Algoritmos de Busca Local

Prof. Dr. Rafael Teixeira Sousa

UFMT

Outline

- Meta-Heurísticas
- Espaço de Busca
- Busca e Otimização
  - Hill-Climbing
  - Simulated Annealing

Algoritmos de Busca

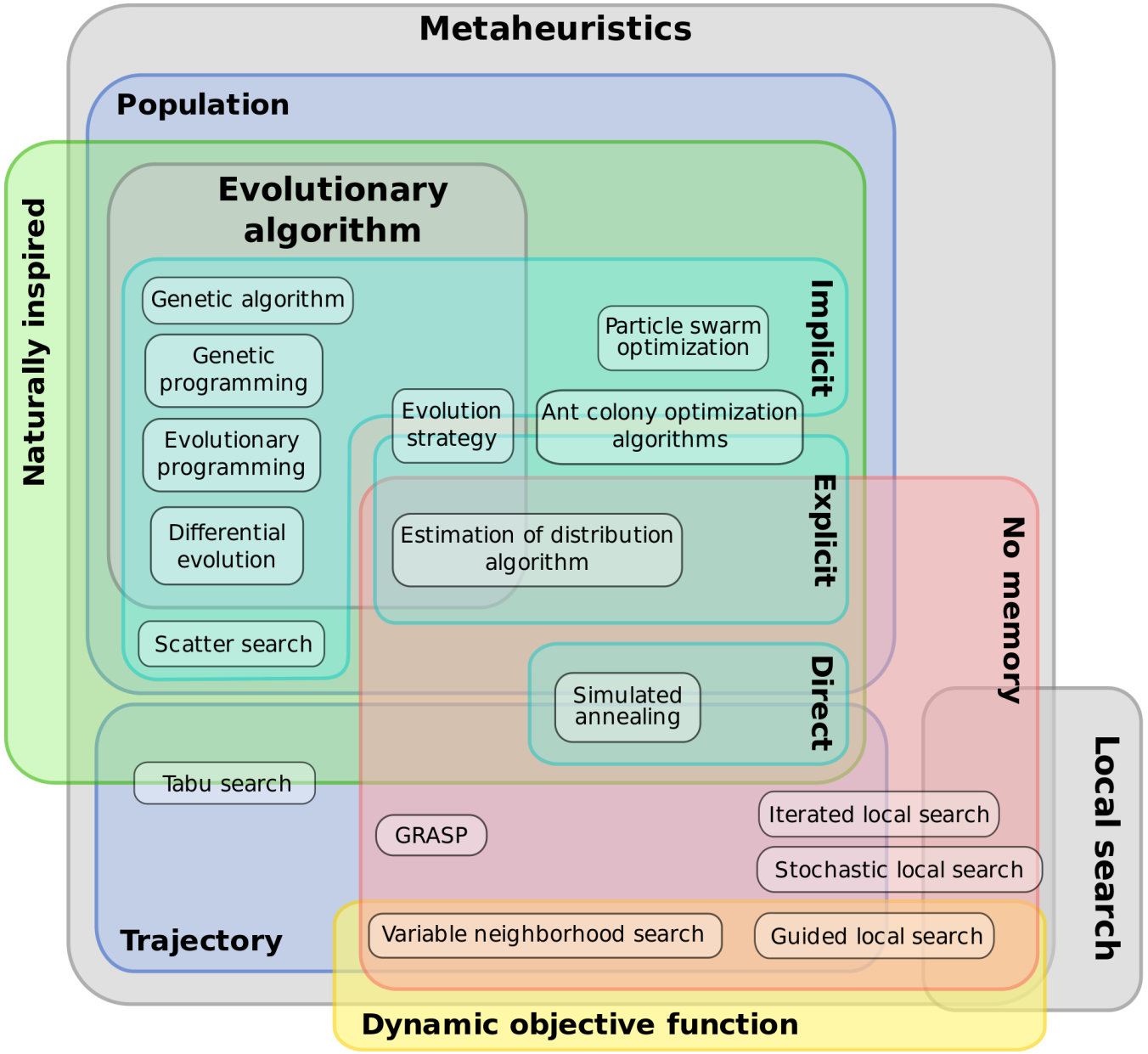
- Até agora: Exploração sistemática do espaço de busca
  - Mantendo nós na memória
  - Registrando alternativas exploradas e não exploradas

- Em vários problemas de otimização o **caminho** é irrelevante e o estado objetivo é a **solução**
- Ex:
  - Projeto de circuitos
  - Escalonamento de trabalhos
  - Otimização de redes

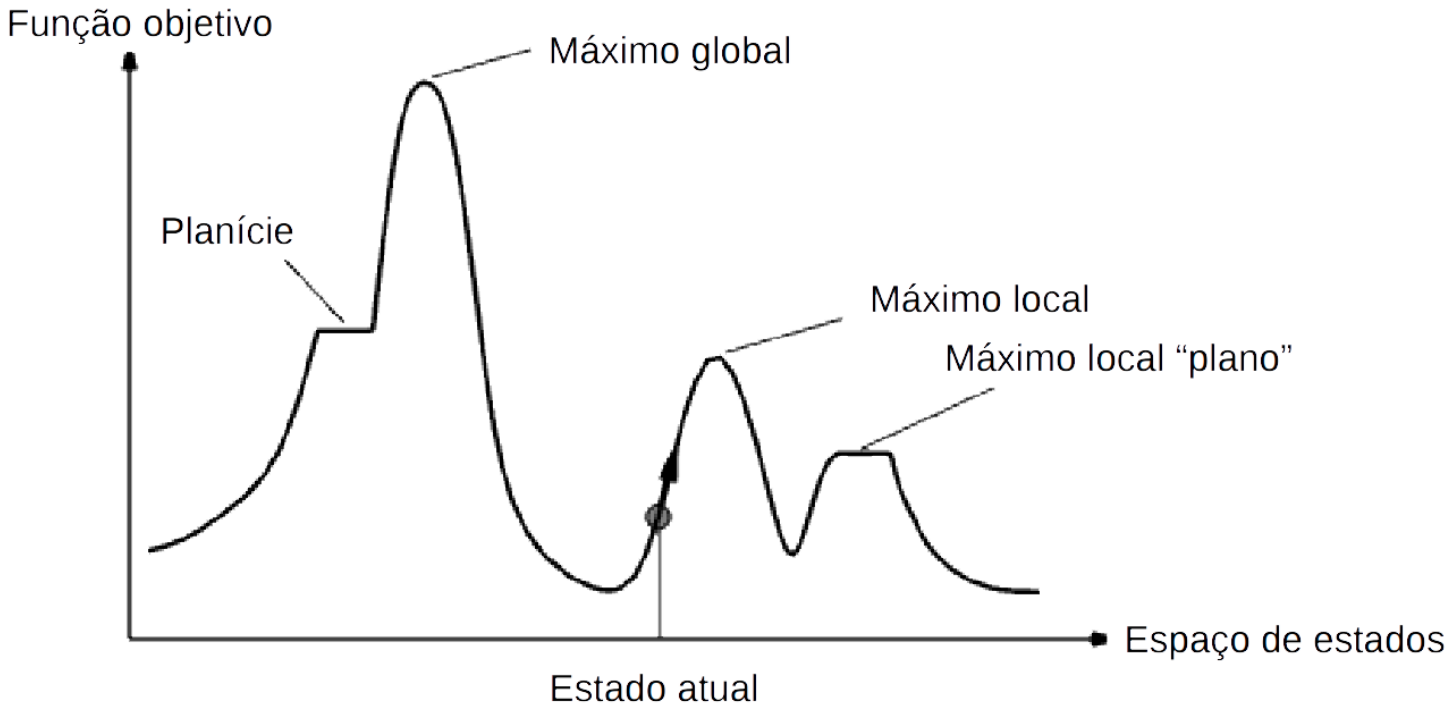
Meta-Heurísticas

- Método heurístico para resolver de forma genérica problemas de busca e otimização
  - Não necessariamente guardam caminho à solução
  - Podem encontrar soluções razoáveis para problemas grandes
    - Até mesmo infinitos

  - **Estado**: vetor de variáveis
  - **Função objetivo**: $f: Estado \rightarrow \Re$
  - **Objetivo**: achar estado que **maximize** ou **minimize** a função objetivo


Metaheuristics
Population
Naturally inspired
Evolutionary algorithm
Genetic algorithm
Genetic programming
Evolutionary programming
Differential evolution
Scatter search
Implicit
Particle swarm optimization
Evolution strategy
Ant colony optimization algorithms
Explicit
Estimation of distribution algorithm
Direct
Simulated annealing
No memory
Local search
Tabu search
GRASP
Iterated local search
Stochastic local search
Trajectory
Variable neighborhood search
Guided local search
Dynamic objective function

Espaço de Busca



- Algoritmo **completo** sempre encontra uma solução, caso ela exista
- Algoritmo **ótimo** sempre encontra mínimo/máximo global

Busca Iterativa

- Começar de um estado inicial e **melhorá-lo iterativamente** (encontrar outros estados que mim/max objetivo)
- Caso Max:
  - Busca pelo ponto mais alto (máximo global) correspondente à solução ótima
- Caso Min:
  - Busca pelo ponto mais baixo (mínimo global) correspondente à solução ótima

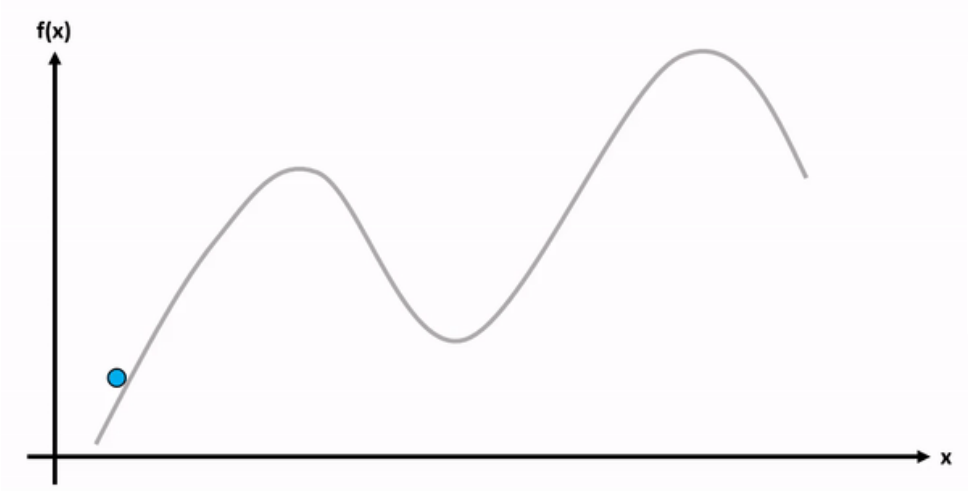
Busca Iterativa

- Métodos que guardam apenas o **estado atual**, e não veem além dos **vizinhos imediatos** do estado
- Dificilmente encontram solução ótima, mas são capazes de lidar com problemas complexos

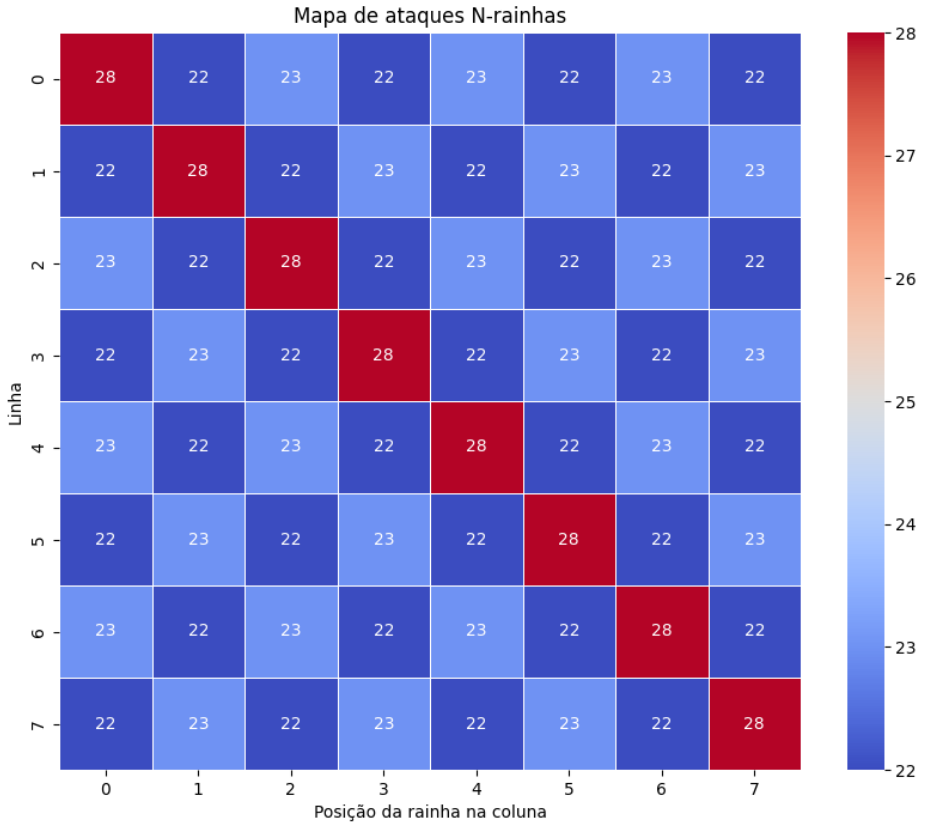
- Hill-Climbing: Subida de Encosta
  - Melhora estado com base na vizinhança
- Simulated Annealing: Têmpera Simulada
  - Aceita temporariamente a piora do estado para tentar melhorar posteriormente

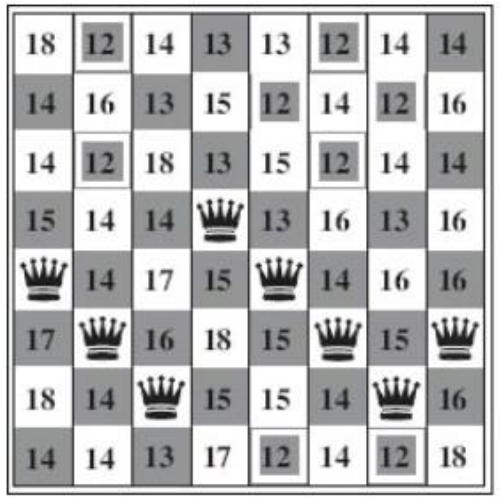
Hill-Climbing

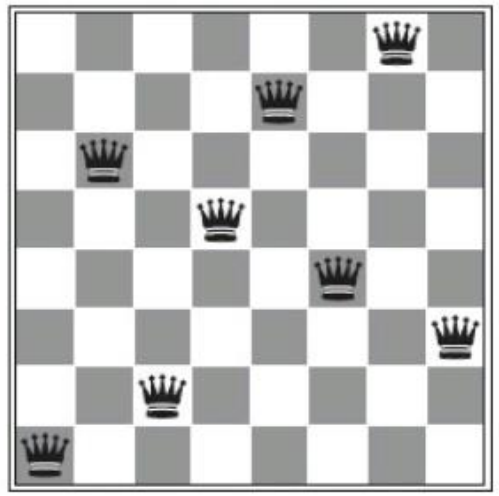
- Se move repetidamente ao vizinho mais “alto”
- Não mantém memória (árvore de busca)
- Para quando encontra algum pico




```
função SUBIDA-DE-ENCOSTA(problema) retorna um estado que é um máximo local
  corrente ← CRIAR-NÓ(ESTADO-INICIAL[problema])
  repita
    vizinho ← um sucessor de corrente com valor mais alto
    se VALOR[vizinho] VALOR[corrente] então retornar ESTADO[corrente]
    corrente ← vizinho
```





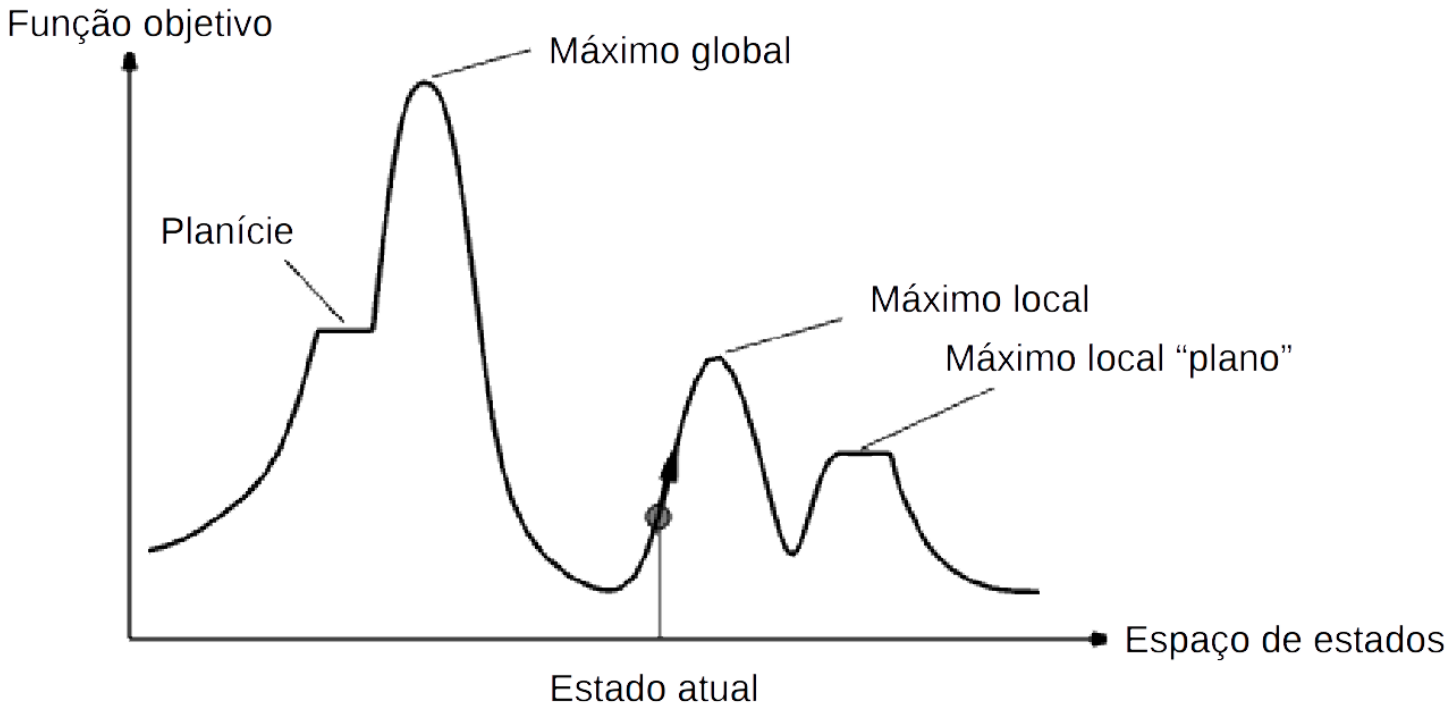


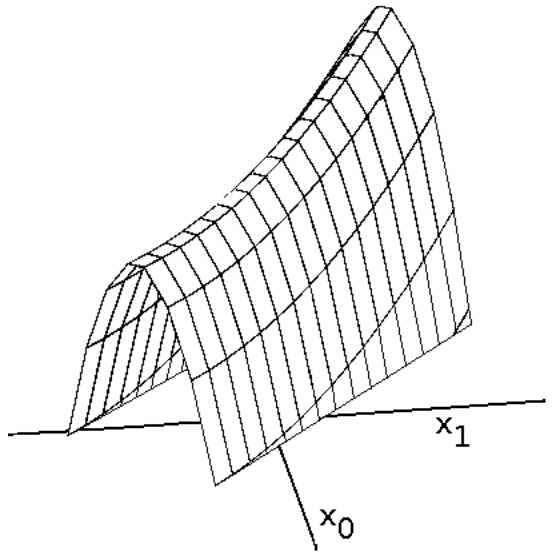
Exemplo: n-Rainhas

- Encontrar arranjo onde todas as rainhas não possam se atacar
- Função heurística *h*: número de pares de rainhas que estão atacando umas às outras
- Mínimo global: 0
- Máximo: 28

Hill-Climbing

- Busca Gulosa Local
- Pode parar em mínimos/máximos locais e em platôs





Exemplo: n-Rainhas

- Estados iniciais aleatórios
  - 86% das vezes a busca fica paralisada sem encontrar mínimo global
- Mas é rápida:
  - Em média encontra a solução em 4 passos
- Espaço de estados:
  - ~17 milhões de possibilidades

Corrigindo platôs

- Permitir a mudança de estados quando vizinhos têm avaliações iguais

- 8-Rainhas:
  - Passa a resolver 94% das vezes
  - Tempo:
    - 21 passos em média quando tem sucesso

Hill-Climbing

- Sucesso depende da topologia do espaço de estados
- Máximos/mínimos locais
- Platôs grandes

- Alternativas:
  - Stochastic Hill-Climbing

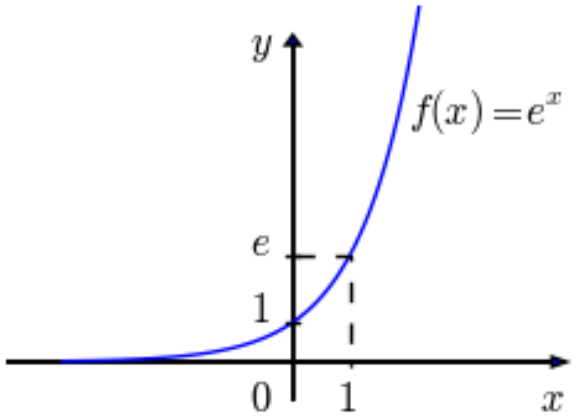
Simulated Annealing

1. Esquentar um material
2. Esfriar de maneira lenta e controlada

- No esfriamento os átomos se arranjam de maneira mais resistente

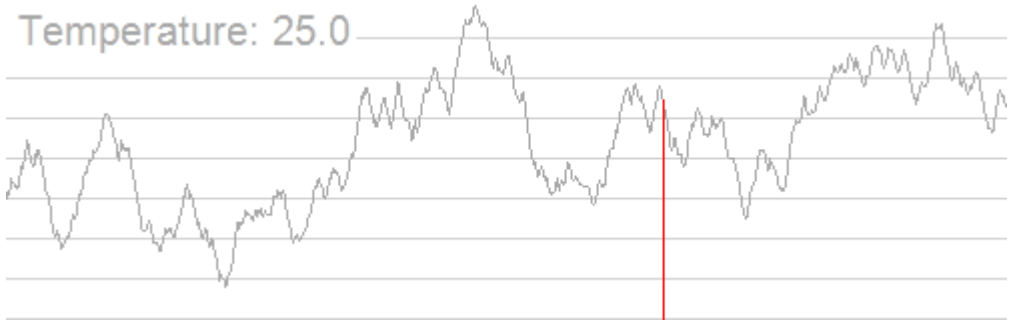
Simulated Annealing

função TÊMPERA-SIMULADA(*problema, escalonamento*) **retorna** um estado solução
  entradas: *problema*, um problema
escalonamento, um mapeamento de tempo para "temperatura"
  atual ← CRIAR-NÓ(*problema*.ESTADO-INICIAL)
  para $t = 1$ **até** ∞ **faça**
    $T \leftarrow$ *escalonamento*[t]
    se $T = 0$ **então retornar** *corrente*
    próximo ← um sucessor de *atual* selecionado aleatoriamente
    $\Delta E \leftarrow$ *próximo*.VALOR – *atual*.VALOR
    se $\Delta E > 0$ **então** *atual* ← *próximo*
    senão *atual* ← *próximo* somente com probabilidade $e^{\Delta E/T}$



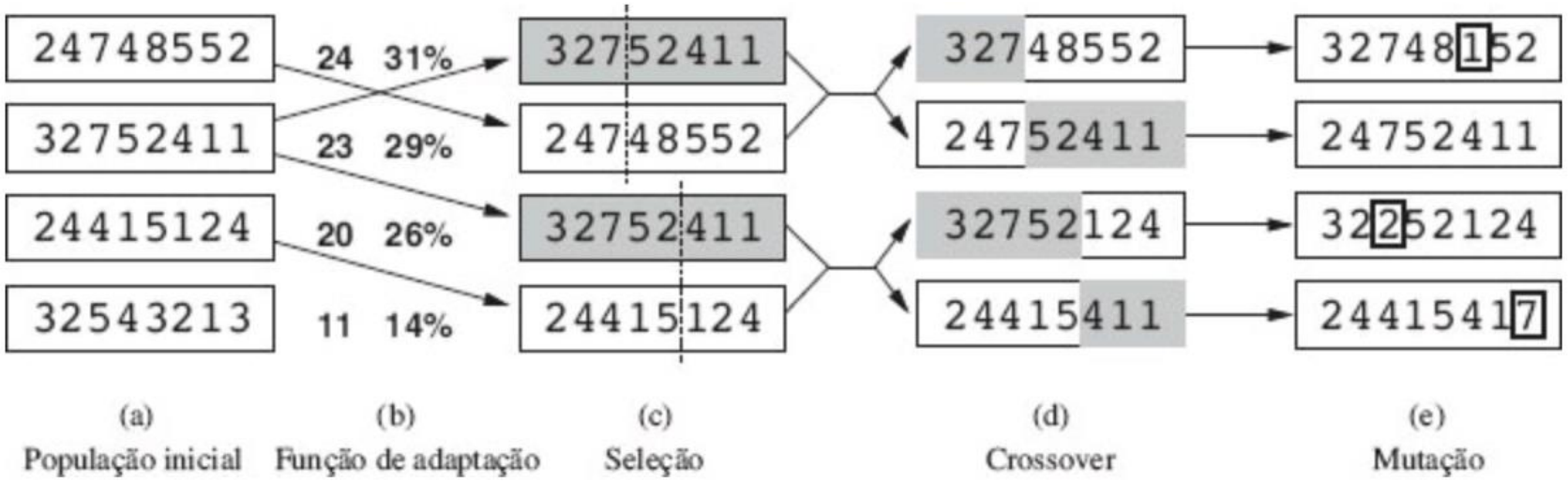
Simulated Annealing

função TÊMPERA-SIMULADA(*problema, escalonamento*) **retorna** um estado solução
  entradas: *problema*, um problema
escalonamento, um mapeamento de tempo para "temperatura"
  atual ← CRIAR-NÓ(*problema*.ESTADO-INICIAL)
  para $t = 1$ **até** ∞ **faça**
    $T \leftarrow$ *escalonamento*[t]
    se $T = 0$ **então retornar** *corrente*
    próximo ← um sucessor de *atual* selecionado aleatoriamente
    $\Delta E \leftarrow$ *próximo*.VALOR – *atual*.VALOR
    se $\Delta E > 0$ **então** *atual* ← *próximo*
    senão *atual* ← *próximo* somente com probabilidade $e^{\Delta E/T}$



Algoritmos Genéticos

- Estado sucessores são gerados a partir da combinação de dois estados pais.
- Analogia a seleção natural



Recapitulando...

- Espaço de Busca
- Hill-Climbing:
  - Encontra solução rápida em problemas complexos
  - Não é ótimo
- Simulated Annealing
  - Maior tempo de busca
  - Costuma encontrar soluções melhores