Inteligência Artificial, Otimização Combinatória: Uma Apresentação

Claudio Cesar de Sá[1]

Departamento de Ciência da Computação
Centro de Ciências e Tecnológias
Universidade do Estado de Santa Catarina

Agenda

1. Um sobrevôo na IA
2. O que é complexidade computacional?
3. Um sobrevôo em otimização
4. Um exemplo: modelagem, código e resultados
5. Tendências

UDESC

Caos + Complexidade + Inteligência . . . = Comportamento Emergente

Figura: Comportamento emergente, inteligente, complexo ou caótico?

Caos no Cotidiano

- Crescimento das cidades
- Mudanças ambientais (em parte previsível)
- Desastres ecológicos – Katrina
- Poluição, lixo, ..., etc
- Comunicação Social:

Caos no Cotidiano

- Crescimento das cidades
- Mudanças ambientais (em parte previsível)
- Desastres ecológicos – Katrina
- Poluição, lixo, ..., etc
- Comunicação Social:
- Há uma aleatoriedade *embutida* nestes eventos!

Caos no Cotidiano

- Crescimento das cidades
- Mudanças ambientais (em parte previsível)
- Desastres ecológicos – Katrina
- Poluição, lixo, ..., etc
- Comunicação Social:
- Há uma aleatoriedade *embutida* nestes eventos!
- Embora computáveis, alguns distantes de terem uma prática!

Complexo = Difícil (achar a cifragem do Enigma)

Figura: *Precisamos de uma máquina que calcule sobre outra máquina!*

Origens da Inteligência Artificial

Histórico – alguns fatos

- Sim, foi logo após a morte de Alan Turing (1954)
- Motivação: máquinas que apresentassem *comportamentos inteligentes*. Exemplo: jogo de xadrez

Origens da Inteligência Artificial

Histórico – alguns fatos

- Sim, foi logo após a morte de Alan Turing (1954)
- Motivação: máquinas que apresentassem *comportamentos inteligentes*. Exemplo: jogo de xadrez
- Máquinas que provassem teoremas (verdades matemáticas)
- Usassem um senso-comum de um ser humano (as *lógicas*)

Origens da Inteligência Artificial

Histórico – alguns fatos

- Sim, foi logo após a morte de Alan Turing (1954)
- Motivação: máquinas que apresentassem *comportamentos inteligentes*. Exemplo: jogo de xadrez
- Máquinas que provassem teoremas (verdades matemáticas)
- Usassem um senso-comum de um ser humano (as *lógicas*)
- Logo, há um *mix* de áreas: psicologia, matemática, lógica (há lugares tratado apenas pela filosofia), ..., e computação!

Origens da Inteligência Artificial

Histórico – alguns fatos

- Sim, foi logo após a morte de Alan Turing (1954)
- Motivação: máquinas que apresentassem *comportamentos inteligentes*. Exemplo: jogo de xadrez
- Máquinas que provassem teoremas (verdades matemáticas)
- Usassem um senso-comum de um ser humano (as *lógicas*)
- Logo, há um *mix* de áreas: psicologia, matemática, lógica (há lugares tratado apenas pela filosofia), ..., e computação!
- Há uma complexidade nisto tudo!

UDESC

Motivações aos *Toys-Problem*

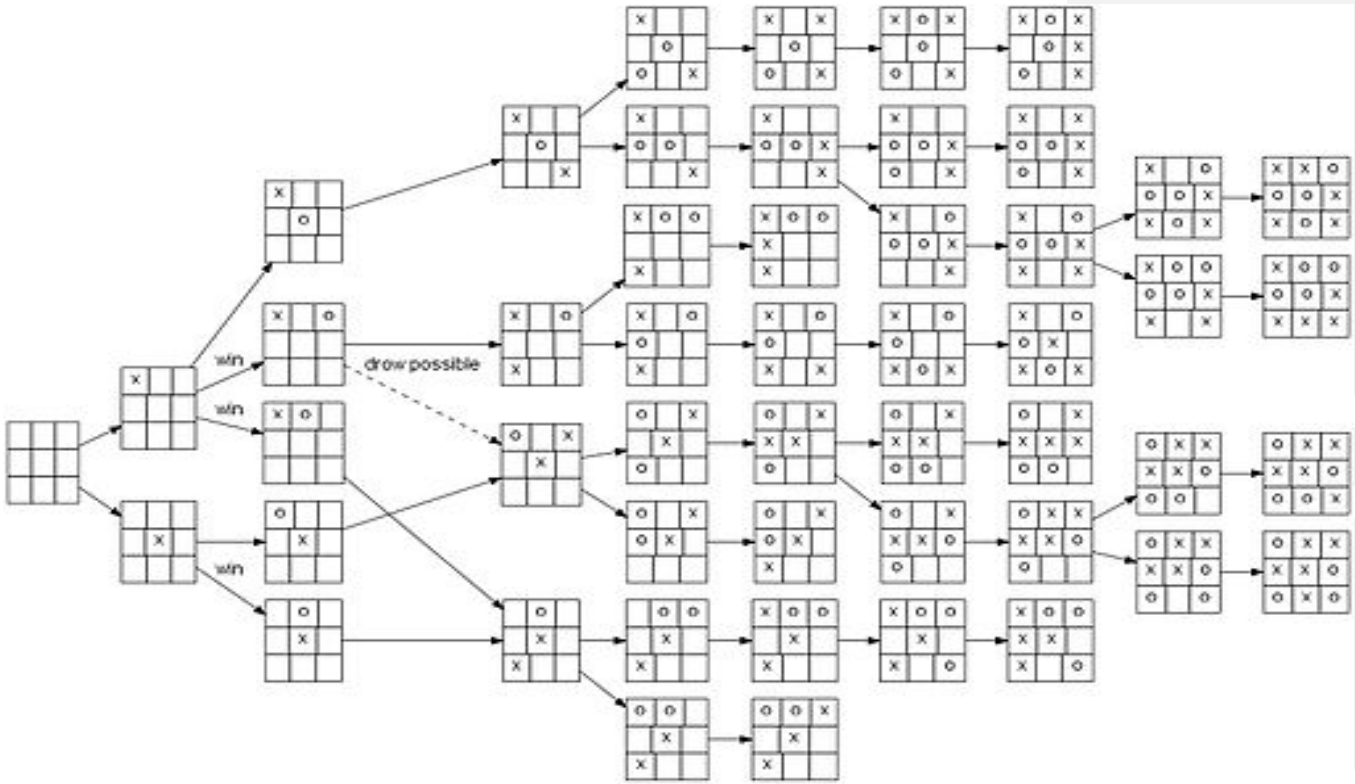


Figura: Problemas que usassem uma habilidade de pensar do ser humano, reproduzir este conhecimento em uma máquina!

Principais Áreas da IA

- Lógica: visa automatizar a cognição humana
- Redes Neurais: reproduz o comportamento neurônios
- Aprendizagem de Máquinas: adquire um conhecimento e exibe o que foi aprendido
- Raciocínio Incerto: visa inferir uma valoração ao conhecimento
- Agentes: estabeleceu paradigmas de autonomia e inteligência
- Robótica: tudo embarcado em um hardware que faça algo
- Ferramentas: como implementar tudo isto. Linguagens de programação: LISP (1960) e PROLOG (197X)
- Sim ... mas há muito mais!

UDESC

Principais Áreas da IA

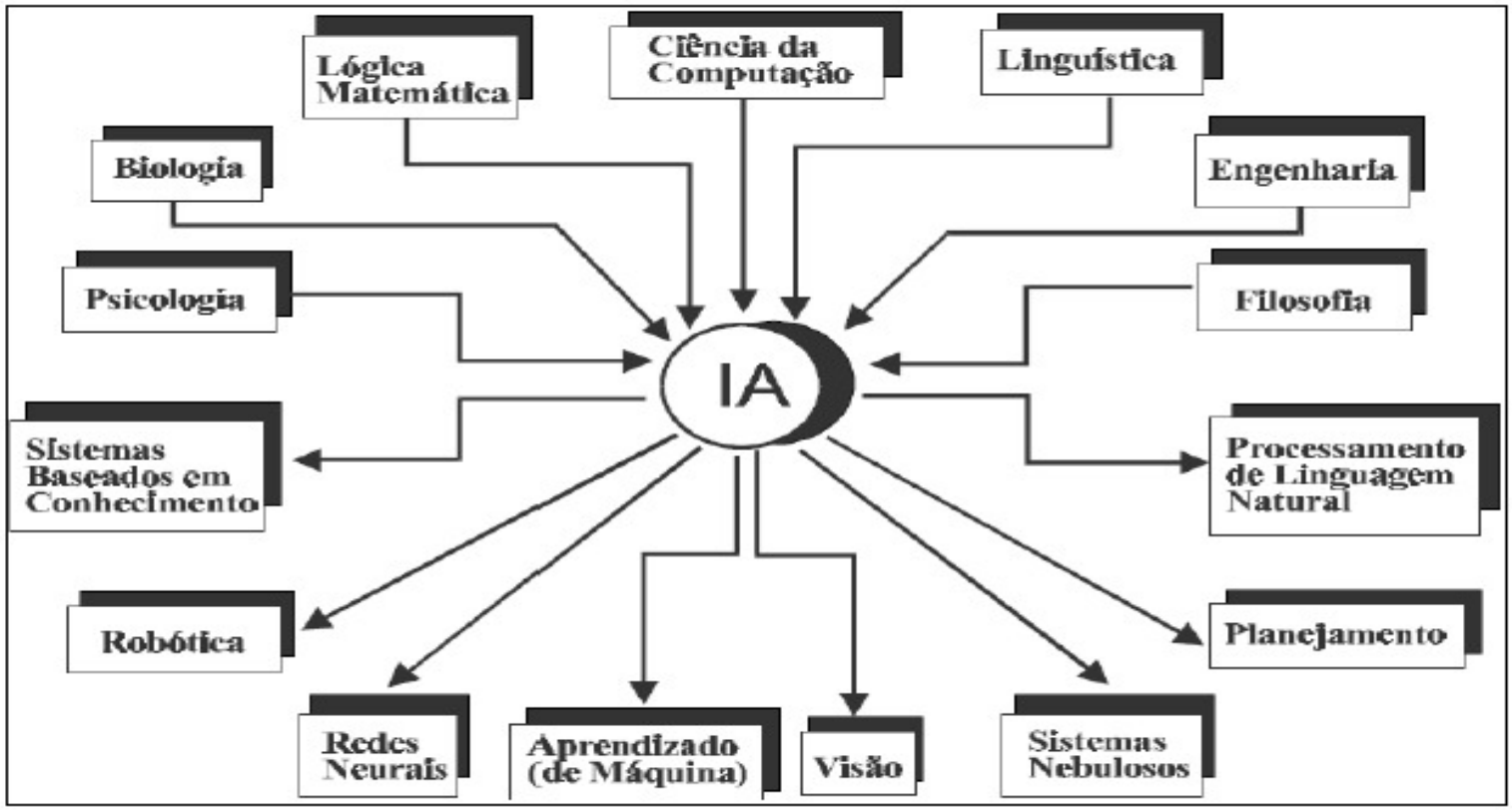


Figura: Observe o sentido das setas !

Sucessos da IA

Figura: Teoria publicada em 1986 ... Sojourner (a esquerda) 1996

Sucessos da IA

Figura: Carro da Google – Sucesso de Marketing

UDESC

Sucessos da IA

Figura: Dava-se um fato ou a resposta, a idéia era encontrar a pergunta certa!

Sucessos da IA

Figura: Impulsão bélica: os drones!

Outros Sucessos da IA

- Mineração de Dados: aprendizagem a partir de dados
  Exemplos: Google, Facebook, etc.
- Reconhecimento de Padrões: imagens, voz, movimentos
- Sistemas de Segurança: Biometria

UDESC

Computação Natural ⊃ IA

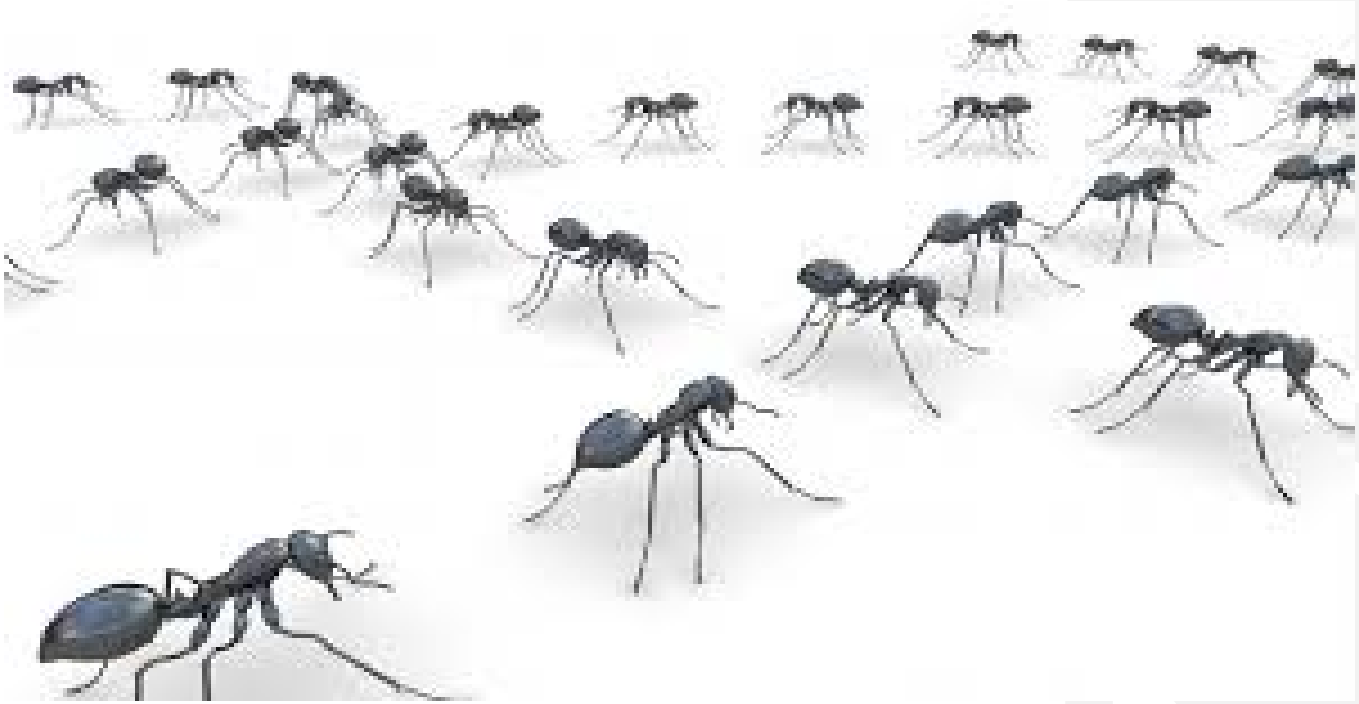
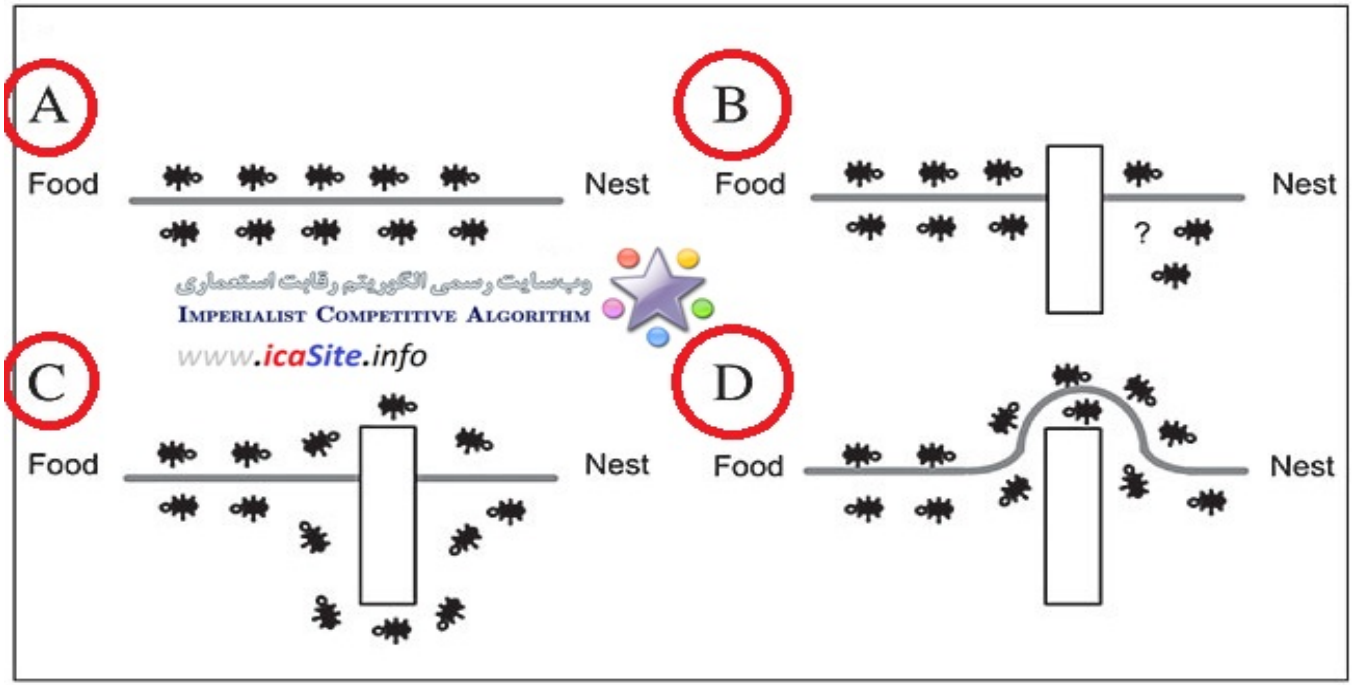


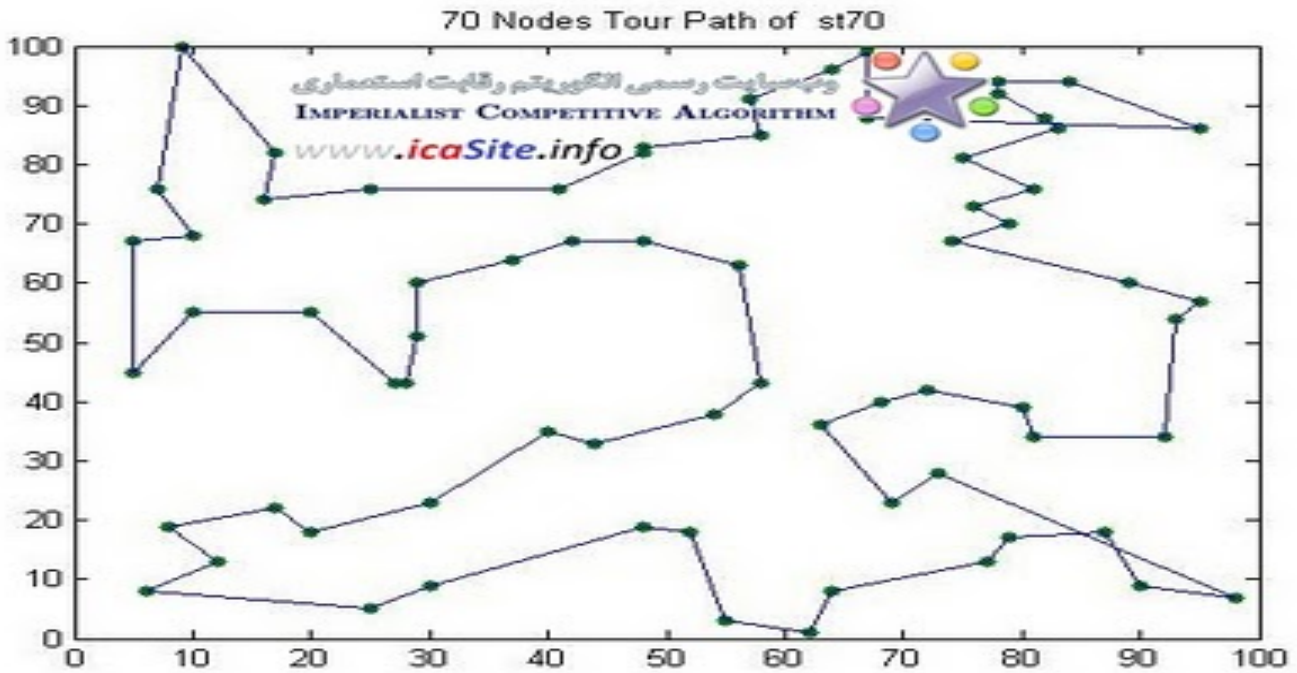
Figura: Computação Inspirada em Seres Biológicos (sua evolução *Darwiniana*)

UDESC

Computação via Colônia de Formigas



Aplicação 1: TSP, um problema difícil



Aplicação 2: encontrar um *ponto de maior ganho*!

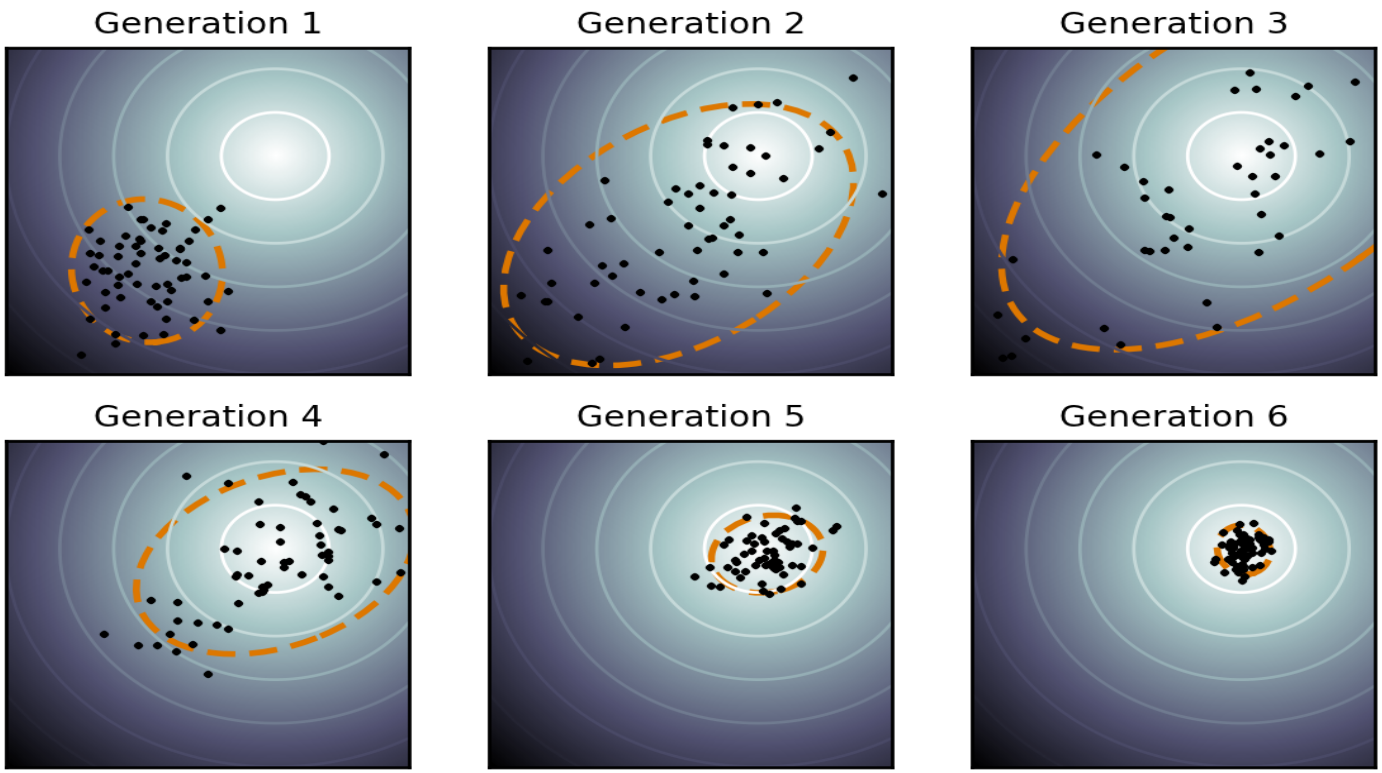


Figura: Computação Inspirada em Seres Biológicos (a evolução *Darwiniana*)

IA × Otimização Combinatória (OC)

IA ⇔ OC:

- Ambas as áreas abordam problemas complexos!
- IA: diversas direções e muitos paradigmas
- Otimização Combinatória (OC): usa modelos como a IA, rígidos, espaços definidos ...
- A área de **Otimização** tem uma divisão: **Discreta** ou **Combinatória** e **Contínua** ou **Numérica** (não é o foco – funções deriváveis)
- Atacar problemas difíceis! Contudo, o que é *difícil*?
- Vamos definir *uma medida de* ***complexo***

Definindo uma Medida Complexidade

- Na área de CC tem uma medida clássica
- Classificar os problemas em **polinomiais** e **exponenciais**
- Assim, os problemas **exponenciais** são os mais intrigantes ...
- O que é isto?

Problema da Satisfatibilidade

Fórmulas Lógicas

Seja uma fórmula $\varphi_1(x)$ sobre o domínio $\{0, 1\}$, temos a sua interpretação dada Tabela Verdade abaixo:

x	$\varphi_1(x)$
1	**1**
0	**0**

Árvore semântica

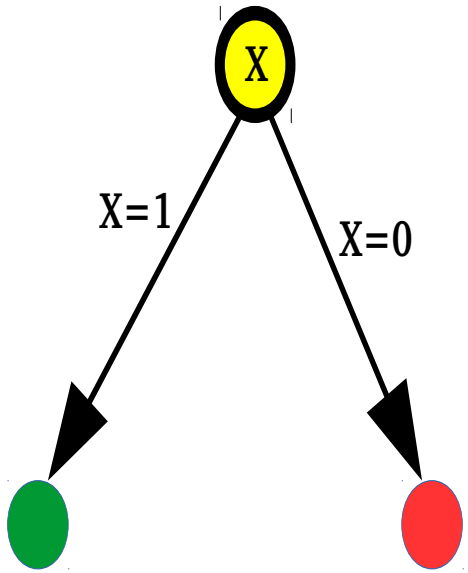


Figura: Representando as validades da função $\varphi_1(x)$

Fórmulas Lógicas

Seja uma fórmula $\varphi_2(x, y) = (\sim x \wedge y) \vee (x \wedge \sim y)$ sobre o domínio $\{0, 1\}$, temos a sua interpretação dada Tabela Verdade abaixo:

x	y	$(\neg$	x	$\wedge$	$y)$	$\vee$	$(x$	$\wedge$	$\neg$	$y)$	$\varphi_2(x, y)$
1	1	0	1	0	1	**0**	1	0	0	1	**0**
1	0	0	1	0	0	**1**	1	1	1	0	**1**
0	1	1	0	1	1	**1**	0	0	0	1	**1**
0	0	1	0	0	0	**0**	0	0	1	0	**0**

Sua árvore semântica é dada por:

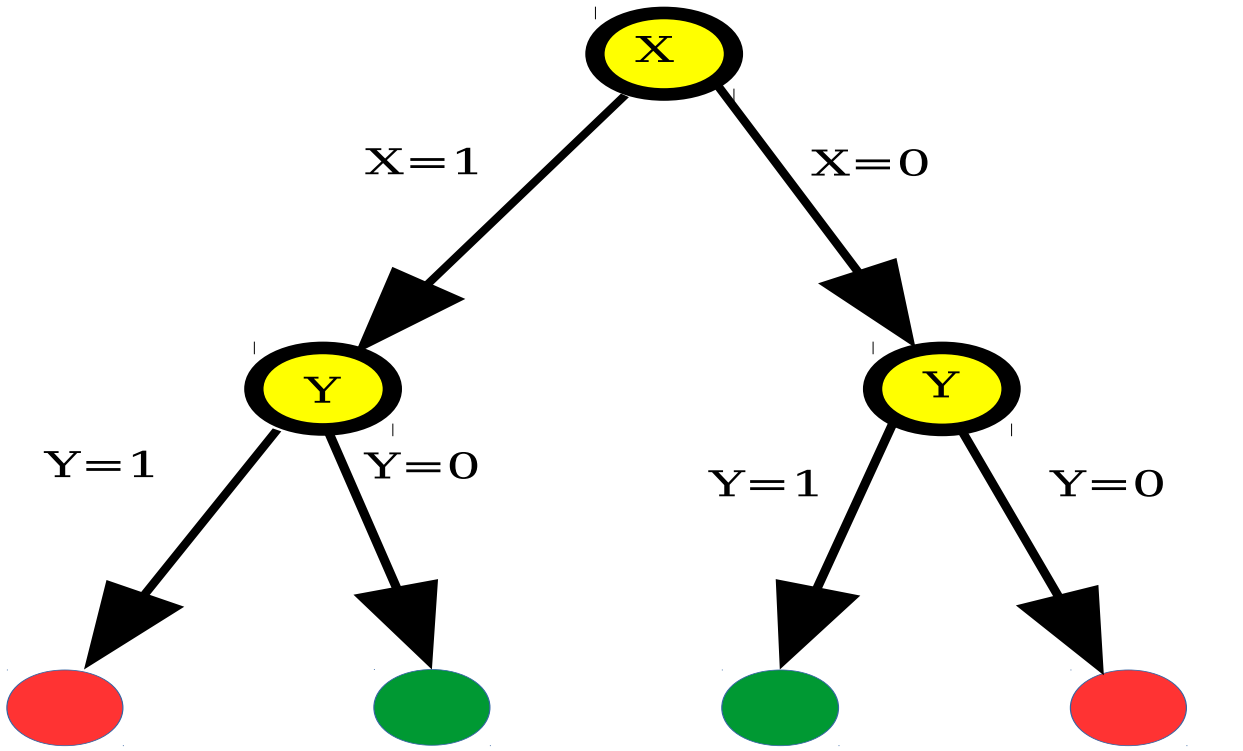


Figura: Representando as validades da função $\varphi_2(x, y)$

Fórmulas Lógicas

Seja uma fórmula $\varphi_3(x, y, z) = (\sim x \vee y) \wedge (\sim y \vee z)$ sobre o domínio $\{0, 1\}$, sua tabela verdade:

x	y	z	$(\neg$	x	$\vee$	$y)$	$\wedge$	$(\neg$	y	$\vee$	$z)$	$\varphi_3(x, y, z)$
1	1	1	0	1	1	1	**1**	0	1	1	1	**1**
1	1	0	0	1	1	1	**0**	0	1	0	0	**0**
1	0	1	0	1	0	0	**0**	1	0	1	1	**0**
1	0	0	0	1	0	0	**0**	1	0	1	0	**0**
0	1	1	1	0	1	1	**1**	0	1	1	1	**1**
0	1	0	1	0	1	1	**0**	0	1	0	0	**0**
0	0	1	1	0	1	0	**1**	1	0	1	1	**1**
0	0	0	1	0	1	0	**1**	1	0	1	0	**1**

Sua árvore semântica:

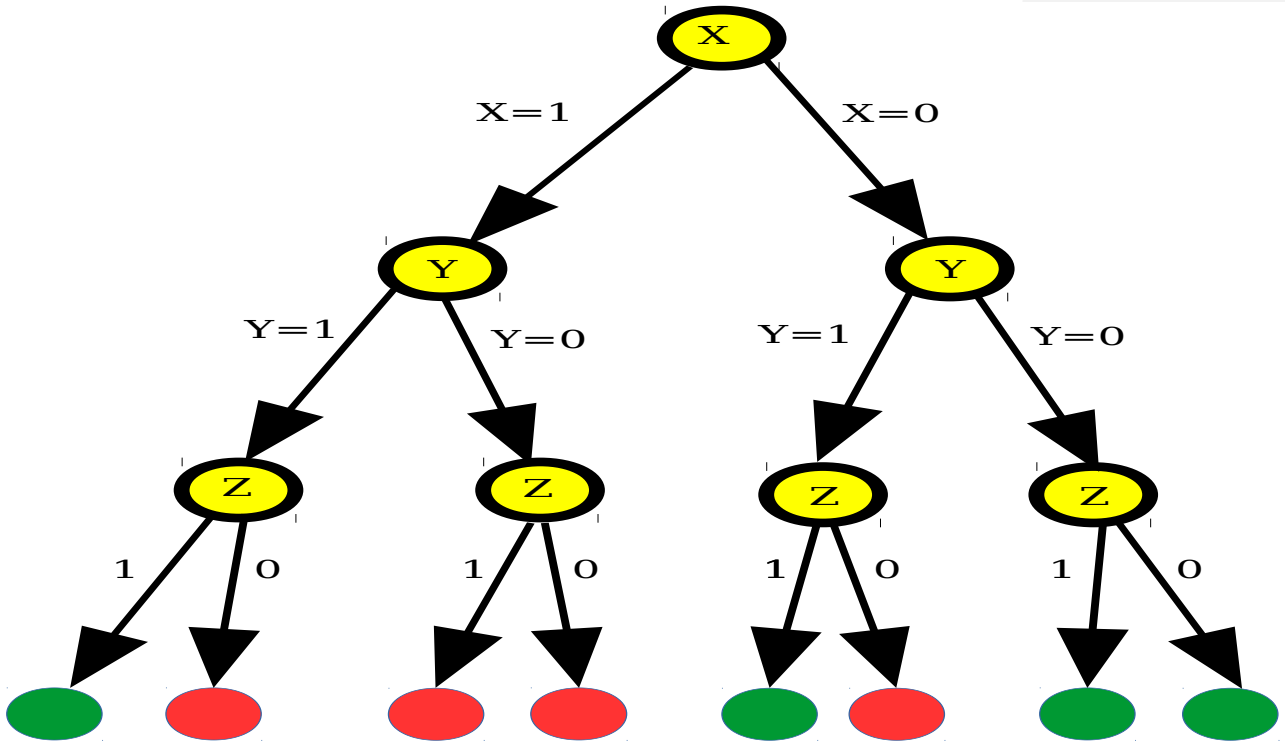


Figura: Representando as validades da função $\varphi_3(x, y, z)$

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

- Claro, o mesmo domínio: $\{0, 1\}$
- O número de linhas cresceu ....

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

- Claro, o mesmo domínio: $\{0, 1\}$
- O número de linhas cresceu ....
- Sim, o número de linhas cresceu e exponencialmente: 2^n onde é o número de variáveis
- Quanto a base 2 veio tamanho do domínio: $\{0, 1\}$

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

- Claro, o mesmo domínio: $\{0, 1\}$
- O número de linhas cresceu ....
- Sim, o número de linhas cresceu e exponencialmente: 2^n onde é o número de variáveis
- Quanto a base 2 veio tamanho do domínio: $\{0, 1\}$
- E quando este número de variáveis e domínio forem maiores?

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

- Claro, o mesmo domínio: $\{0, 1\}$
- O número de linhas cresceu ....
- Sim, o número de linhas cresceu e exponencialmente: 2^n onde é o número de variáveis
- Quanto a base 2 veio tamanho do domínio: $\{0, 1\}$
- E quando este número de variáveis e domínio forem maiores?
- Isto mesmo, temos $\mid D \mid^n$, uma exponencial!

O que temos em comum entre $\varphi_1(x)$, $\varphi_2(x, y)$ $\varphi_3(x, y, z)$?

- Claro, o mesmo domínio: $\{0, 1\}$
- O número de linhas cresceu ....
- Sim, o número de linhas cresceu e exponencialmente: 2^n onde é o número de variáveis
- Quanto a base 2 veio tamanho do domínio: $\{0, 1\}$
- E quando este número de variáveis e domínio forem maiores?
- Isto mesmo, temos $\mid D \mid^n$, uma exponencial!
- Logo, problemas que tenham uma ordem maior ou igual a $2^{O(n)}$ são exponenciais, consequentemente, difíceis!

UDESC

Classe de Problemas e o interesse da IA

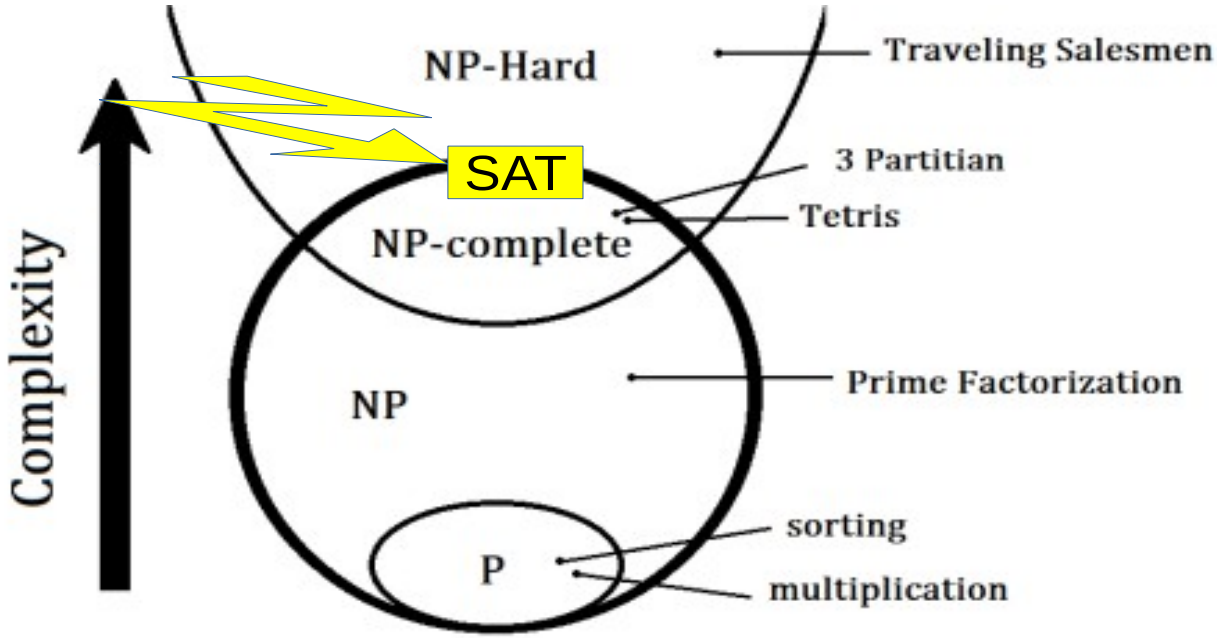


Figura: Problemas e suas complexidades

Introdução à Otimização

Classes de Problemas

Problemas de otimização são geralmente divididos em dois tipos: *otimização combinatorial* (*discreta*) e *otimização numérica* (*contínua*)

Combinatorial Problemas definidos em um espaço de estados finito (ou infinito mas enumerável)

Numérica Definidos em subespaços infinitos e não enumeráveis, como os números reais e complexos

UDESC

Elementos de uma Otimização

Min ou Max

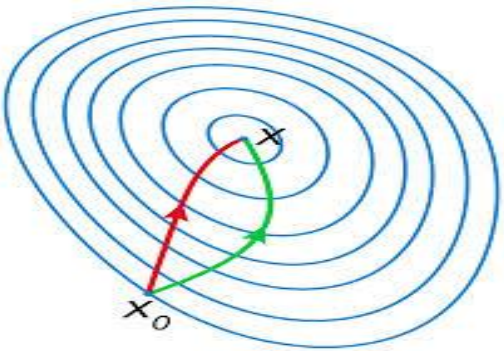
$$f(x)$$

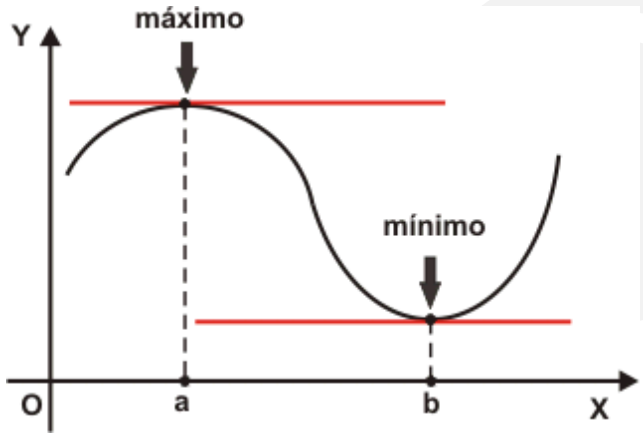
Sujeito a

$$h_k(x) = 0 \qquad k = 1, ..., K$$
$$g_j(x) \geq 0 \qquad j = 1, ..., J$$
$$x_i^{(U)} \geq x_i \geq x_i^{(L)} \qquad i = 1, ..., N$$

Tipos de Variáveis:

- Contínua
- Combinatória/Discreta
- Mista





Mínimo/Máximo: Local x Global

Otimização Combinatorial - Exemplo

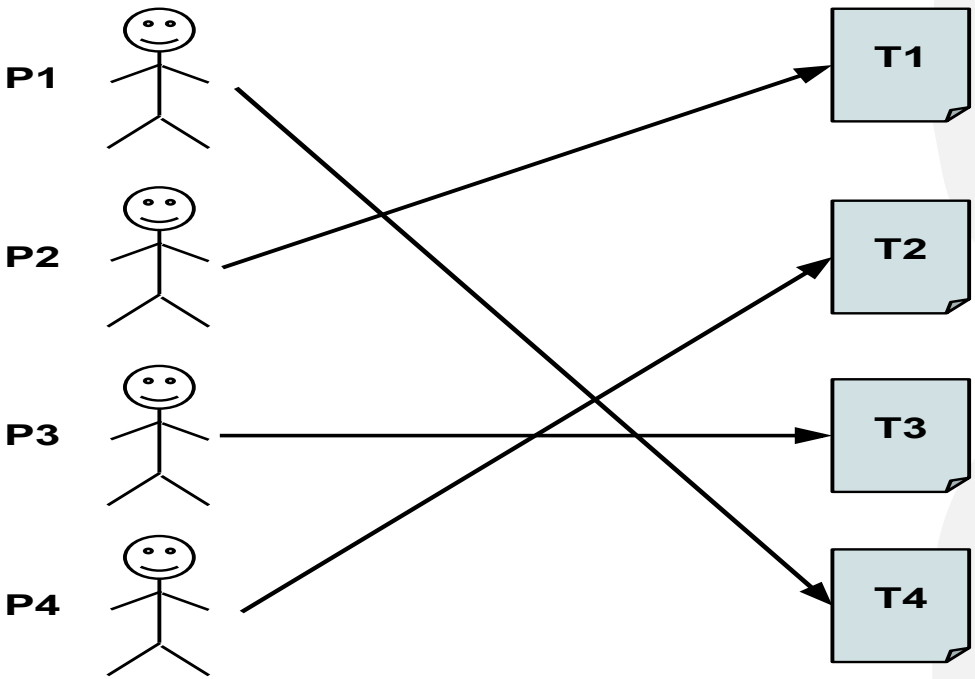
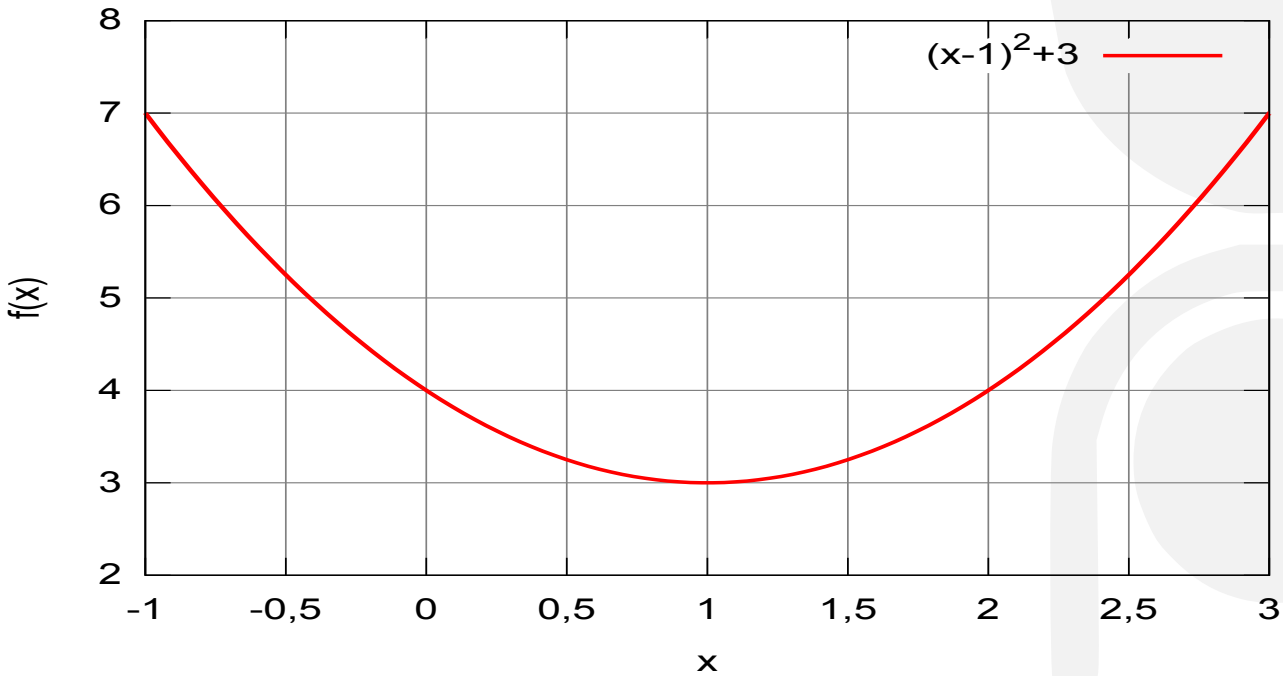


Figura: Atribuição ou designação de trabalhos

Otimização Numérica - Exemplo

Minimizar a função $f(x) = (x-1)^2 + 3$.



As técnicas:

Combinatória:
- Busca Local
- Métodos Gulosos: busca tipo subida a encosta (*hill-climbing*), recozimento simulado (*simulated annealing*), busca tabu, etc
- Programação Dinâmica
- **Programação por Restrições**
- Redes de Fluxo
- .....

Numérica:
- Descida do Gradiente
- Gauss-Newton
- Lavemberg-Marquardt
- .....

Um Problema NP-Difícil: Cabo de Guerra

Critério de escolha do times: por peso

Figura: *O mais pesado tem mais força!*

Especificando o problema

Que seja feita a divisão:

$Joao_1$	$Pedro_2$	$Manoel_3$	....	$Zeca_n$
45	39	79	....	42

- Divisão por peso
- Respeitar critérios como: $|N_A - N_B| \leq 1$
- Todos devem brincar
- Bem, esta simples **restrição** ($|N_A - N_B| \leq 1$), de nosso cotidiano tornou um simples problema em mais uma questão combinatória. Um arranjo da ordem de $\frac{n!}{(n/2)!}$. Casualmente, nada trivial para grandes valores!

Estratégia de Modelagem

- Variável de Decisão: análogo a árvore do SAT

Nomes (n_i):	n_1	n_2	n_3	....	n_n
Peso (p_i):	45	39	79	....	42
Binária (x_i):	0/1	0/1	0/1	....	0/1

- Assim $N_A \approx N/2$, $N_B \approx N/2$ e $|N_A - N_B| \leq 1$
- $x_i = 0$: n_i fica para o time A
- $x_i = 1$: n_i fica para o time B
- Logo a soma:

$$\sum_{i=1}^{n} x_i p_i$$

é o peso total do time B (P_B)

Modelagem das Restrições

- Falta encontrar peso total do time A (P_A), dado por:
- $P_A = P_{total} - P_B$
- ou

$$P_A = \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} x_i p_i$$

- Finalmente, aplicar uma minimização na diferença: $|P_A - P_B|$

Uma Estratégia de Implementação I

Um vetor binário de n posições representando a escolha de cada um no time:

$x_1 : 0/1$	$x_2 : 0/1$	$x_3 : 0/1$	$x_4 : 0/1$	....	$x_n : 0/1$

Uma Estratégia de Implementação II

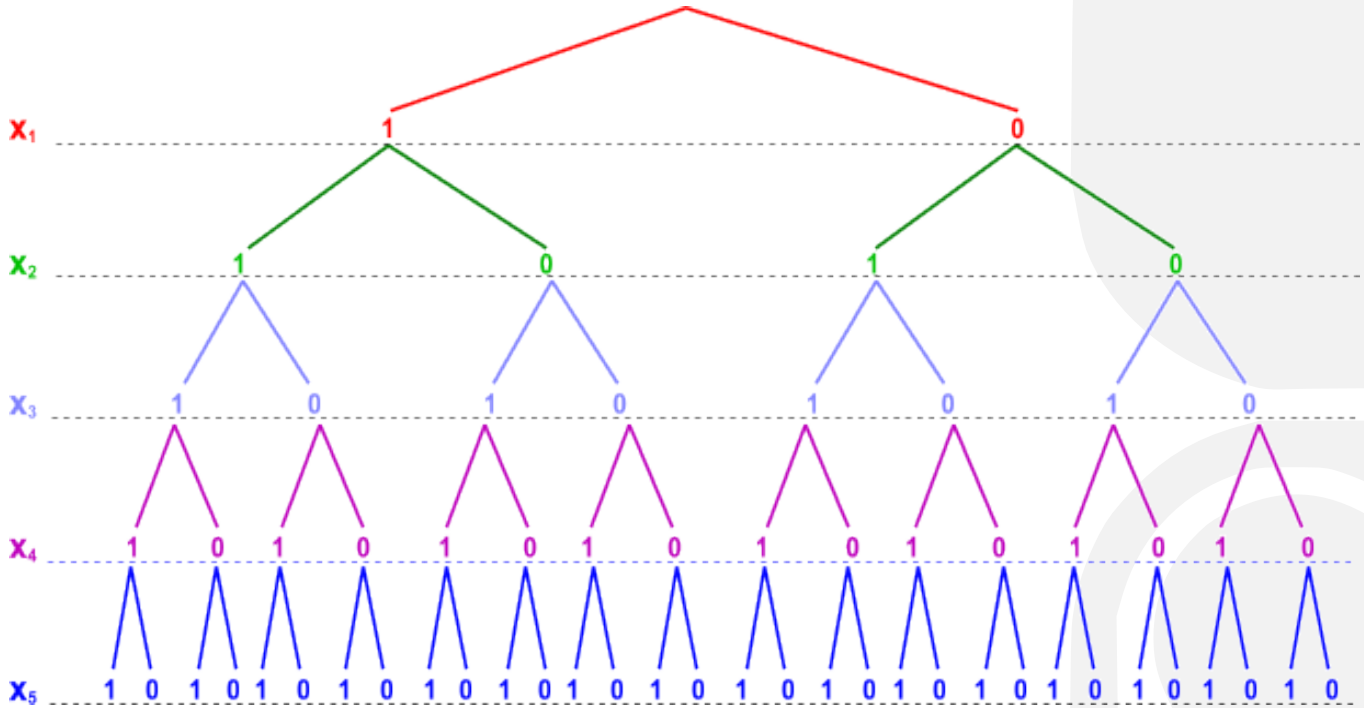


Figura: Se $x_i = 0$, então n_i segue para o time A, caso $x_i = 1$, então n_i vai para o time B

Implementação em Minizinc I

```
1  %%%% CABO DE GUERRA
2  int:  n = 8;  %% total de pessoas
3
4  %%% quantidade de pessoas e seu peso
5  array[1..n] of int : peso;
6  peso = [ 77, 97, 120, 45, 57, 96, 100, 59 ];
7
8  %% Quantas pessoas em cada lado
9  var int: NA;
10 var int: NB;
11
12 %% Variavel de decisao BINARIA
13 array[1..n] of var 0..1 : V_DEC;
14 var int: PESO_TOTAL;
15 var int: PA;
16 var int: PB;
17
```

Implementação em Minizinc II

```
18 %%% Ilustrando uma funcao em MINIZINC
19 function var int: metade( int: n) = n div 2 ;
20
21 %% quantos em cada lado
22 constraint
23    NA == metade(n)    /\ %% and
24    NB == (n - NA); %% LOGO:  NB >= NA
25
26 constraint %% B ... selecionados
27   NB == sum( i in 1..n ) ( V_DEC[i] );
28
29 constraint
30   PESO_TOTAL = sum([peso[i] | i in 1..n]) ;
31
32 constraint
33   PB = sum([V_DEC[i]*peso[i] | i in 1..n]);
34
```

Implementação em Minizinc III

```
35 constraint
36   PA == (PESO_TOTAL - PB);
37
38 % minimizar a diferenca entre os PESOS -- F_OBJETIVO
39 solve minimize abs(PA - PB);
40
41 output [
42   " Peso Total: ", show(PESO_TOTAL), "\t PA : ", show( PA ),
43   "\t PB : ", show( PB ),
44   "\n Vetor de pesos: ", show(peso),
45   "\n V_BIN_TIMES B/1 A/0: ", show(V_DEC )] ++
46   ["\n TIME A: ",
47    show([peso[i] | i in 1..n where fix(V_DEC[i]) == 0 ])
48   ]  ++  ["\n TIME B: ",
49    show([V_DEC[i]*peso[i] | i in 1..n where fix(V_DEC[i]) == 1 ])
50   ];
```

Resultados e Análise

```
Finished in 50msec
Compiling cabo_de_guerra.mzn
Running cabo_de_guerra.mzn
 Peso Total: 651  PA : 312  PB : 339
 Vetor de pesos: [77, 97, 120, 45, 57, 96, 100, 59]
 V_BIN_TIMES B/1 A/0: [1, 1, 1, 1, 0, 0, 0, 0]
----------
 Peso Total: 651  PA : 332  PB : 319
 Vetor de pesos: [77, 97, 120, 45, 57, 96, 100, 59]
 V_BIN_TIMES B/1 A/0: [0, 1, 1, 1, 1, 0, 0, 0]
----------
 Peso Total: 651  PA : 324  PB : 327
 Vetor de pesos: [77, 97, 120, 45, 57, 96, 100, 59]
 V_BIN_TIMES B/1 A/0: [1, 1, 0, 0, 1, 1, 0, 0]
 TIME A: [120, 45, 100, 59]
 TIME B: [77, 97, 57, 96]
----------
```

Resultados e Análise

Números aleatórios aos pesos: 1 a 150

Usando um *solver* médio do Minizinc (*G12 lazyfd*) padrão:

n	tempo	P_A	P_B
5	40msec	276	278
10	46msec	518	519
25	98msec	1198	1197
50	411msec	2290	2291
75	**2s 485msec**	3133	3133
100	470msec	4142	4142
125	**7s 2msec**	4992	4992
150	605msec	5823	5823
175	642msec	6777	6778
200	$>$ 10min	–	–

Reflexões

- Enfim, este problema é uma variação de clássicos NPs, mais especificamente o *sub-set-sum*
- Leia-se: Problema da Mochila
- Implemente este problema usando Programação Dinâmica (PD)

UDESC

Finalizando estes exponenciais

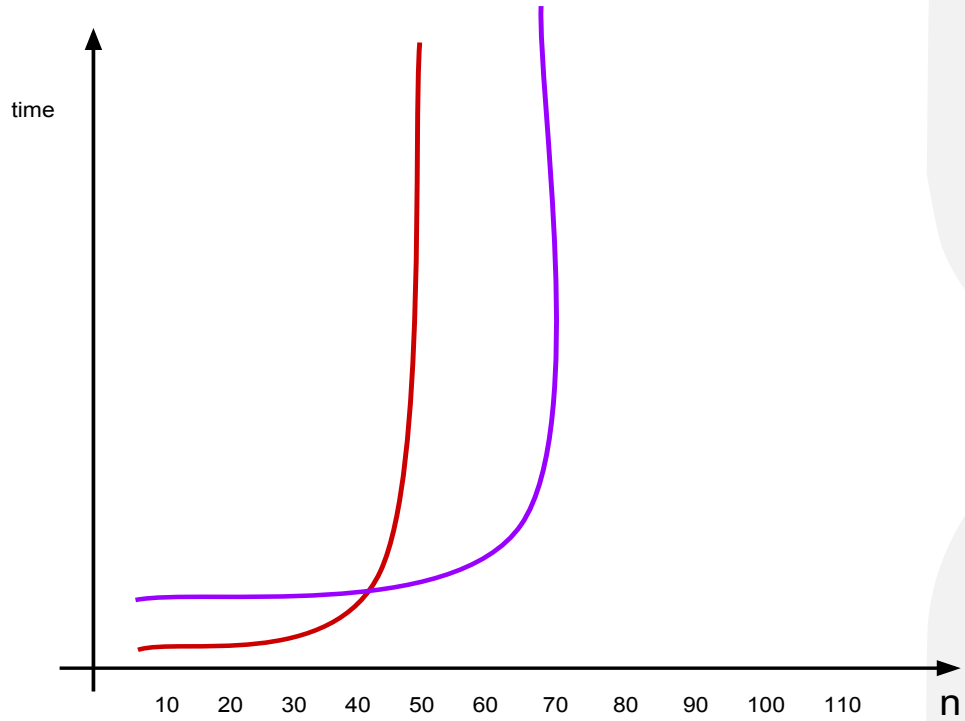


Figura: O limite dos NPs

UDESC

Empurrando o *muro* dos exponenciais

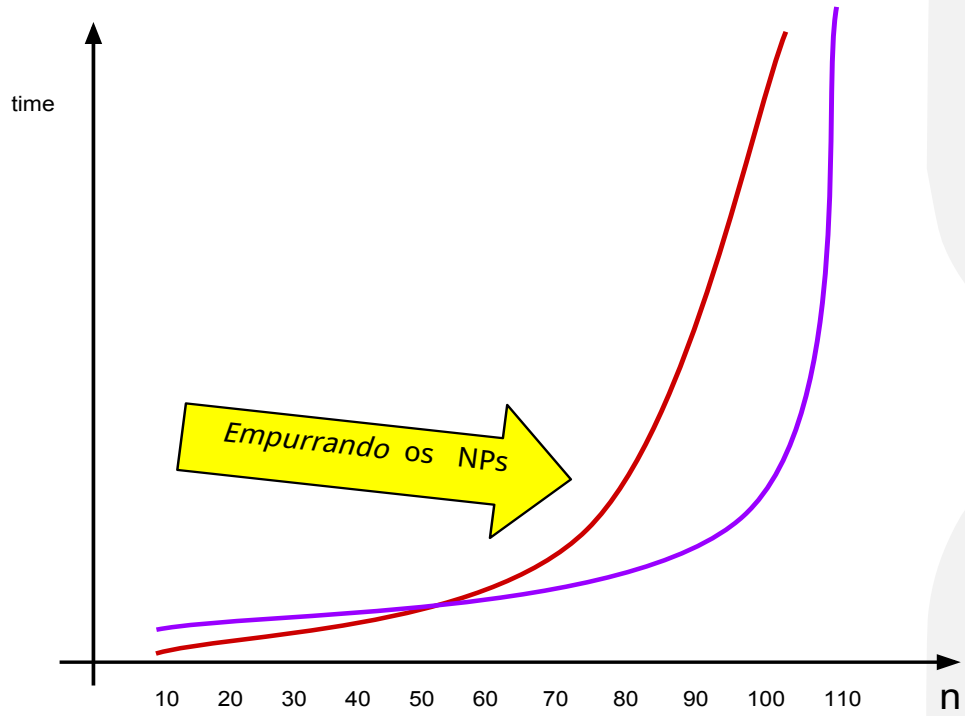


Figura: *Empurrando* o limite dos NPs

Conclusões

- A área computação evolucionária tem apresentado resultados expressivos, são resultados *quase-ótimos*, mas magnetudes acima das demais técnicas;

Conclusões

- A área computação evolucionária tem apresentado resultados expressivos, são resultados *quase-ótimos*, mas magnetudes acima das demais técnicas;
- O hardware com IA embarcada sempre foi um paradigma da construção de uma *inteligência*

Conclusões

- A área computação evolucionária tem apresentado resultados expressivos, são resultados *quase-ótimos*, mas magnetudes acima das demais técnicas;
- O hardware com IA embarcada sempre foi um paradigma da construção de uma *inteligência*
- Os *rápidos*, *baratos* e *velozes*, agora formam um sociedade de agentes inteligentes

Conclusões

- A área computação evolucionária tem apresentado resultados expressivos, são resultados *quase-ótimos*, mas magnetudes acima das demais técnicas;
- O hardware com IA embarcada sempre foi um paradigma da construção de uma *inteligência*
- Os *rápidos*, *baratos* e *velozes*, agora formam um sociedade de agentes inteligentes
- Os problemas solucionados com a *combinatória* tem sido colocados em prática há muitos anos, e ao que parece, devem continuar, ...

Conclusões

- A área computação evolucionária tem apresentado resultados expressivos, são resultados *quase-ótimos*, mas magnetudes acima das demais técnicas;
- O hardware com IA embarcada sempre foi um paradigma da construção de uma *inteligência*
- Os *rápidos*, *baratos* e *velozes*, agora formam um sociedade de agentes inteligentes
- Os problemas solucionados com a *combinatória* tem sido colocados em prática há muitos anos, e ao que parece, devem continuar, ...
- Mas, a nível de Brasil, estamos tímidos!

Perguntas e Referências I

Perguntas e Referências II

- `http://www.joinville.udesc.br/coca/`
- `https://github.com/claudiosa`
- Email: `claudio.sa@udesc.br`
- *Thank you so much*!