Inteligência Artificial

Além da Busca Clássica Parte 2

Simulated Annealing

Prof. Jefferson Morais

Simulated Annealing

- O algoritmo Hill climbing é incompleto, pois pode ficar preso em um máximo local

- Em contraste, um percurso puramente aleatório (sucessor escolhido uniformemente ao acaso) é completo, mas extremamente ineficiente

- Combinação
  - **Têmpera simulada**: tenta combinar a subida de encosta com um percurso aleatório que resulte de algum modo em eficiência e completeza
  - É também considerado uma implementação do *stochastic hill climbing*

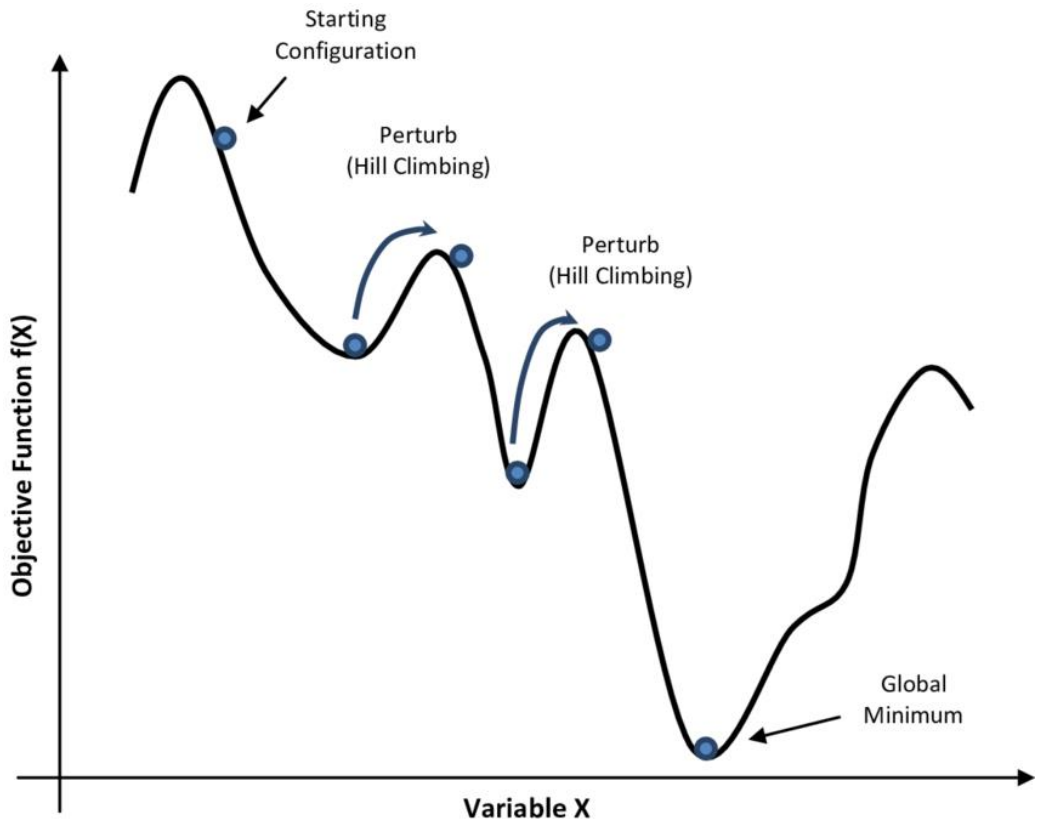
- **Têmpera (recozimento)** é o processo usado para temperar ou endurecer metais e vidro aquecendo-os a alta temperatura e depois esfriando-os gradualmente, permitindo assim que o material alcance um estado cristalino de baixa energia

Simulated Annealing

- Para explicar a Têmpera simulada, vamos mudar nosso ponto de vista de subida de encosta para **descida de gradiente**
  - isto é, minimização do custo
- Imagine a tarefa de colocar um bola de pingue-pongue na fenda mais profunda em uma superfície acidentada
  - Se simplesmente deixarmos a bola rolar, ela acabará em um mínimo local
  - Se agitarmos a superfície, poderemos fazer a bola quicar para fora do mínimo local
  - A solução do algoritmo simulated Annealing é **começar a agitar com força** (alta temperatura) e depois **reduzir gradualmente** a intensidade da agitação (baixar a temperatura)
    - **Agitação = aleatoriedade**

Simulated Annealing

- Descida de gradiente (minimização do custo) e o processo de agitação



Simulated Annealing

- O algoritmo começa a agitar com força (alta temperatura) e depois reduz gradualmente a intensidade da agitação (baixa temperatura)
  - **Agitação = aleatoriedade**

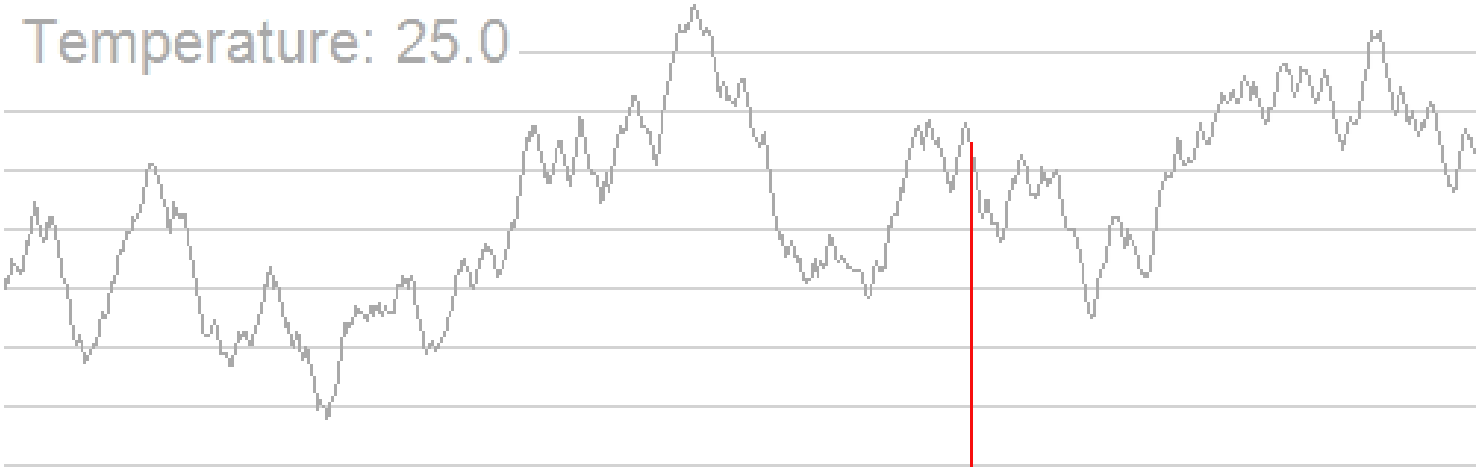
function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
  inputs: *problem*, a problem
    schedule, a mapping from time to "temperature"

  current ← MAKE-NODE(*problem*.INITIAL-STATE)
  for $t = 1$ **to** ∞ **do**
    $T \leftarrow schedule(t)$
    if $T = 0$ **then return** *current*
    next ← a randomly selected successor of *current*
    $\Delta E \leftarrow next.\text{VALUE} - current.\text{VALUE}$
    if $\Delta E > 0$ **then** *current* ← *next*
    else *current* ← *next* only with probability $e^{\Delta E/T}$

Para problemas de maximização.

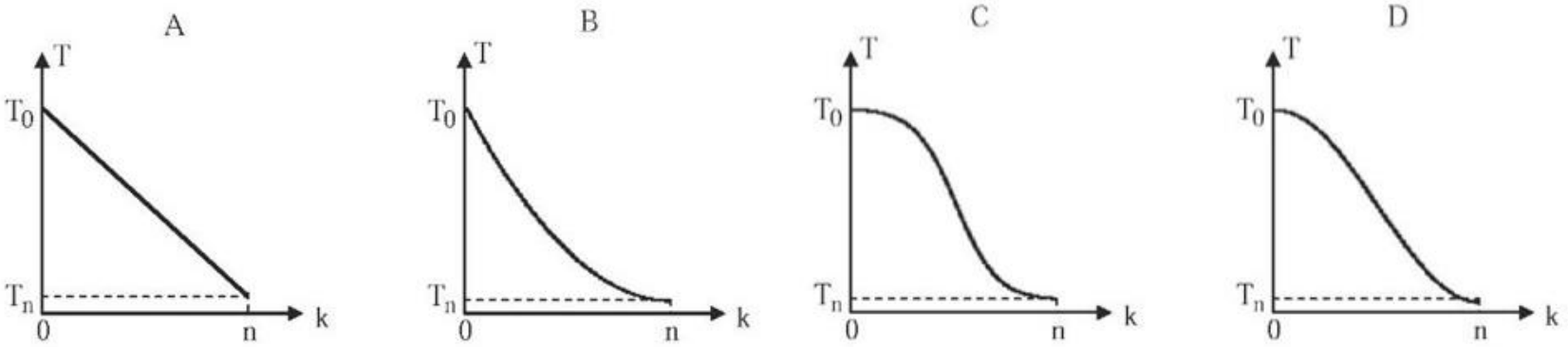
Permite movimentos que pioram o estado atual. Isso ocorre mais vezes nas primeiras iterações e menos nas iterações finais.

Simulated Annealing - Exemplo Maximização

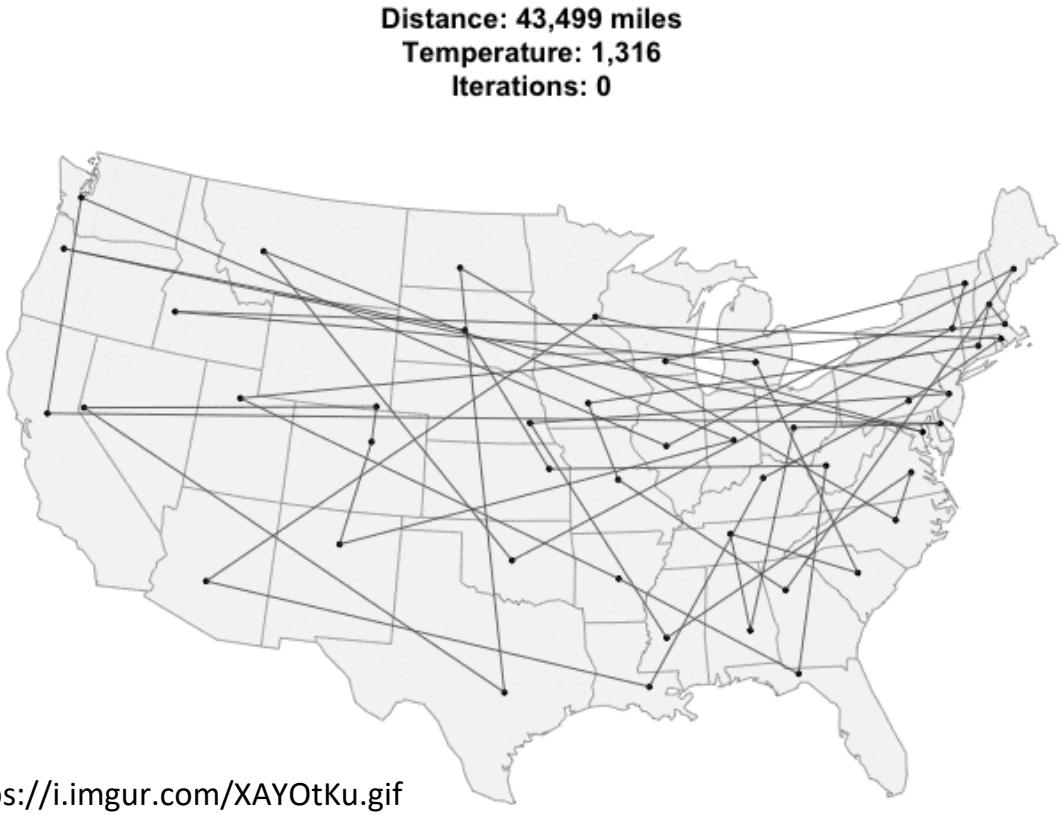


Referência: https://upload.wikimedia.org/wikipedia/commons/d/d5/Hill_Climbing_with_Simulated_Annealing.gif

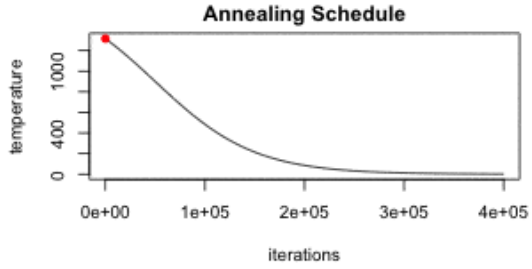
Tipos de decaimentos de temperatura:



Simulated Annealing - Exemplo Minimização



Referência: https://i.imgur.com/XAYOtKu.gif



Simulated Annealing

- **Próxima Aula:**

Algoritmo Genético