Computação Bioinspirada

Introdução

Paulo Henrique Ribeiro Gabriel

Faculdade de Computação
Universidade Federal de Uberlândia

2023/1

Nesta aula

- O que é a Computação Bioinspirada?
- Por que estudar/usar a Computação Bioinspirada?
- Exemplos de Computação Bioinspirada

O que é Computação Bioinspirada?

- A Computação Bioinspirada é uma importante área de pesquisa da Ciência de Computação
- Tem como foco o aprendizado de máquina e técnicas de otimização
- É inspirada por princípios biológicos, os quais podem ser utilizados para resolver problemas complexos
  - Uso de sistemas inteligentes

O que é Computação Bioinspirada?

Exemplos dessas técnicas:

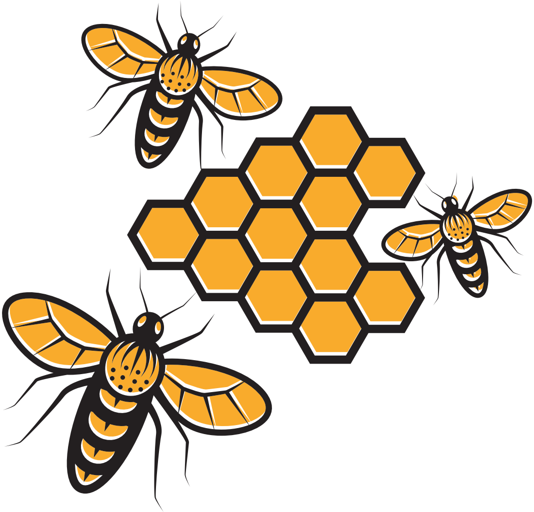
- Redes neurais artificiais
- Algoritmos genéticos
- Inteligência coletiva
- Sistemas imunológicos artificiais
- Técnicas de híbridas

O que é Computação Bioinspirada?

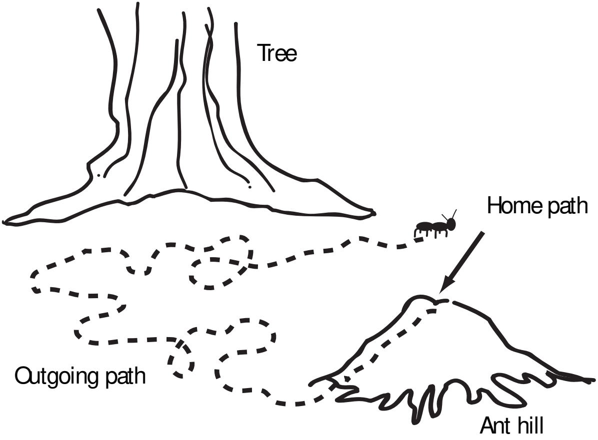
Exemplos de aplicações:

- Bioinformática
- Finanças
- Robótica
- Modelagem e predição de séries temporais
- Modelagem de redes complexas
- Mineração/Agrupamento de dados

Inspiração na natureza

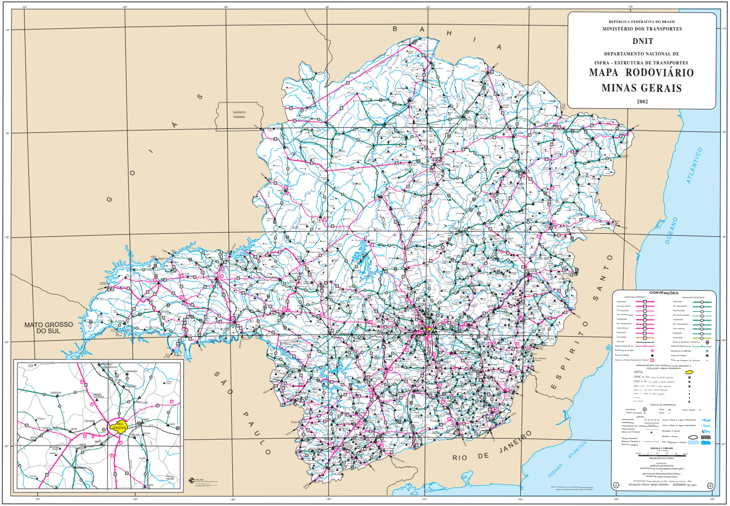


Inspiração na natureza



Por que usar a Computação Bioinspirada?

- No mundo real, existem diversos problemas que podem ser resolvidos utilizando algoritmos e computadores
- Porém, muitos desses problemas são complexos, não podendo ser resolvidos por métodos "convencionais"
- Vamos acompanhar um exemplo...

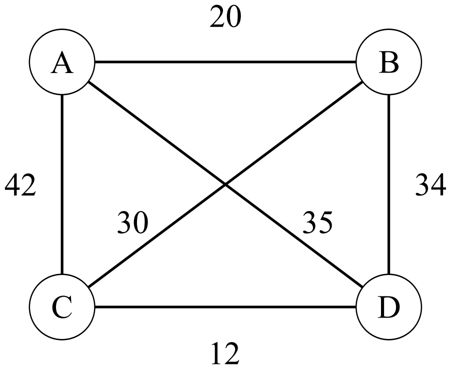

REPÚBLICA FEDERATIVA DO BRASIL
MINISTÉRIO DOS TRANSPORTES
DNIT
DEPARTAMENTO NACIONAL DE
INFRA - ESTRUTURA DE TRANSPORTES
MAPA RODOVIÁRIO
MINAS GERAIS
2002
B A H I A
G O I Á S
DISTRITO FEDERAL
MATO GROSSO DO SUL
S Ã O P A U L O
RIO DE JANEIRO
E S P Í R I T O S A N T O
OCEANO ATLÂNTICO
CONVENÇÕES

phrg@ufu.br

Problema do Caixeiro Viajante

- Um vendedor (caixeiro viajante) tem que visitar n cidades diferentes
  - A viagem começa e termina na *mesma cidade*
- Não importa a *ordem* com que as cidades são visitadas
- De cada uma delas é possível ir diretamente a qualquer outra
- O problema do caixeiro viajante consiste em descobrir a rota que torna *mínima* a viagem total

Problema do Caixeiro Viajante



Para $n = 4$

1. ABCDA
2. ABDCA
3. ACBDA
4. ACDBA
5. ADBCA
6. ADCBA

Problema do Caixeiro Viajante

- O problema do caixeiro é um clássico exemplo de **problema de otimização combinatória**
- Podemos reduzi-lo a um **problema de enumeração**
  - Achamos todas as rotas possíveis
  - Calculamos o comprimento de cada uma delas
  - Escolhemos a melhor

Problema do Caixeiro Viajante

Seja $R(n)$ o número de rotas para n cidades

- A primeira e a última são as mesmas, logo, não precisam entrar no cálculo
- Para a segunda posição, temos $n - 1$ possíveis cidades
- Para a terceira posição, temos $n - 2$ possíveis cidades
- ...
- Para a penúltima cidade, sobrará apenas 1 opção

Logo, temos:

$$R(n) = (n-1) \times (n-2) \times \ldots \times 2 \times 1 = (n-1)!$$

Problema do Caixeiro Viajante

É fácil encontrar $(n-1)!$ rotas?

- Considere o caso em que $n = 20$
- Considere um computador capaz de fazer 1 bilhão de adições por segundo (10^9)
  - Isso significa que, para calcular o comprimento (custo) de uma rota entre 20 cidades, precisamos de 19 somas
  - Ou seja, podemos computar: $10^9/19 \approx 53$ milhões de rotas por segundo
- Porém: $19! \approx 1.2 \times 10^{17}$
- Consequentemente, precisamos de

$$\frac{1.2 \times 10^{17}}{53 \times 10^6} \approx 2.3 \times 10^9 \text{ segundos}$$

Problema do Caixeiro Viajante

A quantidade $(n-1)!$ a uma taxa muito maior que a capacidade que nós temos de processá-la

n	rotas por segundo	$(n-1)!$	cálculo total
5	250 milhões	24	insignificante
10	110 milhões	362880	0.003 seg
15	71 milhões	87 bilhões	20 min
20	53 milhões	1.2×10^{17}	73 anos
25	42 milhões	6.2×10^{23}	470 milhões de anos

Problema do Caixeiro Viajante

Vamos reformular esse problema:

- Tempo que um robô industrial gasta para soldar a carcaça de um automóvel
- Custo (ou combustível) na rota da distribuição diária de marmita em uma grande cidade
- Tempo na rota de abastecimento das vários centros logísticos de uma distribuidora

Problema do Caixeiro Viajante

- A existência ou não de um método eficiente para resolver o problema do caixeiro viajante é um dos grandes problemas em aberto da Matemática
- Uma grande quantidade de problemas importantes podem ser reduzidos ao problema do caixeiro

COOK, S. A. The complexity of theorem-proving procedures. Proceedings of the third annual ACM symposium on Theory of computing - STOC '71. Em: THE THIRD ANNUAL ACM SYMPOSIUM. ACM Press, 1971. Disponível em: http://dx.doi.org/10.1145/800157.805047

KARP, R. M. Reducibility among Combinatorial Problems. Complexity of Computer Computations Springer US, 1972. Disponível em: http://dx.doi.org/10.1007/978-1-4684-2001-2_9

Otimização Combinatória

- Um problema de otimização tem como objetivo encontrar a melhor solução possível para uma determinada função sobre algum domínio

Otimização Combinatória

- Um problema de otimização tem como objetivo encontrar a melhor solução possível para uma determinada função sobre algum domínio
- Quando o domínio é finito, dizemos que o problema é de otimização combinatória

Otimização Combinatória

Formalmente

Um problema de otimização combinatória é uma 4-upla $\Pi = (X, Y, F, O)$ tal que:

- X: Espaço de instâncias ou entrada do problema
- Y: Espaço de soluções do problema
- F: Função de viabilidade do problema, $F : X \times Y \to \{0, 1\}$
- O: Função objetivo do problema, $O : X \times Y \to \mathbb{R}$

Otimização Combinatória

Mais alguns termos:

- $x \in X$ é uma instância do problema Π
- $y \in Y$ é uma solução do problema Π
- Uma solução y para uma instância x é factível se $F(x, y) = 1$

Otimização Combinatória

Meta

Encontrar $y^* \in Y$ factível que minimize (ou maximize) a função objetivo $O(x, y)$

- y^* é chamado solução ótima
- $OPT(x)$ é o valor de $O(x, y^*)$

Exemplo

Problema do caixeiro viajante

Entrada: Grafo não-dirigido $\mathcal{G} = (V, E)$ e custos $c_e \geq 0$, para todo $e \in E$.

Saída: Encontrar um circuito de menor custo que visita cada "cidade" $(v \in V)$ exatamente uma vez.

Otimização Combinatória

A partir de um problema $\Pi = (X, Y, F, O)$, podemos definir alguns problemas algorítmicos:

- Problema de busca: Dado $x \in X$, encontrar um ótimo y^*
- Problema computacional: Dado $x \in X$, encontrar $OPT(x)$
- Problema de decisão: Dado $x \in X$ e $k \in \mathbb{R}^+$, existe $OPT(x) \geq k$?

Otimização Combinatória

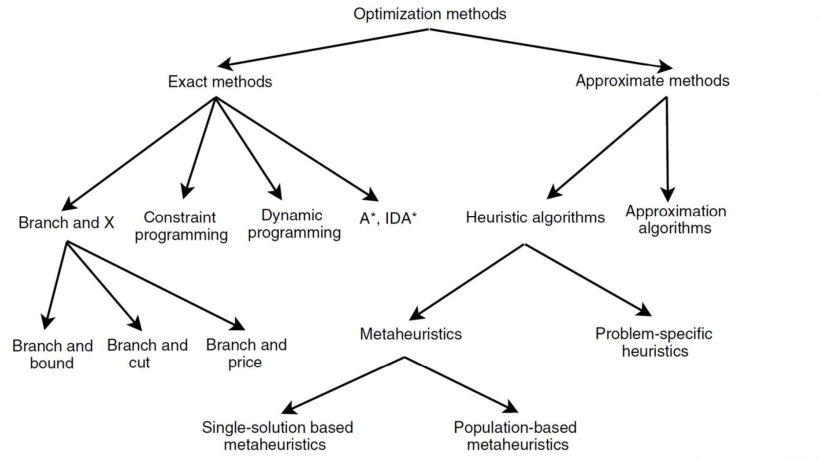
A partir de um problema $\Pi = (X, Y, F, O)$, podemos definir alguns problemas algorítmicos:

- Problema de busca: Dado $x \in X$, encontrar um ótimo y^*
- Problema computacional: Dado $x \in X$, encontrar $OPT(x)$
- Problema de decisão: Dado $x \in X$ e $k \in \mathbb{R}^+$, existe $OPT(x) \geq k$?

Ou seja...

- Para cada problema de otimização combinatória, existe um problema de decisão correspondente que pergunta se existe uma solução factível
- Muitos desses problemas de decisão são do tipo $\mathcal{NP}$-Completos

Métodos de Otimização



Meta-heurísticas

- Técnicas iterativas que buscam melhorar uma ou mais soluções
- Requerem pouco conhecimento do problema
- Precisam distinguir boas soluções
- Geralmente, encontram boas soluções (mas não garantem o ótimo)
- São adaptáveis: possuem parâmetros ajustáveis

Meta-heurísticas

Algorithm 3.1: General Optimization Algorithm

```
set initial control parameters
begin
    t = 0
    initialize candidate(s)
    evaluate initial candidate(s)
    while not termination-condition do
        t = t + 1
        generate new candidate(s)
        evaluate new candidate(s)
        select solution(s) for next iteration
        optional: update control parameters
    end
end
```

Meta-heurísticas

- Grande área: Inteligência Computacional
- Tratam o problema a ser resolvido como um conjunto de informações a partir das quais algo deverá ser extraído ou inferido
- O processo de solução corresponde a uma sequência de ações que levam a um desempenho desejado ou melhoram o desempenho relativo de soluções candidatas
- Este processo de procura por um desempenho desejado é denominado busca

Meta-heuríticas

- Técnicas de inteligência computacional são técnicas alternativas
- Ou seja, existem outras maneiras para se resolver um mesmo problema
- É preciso avaliar com cuidado se há ou não a necessidade de aplicação dessas técnicas a um dado problema

Meta-heurísticas: Quando usar?

- Em problemas para os quais existe pouca informação
- Quando uma exploração completa é impossível devido ao tamanho do espaço
- Porém se você tem uma solução candidata, ela pode ser avaliada

phrg@ufu.br

Agradecimentos

Esse material foi baseados em notas escritas pelos professores Dra. Aurora Pozo (UFPR)