

Computação Evolutiva



Eduardo do Valle Simões
Renato Tinós

ICMC - USP

Computação Evolutiva 1

Principais Tópicos

- ≍ Introdução
- ≍ Evolução Natural
- ≍ Computação Evolutiva
- ≍ Algoritmos Genéticos
- ≍ Aplicações
- ≍ Conclusão

Computação Evolutiva 2

Introdução



<http://www.formula-1.com/>

Como otimizar soluções para um processo complexo com um grande número de variáveis?

Computação Evolutiva 3

Evolução natural

- ≍ A evolução natural pode ser vista como um processo de otimização no qual:
 - ≍ Indivíduos e populações competem entre si por recursos
 - ≍ Alimento
 - ≍ Água
 - ≍ Abrigo

Computação Evolutiva 4

Evolução natural

- ≍ (continuação)
 - ≍ Indivíduos mais bem sucedidos na sobrevivência e atração de um parceiro terão, relativamente, mais descendentes (espalham seus genes)
 - ≍ Indivíduos mal sucedidos geram poucos ou nenhum descendente

Computação Evolutiva 5

Computação Evolutiva

- ≍ Introdução
 - ≍ Sistemas para a resolução de problemas que utilizam modelos computacionais baseados na teoria da evolução natural
 - ≍ Pesquisas tiveram início na década de 50

Computação Evolutiva 6

Algoritmos Genéticos (AGs)

- ⌘ Métodos adaptativos que podem ser utilizados para resolver problemas de busca e otimização
- ⌘ São baseados nos processos genéticos de organismos biológicos
 - ⌘ Populações de soluções evoluem, ao longo das gerações, de acordo com os princípios de seleção natural

Computação Evolutiva 7

Algoritmos Genéticos

- ⌘ Desenvolvido por John Holland e sua equipe (popularizado por David Goldberg)
- ⌘ Objetivo:
 - ⌘ Desenvolver sistemas artificiais baseados nos mecanismos dos sistemas naturais

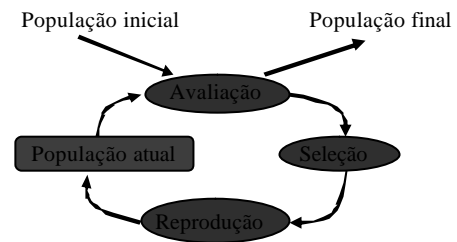
Computação Evolutiva 8

Algoritmos Genéticos

- ⌘ Podem “evoluir” soluções para problemas do mundo real
 - ⌘ Problemas devem ser adequadamente codificados
 - ⌘ Deve haver uma forma de avaliar as soluções apresentadas

Computação Evolutiva 9

Algoritmos Genéticos



Computação Evolutiva 10

Algoritmos Genéticos

- ⌘ Utilizam uma população de soluções candidatas (indivíduos)
- ⌘ Otimização ocorre em várias gerações
 - ⌘ A cada geração
 - ⌘ Mecanismos de seleção selecionam os indivíduos mais aptos
 - ⌘ Operadores de reprodução geram novos indivíduos

Computação Evolutiva 11

Algoritmos Genéticos

- ⌘ Cada indivíduo representa uma possível solução para um dado problema
- ⌘ A cada indivíduo é associado um escore de aptidão, que mede o quão boa é a solução que ele representa
- ⌘ Indivíduos mais aptos têm mais oportunidades de serem reproduzidos

Computação Evolutiva 12

Princípios básicos

- ⌘ Indivíduo
- ⌘ Codificação
- ⌘ Função de aptidão
- ⌘ Reprodução

Computação Evolutiva 13

Indivíduo

- ⌘ Possível solução para um dado problema
 - ⌘ Também chamado de cromossomo ou string
- ⌘ Codificado como vetor de características
- ⌘ População
 - ⌘ Conjunto de indivíduos

Computação Evolutiva 14

Codificação

- ⌘ Cada indivíduo é codificado por um conjunto de parâmetros (genes)
 - ⌘ Genes podem assumir valores:
 - ⌘ Binários (0; 1)
 - ⌘ Inteiros (-2; -1; 0; 1; 2; 3...)
 - ⌘ Reais (-2,33; 0; 3,45; 2,5 x 10⁴)
- ⌘ Parâmetros são combinados para formar strings ou vetores (cromossomos)
 - ⌘ Exemplo:

$$X_i = [2 \ 1 \ 8 \ 0 \ -2 \ -4 \ 1]$$

Computação Evolutiva 15

Codificação

- ⌘ Genótipo
 - ⌘ Conjunto de parâmetros representado por um cromossomo
- ⌘ Fenótipo
 - ⌘ Produto da interação de todos os genes

Computação Evolutiva 16

Função de aptidão

- ⌘ Mede o grau de aptidão de um indivíduo
 - ⌘ Aptidão = probabilidade do indivíduo sobreviver para a próxima geração
- ⌘ Ex. projeto de ponte
 - ⌘ Menor Custo
 - ⌘ Menor tempo de construção
 - ⌘ Maior capacidade de carga

Computação Evolutiva 17

Função de aptidão

- ⌘ É aplicada ao fenótipo do indivíduo
 - ⌘ O genótipo precisa ser decodificado, recuperando o fenótipo associado
- ⌘ Cada aplicação tem sua própria função de aptidão

Computação Evolutiva 18

Reprodução

- ⌘ Permite obtenção de novos indivíduos
- ⌘ Utiliza operadores genéticos
 - ⌘ Transformam a população
 - ⌘ Crossover (cruzamento ou recombinação)
 - ⌘ Mutação

Computação Evolutiva 19

Crossover

- ⌘ Recombinação de características dos pais durante a reprodução
 - ⌘ Permite que as próximas gerações herdem essas características
- ⌘ Funcionamento
 - ⌘ Escolhe dois indivíduos e troca trechos dos cromossomos entre eles
 - ⌘ Exploração rápida do espaço de busca

Computação Evolutiva 20

Crossover

- ⌘ Diversas variações
 - ⌘ Um ponto
 - ⌘ Mais comum
 - ⌘ Dois pontos
 - ⌘ Multi-pontos
 - ⌘ Uniforme

Computação Evolutiva 21

Crossover 1 ponto

Ponto de crossover

Pai 1 Pai 2

0100011 Pais 0010101

Filho A Filho B

0100101 Filhos 0010011

Computação Evolutiva 22

Crossover de 2 pontos

Pai 1 Pai 2

0100011 Pais 0010101

Filho A Filho B

0100101 Filhos 0010011

Computação Evolutiva 23

Crossover uniforme

Mascara: 0 1 0 1 0 0 0

Pai 1 Pai 2

0100011 Pais 0010101

Filho A Filho B

0110101 Filhos 0000011

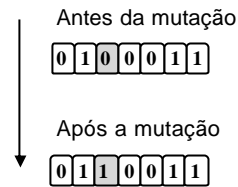
Computação Evolutiva 24

Mutação

- ⌘ Introdução e manutenção da diversidade genética
 - ⌘ Aplicado a cada indivíduo após crossover
- ⌘ Altera aleatoriamente um ou mais genes no cromossomo
- ⌘ Assegura que a probabilidade de atingir qualquer ponto do espaço de busca nunca será zero
- ⌘ Taxa de mutação pequena $P_m \approx 0.001$

Computação Evolutiva 25

Mutação



Computação Evolutiva 26

Seleção

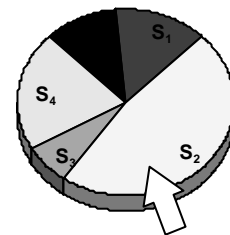
- ⌘ Escolhe preferencialmente, embora não exclusivamente, indivíduos com maiores notas de aptidão
 - ⌘ Procura manter a diversidade da população
- ⌘ Indivíduos mais aptos têm mais oportunidades de serem reproduzidos

Computação Evolutiva 27

Seleção pela roleta

Método da Roleta baseado em Aptidão Relativa

Indivíduo	Aptidão	Aptidão
S_i	$f(S_i)$	Relativa
S_1	10110	2.23
S_2	11000	7.27
S_3	11110	1.05
S_4	01001	3.35
S_5	00110	1.69



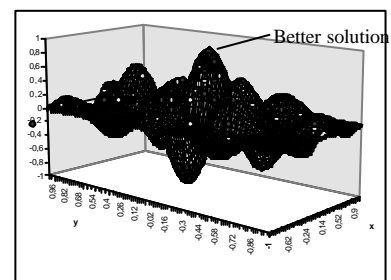
Computação Evolutiva 28

Elitismo

- ⌘ Indivíduo de maior desempenho é automaticamente selecionado
- ⌘ Evita modificações deste indivíduo pelos operadores genéticos
 - ⌘ Utilizado para que os melhores indivíduos não desapareçam da população pela manipulação dos operadores genéticos

Computação Evolutiva 29

Espaço de Busca



Computação Evolutiva 30

Observações

- ⌘ Se o AG estiver corretamente implementado, a população deve evoluir em gerações sucessivas
- ⌘ Aptidão do melhor indivíduo e da média da população devem aumentar em direção a um ótimo global
- ⌘ Importância da codificação na convergência

Computação Evolutiva 31

Critério de parada

- ⌘ Tempo de execução
- ⌘ Número de gerações
- ⌘ Valor de aptidão mínimo e/ou médio
- ⌘ Convergência
 - ⌘ Nas últimas k iterações não houve melhora nas aptidões

Computação Evolutiva 32

Escolha de parâmetros

- ⌘ Escolhidos de acordo com o problema
 - ⌘ Quantos cromossomos em uma população
 - ⌘ Poucos ? Efeito pequeno do *crossover*
 - ⌘ Muitos ? Aumenta tempo de computação
 - ⌘ Taxa de mutação
 - ⌘ Baixa ? Mudanças lentas
 - ⌘ Alta ? Traços desejados não são mantidos (caos)

Computação Evolutiva 33

Escolha de parâmetros

- ⌘ Outros parâmetros
 - ⌘ Quantos indivíduos selecionados para reprodução?
 - ⌘ Quantos pontos de *crossover*?
 - ⌘ Critério para medir aptidão?
- ⌘ Manter limites no tamanho da população e complexidade da análise
 - ⌘ Algoritmo pode se tornar ineficiente

Computação Evolutiva 34

Aplicações

- ⌘ Otimização de função numérica
- ⌘ Otimização combinatória
 - ⌘ Determinação de Árvores Filogenéticas
- ⌘ Projetos
 - ⌘ Projeto de pontes
- ⌘ Aprendizado de Máquina
 - ⌘ Determinação dos parâmetros de Redes Neurais Artificiais em problemas de Bioinformática

Computação Evolutiva 35

Exemplo1: preparo de biscoitos

- ⌘ Otimizar quantidade de açúcar e farinha de trigo para preparar biscoitos
- ⌘ Passos
 - ⌘ Criar população inicial
 - ⌘ Codificar strings ou cromossomos
 - ⌘ Definir função de aptidão
 - ⌘ Reprodução

Computação Evolutiva 36

Exemplo1: preparo de biscoitos

Qualidade do biscoito (q):

açúcar (kg)

farinha (kg)

	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	4	3	2	1
2	2	3	4	5	6	5	4	3	2
3	3	4	5	6	7	6	5	4	3
4	4	5	6	7	8	7	6	5	4
5	5	6	7	8	9	8	7	6	5
6	4	5	6	7	8	7	6	5	4
7	3	4	5	6	7	6	5	4	3
8	2	3	4	5	6	5	4	3	2
9	1	2	3	4	5	4	3	2	1

Comunicação Evolutiva 37

Exemplo1: preparo de biscoitos

Codificação do cromossomo

Quantidade de farinha de trigo e de açúcar

açúcar (kg)

farinha (kg)

2 4

Comunicação Evolutiva 38

Exemplo1: preparo de biscoitos

Função de aptidão:

$$f_i = \frac{q_i}{q_j}$$

CROMOSSOMO	GRAU	APT.	PADRÃO
1 4	4	0.4	
3 1	3	0.3	
1 2	2	0.2	
1 1	1	0.1	

Comunicação Evolutiva 39

Exemplo1: preparo de biscoitos

Mutação:

Pai: 2 5

±1

Filho: 3 5

Comunicação Evolutiva 40

Exemplo1: preparo de biscoitos

Regras:

- Cada cromossomo pode aparecer somente uma vez
- Tamanho máximo da população: 4
- Nova população: melhor indivíduo (elitismo) + indivíduos restantes escolhidos aleatoriamente

Comunicação Evolutiva 41

Exemplo1: preparo de biscoitos

Caso 1 (sem crossover)

Geracao	Cromossomo	Qualidade	Filho	Qualidade
0	[1 1]	1	[1 2]	2
1	[1 2]	2	[1 3]	3
	[1 1]	1	[1 2]	2
2	[1 3]	3	[1 4]	4
	[1 2]	2	[2 2]	3
	[1 1]	1	[2 1]	2
3	[1 4]	4	[2 4]	5
	[1 3]	3	[2 3]	4
	[1 2]	2	[1 3]	3
	[2 1]	2	[3 1]	3

Comunicação Evolutiva 42

Exemplo1: preparo de biscoitos Caso 1 (sem crossover)

• Geração 4:

[2 4]	5	—	(2 5)	6
[1 4]	4	—	(1 5)	5
[1 3]	3	—	(2 3)	4
[2 1]	2	—	(2 2)	3

• Geração 5:

(2 5)	6	—	(3 5)	7
(1 5)	5	—	(1 4)	6
(2 3)	4	—	(2 2)	3
[2 2]	3	—	(3 2)	4

• Geração 8:

[5 5]	9
[4 5]	8
[2 5]	6
[2 1]	2

Computação Evolutiva 43

Exemplo1: preparo de biscoitos Caso 1 (sem crossover)

≅ Qualidade do biscoito (q):

		farinha (kg)								
		1	2	3	4	5	6	7	8	9
açúcar (kg)	1	1	2	3	4	5	4	3	2	1
	2	2	3	4	5	6	5	4	3	2
	3	3	4	5	6	7	6	5	4	3
	4	4	5	6	7	8	7	6	5	4
	5	5	6	7	8	9	8	7	6	5
	6	4	5	6	7	8	7	6	5	4
	7	3	4	5	6	7	6	5	4	3
	8	2	3	4	5	6	5	4	3	2
	9	1	2	3	4	5	4	3	2	1

Computação Evolutiva 44

Exemplo1: preparo de biscoitos

≅ Qualidade do biscoito (q):

		farinha (kg)								
		1	2	3	4	5	6	7	8	9
açúcar (kg)	1	1	2	3	4	5	4	3	2	1
	2	2	0	0	0	0	0	0	0	2
	3	3	0	0	0	0	0	0	0	3
	4	4	0	0	7	8	7	0	0	4
	5	5	0	0	8	9	8	0	0	5
	6	4	0	0	7	8	7	0	0	4
	7	3	0	0	0	0	0	0	0	3
	8	2	0	0	0	0	0	0	0	2
	9	1	2	3	4	5	4	3	2	1

Computação Evolutiva 45

Exemplo1: preparo de biscoitos

• Crossover:

Pai 1:	2	5
Pai 2:	1	4
Filho 1:	1	5
Filho 2:	2	4

Computação Evolutiva 46

Exemplo1: preparo de biscoitos Caso 2 (com crossover)

	Cromossomo	Qualidade	Filho	Qualidade
• Geração 0:	(1 1)	1	(1 2)	2
• Geração 1:	(2 1)	2	(3 1)	3
	(1 1)	1	(1 2)	
			(2 1)	
			(1 1)	
• Geração 7:	[5 5]	9		
	[1 4]	4		
	[3 1]	2		
	[5 2]	0		

Computação Evolutiva 47

Exemplo1: preparo de biscoitos Caso 2 (com crossover)

≅ Qualidade do biscoito (q):

		farinha (kg)								
		1	2	3	4	5	6	7	8	9
açúcar (kg)	1	1	2	3	4	5	4	3	2	1
	2	2	0	0	0	0	0	0	0	2
	3	3	0	0	0	0	0	0	0	3
	4	4	0	0	7	8	7	0	0	4
	5	5	0	0	8	9	8	0	0	5
	6	4	0	0	7	8	7	0	0	4
	7	3	0	0	0	0	0	0	0	3
	8	2	0	0	0	0	0	0	0	2
	9	1	2	3	4	5	4	3	2	1

Computação Evolutiva 48

Exemplo2: Robótica Evolutiva

≅ Sistemas Evolutivos

≅ Teste de um critério definido e interrupção do processo quando um desempenho aceitável é produzido.

Computação Evolutiva 49



Exemplo2: Robótica Evolutiva

≅ Configuração da Rede Neural

≅ Velocidade de movimento

≅ Seleção dos Sensores

Computação Evolutiva 51

Exemplo2: Robótica Evolutiva

Arquitetura do Robô

Computação Evolutiva 52

Exemplo2: Robótica Evolutiva

≅ Objetivo: Navegação sem Colisões

Computação Evolutiva 53

Exemplo2: Robótica Evolutiva

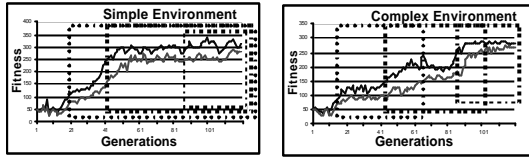
≅ 120 Gerações: (1 min.)

≅ Pontuação do Melhor Robô

≅ Média da População

Computação Evolutiva 54

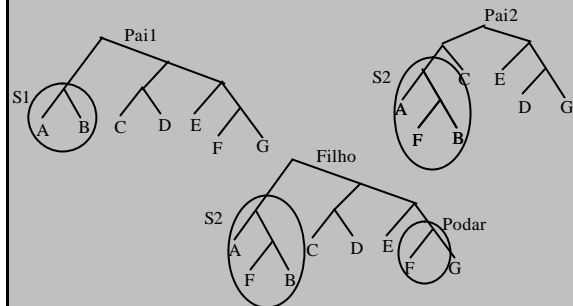
Exemplo2: Robótica Evolutiva



- "Espécie" 1 – Um sensor frontal
- "Espécie" 2 – Dois sensores, um frontal e outro lateral
- "Espécie" 3 – Três sensores, um frontal e dois laterais

Computação Evolutiva 55

Exemplo3: Árvore Filogenética



Conclusão

- ≈ Conceitos básicos
- ≈ Evolução Natural
- ≈ Algoritmos genéticos
 - ≈ Codificação
 - ≈ Função de aptidão
 - ≈ Operadores Genéticos
 - ≈ Reprodução

Computação Evolutiva 57

FIM

Cópia das transparências e referências bibliográficas podem ser obtidas no site:

<http://www.icmc.usp.br/~simoes/seminars/icobicobi/>

email: Eduardo Simões – simoes@icmc.usp.br
Renato Tinós – rtinos@icmc.usp.br

Computação Evolutiva 58