

RACIOCÍNIO AUTOMÁTICO PARA OPTIMIZAÇÃO

Luís Morgado

ISEL-ADEETC

OPTIMIZAÇÃO

- **Seleccção da melhor opção a partir de um conjunto de opções possíveis** de acordo com um critério de avaliação de opções
 - Maximização
 - Minimização
- **Critério de avaliação**
 - Medida de ganho (*performance*)
 - Medida de perda (*loss*)
 - Medida de adequação (*fitness*)

OPTIMIZAÇÃO

Definição formal

A – Conjunto de opções possíveis

f – Função de avaliação (define critério de avaliação)

$$f: A \rightarrow \mathbb{R}$$

$$\mathbf{x}_0 \in A$$

$$f(\mathbf{x}_0) \geq f(\mathbf{x}) \quad \forall \mathbf{x} \in A$$

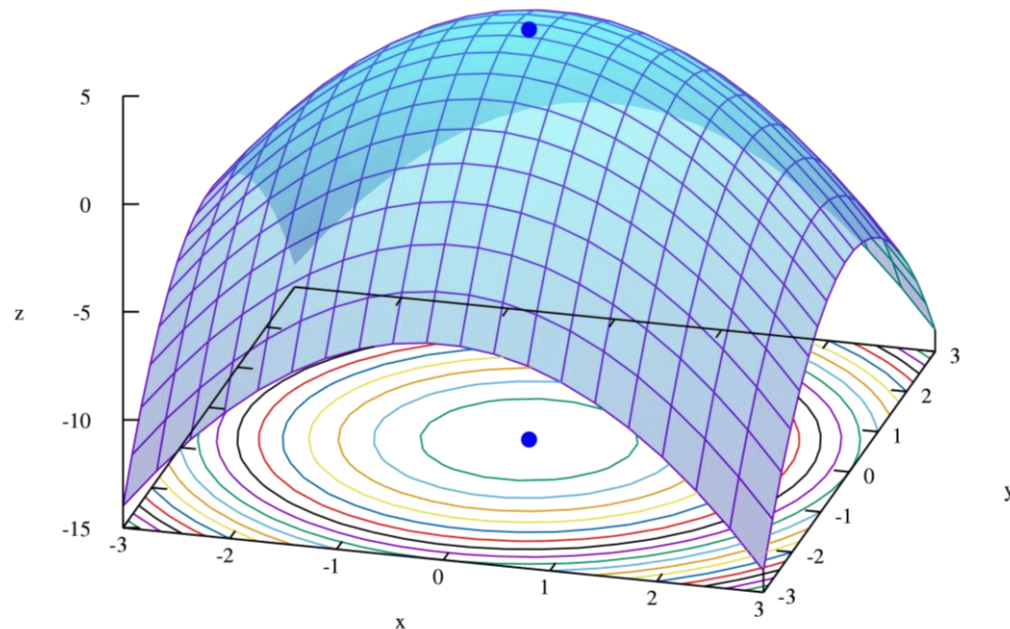
$$f(\mathbf{x}_0) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in A$$

OPTIMIZAÇÃO

Exemplo

$$z = f(x, y) = -(x^2 + y^2) + 4$$

Máximo global: $(x, y) = (0, 0)$, $z = 4$

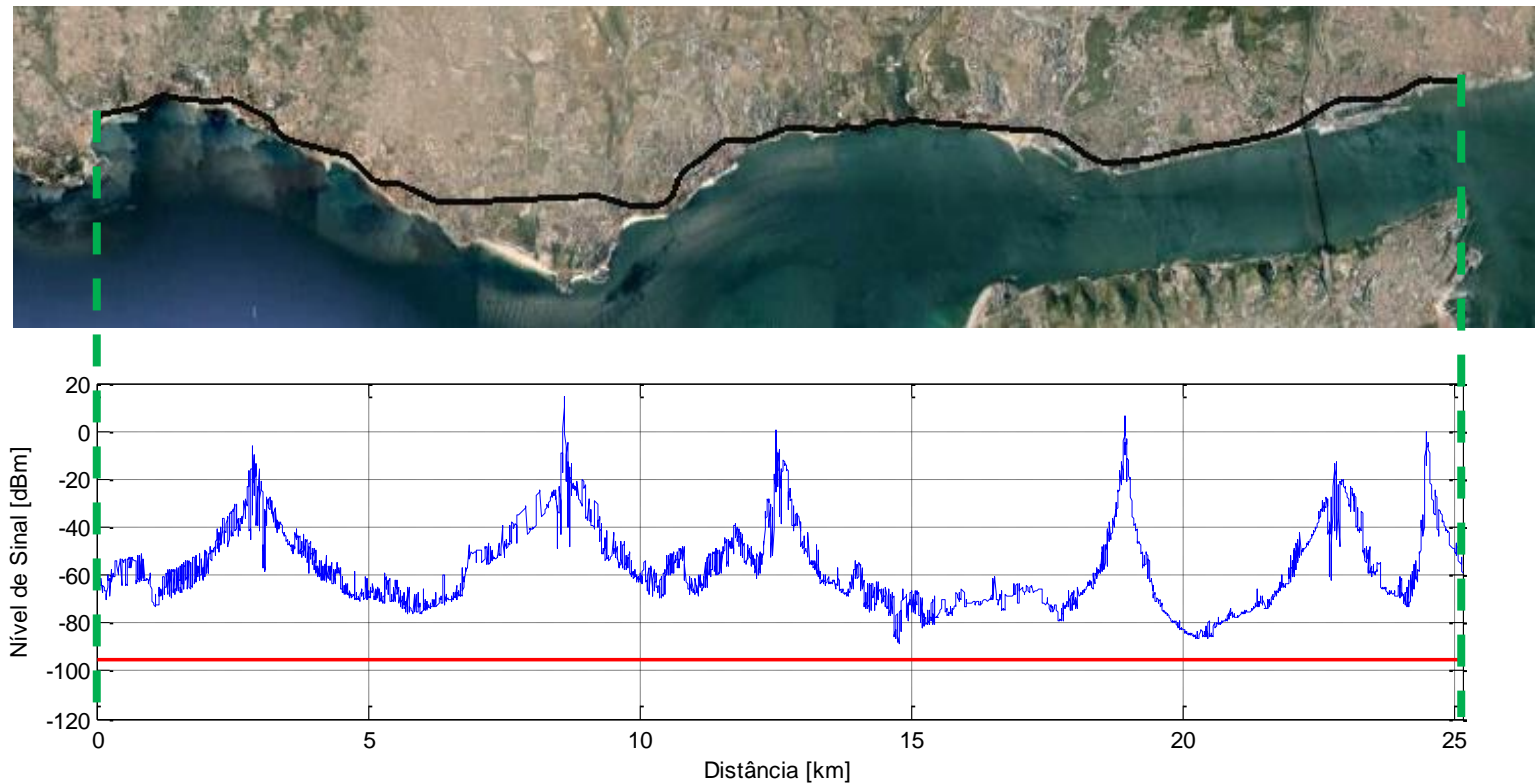


OPTIMIZAÇÃO

- **Métodos computacionais de otimização**
 - **Otimização matemática**
 - Exemplos
 - Otimização linear
 - Descida de gradiente
 - **Meta-heurísticas**
 - Formas de procura em espaços de estados
 - Exemplos
 - Procura local sôfrega (*Hill-climbing*)
 - Têmpera simulada (*Simulated annealing*)
 - Algoritmos genéticos

Exemplo: Otimização de Cobertura Rádio

- Otimizar o nível de sinal ao longo da linha através da estimação das estações base necessárias para garantir cobertura em toda a linha



Exemplo: Otimização de Estruturas

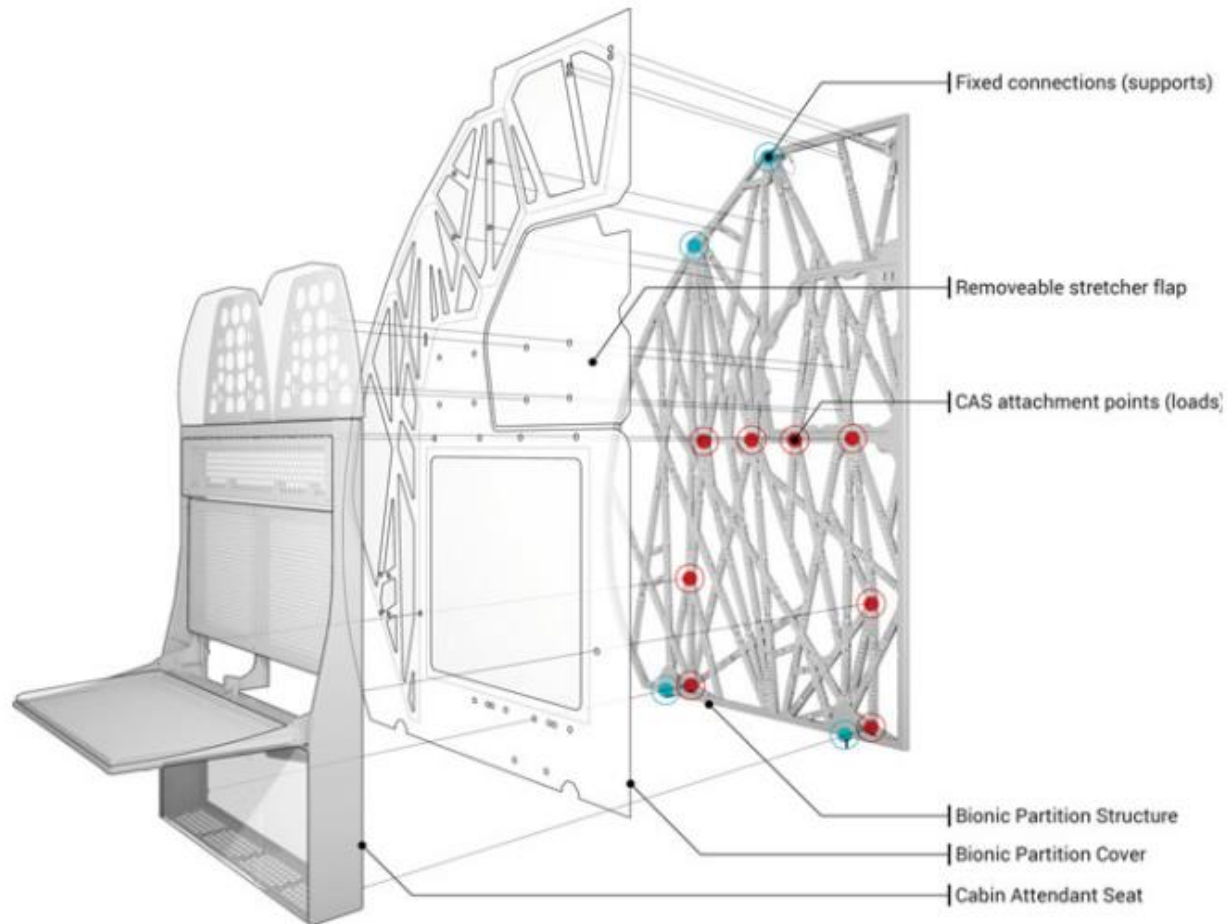
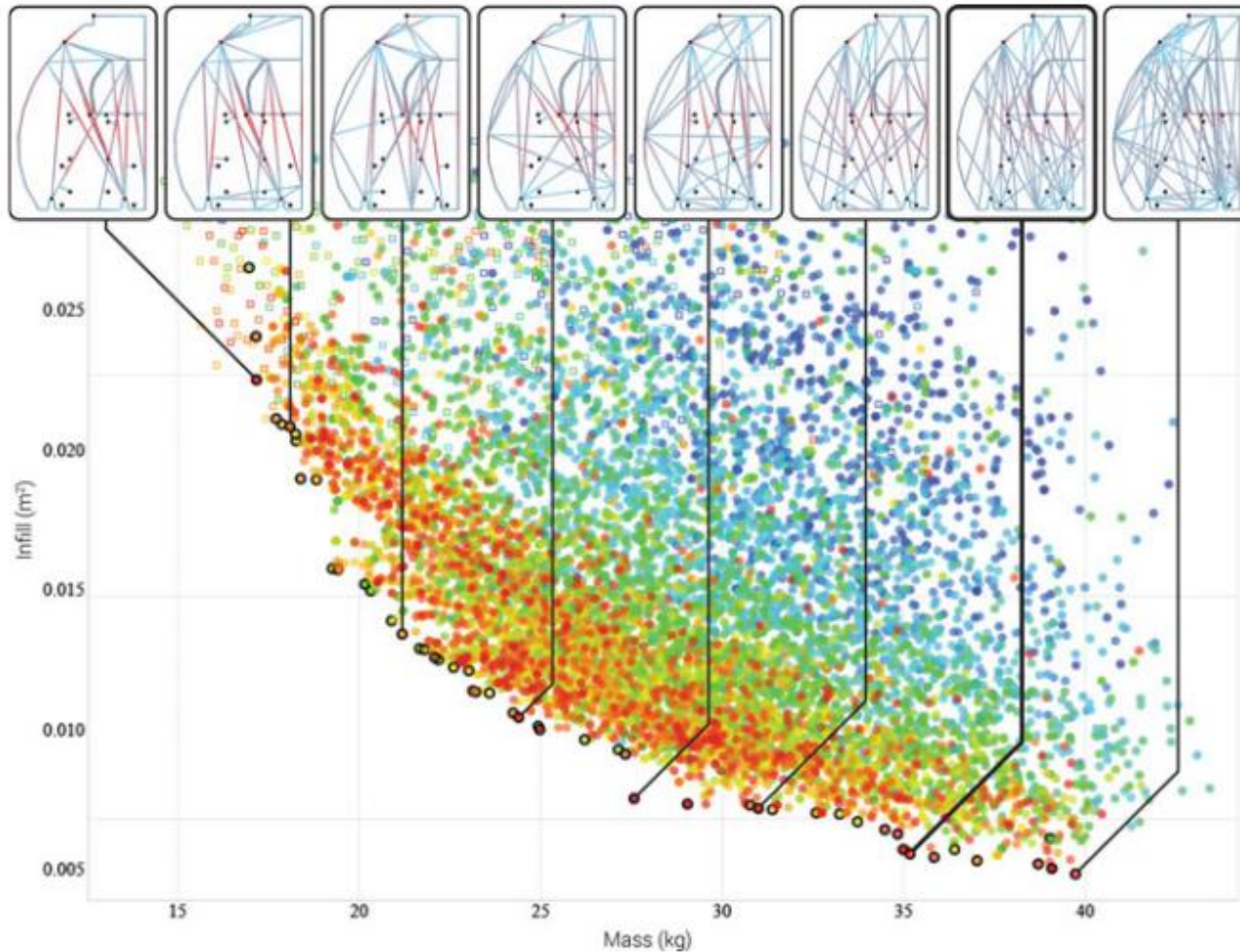


Fig. 1. Description of aircraft cabin partition design problem

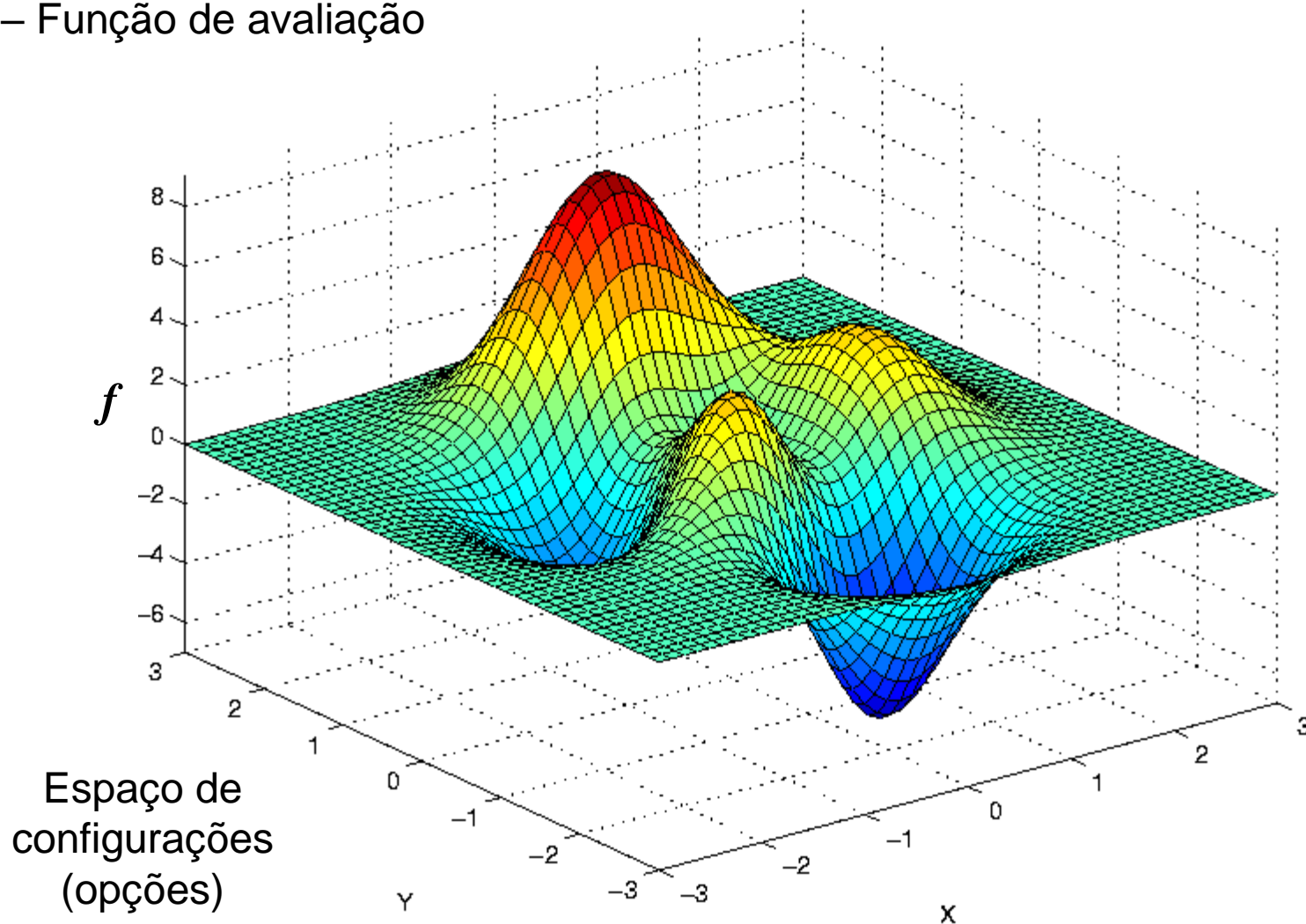
Exemplo: Otimização de Estruturas



OPTIMIZAÇÃO

CRITÉRIO DE AVALIAÇÃO

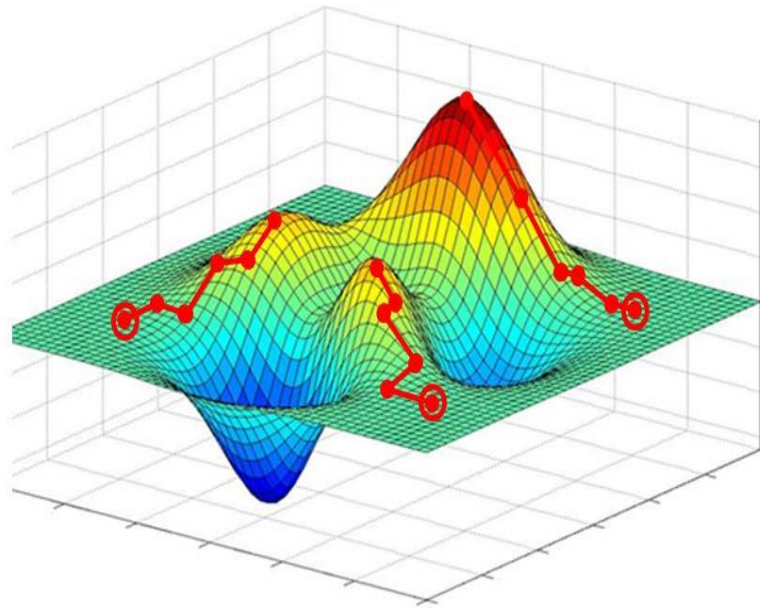
f – Função de avaliação



RACIOCÍNIO AUTOMÁTICO

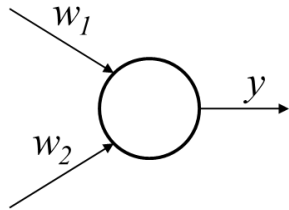
RESOLUÇÃO DE PROBLEMAS DE OPTIMIZAÇÃO

- Saber qual a melhor configuração de parâmetros de modo a maximizar uma função de valor ou de adequação
- Resultado: Configuração de parâmetros (estado)



Procura exaustiva não é viável → Procura local

Exemplo: Redes neuronais artificiais



Configuração: $\mathbf{w} = (w_1, w_2)$

↓
Estado

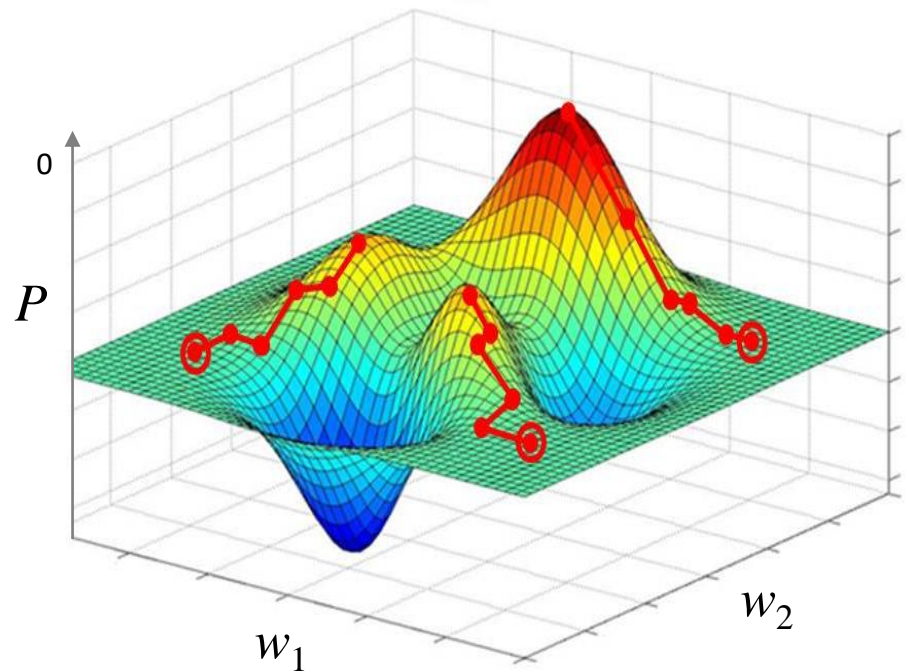
**Vector de transformação
de estado:**

$$\Delta \mathbf{w} = (\Delta w_1, \Delta w_2)$$

Critério de avaliação

– Medida de ganho (*performance*)

$$P = - \sum_s \left(\sum_z (d_{sz} - o_{sz})^2 \right)$$



RACIOCÍNIO ATRAVÉS DE PROCURA

RESOLUÇÃO DE PROBLEMAS DE OPTIMIZAÇÃO

– Estado

- Representa uma **configuração**
- **Espaço de estados**

– Transição

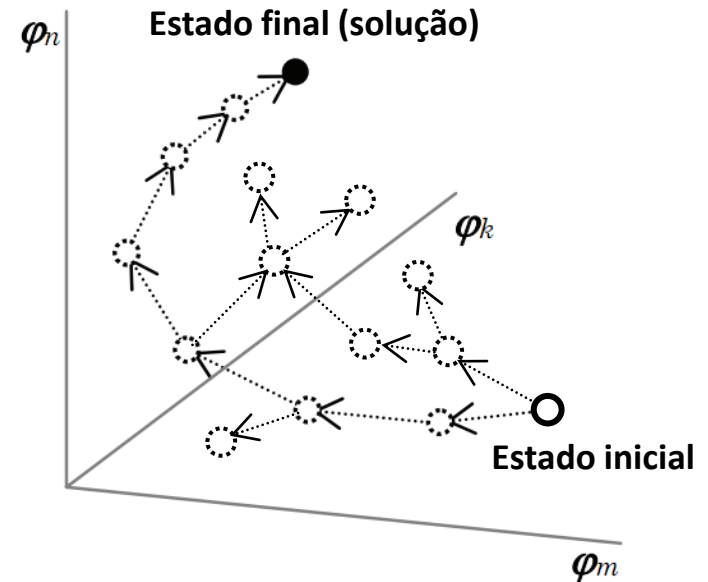
- Representa uma **transformação** de estado
 - **Operador** (de transição de estado)
 - **Vector** (de transição de estado)

– Valor

- Função de valor de estado

– Solução

- Estado final



PROCURA EM ESPAÇOS DE ESTADOS LOCAL

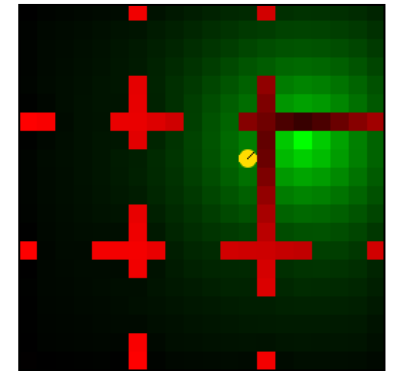
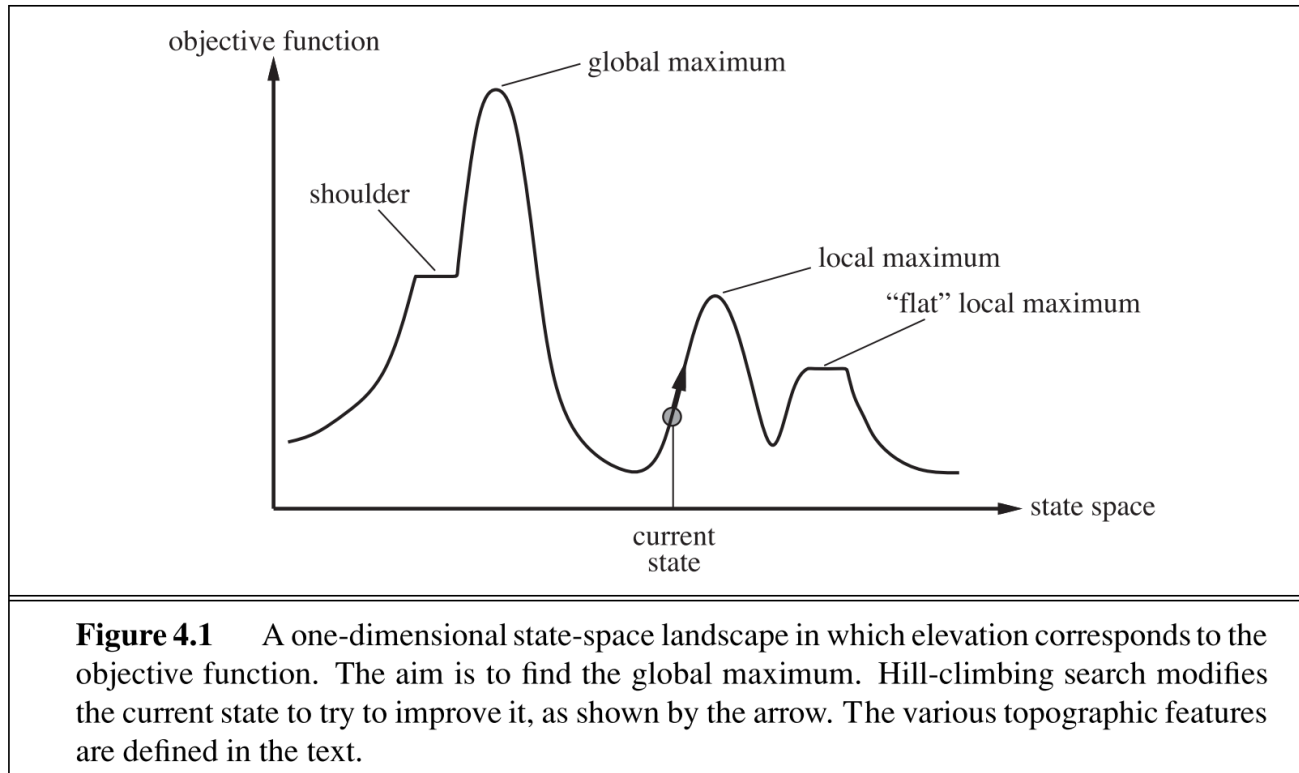
Algoritmo *Hill-Climbing*

```
function HILL-CLIMBING(problem) returns a state that is a local maximum  
  
  current  $\leftarrow$  MAKE-NODE(problem.INITIAL-STATE)  
  loop do  
    neighbor  $\leftarrow$  a highest-valued successor of current  
    if neighbor.VALUE  $\leq$  current.VALUE then return current.STATE  
    current  $\leftarrow$  neighbor
```

Figure 4.2 The hill-climbing search algorithm, which is the most basic local search technique. At each step the current node is replaced by the best neighbor; in this version, that means the neighbor with the highest VALUE, but if a heuristic cost estimate h is used, we would find the neighbor with the lowest h .

ALGORITMO *HILL-CLIMBING*

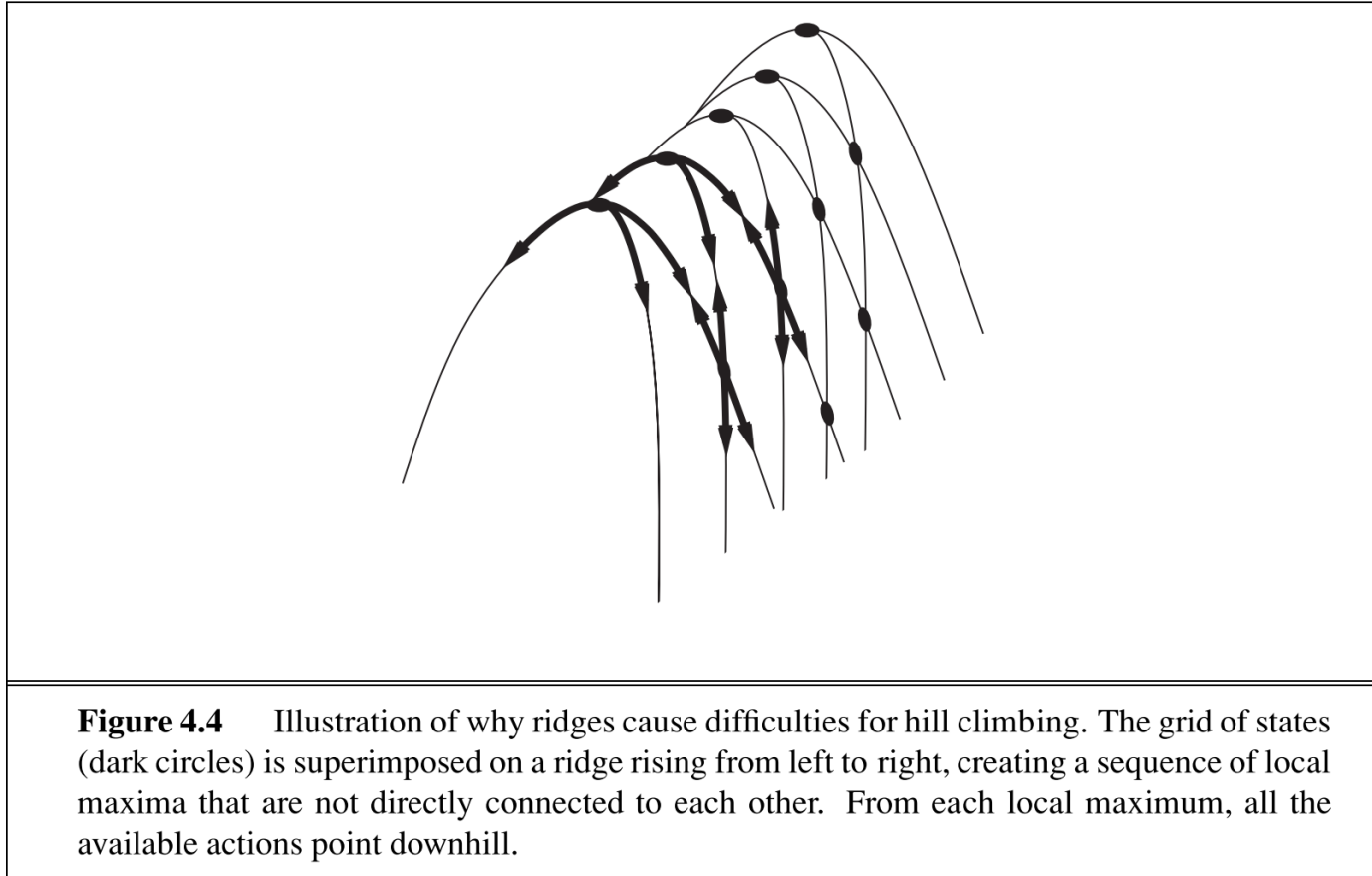
PROBLEMA: Óptimos Locais



[Russel & Norvig, 2010]

ALGORITMO *HILL-CLIMBING*

PROBLEMA: Óptimos Locais



ALGORITMO *HILL-CLIMBING*

Características

- **Completo**
 - Não é completo
- **Ótimo**
 - Não é ótimo
- **Complexidade**
 - Espacial
 - $O(b)$
 - Temporal
 - $O(d)$

PROCURA EM ESPAÇOS DE ESTADOS LOCAL

- **Soluções para o problema dos óptimos locais**
 - Manter memória de estados visitados
 - Complexidade espacial exponencial
 - **Não é viável para problemas gerais com espaços de estados de grande dimensionalidade**
 - Mecanismos de exploração não exaustiva
 - **Amostragem estocástica do espaço de estados**
 - **Controlo de *exploração / aproveitamento***
 - Exploração
 - » Seleccionar operador com um critério de base aleatória
 - Aproveitamento
 - » Gradiente - conhecimento local
 - » Seleccionar operador de seguida de gradiente
 - » Acção sôfrega (*greedy*)

ALGORITMO *HILL-CLIMBING*

Variantes

- ***Hill-Climbing* estocástico**
 - Escolha aleatória entre sucessores que aumentam o valor de estado
 - Convergência mais lenta que *Hill-climbing*
 - Pode encontrar melhores soluções
 - Consoante a topologia do espaço de estados
 - Não completo
- ***Hill-Climbing* estocástico com único sucessor**
 - Sucessores gerados aleatoriamente até um aumentar o valor de estado
 - Útil quando o número de sucessores por estado é muito elevado
 - Não completo
- ***Hill-Climbing* com reinício aleatório**
 - Procura é reiniciada a partir de estados iniciais aleatórios, até ser atingido um objectivo
 - Completo

PROCURA EM ESPAÇOS DE ESTADOS LOCAL

- **Métodos de têmpera simulada**

(Simulated Annealing)

- Adaptação dinâmica de parâmetros do método de procura
- Amostragem
- Restrição temporal
 - Soluções aproximadas
- Analogia com a têmpera do metal
 - Aquecimento
 - Flexibilidade
 - Arrefecimento
 - Restrição

PROCURA EM ESPAÇOS DE ESTADOS LOCAL

Algoritmo *Simulated Annealing*

function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state

inputs: *problem*, a problem

schedule, a mapping from time to “temperature”

current \leftarrow MAKE-NODE(*problem*.INITIAL-STATE)

for $t = 1$ **to** ∞ **do**

$T \leftarrow$ *schedule*(t)

if $T = 0$ **then return** *current*

next \leftarrow a randomly selected successor of *current*

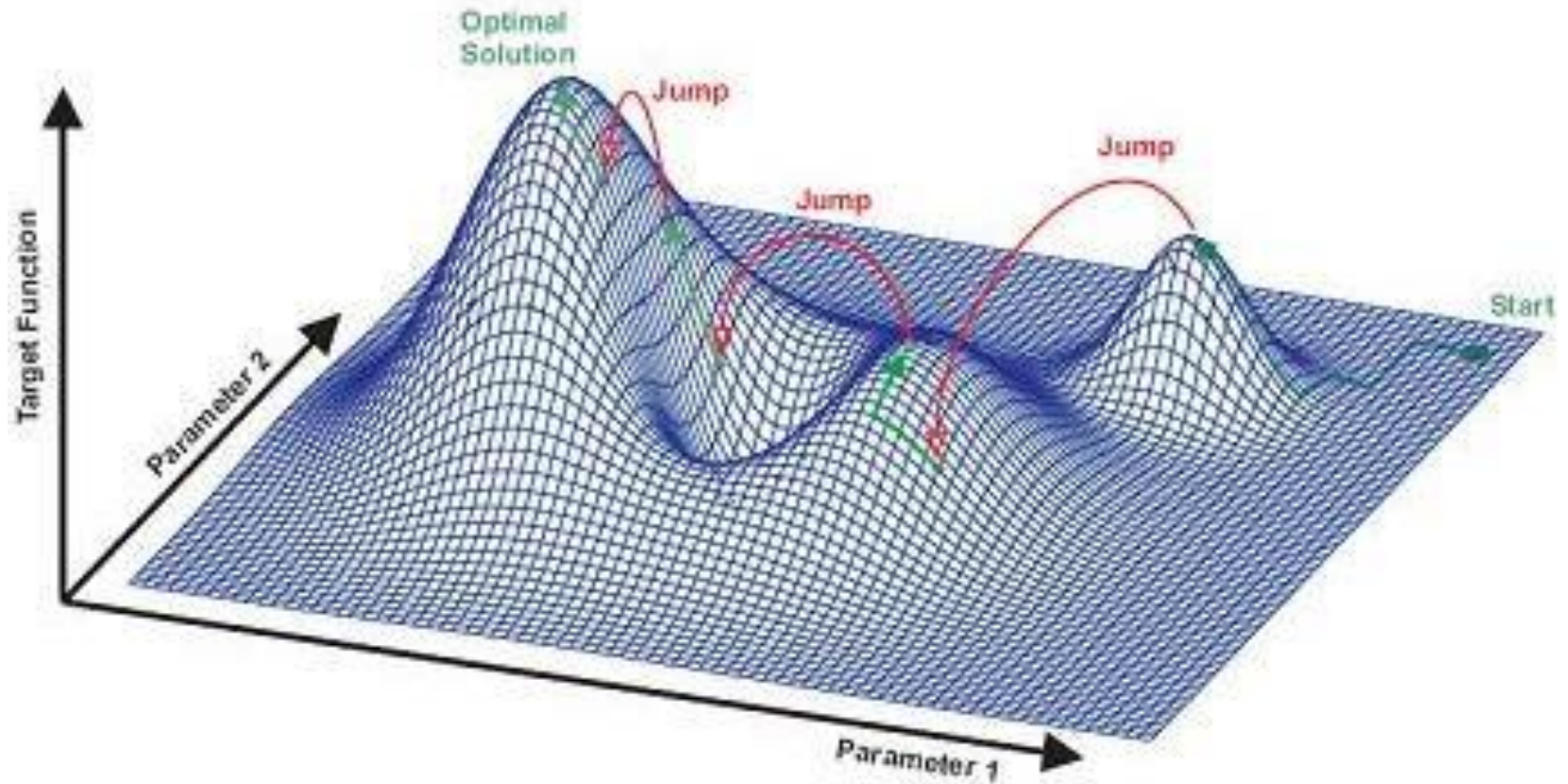
$\Delta E \leftarrow$ *next*.VALUE – *current*.VALUE

if $\Delta E > 0$ **then** *current* \leftarrow *next*

else *current* \leftarrow *next* only with probability $e^{\Delta E/T}$

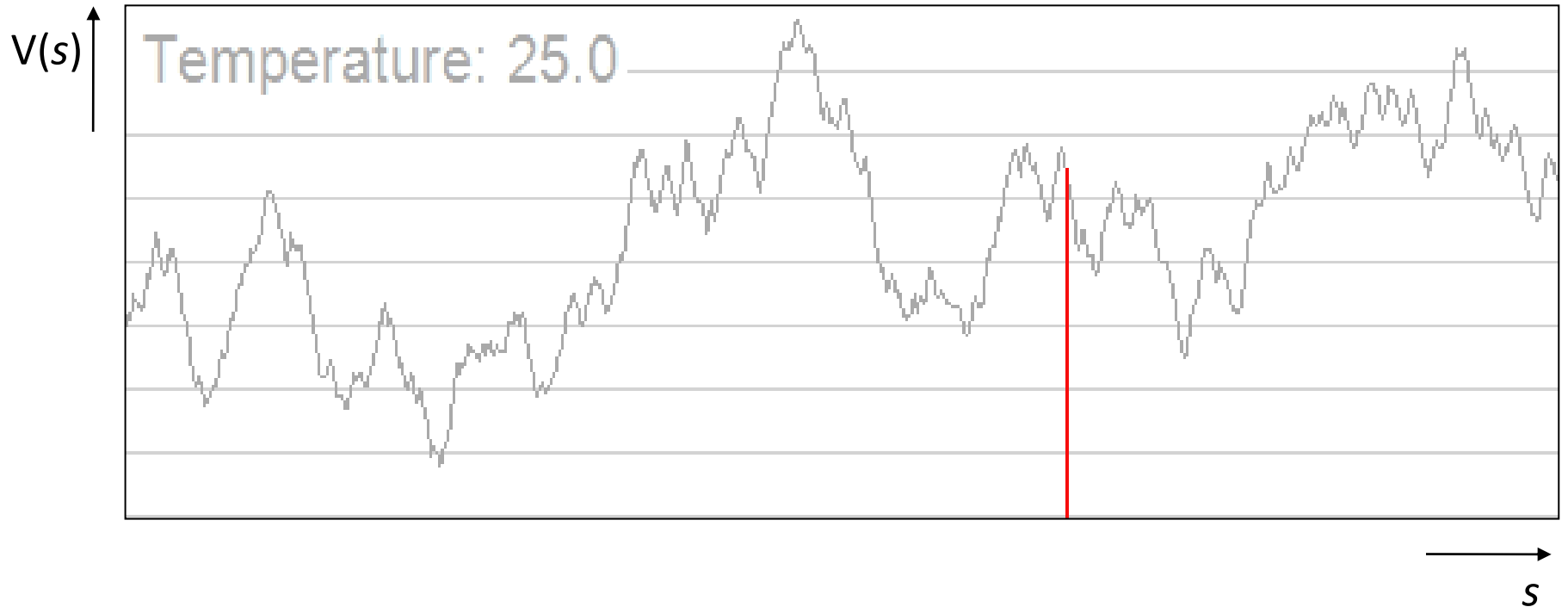
Figure 4.5 The simulated annealing algorithm, a version of stochastic hill climbing where some downhill moves are allowed. Downhill moves are accepted readily early in the annealing schedule and then less often as time goes on. The *schedule* input determines the value of the temperature T as a function of time.

ALGORITMO *SIMULATED ANNEALING*



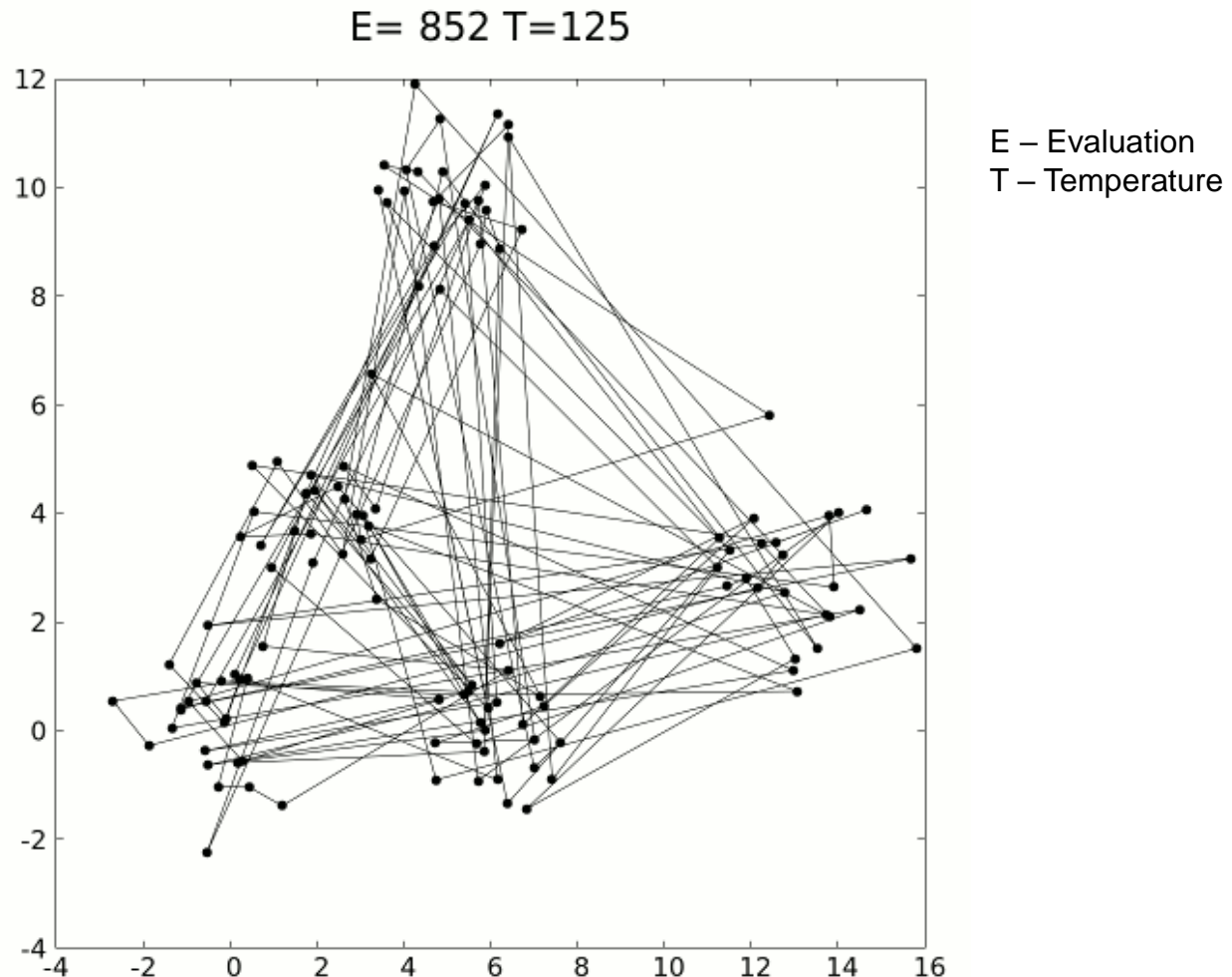
ALGORITMO *SIMULATED ANNEALING*

Exemplo



PROBLEMA DO CAIXEIRO VIAJANTE

Determinar a menor rota para visitar um conjunto de cidades, retornando à cidade de origem



RESOLUÇÃO DE UM PROBLEMA

Elementos a considerar na resolução de um problema de optimização

- **Problema**

- Representa informação de contexto do problema
 - e.g. distâncias entre cidades (problema do caixeiro viajante)
- Função de avaliação de estado
- Estado inicial
- Operadores de transformação de estado

- **Estado**

- Representa uma configuração de resolução do problema
 - e.g. configuração de rainhas no tabuleiro (problema das N-Rainhas)
- Realiza encapsulamento de manipulação da configuração de estado

RESOLUÇÃO DE UM PROBLEMA

- **Operador**

- Representa transformação de estado
- Aplicado a um estado, retorna
 - Um estado sucessor, ou
 - Conjunto de estados sucessores
- Deve produzir transformações locais de estado, reflectindo as características do domínio do problema
 - e.g. movimentação de uma rainha (problema das N-Rainhas)
 - e.g. alteração de sequência parcial do percurso (problema do caixeiro viajante)
- Transformações de estado com carácter aleatório correspondem à introdução de um pendor de amostragem global
 - Degeneram numa procura puramente estocástica
 - Degradação de desempenho

- **Resultados**

- Número de iterações para obter solução
- Qualidade da solução

REFERÊNCIAS

[Russel & Norvig, 2010]

S. Russell, P. Norvig, “Artificial Intelligence: A Modern Approach”, 3rd Ed., Prentice Hall, 2010

[Beire *et. Al.*, 2014]

A. Beire, N. Cota, H. Pita, A. Rodrigues, “Automatic tuning of Okumura-Hata model on railway communications”, International Symposium on Wireless Personal Multimedia Communications (WPMC), 2014

[Nagy *et al.*, 2018]

D. Nagy, D. Zhao, D. Benjamin, “Nature-Based Hybrid Computational Geometry System for Optimizing Component Structure”, Humanizing Digital Reality, Springer Nature, 2018