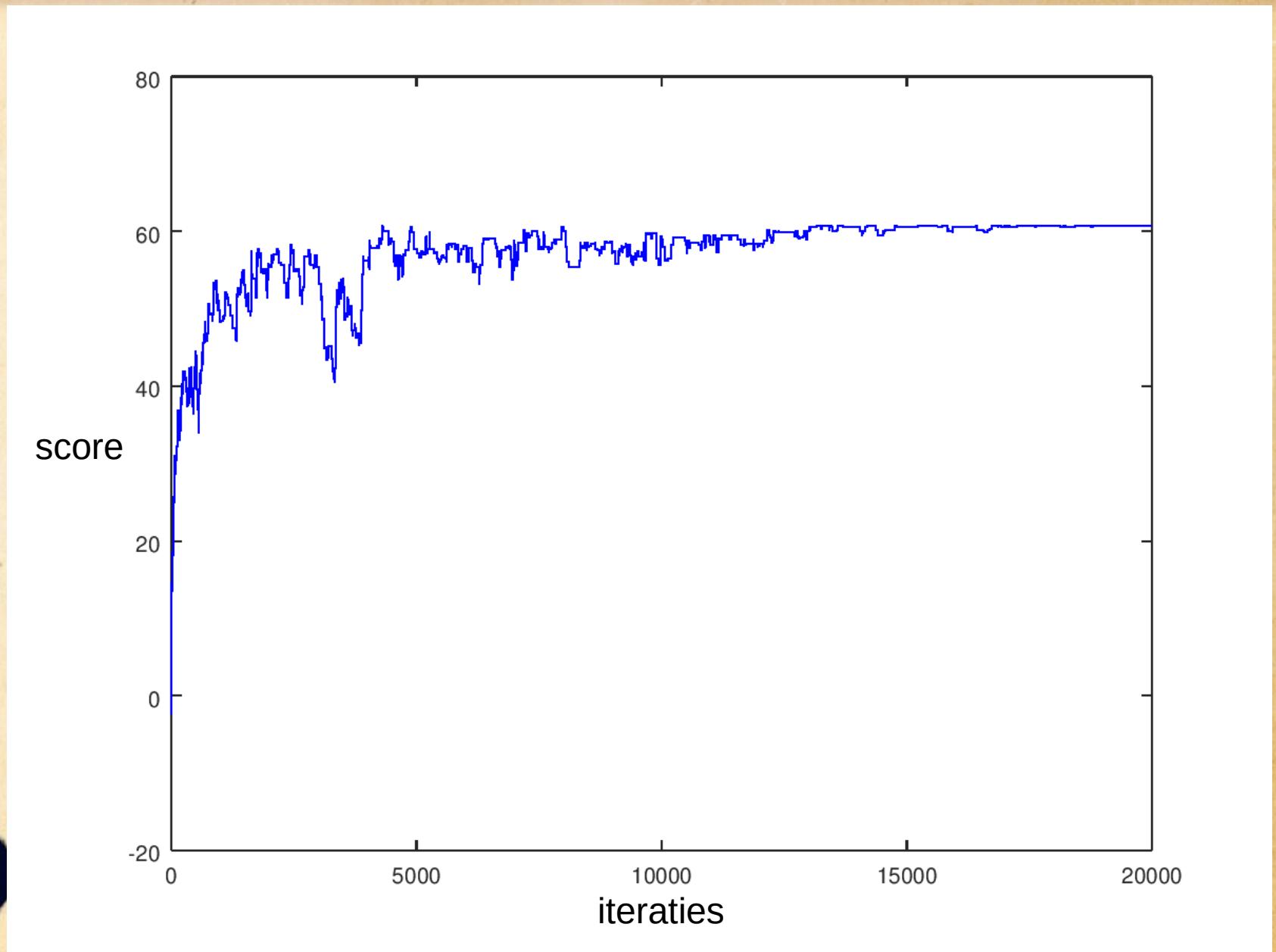
Doe een kleine random aanpassing

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

 Maak de aanpassing ongedaan

Verlaag temperatuur

Simulated Annealing



Simulated Annealing, Temperatuur

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

$$2^{V/T} = \frac{1}{2} \quad (50\%)$$

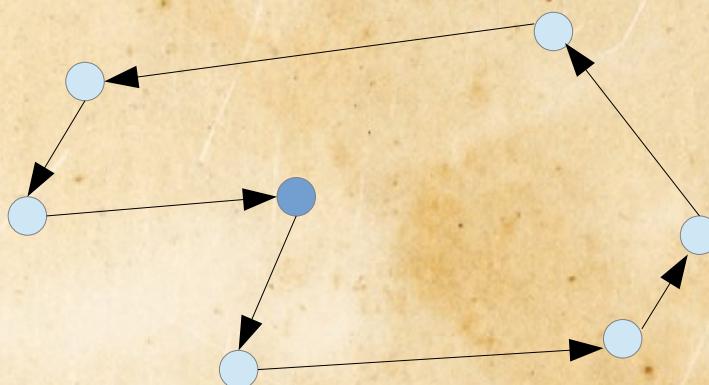
$$T = -V = 20$$

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

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

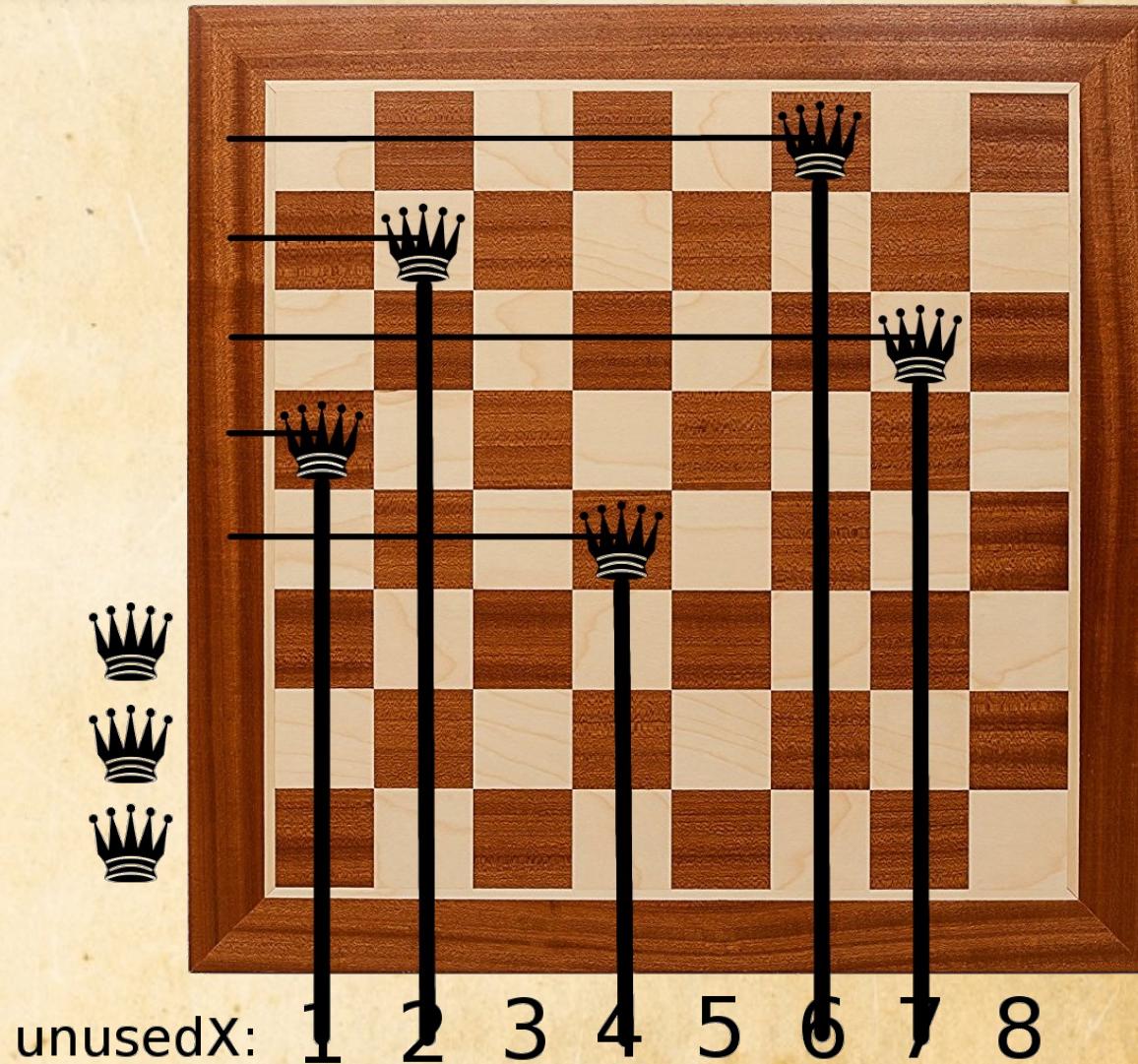
Traveling Salesman, Demo

- 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	

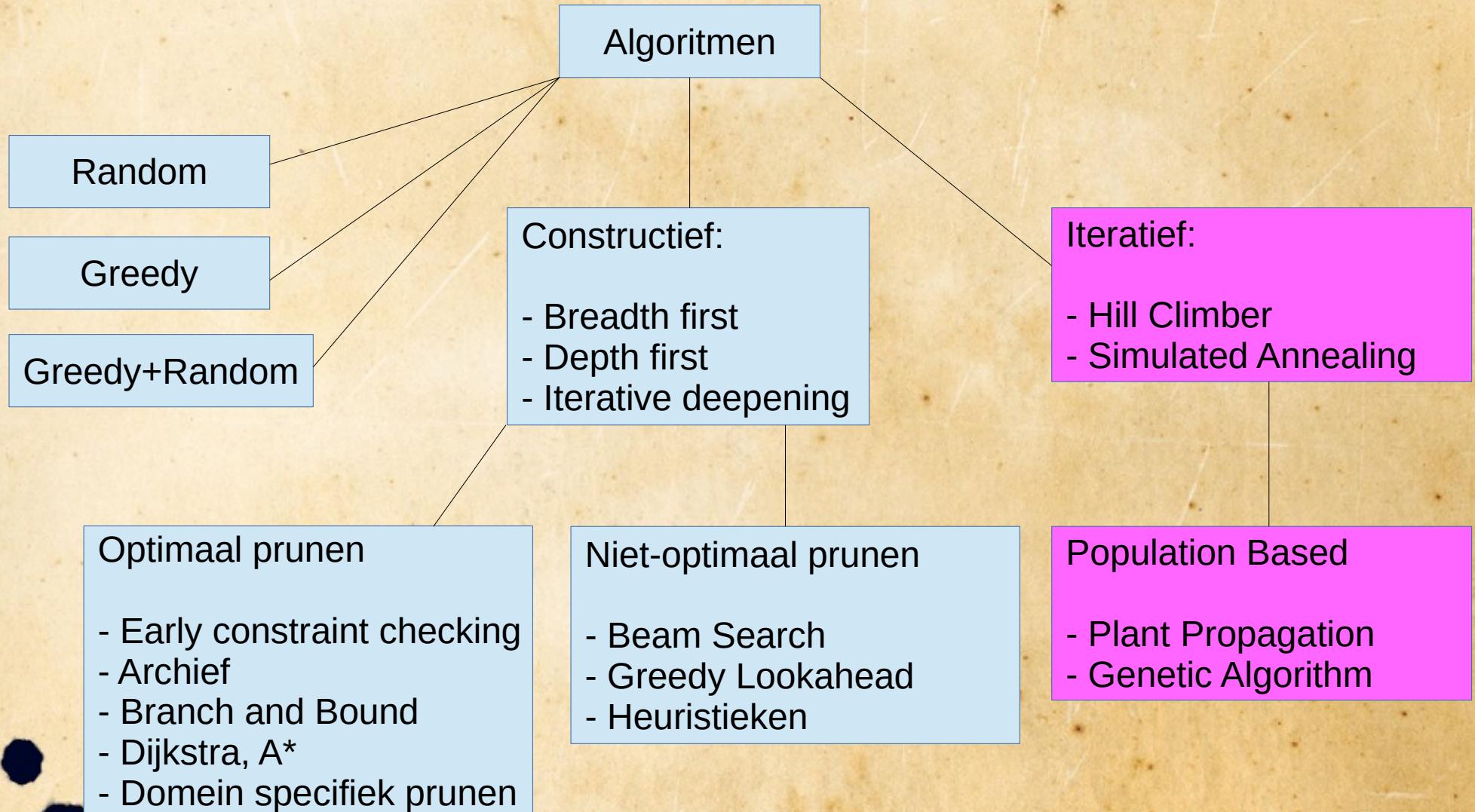
N-Queens, Demo

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

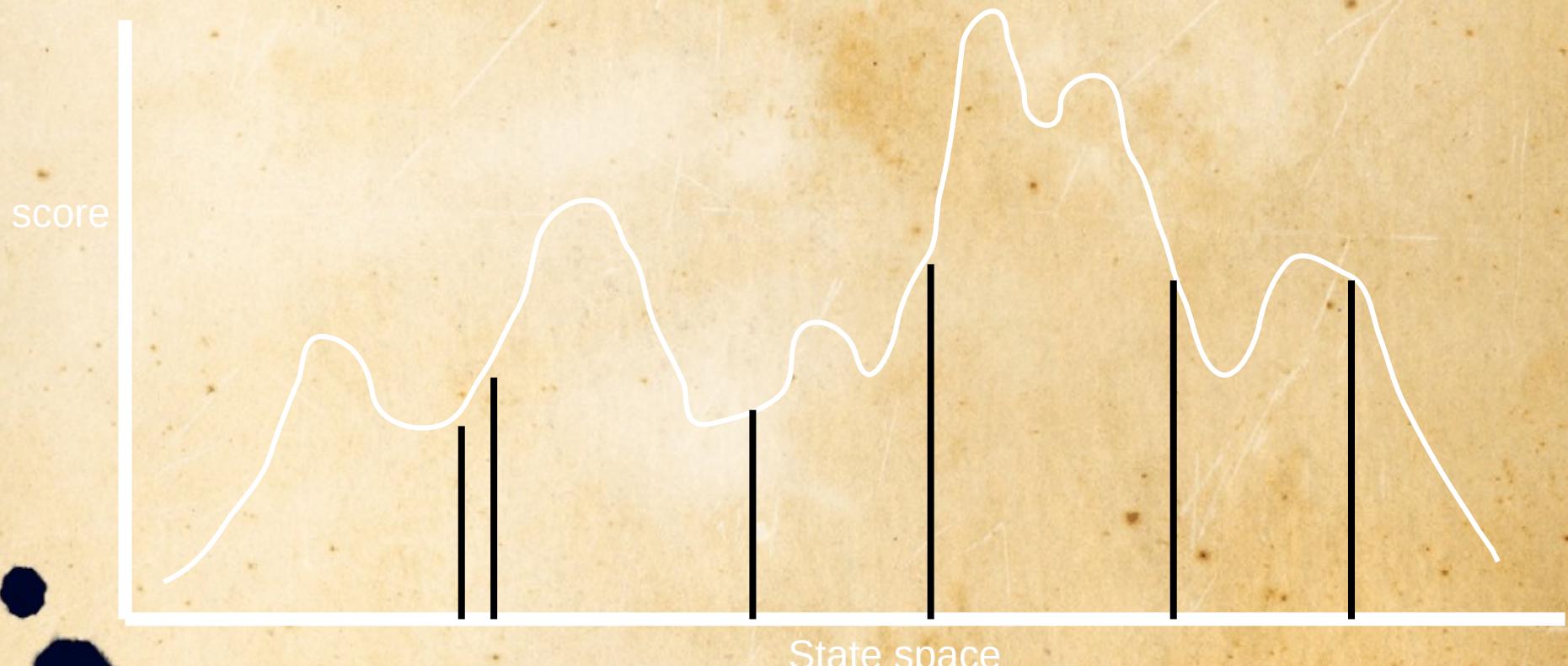
```
python3 -m NQueens.ConstructiveAlgorithms.Random 20 p  
queensX: [16, 2, 7, 5, 18, 15, 9, 6, 3, 17, 8, 1, 19, 10, 14, 4, 0, 13, 11, 12]
```

A diagram on a grid showing two parallel red lines. Several points labeled 'Q' are marked along these lines. A point labeled 'Q' is also marked at the intersection of a horizontal red line and a vertical red line.

Algoritmen



Population based algorithms

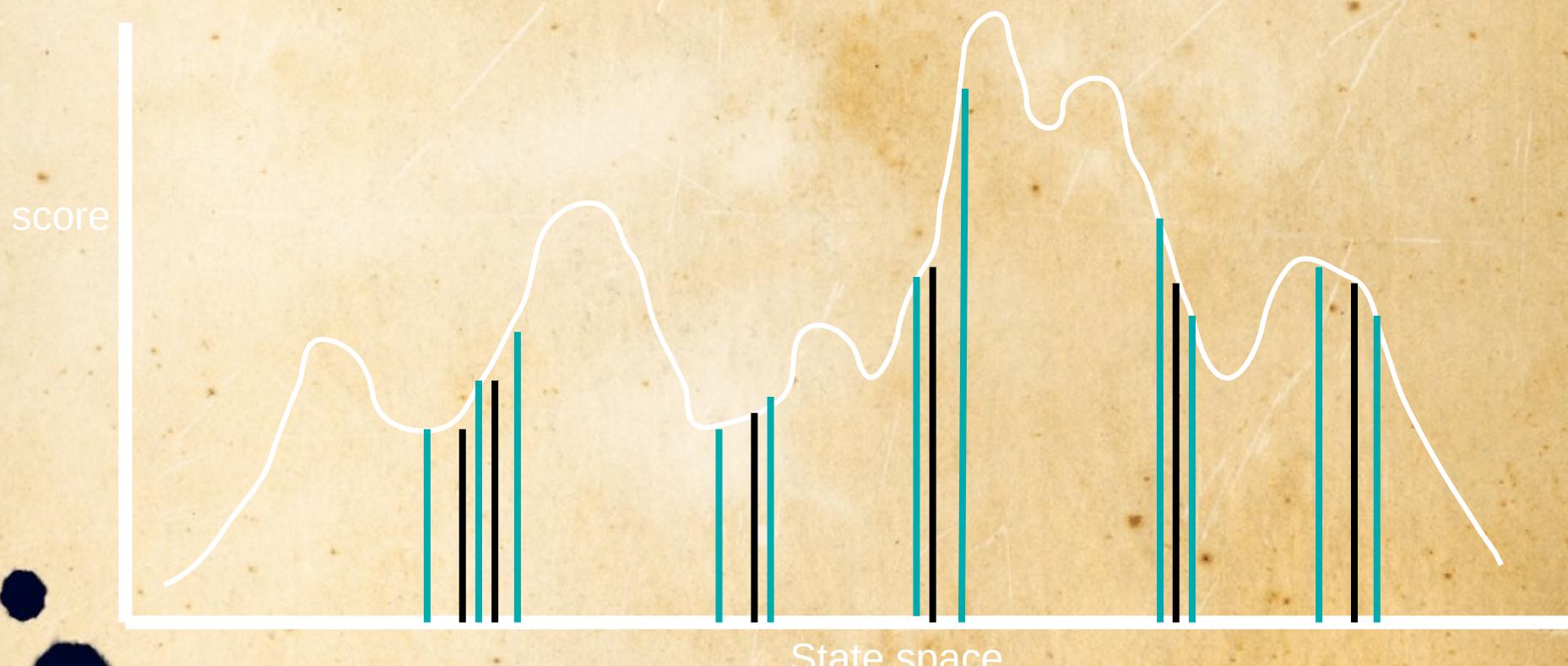


Populatie van Hill Climbers: 6

Population based algorithms



Populatie krijgt nieuwe kinderen

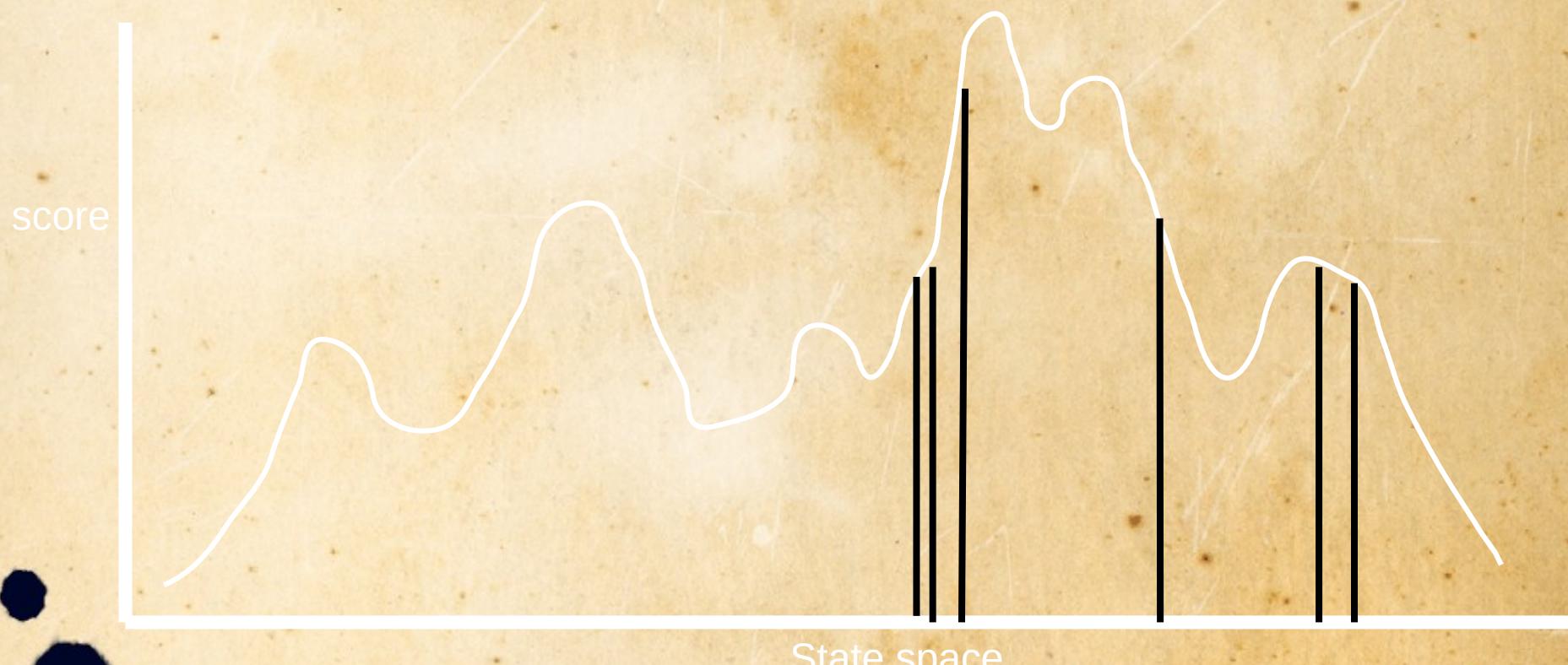


Populatie van Hill Climbers: 6

Population based algorithms



Survival of the fittest



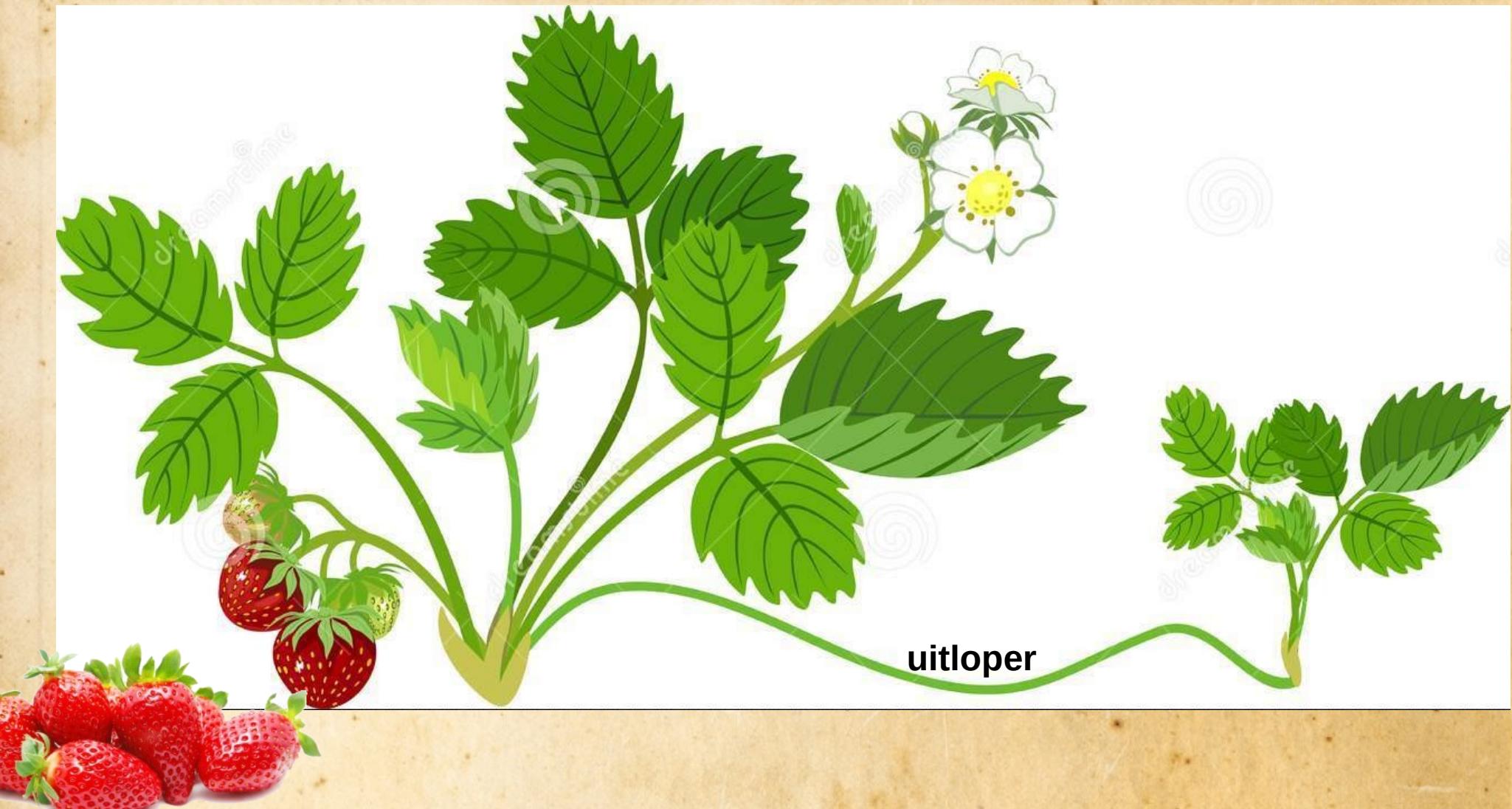
Populatie van Hill Climbers: 6

Plant Propagation Algorithm

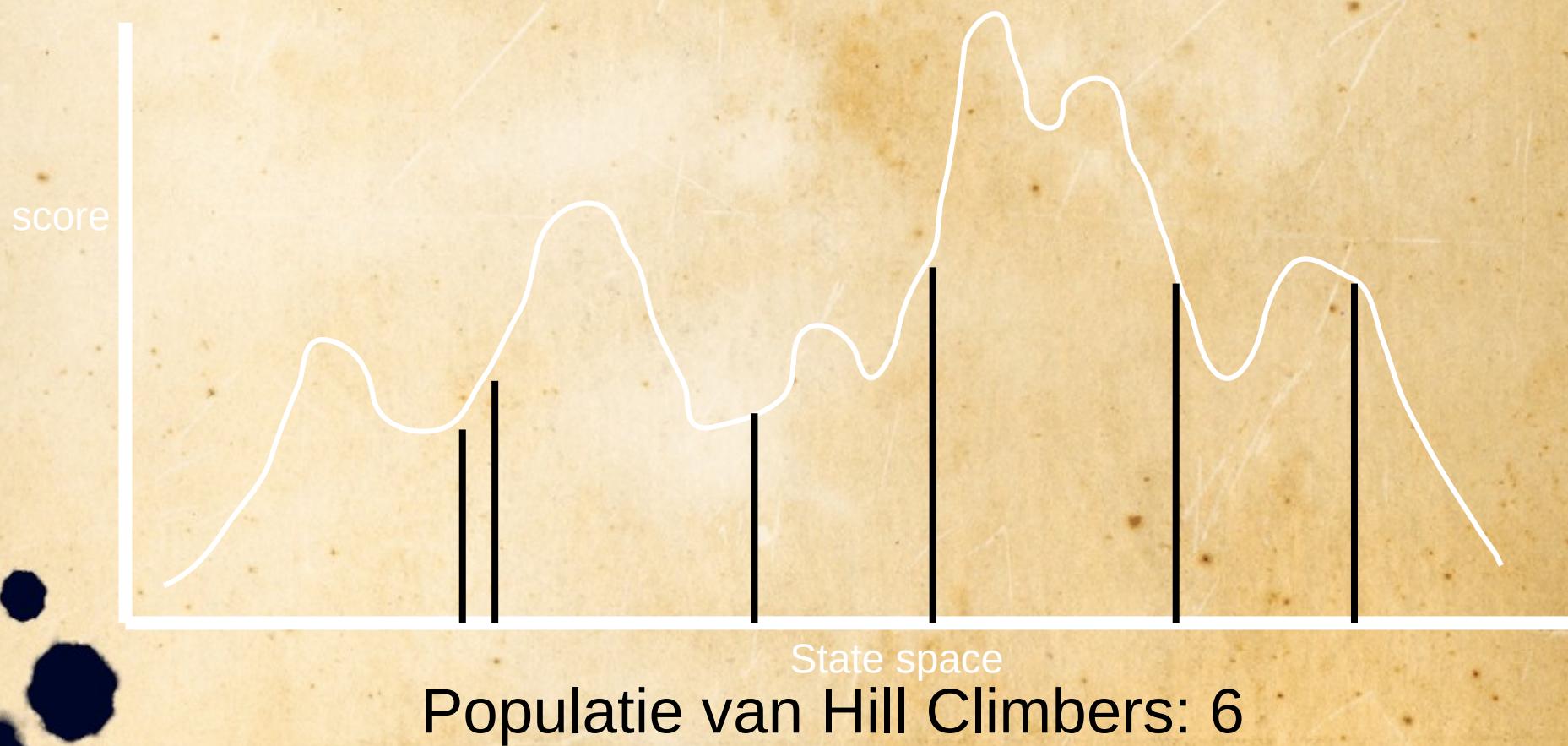


- Paper:
 - **Nature-Inspired Optimisation Approaches and the New Plant Propagation Algorithm**
 - Abdellah Salhi, Eric S Fraga

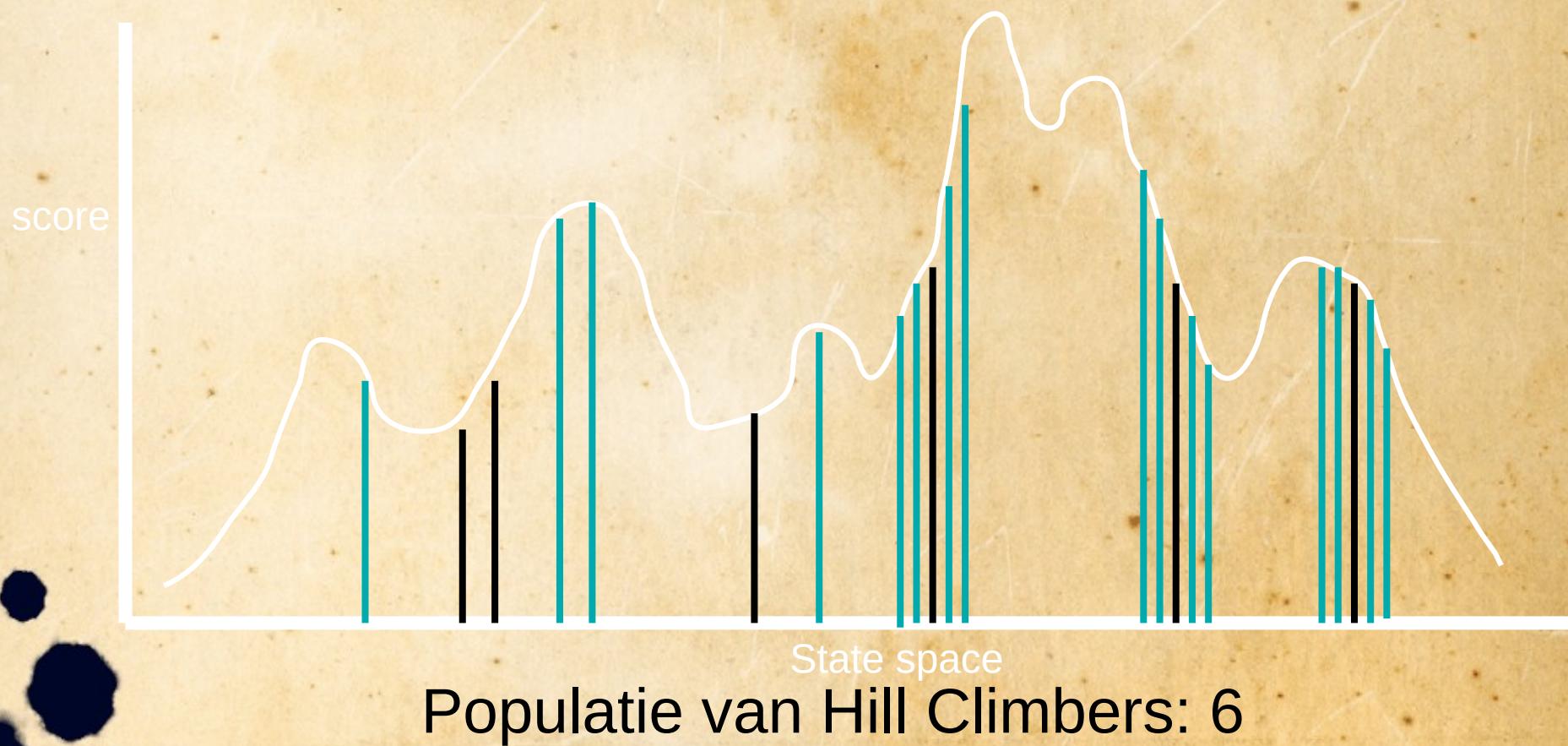
Plant Propagation Algorithm



Plant Propagation Algorithm

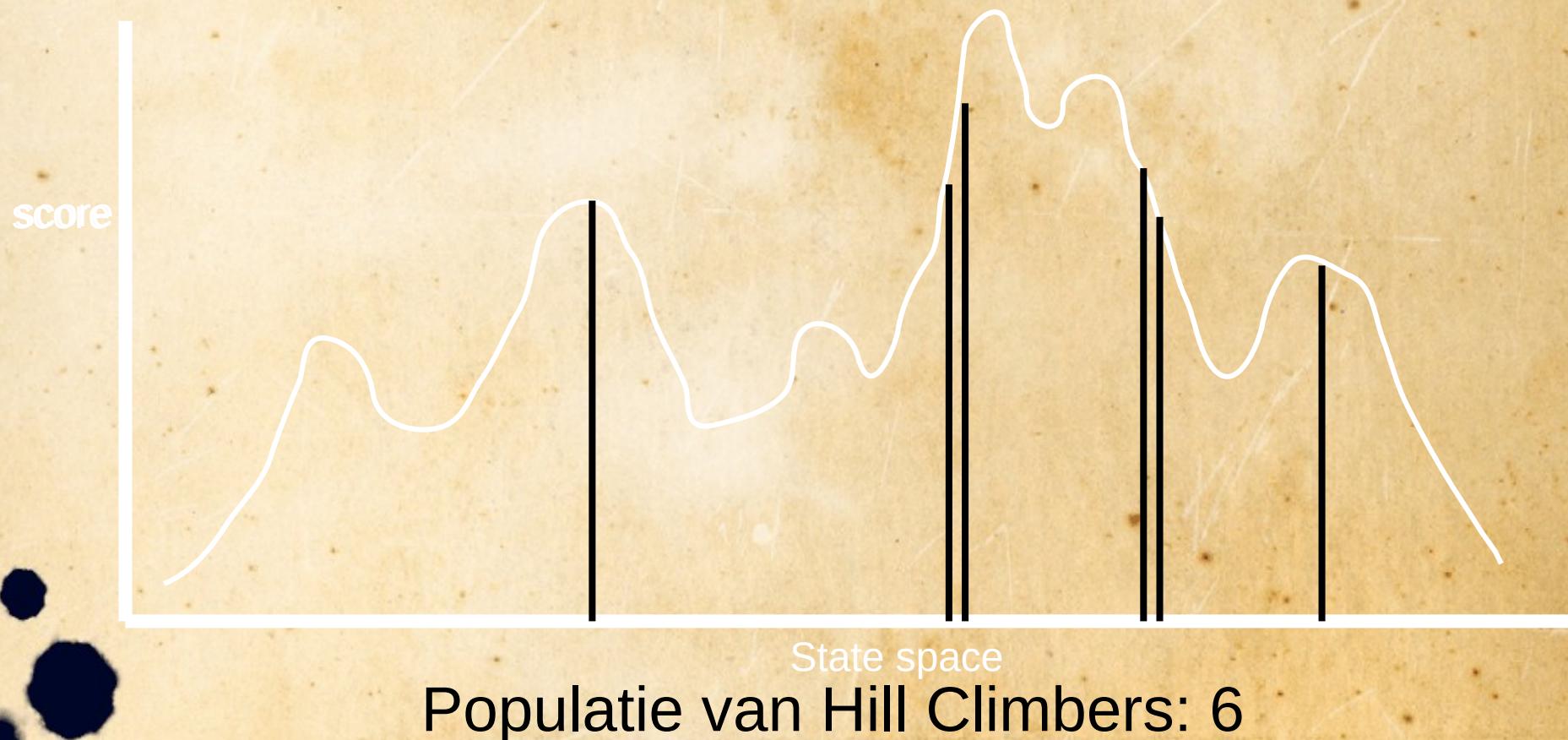


Plant Propagation Algorithm



Plant Propagation Algorithm

Selecteer beste 6



Plant Propagation Algorithm

Require: objective $f(x)$, $x \in \mathcal{R}^n$

Generate a population $P = \{p_i, i = 1, \dots, m\}$

$g \leftarrow 1$

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 $p_i, i = 1, \dots, m$ **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}

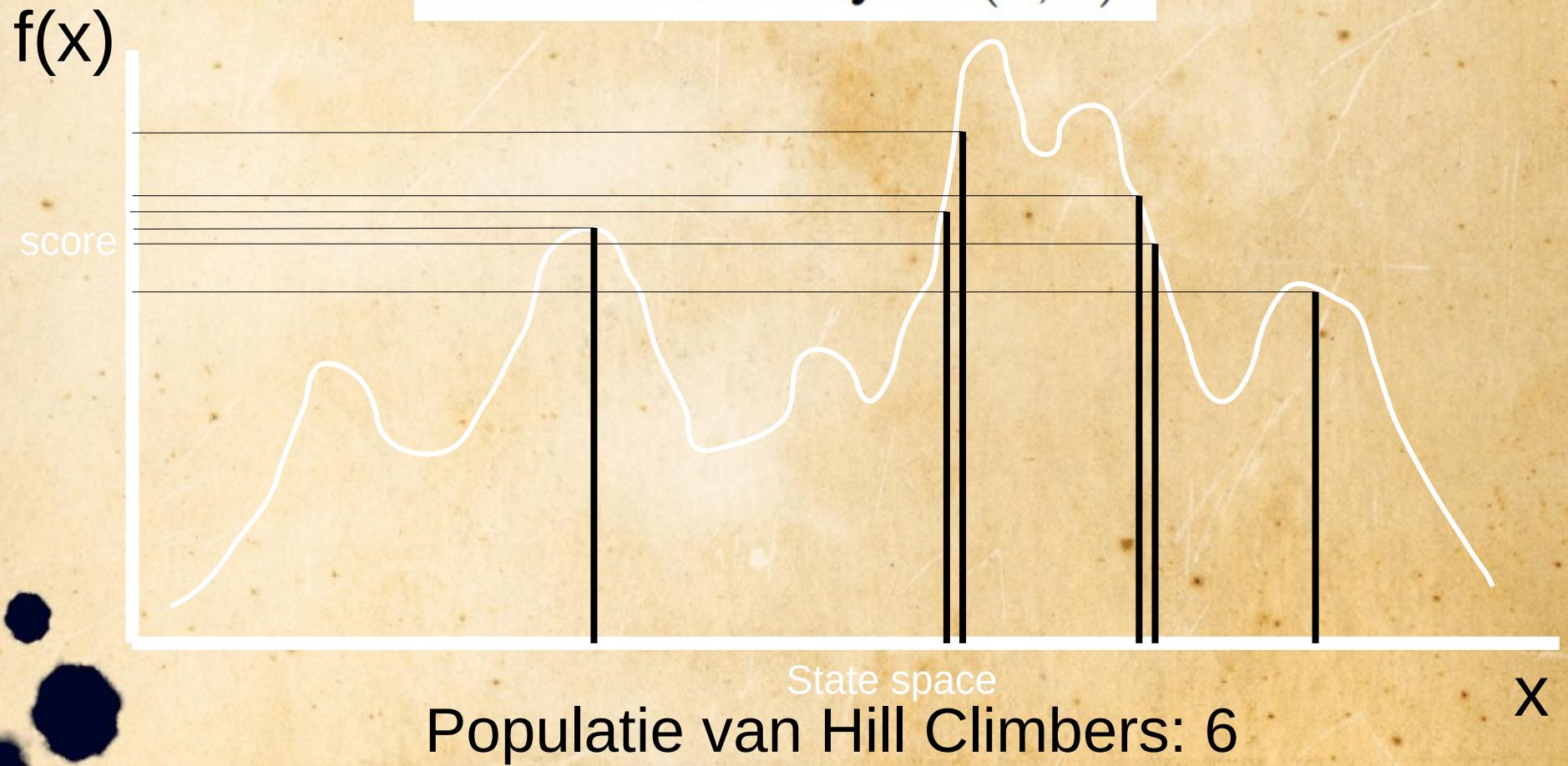
end for

return P , the population of solutions

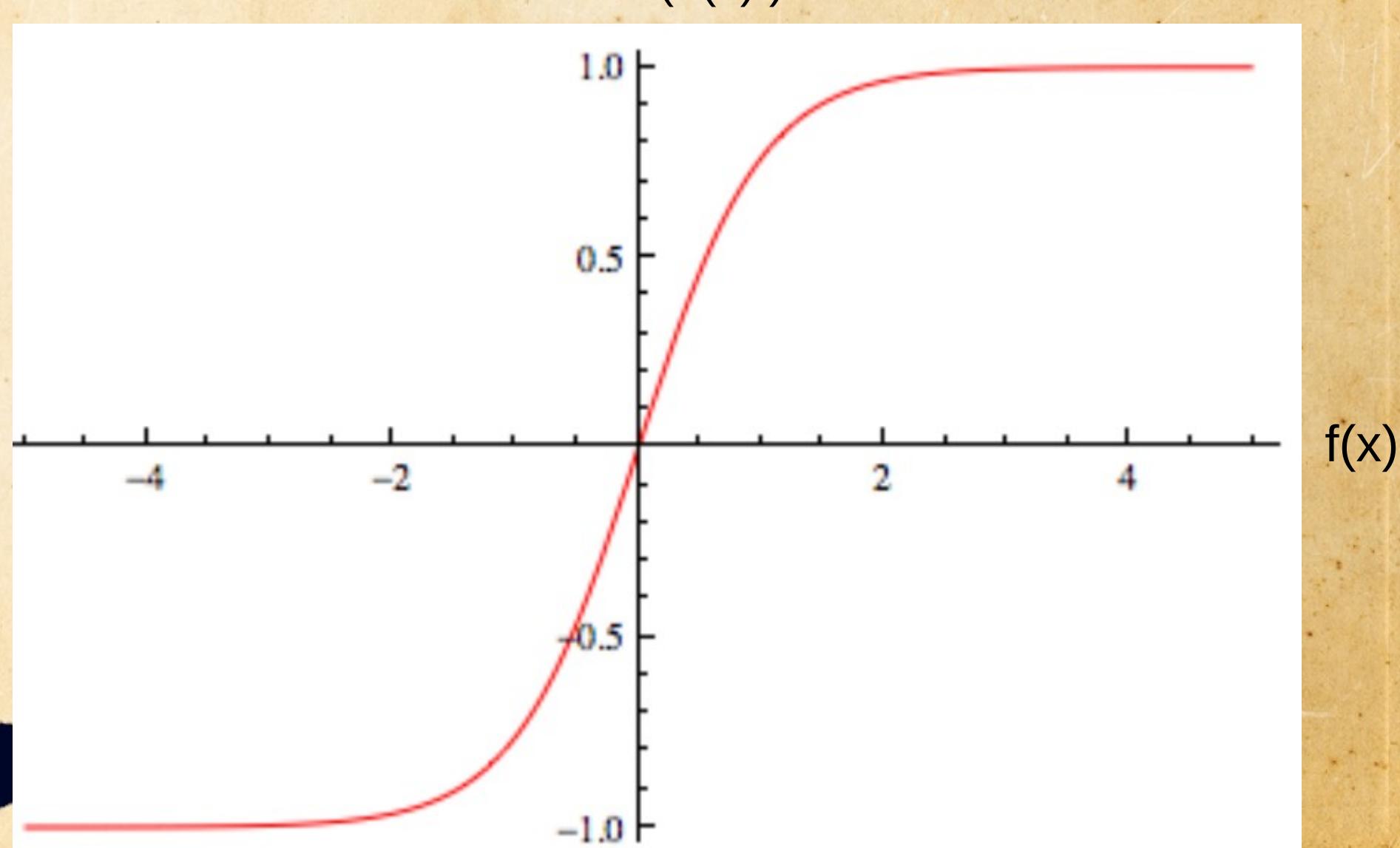
Plant Propagation Algorithm

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

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



Plant Propagation Algorithm



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

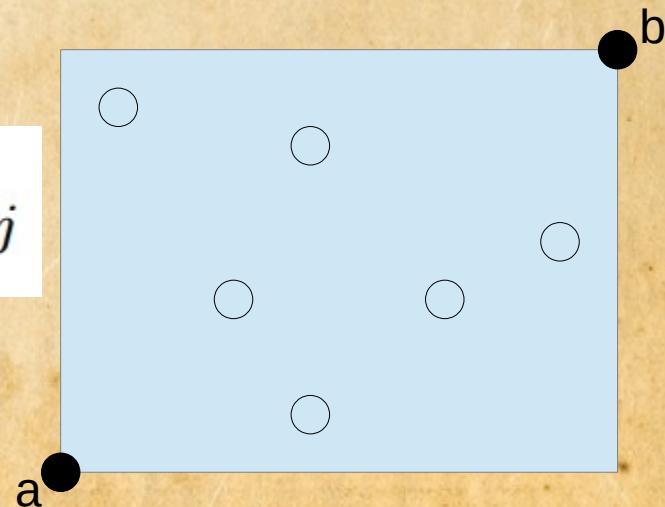
Distance of runner:

$$d_{r,j} = 2(1 - N_i)(r - 0.5)$$

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

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

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

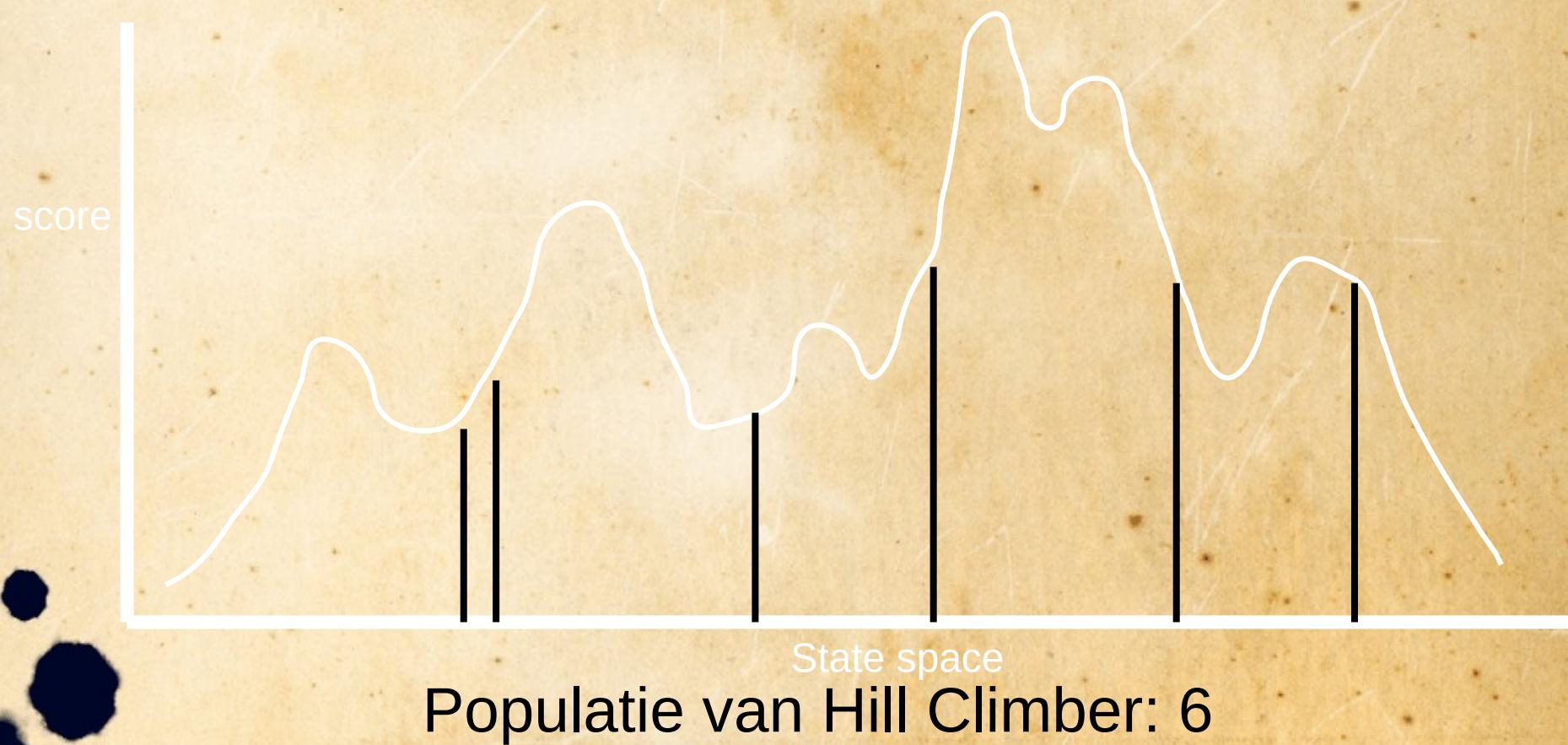


Genetic Algorithm

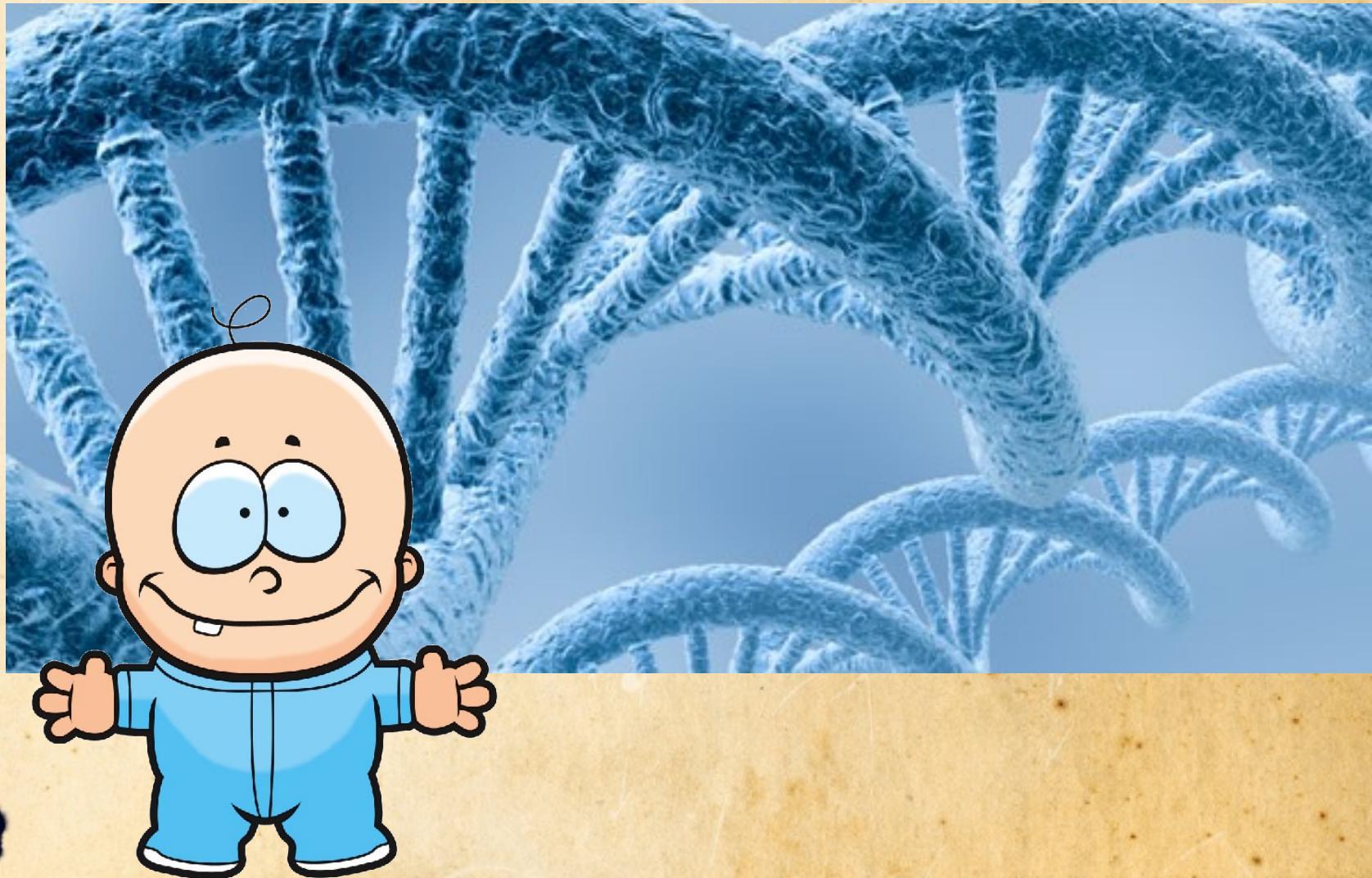


- Website:
 - https://www.tutorialspoint.com/genetic_algorithms

Genetic Algorithm

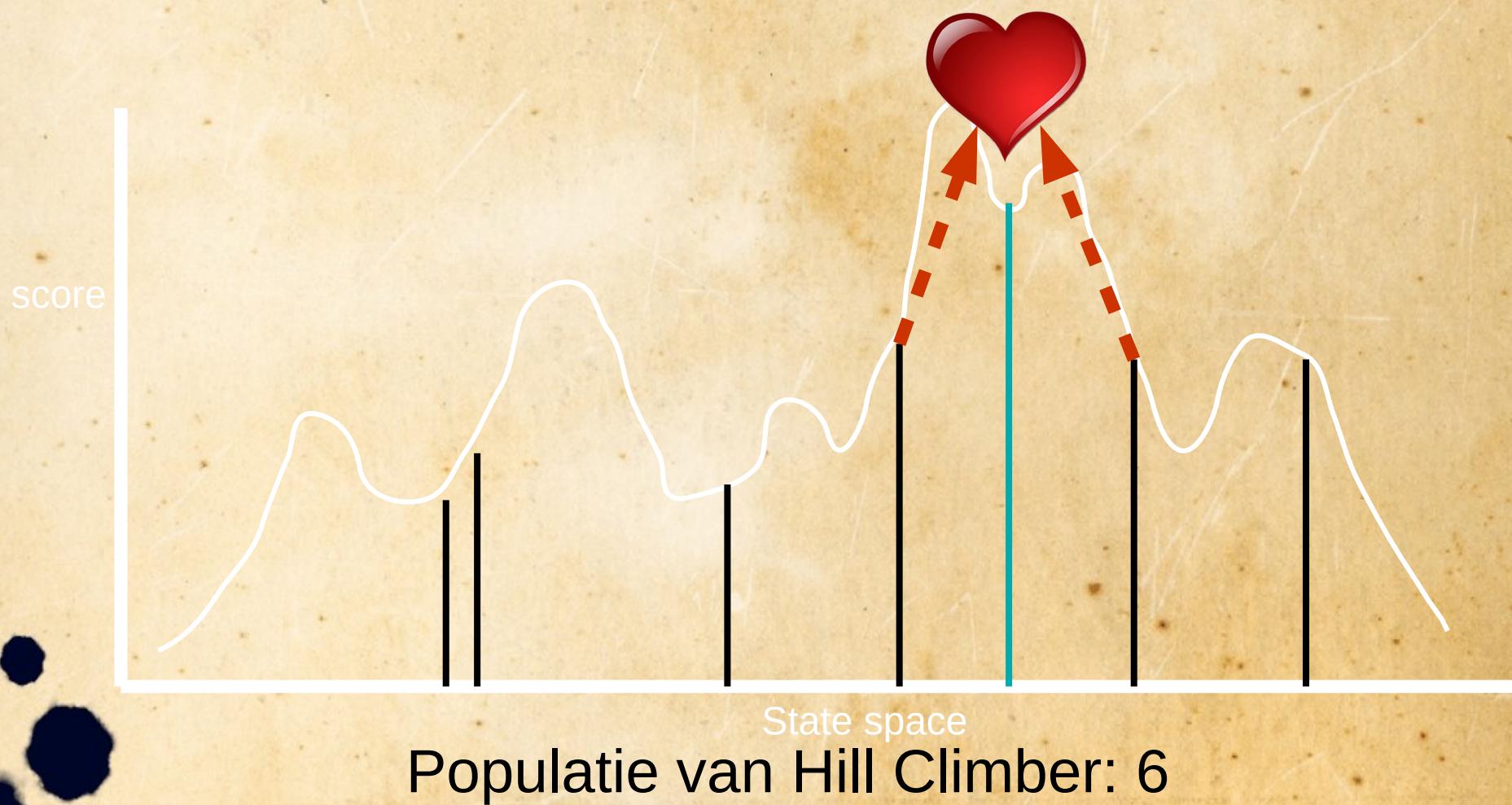


Genetic Algorithm

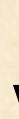


Genetic Algorithm

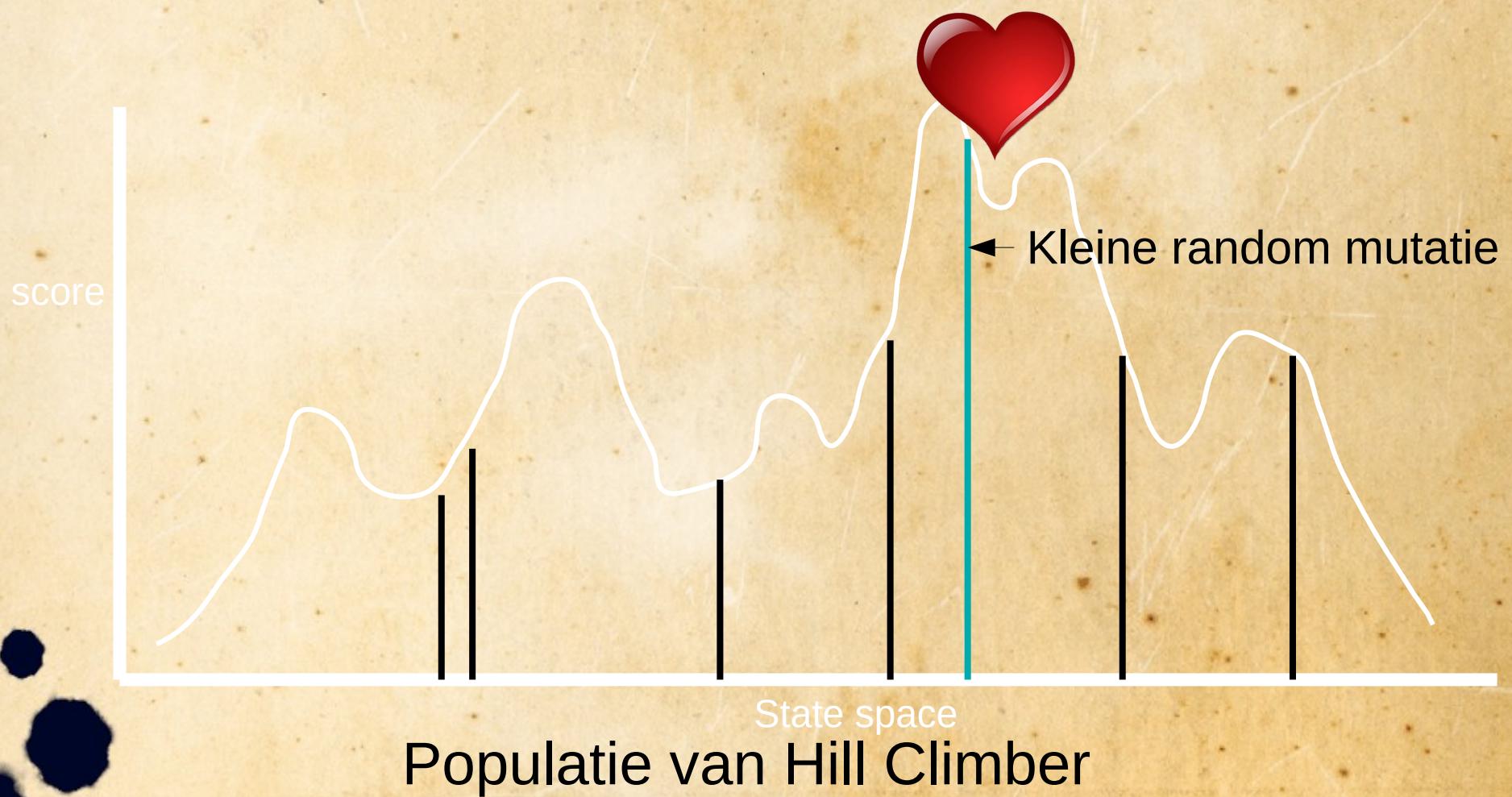
Crossover



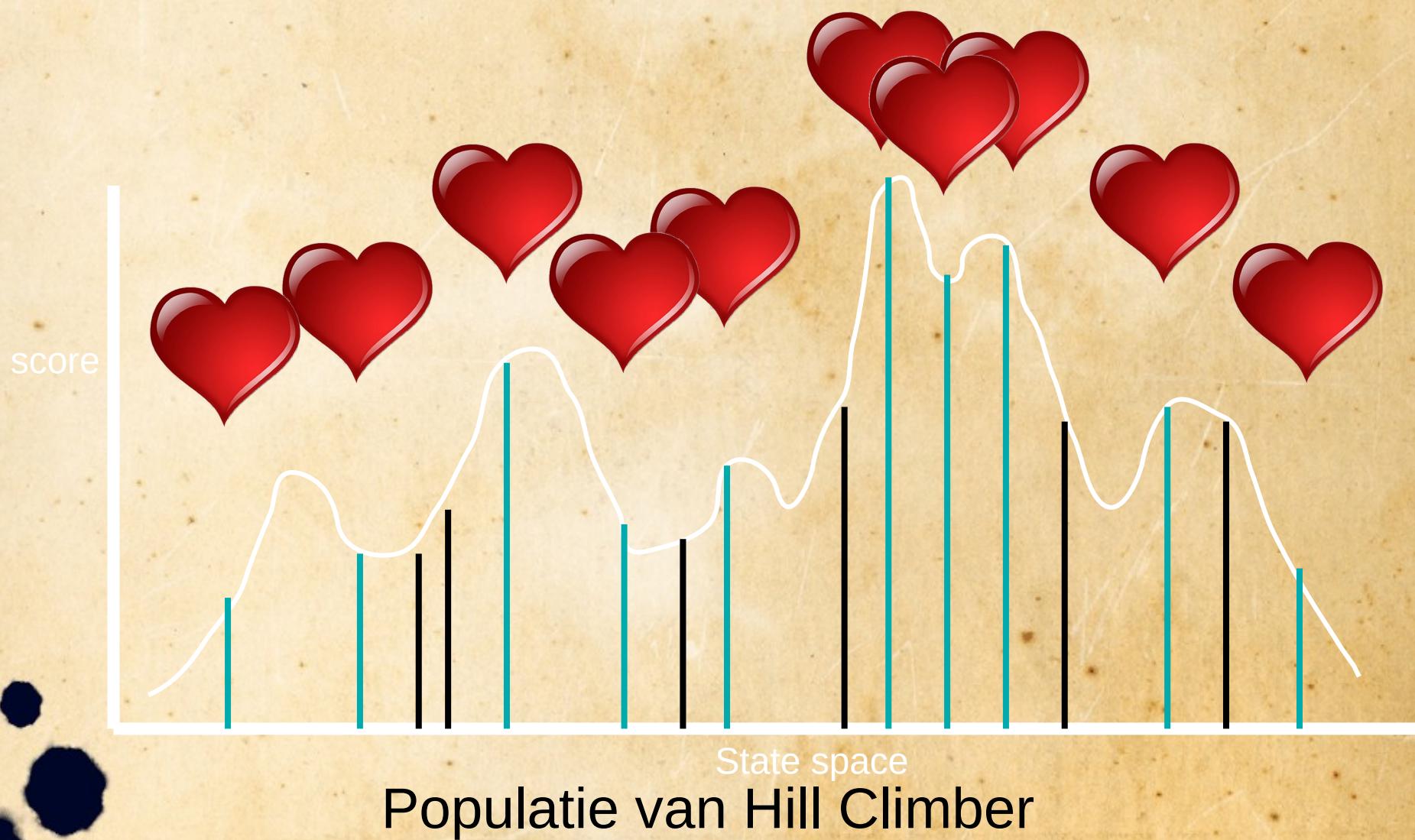
Genetic Algorithm



Genetic Algorithm

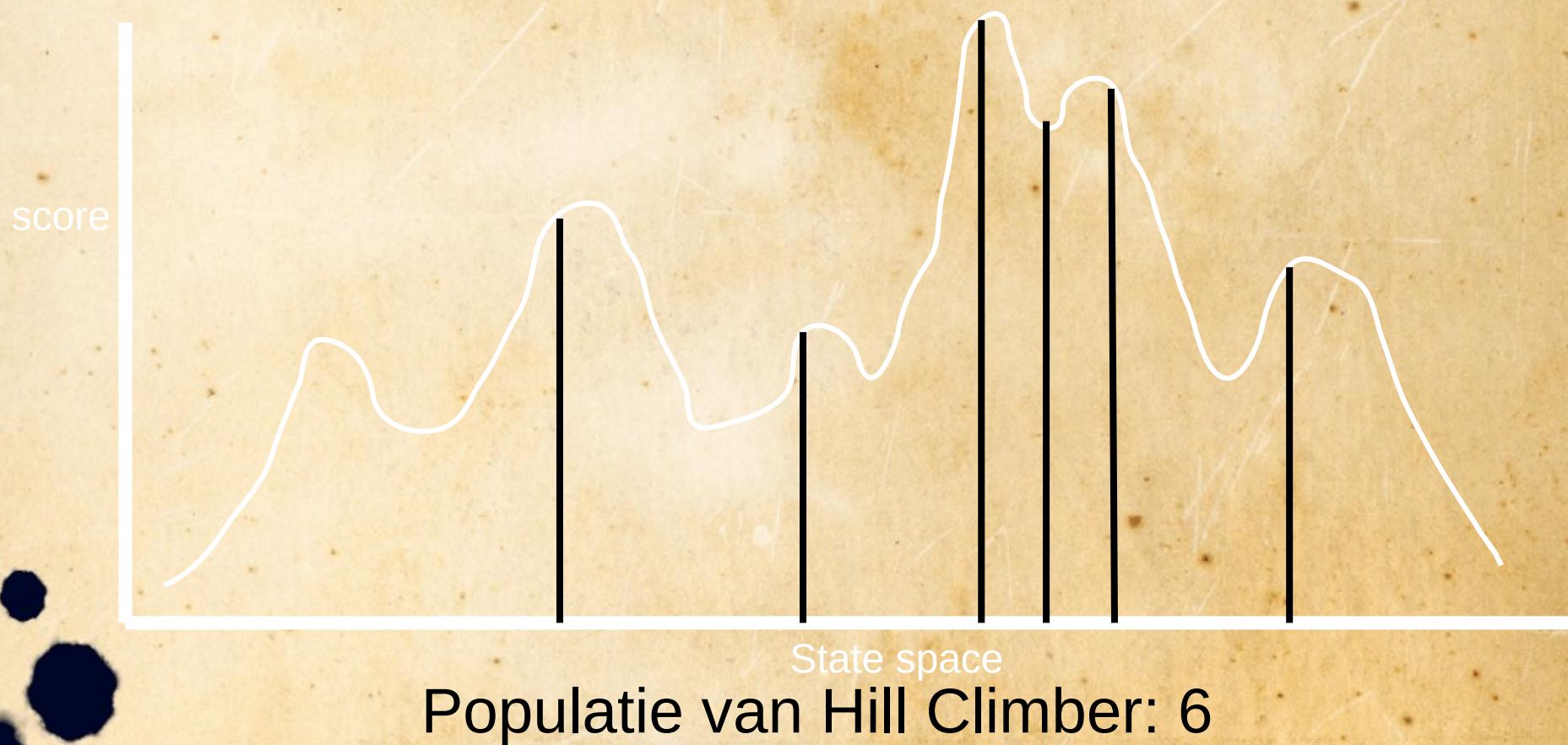


Genetic Algorithm

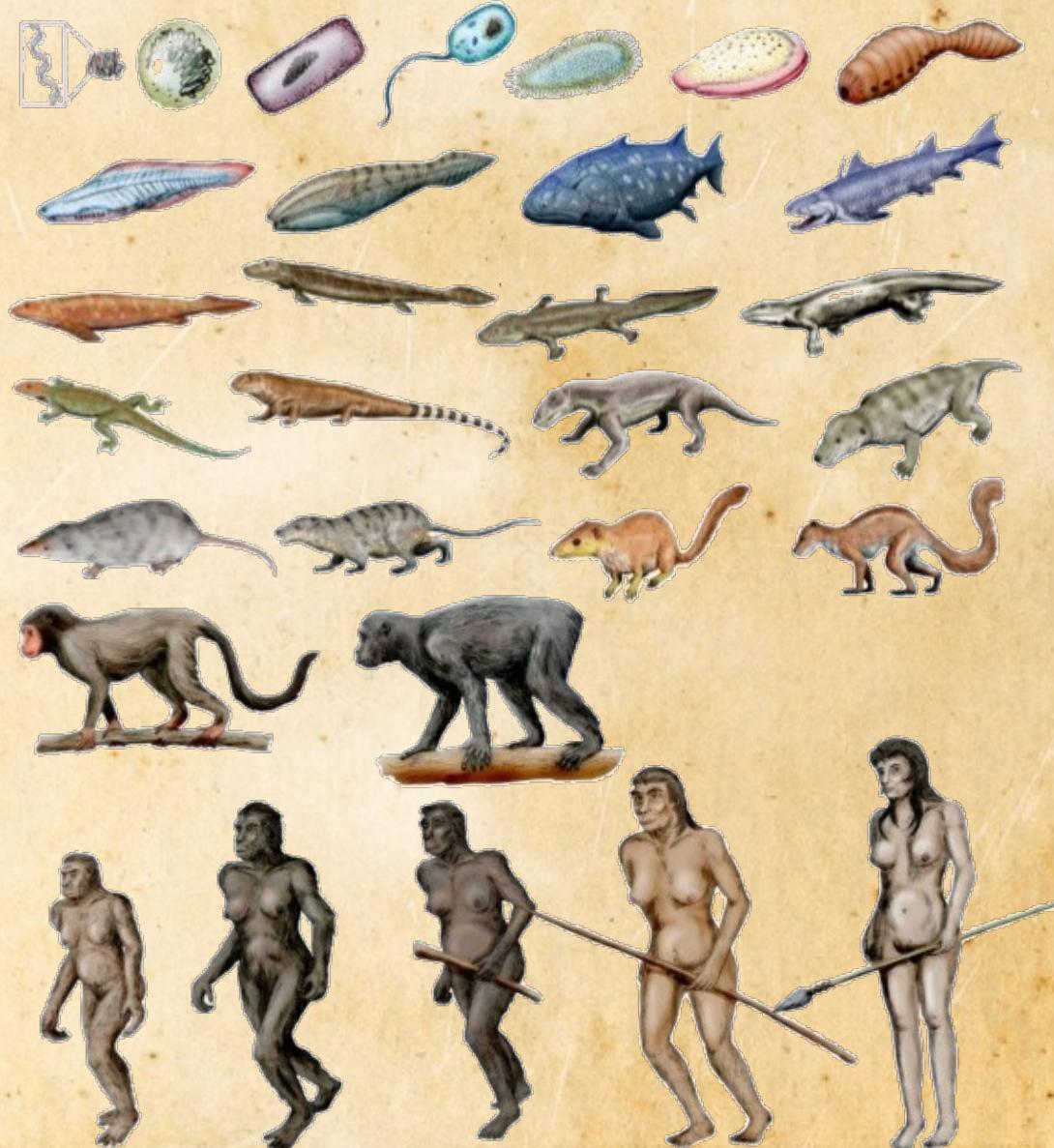


Genetic Algorithm

Select the best 6

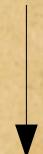
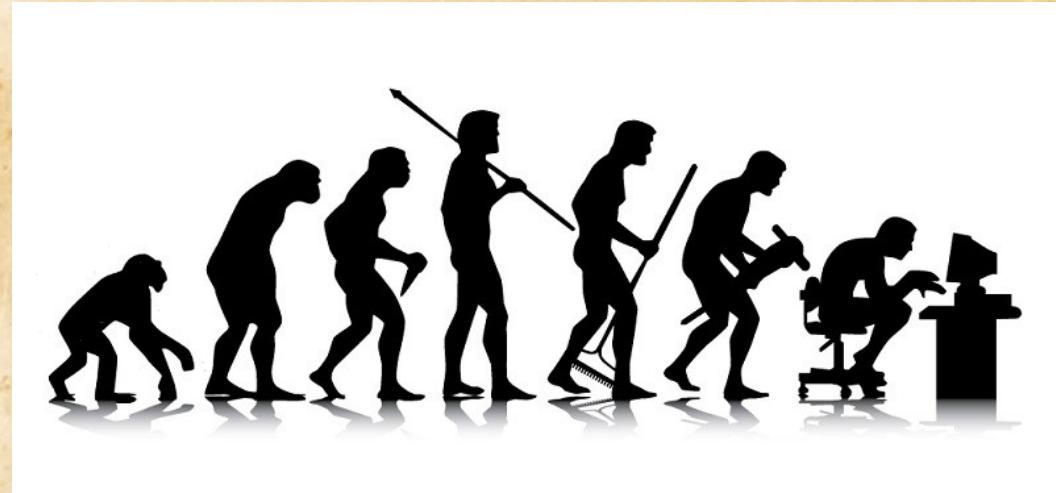


Genetic Algorithm



Genetic Algorithm

```
33     self.logpath = None
34     self.debug = debug
35     self.logger = logging.getLogger(__name__)
36     if path:
37         self.file = open(path, 'w')
38         self.file.write('')
39         self.fingerprints = {}
40
41     @classmethod
42     def from_settings(cls, settings):
43         debug = settings.getbool('logger', 'debug')
44         return cls(job_dir(settings), debug)
45
46     def request_seen(self, request):
47         tp = self.request_fingerprint(request)
48         if tp in self.fingerprints:
49             return True
50         self.fingerprints[tp] = 1
51         if self.file:
52             self.file.write(tp + '\n')
53             self.file.flush()
54         self.request_fingerprint(self, request)
```



Algoritmen en
Heuristieken

Simulation Hypothesis

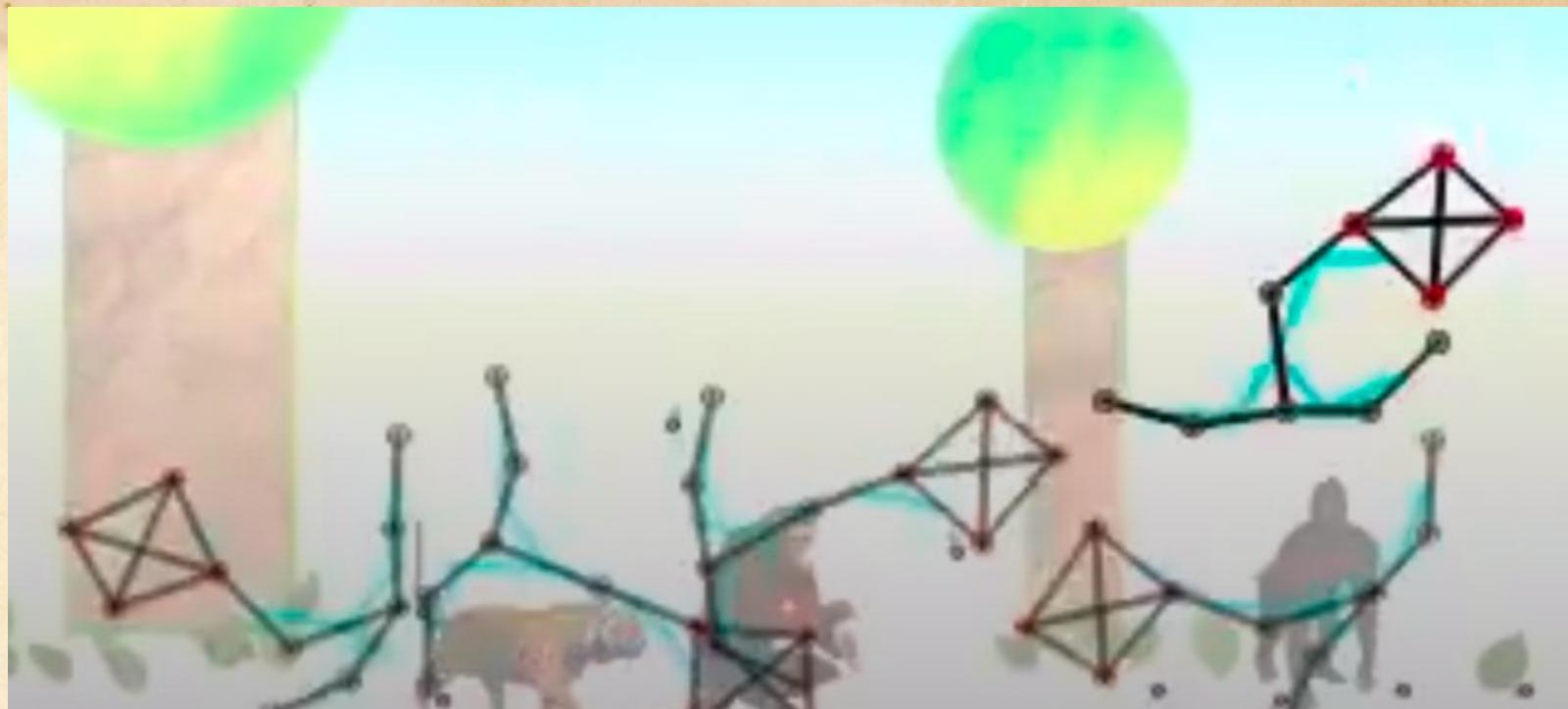


IS THIS A
SIMULATION?

Andere population based algoritmen

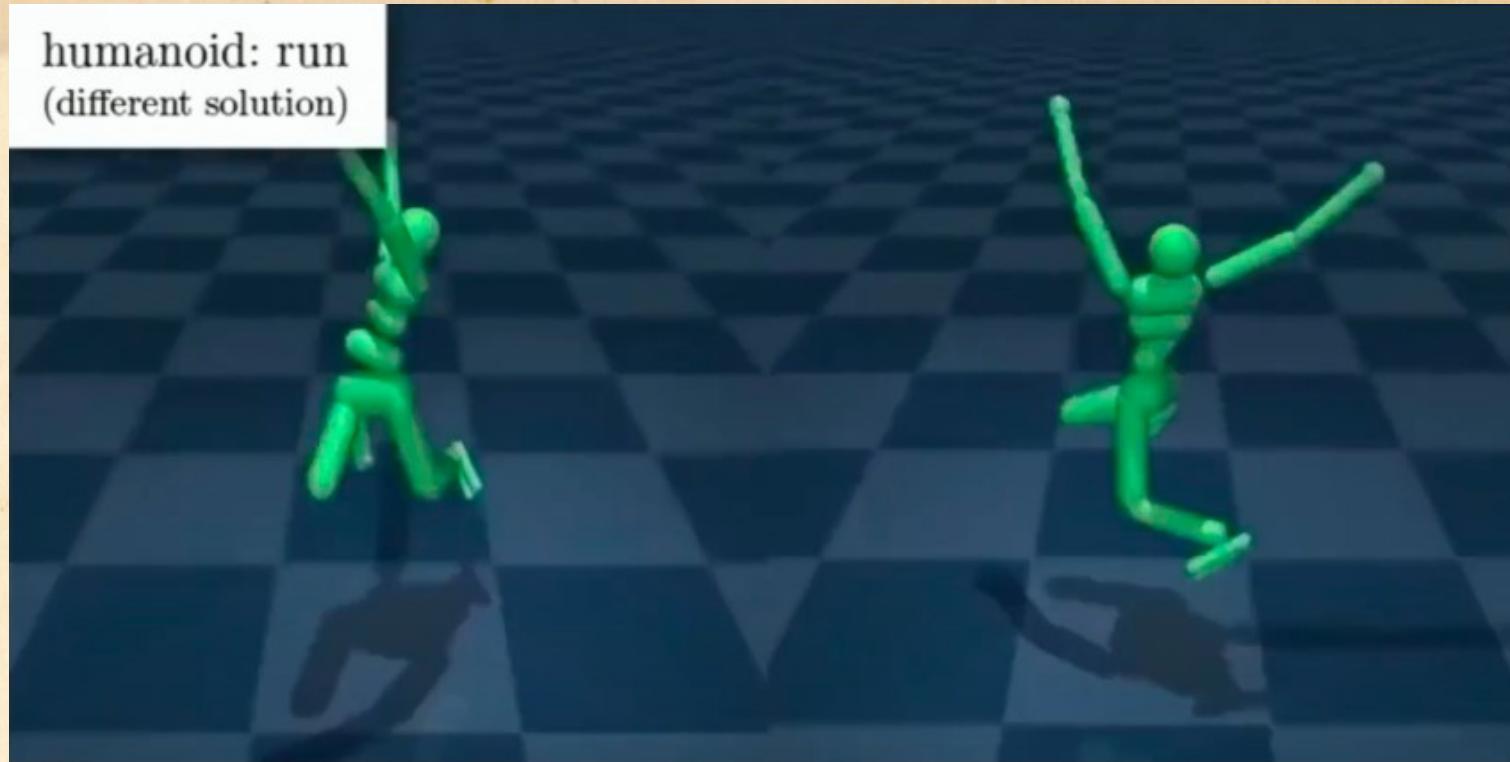
- Ant colony optimization
- Artificial immune system
- Bee colony optimization
- Brain storm optimization
- Fireworks algorithm
- Particle swarm optimization
- ...

Population based, Demo



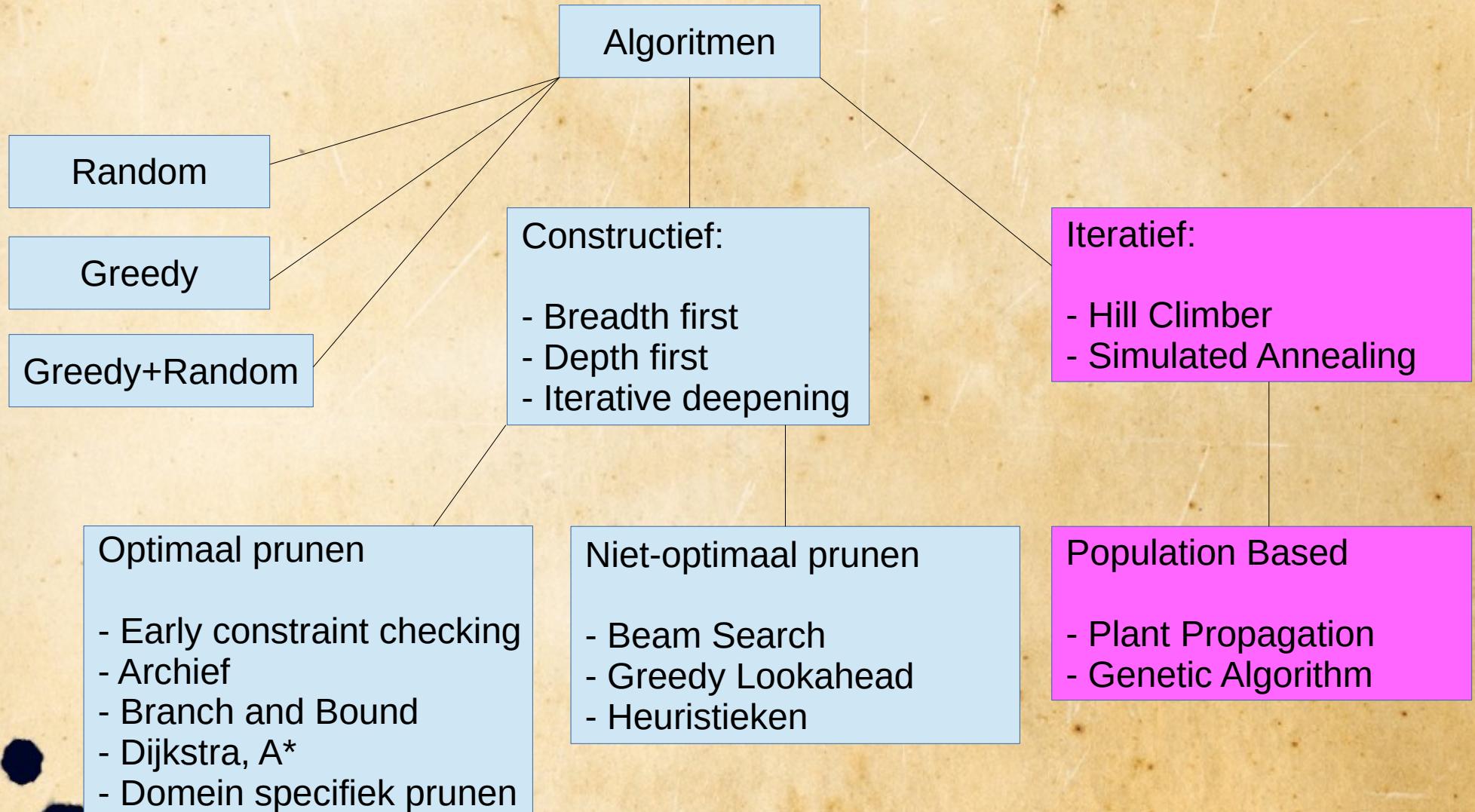
- Learn to walk, Roberto Mior (miorsoft.itch.io)
 - Youtube video:
 - <https://youtu.be/qtmG8mMgbpo>

Population based, solution?



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

Algoritmen



State of the art: zelf-lerende algoritmen

- Bijvoorbeeld: Deepmind

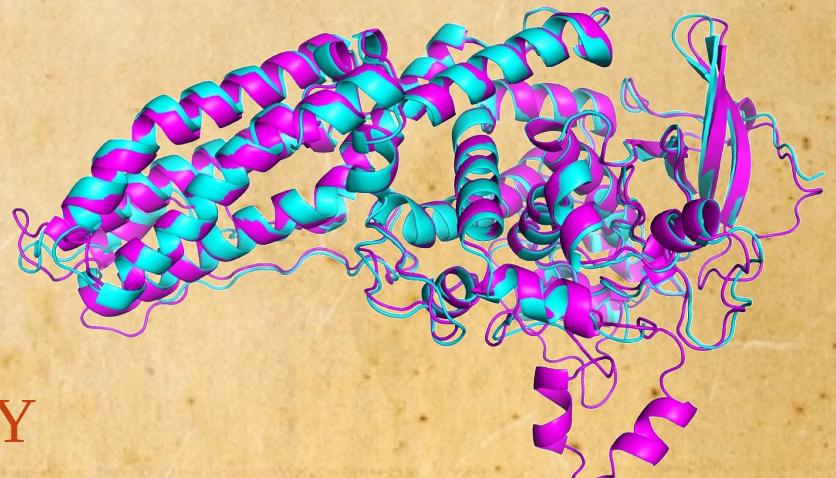
-AlphaGo - The Movie (1:30uur):

<https://youtu.be/WXuK6gekU1Y>



-AlphaFold (0:07uur):

<https://youtu.be/WXuK6gekU1Y>



State of the art: zelf-lerende algoritmen

- ChatGTP

<https://chat.openai.com/>



You

What is ChatGPT?



ChatGPT

ChatGPT is a conversational AI model developed by OpenAI, based on the GPT (Generative Pre-trained Transformer) architecture. It's designed to generate human-like text responses to natural language input, making it suitable for a wide range of natural language processing tasks, including answering questions, providing information, engaging in discussions, and more.

- Waymo

Self Driving Taxis (San Francisco)

<https://youtu.be/Y8qfHlpe31k>



Literatuur

Artificial Intelligence: A Modern Approach Third Edition

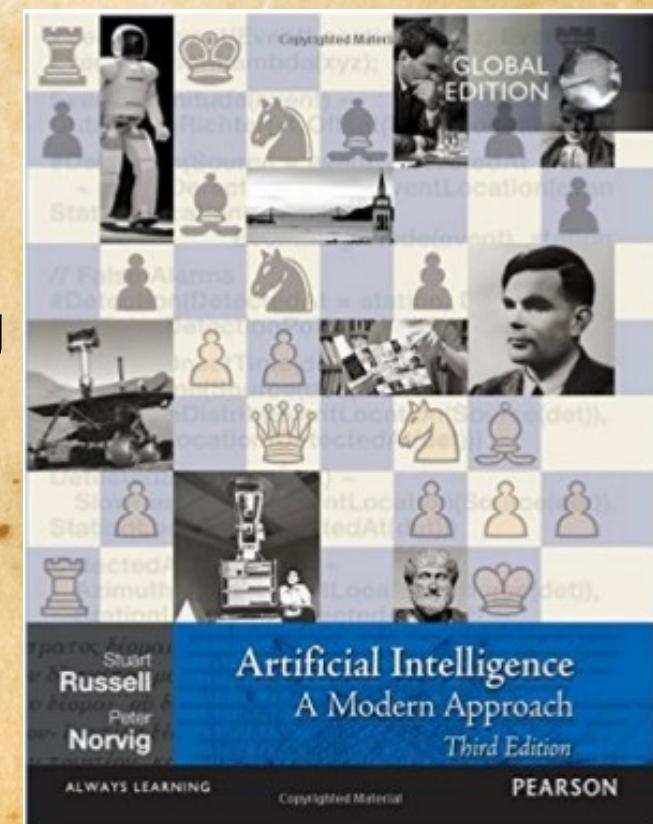
Stuart J. Russell and Peter Norvig

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>

Python Data Model

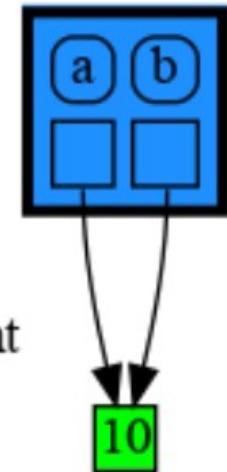
- Een verkeerd begrip van het Python Data Model kan leiden tot **bugs die moeilijk te fixen zijn!**
- Python Data Model, 2 verschillende soorten types:
 - **Immutable**: bool, int, float, complex, str, tuple, bytes, frozenset
 - **Mutable**: list, dict, set, class, ... (all other types)
- Gebruik **memory_graph** voor Data Model visualisatie:
 - Zie install: <https://pypi.org/project/memory-graph/>

Immutable Type: int

```
import memory_graph  
memory_graph.config.not_node_types.remove(int)  
  
a = 10  
b = a  
  
memory_graph.render(locals(), 'immutable1.png')  
a += 1  
  
memory_graph.render(locals(), 'immutable2.png')
```

Waarden van immutable type kunnen **niet** ter plekken worden aangepast, maar worden bij aanpassing gekopieerd.

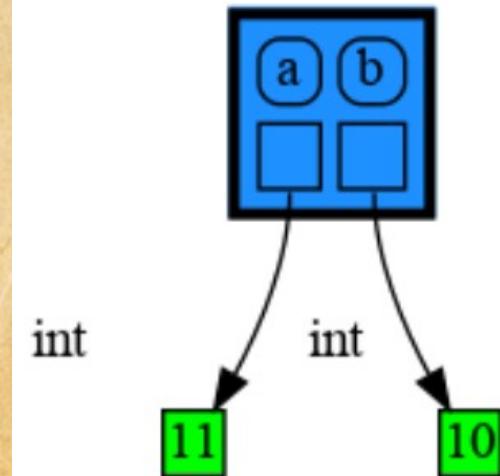
dict



int

immutable1.png

dict



int

int

immutable2.png

Mutable Type: list

```
import memory_graph

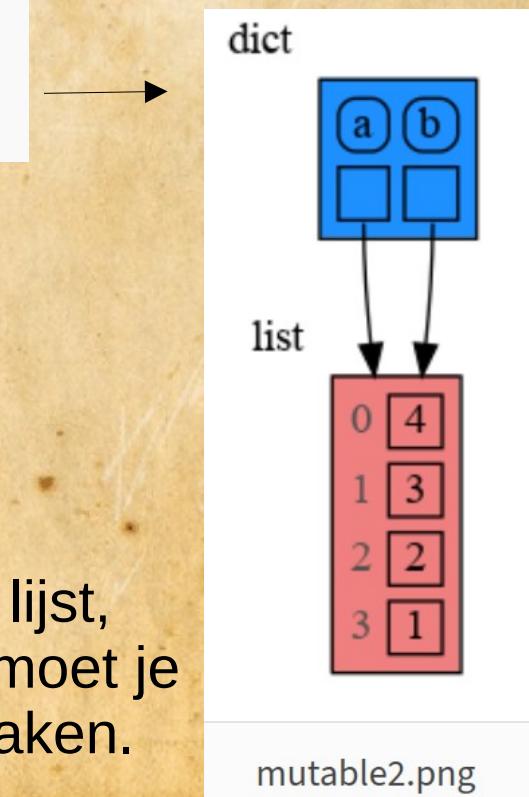
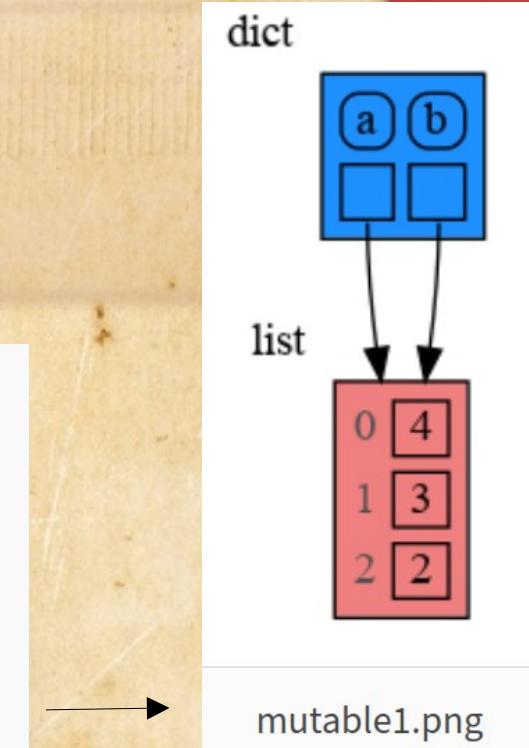
a = [4, 3, 2]
b = a

memory_graph.render(locals(), 'mutable1.png')
a.append(1)

memory_graph.render(locals(), 'mutable2.png')
```

Waarden van mutable type kunnen **wel** ter plekken worden aangepast, er wordt dus niet eerst een kopie gemaakt (het zou traag zijn om bij elke lijst-aanpassing eerst een kopie van de lijst te moeten maken).

‘a’ en ‘b’ delen de lijst,
als je dat niet wil moet je
zelf een kopie maken.

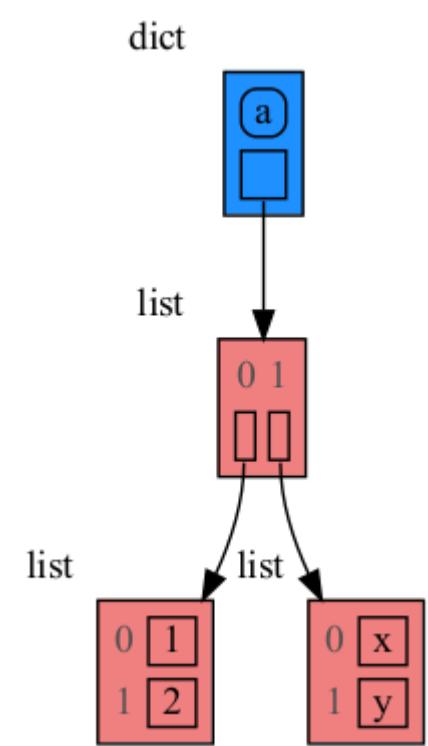


Mutable Type: copy nested list

```
import memory_graph
import copy

a = [ [1, 2], ['x', 'y'] ] # a nested list (a list containing lists)

memory_graph.show(locals())
```



Mutable Type: 3 different copies

```
import memory_graph
import copy

a = [ [1, 2], ['x', 'y'] ] # a nested list (a list containing lists)

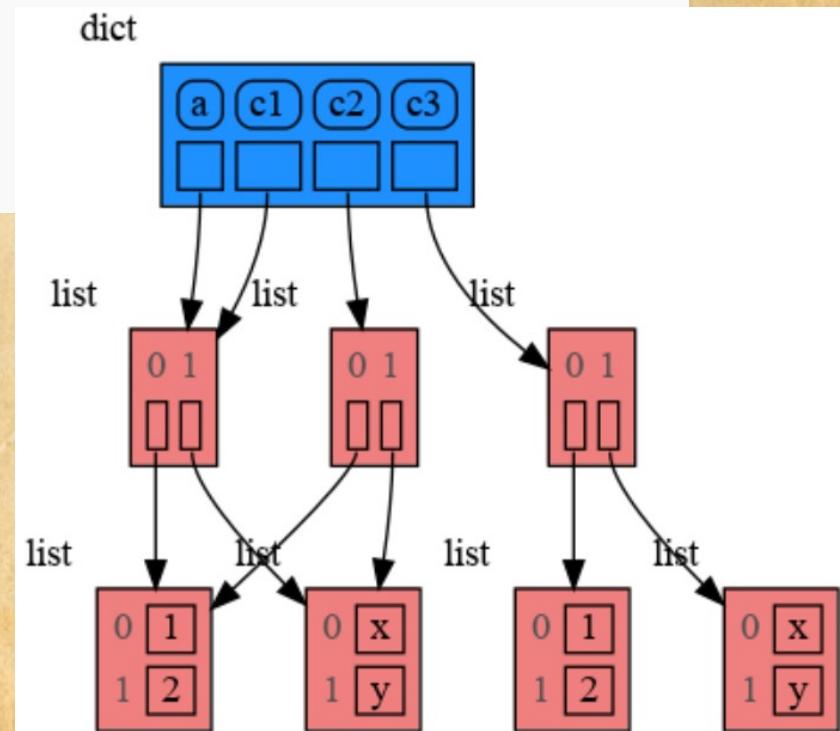
# three different ways to make a "copy" of 'a':
c1 = a
c2 = copy.copy(a) # equivalent to: a.copy() a[:] list(a)
c3 = copy.deepcopy(a)

memory_graph.show(locals())
```

c1: assignment, alles is gedeeld

c2: shallow copy, eerst verwijzing wordt gekopieerd
maar onderliggende is gedeeld

c3: deep copy, alles wordt gekopieerd



Eigen copy methode

```
import memory_graph
import copy

class My_Class:

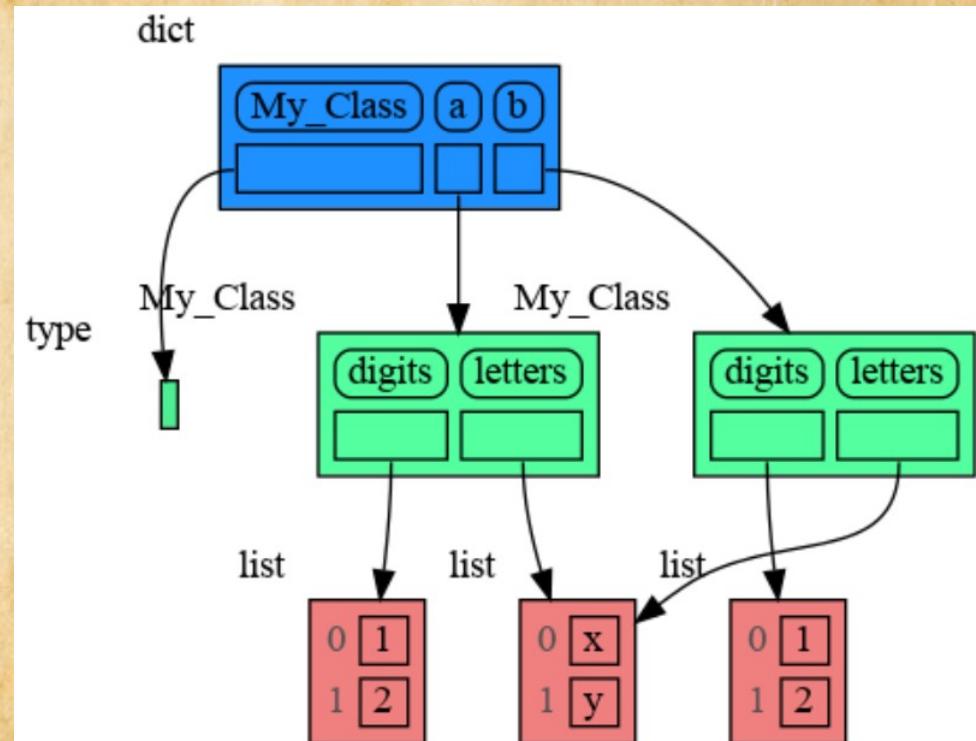
    def __init__(self):
        self.digits = [1, 2]
        self.letters = ['x', 'y']

    def copy(self): # custom copy method
        c = copy.copy(self)
        c.digits = copy.copy(self.digits)
        return c

a = My_Class()
b = a.copy()

memory_graph.show(locals())
```

De eigen copy methode kopieert de 'digits' maar deelt de 'letters'



Python Data Model

- Mental model: heb een plaatje in je hoofd met pijltjes bij mutable types.
- Als je een waarde van mutable type kopieert, bedenk dan welke “copy” je nodig hebt:
 - Assignment: alles gedeeld
 - Shallow copy: eerste verwijzing gekopieerd, rest gedeeld
 - Deep copy: alles gekopieerd
 - Custom copy: zelf te bepalen
- Als er bugs zijn: gebruik `memory_graph` om een plaatje te maken van al je data om te zien wat wordt gedeeld:
 - Zie: <https://pypi.org/project/memory-graph/>

The End

- We hebben nog wat tijd over. Opties:
 - 1: Verlaat de zaal
- Blijf nog even om:
 - 2: Te overleggen over welke algoritmen handig zouden kunnen zijn voor uw specifieke case
 - 3: Op je laptop te experimenteren met **memory_graph**:
 - Run voorbeeldcode: <https://pypi.org/project/memory-graph/>
 - Visualiseer de data van je eigen software
- Hoe dan ook: veel succes met uw case!
- Zijn er nog klassikale vragen?