Inteligência Artificial
ACH2016

Aula 6: Busca local

Profa. Karina Valdivia Delgado
EACH-USP

Slides baseados em:

RUSSEL, S.; NORVIG, P. Artificial Intelligence: A modern approach. Third Edition, 2010. Capítulo 4.

Slides do Prof. Edirlei Soares de Lima

Slides da Profa. Leliane Nunes de Barros

Métodos de busca

- **Busca sem informação ou cega ou exaustiva ou força bruta ou sistemática :**
  - Não sabe qual o melhor nó da fronteira a ser expandido. Apenas distingue o estado objetivo dos não objetivos.
- **Busca informada ou heurística:**
  - Estima qual o melhor nó da fronteira a ser expandido com base em funções heurísticas. Sabem se um estado não objetivo é “mais promissor”.
- **Busca local:**
  - Operam em um único estado e movem-se para a vizinhança deste estado.

Busca local

- Em muitos problemas o **caminho para a solução é irrelevante**.
  - **Jogo das n-rainhas:** o que importa é a configuração final e não a ordem em que as rainhas foram acrescentadas.
  - **Outros exemplos:**
  - Projeto de Circuitos eletrônicos;
  - Roteamento de veículos;
  - Escalonamento de salas de aula;
  - Otimização de redes;

Busca local

- Algoritmos de busca local operam sobre um **único estado corrente**, ao invés de vários caminhos.
- Em geral se movem apenas para os **vizinhos** desse estado.
- O caminho seguido pelo algoritmo não é guardado.

Busca local

- **Vantagens:**
  - Ocupam pouquíssima memória (normalmente constante).
  - Podem encontrar soluções razoáveis em grandes ou infinitos espaços de estados.
- **São uteis para resolver problemas de otimização.**
  - Buscar por estados que atendam a uma função objetivo.

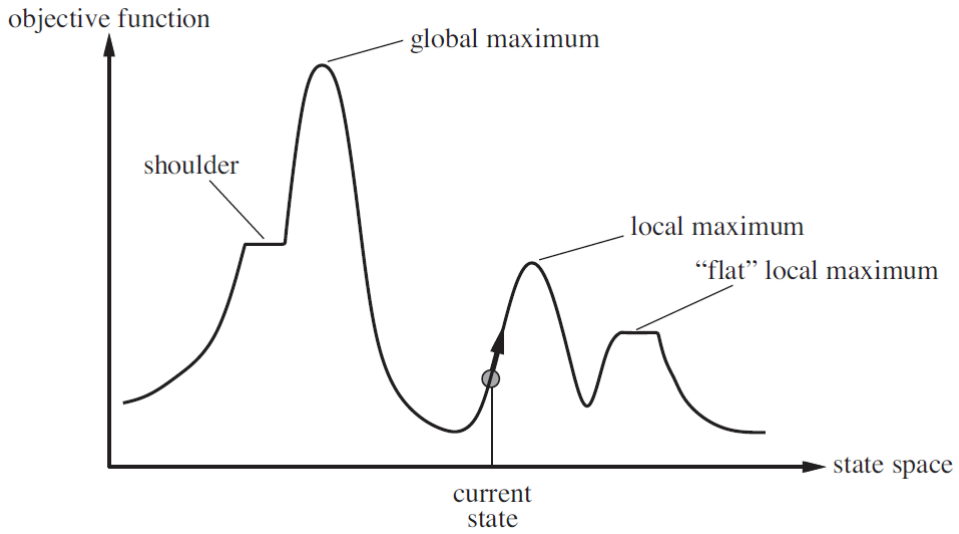
Busca local

- **Topologia do espaço de estados**: o algoritmo de busca local explora essa topologia

- **Posição** = Estado;

Elevação = **valor da função objetivo** ou valor da função de custo da heurística;

- Busca-se o **máximo** ou mínimo global;



Busca local

- **Principais Algoritmos:**
  - Hill Climbing
  - Simulated Annealing
  - Genetic Algorithms

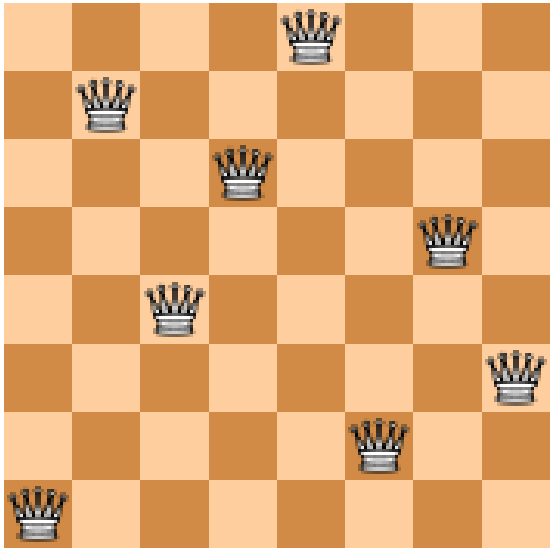
Pseudocódigo – Hill Climbing

```
Função Hill-Climbing(Problema) retorna um estado que é um máximo local

  EstadoAtual ← CRIAR-NÓ(Problema.EstadoInicial)

  repita

     Vizinho ← SucessorDeMaiorValor(EstadoAtual)

     se Vizinho.Valor for menor ou igual EstadoAtual.Valor então

       retorna EstadoAtual

     EstadoAtual ← Vizinho
```

- O algoritmo move-se continuamente para os estados que aumentam o valor.
- Termina quando atinge um "pico" onde nenhum vizinho tem um valor maior.
- Não mantem uma árvore de busca.

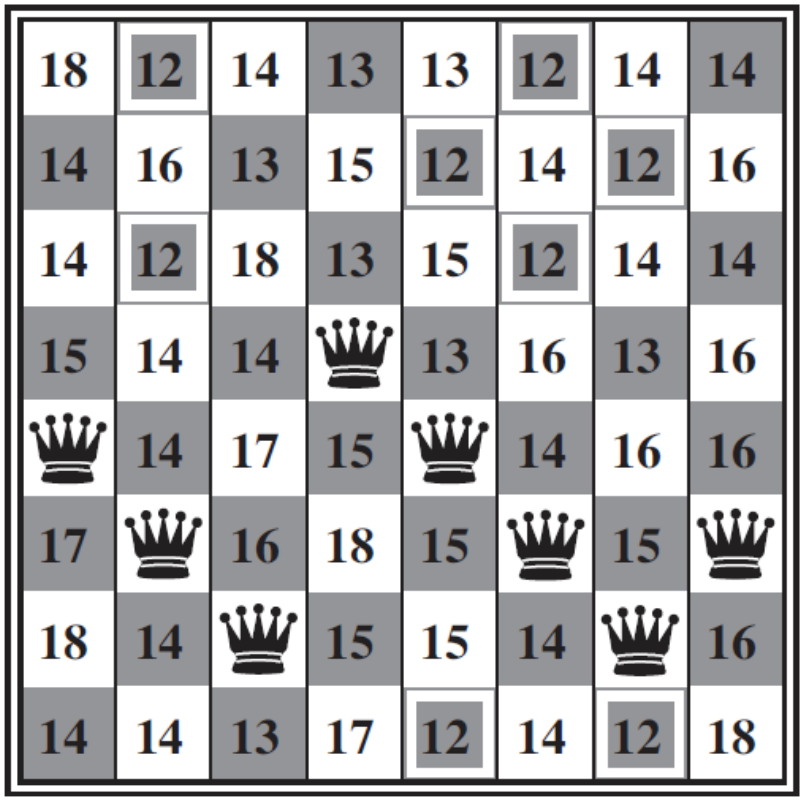
Hill Climbing:exemplo

- **Problema:** 8 Rainhas (estados completos)
  - Em cada estado: 8 rainhas no tabuleiro, uma em cada coluna.
- **Ações:**
  - Mover uma rainha para outro quadrado na mesma coluna.
- **Função Heurística (h):**
  - Número de pares de rainhas que estão sendo atacadas umas às outras.
- **Objetivo:**
  - h = 0 (nenhuma rainha sendo atacada)



Hill Climbing:exemplo

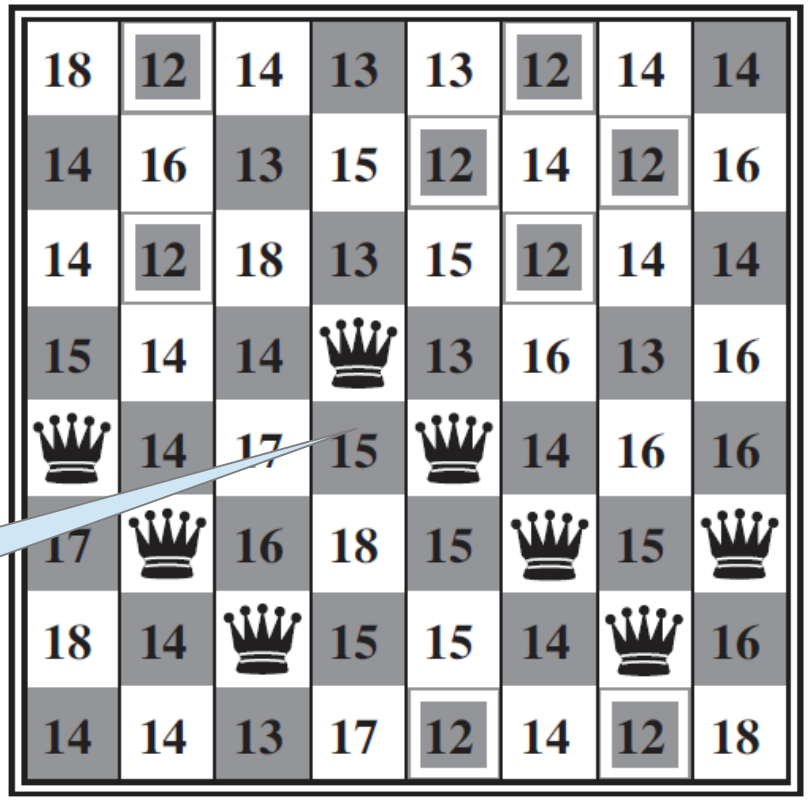
- **h = 17**
- Melhor movimento: 12



Valores de seus sucessores

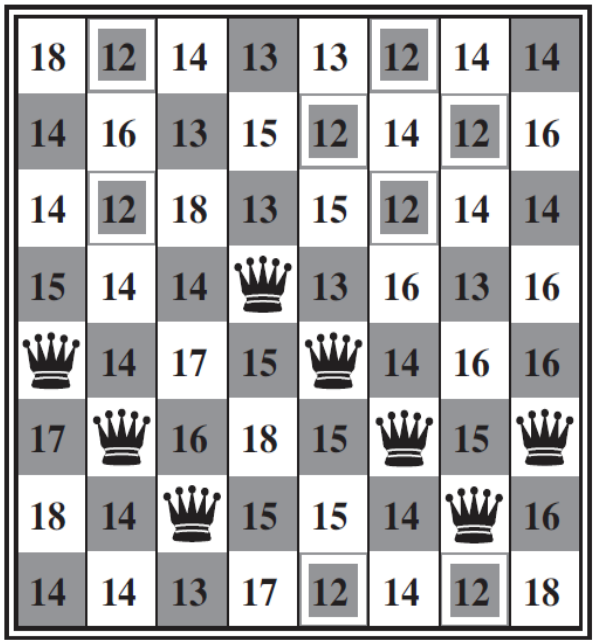
Hill Climbing:exemplo

- **h = 17**
- Melhor movimento: 12

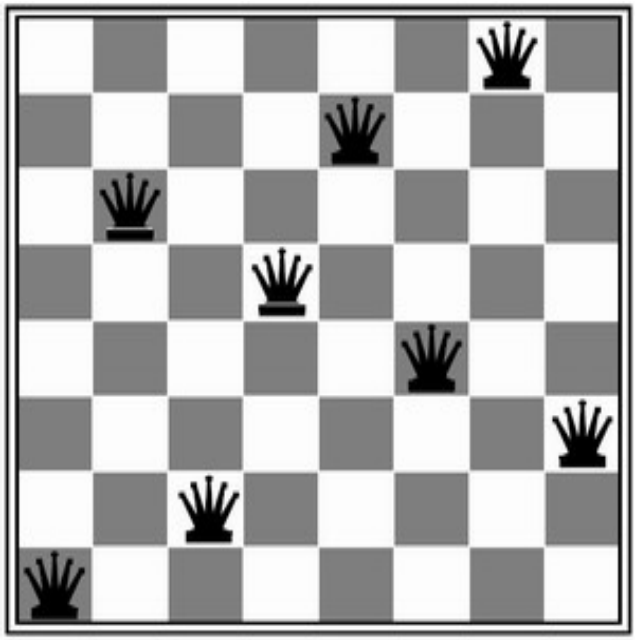


Se movimentamos a rainha da coluna 4 para essa posição temos h=17-2=15

Hill Climbing:exemplo

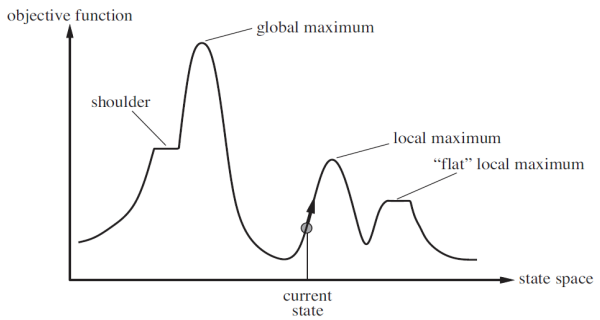


- Em 5 passos alcança o estado com h=1 (mínimo local para o custo h)



Hill Climbing

- É um **algoritmo guloso** – escolhe sempre o primeiro melhor vizinho para progredir na busca.
- Essa abordagem pode ter **bons resultados em alguns problemas**. Sendo capaz de progredir rapidamente para a solução problema.
- Mas, pode ficar paralisada pelas seguintes razões:
  - Máximos locais: Pico mai alto que cada um de seus estados vizinhos. O algoritmo fica presso.
  - Platôs: Áreas planas. O algoritmo tal platô.
    - Máximo local plano
    - Planície



Hill Climbing – Exemplo

- **Problema:** 8 Rainhas
- **Inicializando aleatoriamente o estado inicial:**
  - O algoritmo fica **preso em um máximo local** em 86% das vezes.
  - Resolve apenas 14% das instancias.
- Quando tem sucesso, resolve o problema em **aproximadamente 4 passos** – nada mal para um espaço de estados com 17 milhões de estados.

Hill Climbing

Não é ótimo e não é completo.

O desempenho do Hill Climbing depende muito do **formato da topologia** do espaço de estados.

- **Variações:**

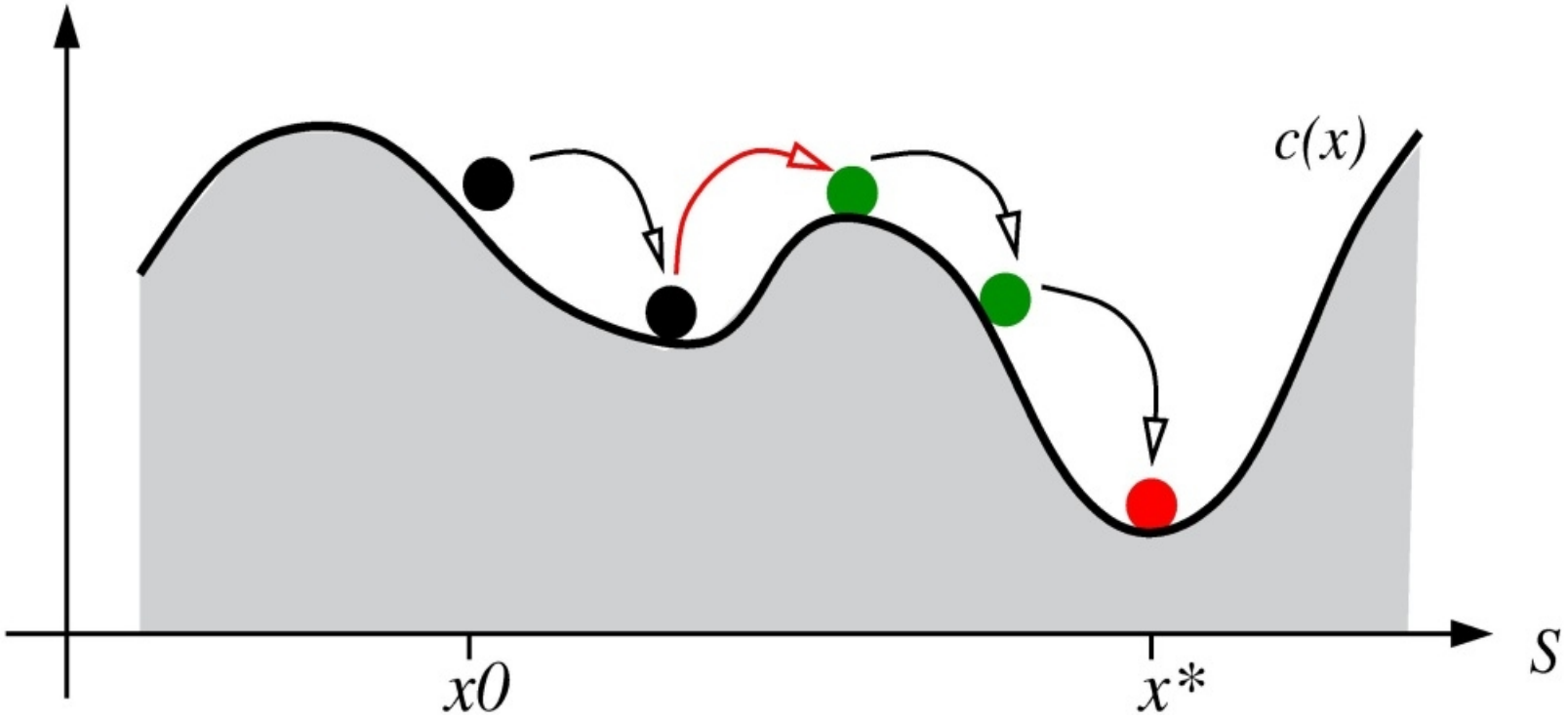
Random-Restart Hill Climbing;

Random-restart Hill-climbing

- Execute uma série de buscas hill-climbing a partir de um estado inicial aleatório.
- Guarda o melhor resultado.
- Pode usar um número fixo de iterações ou continuar até não conseguir superar o melhor resultado guardado considerando um certo número de iterações.
- Sucesso depende da forma da superfície.

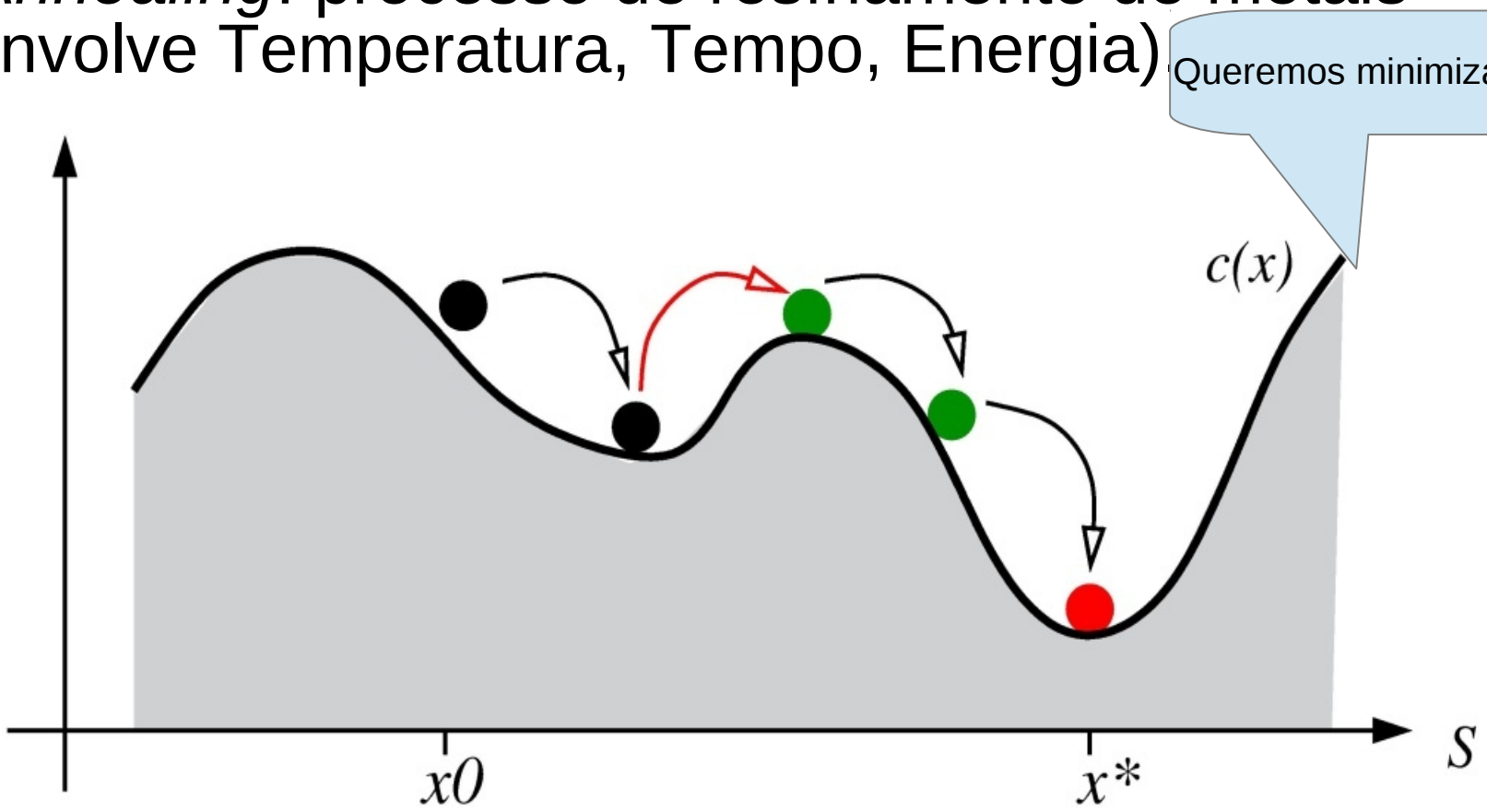
Simulated annealing

- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de metais (envolve Temperatura, Tempo, Energia).



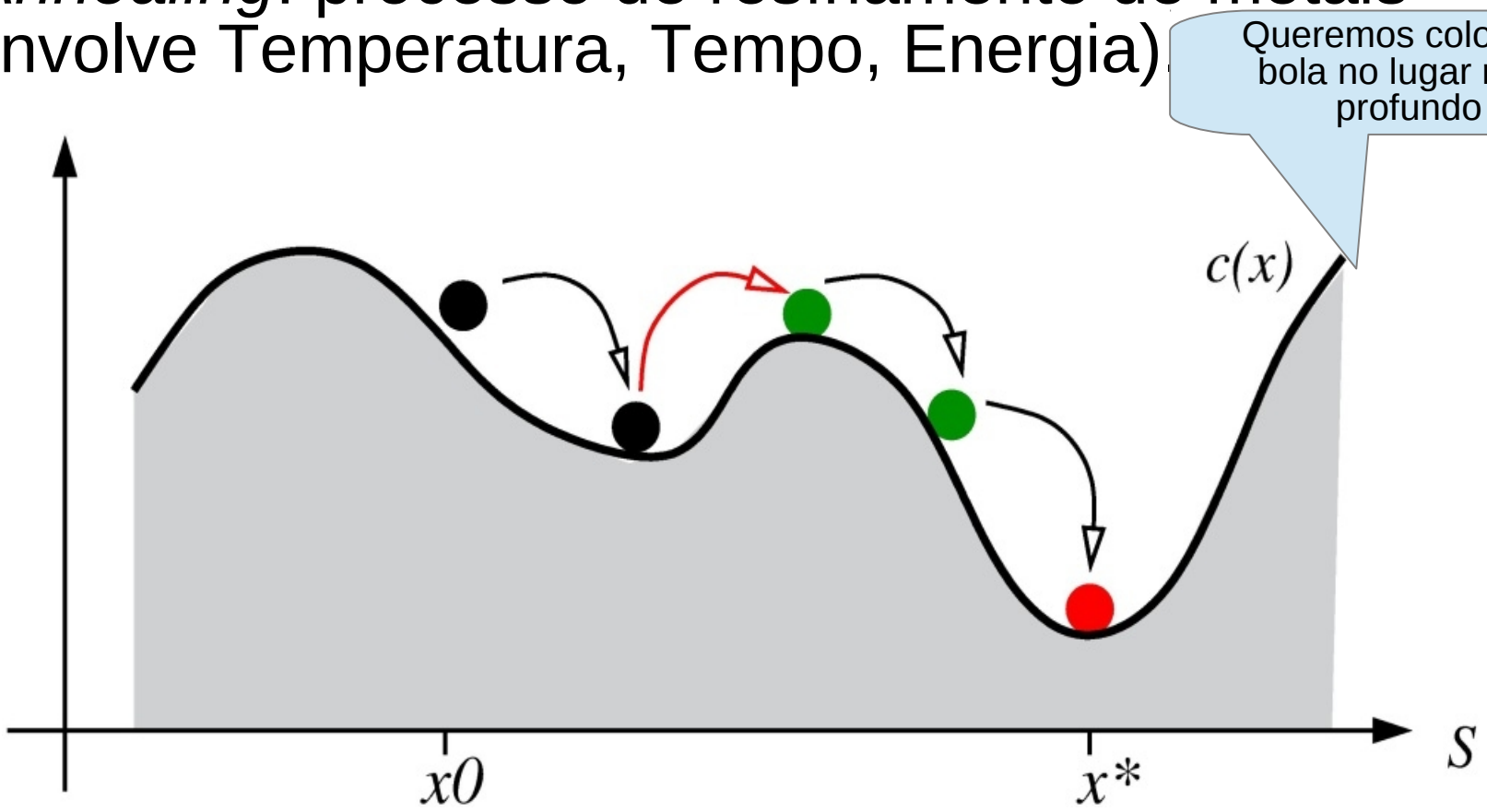
Simulated annealing

- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de metais (envolve Temperatura, Tempo, Energia)



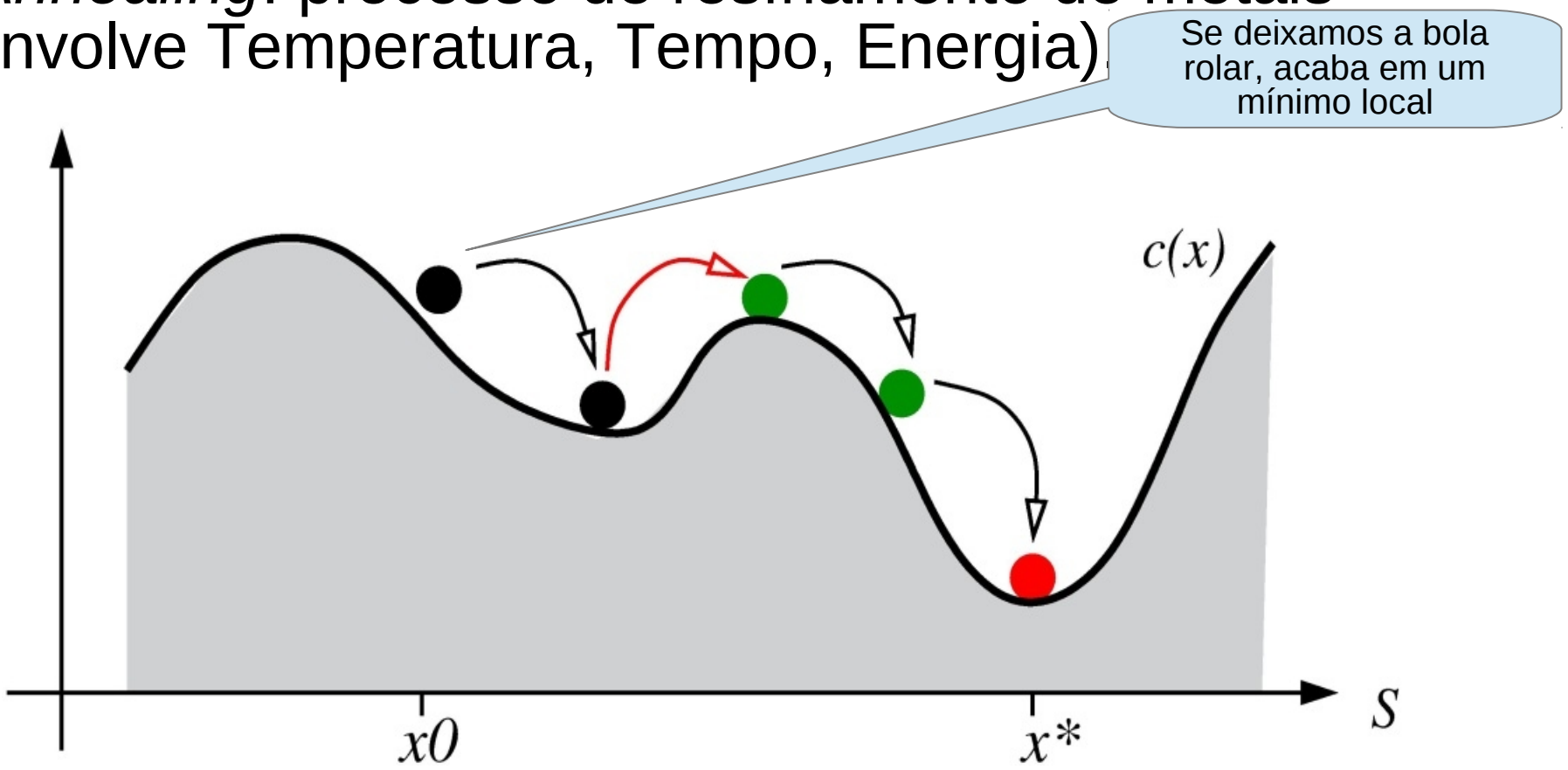
Simulated annealing

- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de metais (envolve Temperatura, Tempo, Energia).



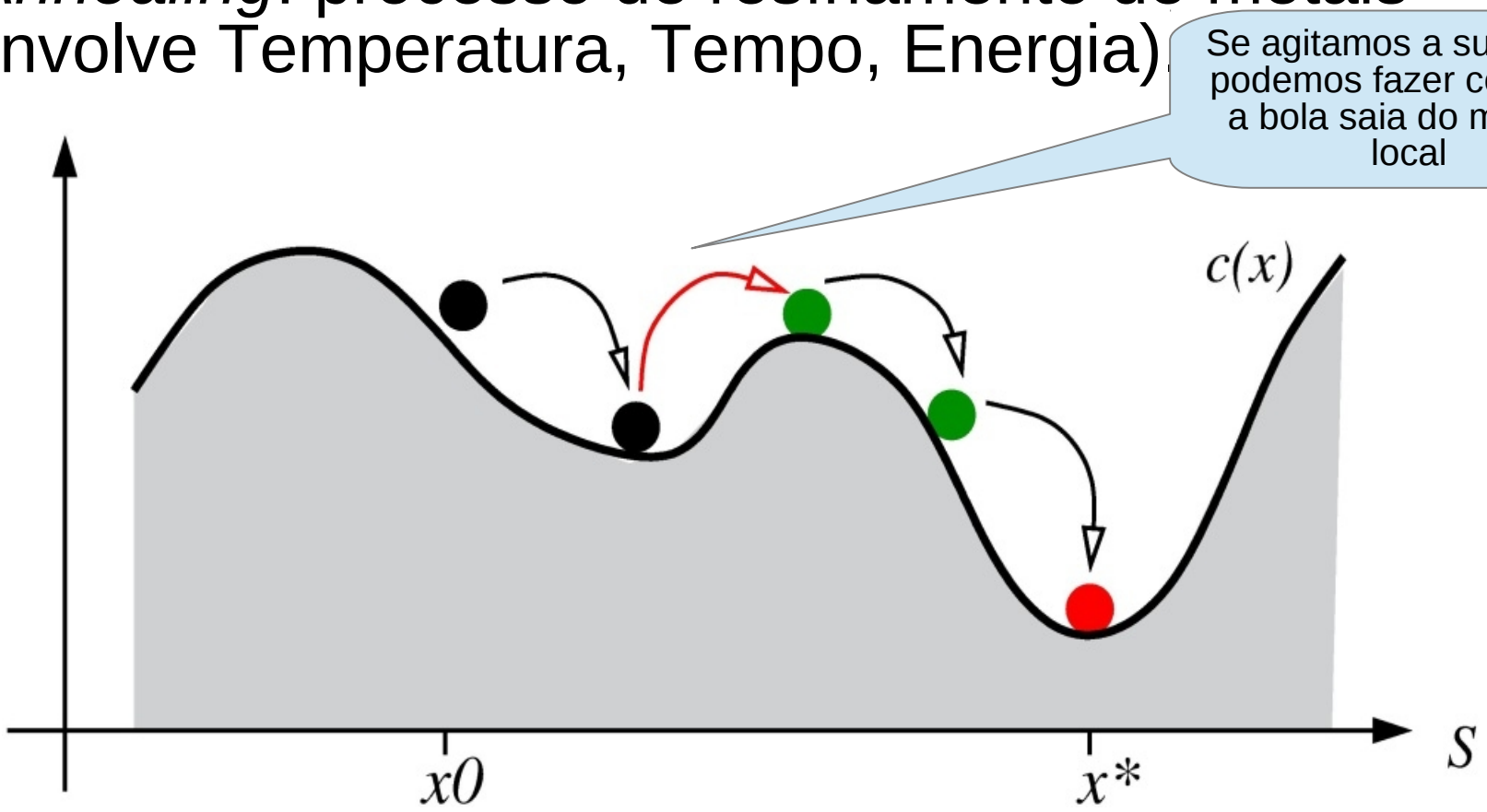
Simulated annealing

- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de metais (envolve Temperatura, Tempo, Energia).



Simulated annealing

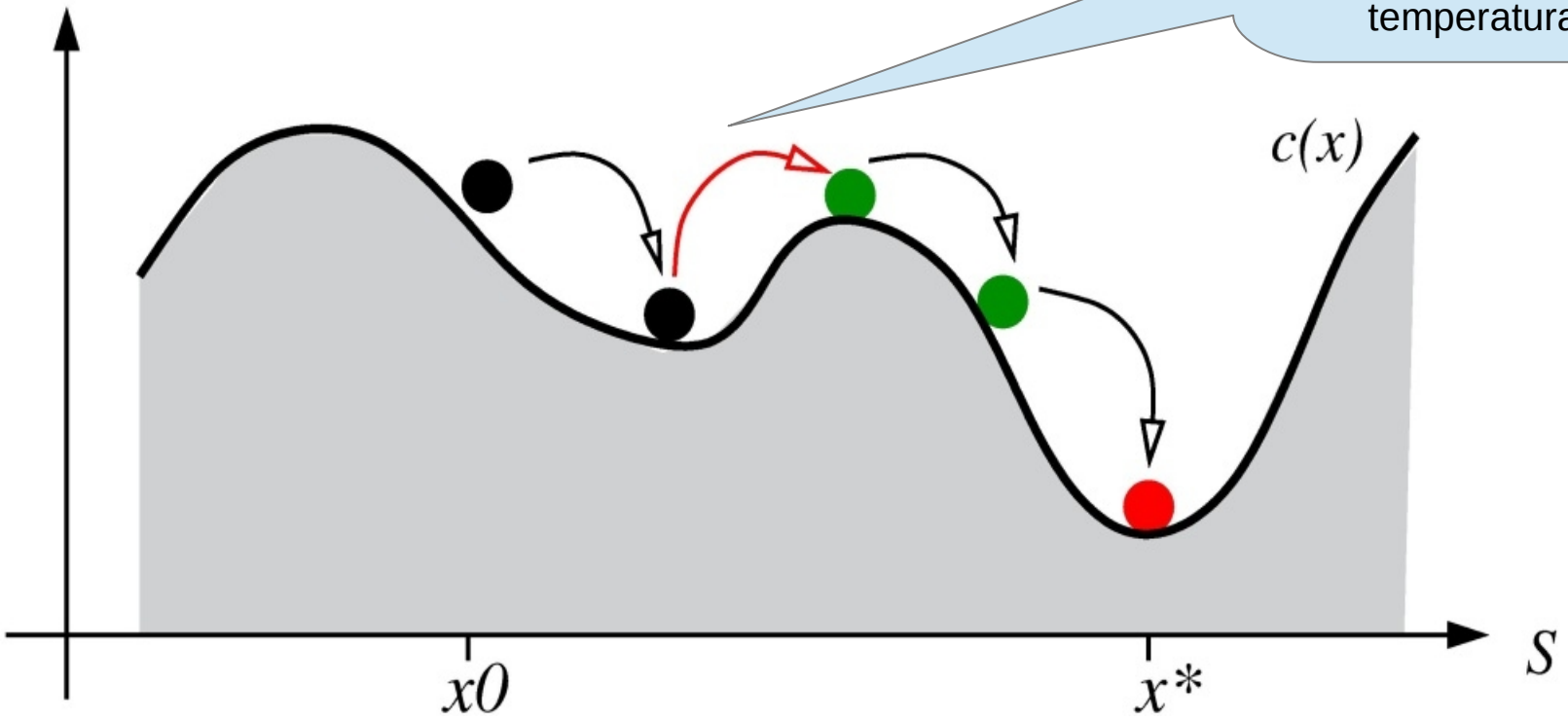
- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de metais (envolve Temperatura, Tempo, Energia).



Simulated annealing

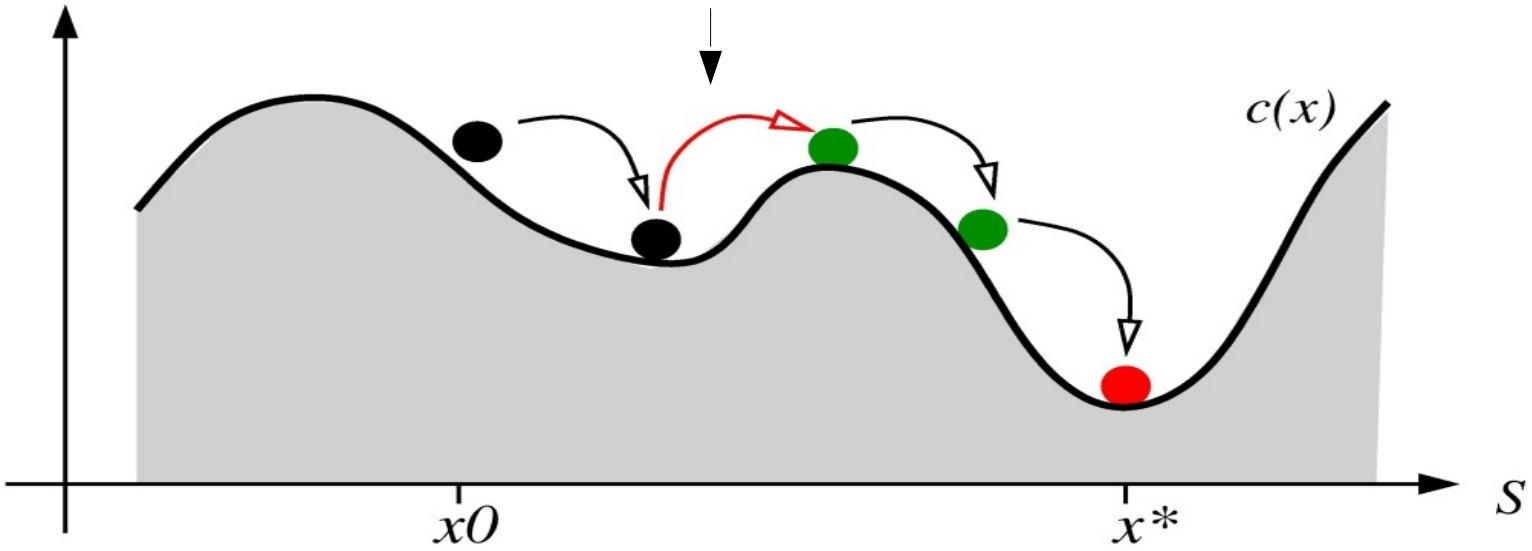
- Tenta combinar o Hill-climbing com um percurso aleatório.
- *Annealing*: processo de resfriamento de
(envolve Temperatura, Tempo, Energia).

O algoritmo começa agitando com força (alta temperatura) e depois reduce gradualmente a intensidade (baixando a temperatura).



Simulated annealing

- Ao invés de começar de novo permite que a busca suba o “morro” alguns paços, i.e, piora para depois melhorar.
- Pulo no espaço de estados não totalmente aleatório.
- Decresce gradualmente o tamanho do pulo e a frequência até alcançar o melhor estado.



Simulated annealing

Função Simulated-Annealing (*problema, escala*) **retorna** *um estado solução*

Entradas: *problema*

escala: um mapeamento de tempo para ”temperatura”

estado-corrente ← **`CRIAR-NÓ`**(problema. Estado-Inicial)

para *t = 1* **to** infinito **faça**

T ← *escala[t]*

se *T = 0* **então retorna** *estado-corrente*

próximo-estado ← *seleção aleatória de um sucessor do estado-corrente*

ΔE ← *próximo-estado.Valor – estado-corrente.*Valor

se $\Delta E > 0$ **então** *estado-corrente* ← *próximo-estado*

senão *estado-corrente* ← *próximo-estado com probabilidade* $e^{\Delta E/T}$

Simulated annealing

Temperatura

Função Simulated-Annealing (*problema, escala*) **retorna** *um estado*

Entradas: *problema*

escala: um mapeamento de tempo para ”temperatura”

estado-corrente ← **CRIAR-NÓ**(problema. Estado-Inicial)

para *t = 1* **to** infinito **faça**

T ← *escala[t]*

se *T = 0* **então retorna** *estado-corrente*

próximo-estado ← *seleção aleatória de um sucessor do estado-corrente*

ΔE ← *próximo-estado.Valor – estado-corrente.*Valor

se $\Delta E > 0$ **então** *estado-corrente* ← *próximo-estado*

senão *estado-corrente* ← *próximo-estado com probabilidade* $e^{\Delta E/T}$

Simulated annealing

Se o movimento melhora a situação, vai para o próximo estado (o movimento é aceito)

Função Simulated-Annealing (*problema, escala*) **retorn**

Entradas: *problema*

escala: um mapeamento de tempo para ”tempe ra”

estado-corrente ← **`CRIAR-NÓ`**(problema. Estado-Inicial)

para *t = 1* **to** infinito **faça**

T ← *escala[t]*

se *T = 0* **então retorna** *estado-corrente*

próximo-estado ← *seleção aleatória de um sucessor do estado-corrente*

ΔE ← *próximo-estado.Valor – estado-corrente.*Valor

se $\Delta E > 0$ **então** *estado-corrente* ← *próximo-estado*

senão *estado-corrente* ← *próximo-estado com probabilidade* $e^{\Delta E/T}$

Simulated annealing

Função Simulated-Annealing (*problema, escala*) **r**

Entradas: *problema*

escala: um mapeamento de tempo pa

estado-corrente ← **`CRIAR-NÓ`**(problema. Estado-I

para *t = 1* **to** infinito **faça**

T ← *escala[t]*

se *T = 0* **então retorna** *estado-corrente*

próximo-estado ← *seleção aleatória de um sucessor do estado-c ente*

ΔE ← *próximo-estado.Valor – estado-corrente.*Valor

se $\Delta E > 0$ **então** *estado-corrente* ← *próximo-estado*

senão *estado-corrente* ← *próximo-estado com probabilidade* $e^{\Delta E/T}$

Senão aceitará o movimento que piora a situação (vai para um próximo estado pior) com probabilidade $e^{\Delta E/T}$; e com probabilidade $1-e^{\Delta E/T}$ fica no mesmo estado.

Simulated annealing

Função Simulated-Annealing (*problema, escala*) **retorna**

Entradas: *problema*

escala: um mapeamento de tempo para ”temperatura”

estado-corrente ← **CRIAR-NÓ**(problema. Estado-Inicial)

para *t = 1* **to** infinito **faça**

T ← *escala[t]*

se *T = 0* **então retorna** *estado-corrente*

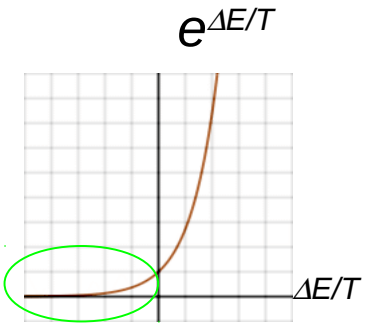
próximo-estado ← *seleção aleatória de um sucessor do estado-corrente*

ΔE ← *próximo-estado.Valor – estado-corrente.*Valor

se $\Delta E > 0$ **então** *estado-corrente* ← *próximo-estado*

senão *estado-corrente* ← *próximo-estado com probabilidade* $e^{\Delta E/T}$

Uma vez ΔE é negativo e T>=0 (i.e., o expoente de e é negativo), a probabilidade decresce à medida que $\Delta E/T$ se reduz.



Busca local

- **Principais Algoritmos:**
  - Hill Climbing
  - Simulated Annealing
  - Genetic Algorithms