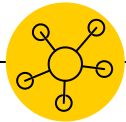


Metaheurísticas: Uma breve introdução





Olá!

Eu sou Caroline Albuquerque

Engenheira de computação e doutoranda no Programa de Engenharia Elétrica e de Computação.

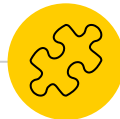
Github [@imcarolalbuquerque](#) | Gmail [carolads](#)

1

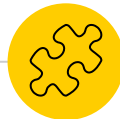
Motivação

Por que metaheurísticas são importantes?

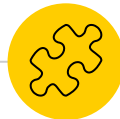
Inteligência Artificial (AI) tem desenvolvido um grande número de ferramentas para resolver os problemas de busca e otimização mais difíceis na ciência da computação e pesquisa operacional.



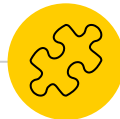
Inteligência computacional (CI) é um sub-divisão da AI, definida como o estudo de mecanismos adaptativos para permitir ou facilitar o comportamento inteligente em ambientes complexos e em constante mudança.



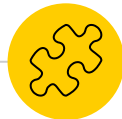
Tais mecanismos incluem os paradigmas da AI que exibem a habilidade de aprender ou adaptar-se a novas situações, generalizar, abstrair, descobrir e associar.



Os quatro principais paradigmas da IC são as redes neurais artificiais (NN), sistemas fuzzy (FS), inteligência de enxame (SI) e computação evolucionária (EC).



SI e EC são normalmente chamadas de metaheurísticas, e são propostas principalmente simulando a natureza ou invocando procedimentos inteligentes aprendidos.



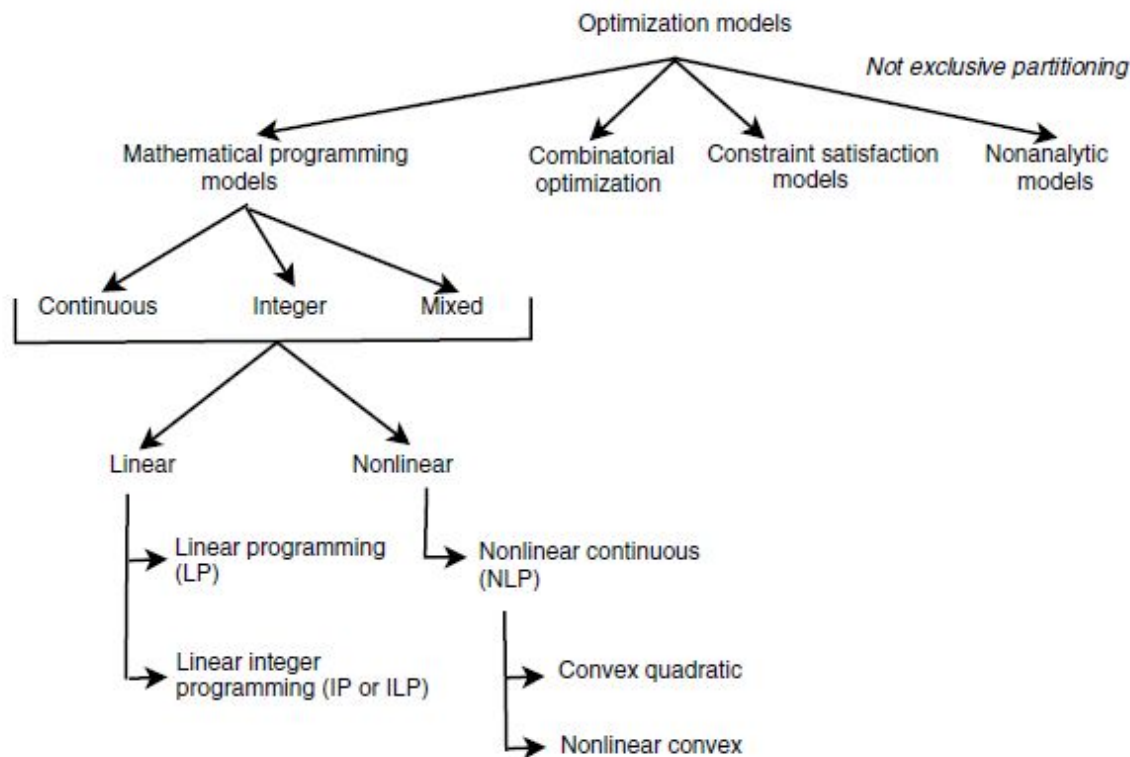
2

Conceitos gerais

Vamos falar um pouco sobre otimização.



Modelos clássicos de otimização



Definição. Uma solução $s^* \in S$ é um ótimo global se tem uma função objetivo melhor que todas as outras soluções no espaço de busca, isto é,

$$\forall s \in S, f(s^*) \leq f(s).$$



“



Complexidade de algoritmos

Definição 1. Notação Big-O. Um algoritmo tem complexidade $f(n) = O(g(n))$ se existem constantes positivas n_0 e c nas quais $\forall n < n_0, f(n) \leq c \cdot g(n)$. Ou seja, a complexidade de $f(n)$ é limitada superiormente por $g(n)$.



Complexidade de algoritmos

Definição 2. Algoritmo de tempo polinomial. Um algoritmo é considerado de tempo polinomial se sua complexidade é $O(p(n))$, em que $p(n)$ é uma função polinomial de n .

Definição 3. Algoritmo de tempo exponencial. Um algoritmo é considerado de tempo exponencial se sua complexidade é $O(c^n)$, onde c é uma constante estritamente maior que 1.



Complexidade de problemas

A complexidade de um problema é equivalente à complexidade do melhor algoritmo que resolve aquele problema. Um problema é **tratável** (ou fácil) se existe um algoritmo de tempo polinomial para resolvê-lo. Um problema é **intratável** se não existe algoritmo de tempo polinomial para resolver o problema.



Complexidade de problemas

Há duas classes de complexidade: **P** e **NP**. **P** representa o conjunto de todos os problemas que podem ser resolvidos por um **algoritmo determinístico** em tempo polinomial. **NP** representa o conjunto de problemas que podem ser resolvidos por um algoritmo **não-determinístico** em tempo polinomial.



Métodos de otimização

Métodos exatos

Obtém soluções ótimas e garantem a otimalidade delas. Para problemas NP, só são viáveis para pequenas instâncias.

Programação dinâmica, A*, Branch and Bound etc.

Métodos aproximativos

Geram soluções de alta qualidade em um tempo adequado para uso prático. Mas não há garantias de que a solução ótima global seja encontrada. São específicos ao problema.

Algoritmos aproximativos e Algoritmos heurísticos

Metaheurísticas

Possuem as mesmas características dos métodos aproximativos, exceto por serem aplicáveis a vários problemas. Eficientes para tratar problemas grandes e complexos.

Algoritmos genéticos, Busca Tabu, Nuvem de partículas etc.



Quando usar metaheurísticas?

Não

- Nos casos em que os algoritmos exatos fornecem um tempo de busca razoável para as instâncias em questão.
- Se o problema, mesmo NP-Completo, possuir instâncias pequenas, ou instâncias maiores com estruturas que possam ser favoráveis ao método exato.

Sim

- Problemas NP-Completo intratáveis por algoritmos exatos dentro do tempo requerido.
- Problemas polinomiais cuja complexidade é muito grande que instâncias reais não podem ser resolvidas em tempo razoável.
- Problemas não-lineares contínuos (NLP), quando os métodos derivativos falham devido ao espaço de busca ser difícil (ruído, descontinuidades, não-linearidade etc.)
- Aplicações em tempo-real.

3

Metaheurísticas

Classificação e uso



A palavra *heurística* vem do grego antigo *heuriskein*, que significa “a arte de descobrir novas estratégias para resolver problemas”. O sufixo *meta*, também vem do grego, e significa “metodologia de mais alto nível”.



Classificação das metaheurísticas

Busca local

- Realizam buscas em um sub-espço do espaço de busca, buscando o mínimo local. A busca local é mais utilizada para aumentar a convergência de algoritmos de busca global.

Hill climbing, simulated annealing, busca tabu, variable neighborhood search, GRASP etc.

Busca global

- Realizam buscas em todo o espaço de busca a fim de encontrar o mínimo global.

Simulated annealing, busca tabu, variable neighborhood search, GRASP, colônia de formigas, nuvem de partículas, algoritmos genéticos



Classificação das metaheurísticas

Solução única

- Focam em modificar e melhorar uma única solução candidata.

Simulated annealing, iterated local search, variable neighborhood search, guided local search etc.

Baseadas em população

- Abordam a manutenção e melhoria de múltiplas soluções candidatas, frequentemente usando características da população para guiar a busca.

Colônia de abelhas, path relinking, colônia de formigas, nuvem de partículas, algoritmos genéticos etc.



Classificação das metaheurísticas

Híbridos

Metaheurísticas combinadas com outros métodos de otimização, como programação linear, aprendizagem de máquina, metaheurística de busca local etc.

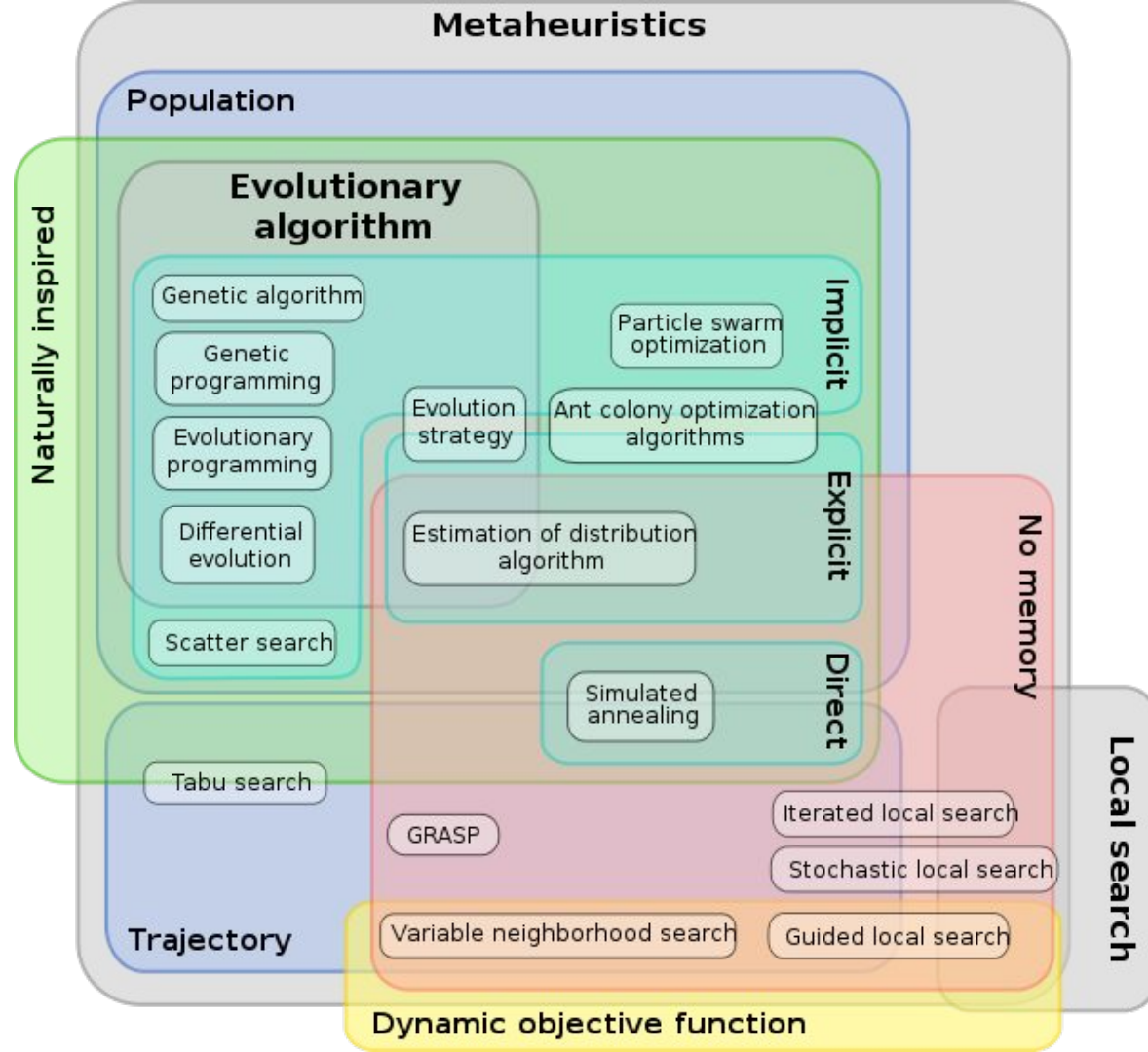
No caso de combinação com **busca local**, é chamado **memético**.

Bio-inspirados

Utilizam metáforas inspiradas em comportamentos observados na natureza para definir suas operações. **Algoritmos evolucionários** e de **inteligência de enxame** são exemplos dessa classe.

Paralelos

Usam técnicas de programação paralela para rodar múltiplas buscas em paralelo. Podem variar de simples esquemas de distribuição até buscas concorrentes que interagem entre si para melhorar a solução global.



4

Alguns exemplos

Algumas das metaheurísticas mais comuns



Busca Tabu

A implementação de busca tabu usa estruturas de memória que descrevem as soluções visitadas ou conjuntos fornecidos pelo usuário de regras.

É possível piorar uma solução se uma melhoria não está disponível (a busca está presa em um mínimo local).

Se uma solução potencial foi já visitada dentro de um determinado período de curto prazo ou se violou uma regra, ela é marcada como "tabu" (proibido), de modo que o algoritmo não considera essa possibilidade repetidamente.



Algoritmos genéticos

Inspirado na teoria da evolução de Darwin, trata cada solução potencial como um indivíduo de uma população genética, realizando cruzamento dos indivíduos mais aptos (melhores soluções) e realizando pequenas e raras mutações, a fim de gerar soluções de alta qualidade.



Nuvem de partículas

Resolve um problema mantendo uma população de soluções candidatas (partículas) e movendo-as em torno do espaço de busca de acordo com fórmulas matemáticas simples a respeito da posição e velocidade da partícula. O movimento de cada partícula é influenciado pela sua melhor posição no local, mas também é orientado em direção às melhores posições globais, que são as posições melhores encontradas por todas as outras partículas. Com isso, é esperado mover o enxame em direção as melhores soluções.



Obrigada!

Alguma pergunta ?

Faça o download dessa apresentação em

<https://github.com/iamcarolalbuquerque/slides>