

COMPUTAÇÃO EVOLUCIONÁRIA ALGORITMOS GENÉTICOS (2)

Cristiano Leite de Castro

crislcastro@ufmg.br

Departamento de Engenharia Elétrica
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil

Componentes de um AG

1. Representação;
2. Operadores de Variação:
 1. Recombinação e Mutação;
3. **Modelos de População;**
4. Mecanismos de Seleção:
 1. Seleção dos Pais
 2. Seleção dos Sobreviventes (substituição);

Modelos de População

- seja μ = tamanho da população; λ = número de descendentes;
- **Modelo Geracional:** ($\mu = \lambda$)
 - cada indivíduo vive por exatamente uma geração;
 - todos os pais são substituídos por seus descendentes.
- **Modelo *Steady-State*:** ($\mu > \lambda$)
 - apenas uma parte dos pais é substituída pelos λ descendentes;
 - no caso extremo, tem-se apenas 1 filho por geração e assim, 1 membro da população é substituído;
 - *Generation Gap*:
 - proporção da população a ser substituída;
 - 1.0 para GGA e λ/μ para SSGA.

Componentes de um AG

1. Representação;
2. Operadores de Variação:
 1. Recombinação e Mutação;
3. Modelos de População;
4. **Mecanismos de Seleção:**
 1. **Seleção dos Pais**
 2. Seleção dos Sobreviventes (substituição);

Seleção dos Pais

- os operadores de seleção funcionam com base somente nas aptidões (*fitness*) individuais das soluções candidatas;
 - isto é, eles são independentes da **forma de representação** adotada para as soluções;
- é importante fazer a distinção entre:
 - **operador**: responsável por definir as probs. de seleção (PS_i);
 - **algoritmo**: responsável por definir como é feita a amostragem a partir das probs. de seleção.;

- Valor esperado para o número de cópias selecionadas do i-ésimo indivíduo:

$$E(n_i) = \lambda \times PS_i$$

(λ = número de pais a serem selecionados; PS_i = prob. de seleção do i-ésimo)

- **Operador de Seleção Proporcional ao *Fitness* (SPF):**

- a probabilidade de seleção PS_i para a i -ésimo indivíduo é

$$PS_i = \frac{f_i}{\sum_{j=1}^{\mu} f_j}$$

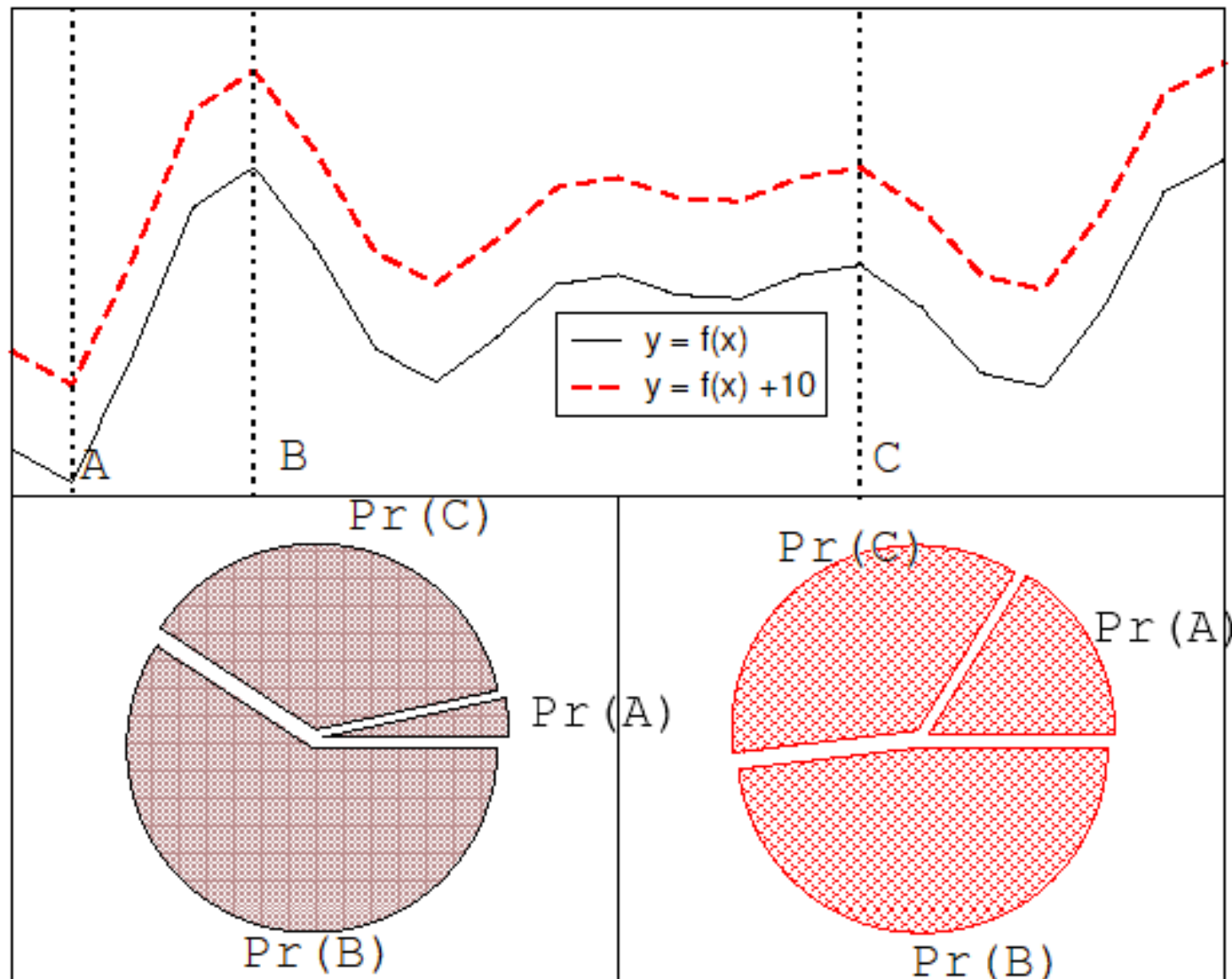
- **convenção:**
 - assume-se maximização da função de *fitness* (função objetivo);
 - assume-se *fitness* não-negativo.

• Problemas do SPF:

1. **convergência prematura:** “super-indivíduos” podem tomar conta da população rapidamente.
2. **perda de pressão na seleção:** qdo os valores de *fitness* são muito próximos, a seleção tende a tornar-se aleatória a partir de uma distribuição uniforme;
3. **susceptibilidade a versões deslocadas da função *fitness*.**

• Escalonamento é comumente usado p/ corrigir os itens 2 e 3:

- **Ex: Janelamento (*Windowing*):** $f'_i = f_i - \beta$
 - onde f_i é o valor do *fitness* para o i -ésimo indivíduo;
 - e β é o pior *fitness* da geração corrente;



Exemplo: susceptibilidade do operador SPF ao deslocamento da função de aptidão.

- **Operador baseado em *Ranking*:**

- tenta remover os problemas do operador SPF por obter as probs. de seleção a partir de **valores relativos** (não-absolutos) de *fitness*.
- ranqueia a população de acordo com o *fitness* e calcula as probs. com base nesse *ranking*, o qual geralmente varia entre:
 - $\mu - 1$ (melhor indivíduo) e 0 (pior indivíduo).
- impõe um *overhead* de ordenação. Porém, esta operação é usualmente simples quando comparada ao tempo gasto para avaliação do *fitness*.

Ranking Linear

$$P_{lin-rank}(i) = \frac{(2-s)}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

- parametrizado pelo fator s : $1.0 < s \leq 2.0$
 - s determina a inclinação da reta;
 - em GGAs (geracionais) s representa o número de cópias a serem selecionadas do indivíduo de melhor *fitness*.
 - $s_{max} = 2$, para que na média o indivíduo de *rank* mediano seja capaz de ser selecionado pelo menos uma vez.

- Exemplo:

	Fitness	rank	SPF	RL(s=2)	RL(s=1.5)	RL(s=1.1)
A	1	0	0.1	0	0.167	0.3
B	5	2	0.5	0.67	0.5	0.37
C	4	1	0.4	0.33	0.33	0.33
soma	10		1.0	1.0	1.0	1.0

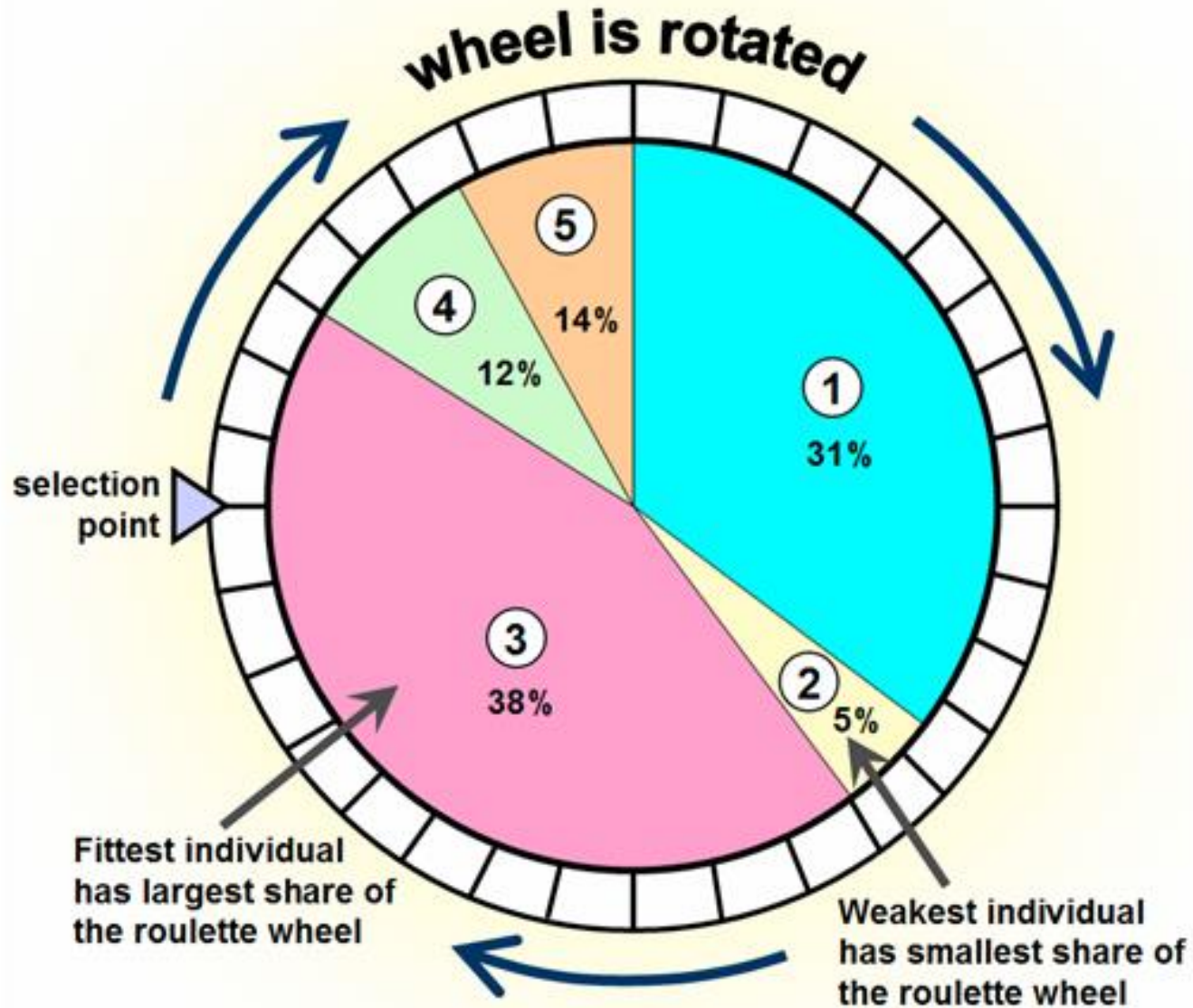
Ranking Exponencial

$$Pre_i = \frac{1 - e^i}{c}$$

$$c = \sum_{j=1}^{\mu} Pre_i$$

- Conforme visto, a pressão imposta pelo *ranking linear* na seleção dos pais é limitada por $s_{max} = 2$;
- *Ranking exponencial* pode ser usado p/ se obter mais de 2 cópias p/ o indivíduo de melhor *fitness*;
- fator de normalização c é função do tamanho da população (μ).
 - ele garante que a soma das probabilidades seja igual a 1.

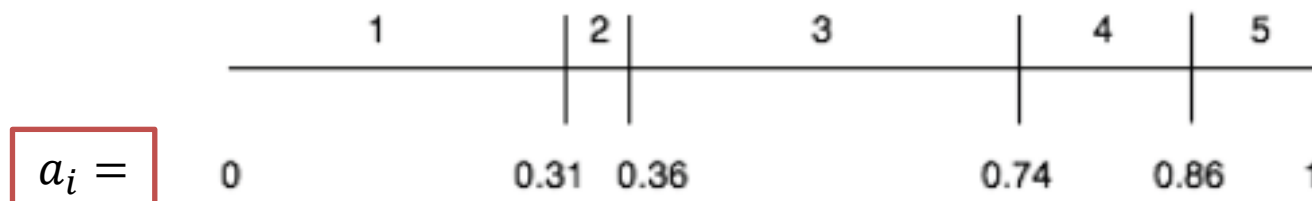
- Foi mostrado que $E(n_i) = \lambda \times PS_i$;
- Entretanto na prática, como λ é pequeno, o número de cópias selecionadas de cada indivíduo i geralmente não corresponde ao valor esperado $E(n_i)$;
- métodos comumente usados para amostragem de indivíduos a partir das probs. de seleção:
 1. **Roleta;**
 2. **Amostragem Universal Estocástica (SUS);**



- indivíduos são mapeados para segmentos de reta contíguos no intervalo $[0,1]$;
- o tamanho do segmento de cada indivíduo é proporcional a sua **prob. de seleção**;
- Obtenha $a_0 = 0$, $a_i = PS_i + \sum_{k=1}^{i-1} PS_k$ p/ $i = 1 \dots \mu$
- um número aleatório r é sorteado e o indivíduo cujo segmento contém r é então selecionado para ser um pai;
 - esse processo é análogo a girar uma **Roleta** com um único ponto de seleção.
- roda-se a Roleta λ vezes para selecionar λ indivíduos;

Exemplo:

No.	chromosome	fitness	fraction of total
1	0100010001	6.82	0.31
2	1100101001	1.11	0.05
3	1100111001	8.48	0.38
4	0101011111	2.57	0.12
5	1100100100	3.08	0.14
Totals:		22.05	1.00



```
BEGIN
  set current_member = 1;
  WHILE ( current_member  $\leq$   $\lambda$  ) DO
    Pick a random value r uniformly from [0,1];
    set i = 1;
    WHILE ( ai < r ) DO
      set i = i + 1;
    OD
    set mating_pool[current_member] = parents[i];
    set current_member = current_member + 1;
  OD
END
```

Fig. 3.20. Pseudocode for the roulette wheel algorithm

Amostragem Universal Estocástica (SUS)

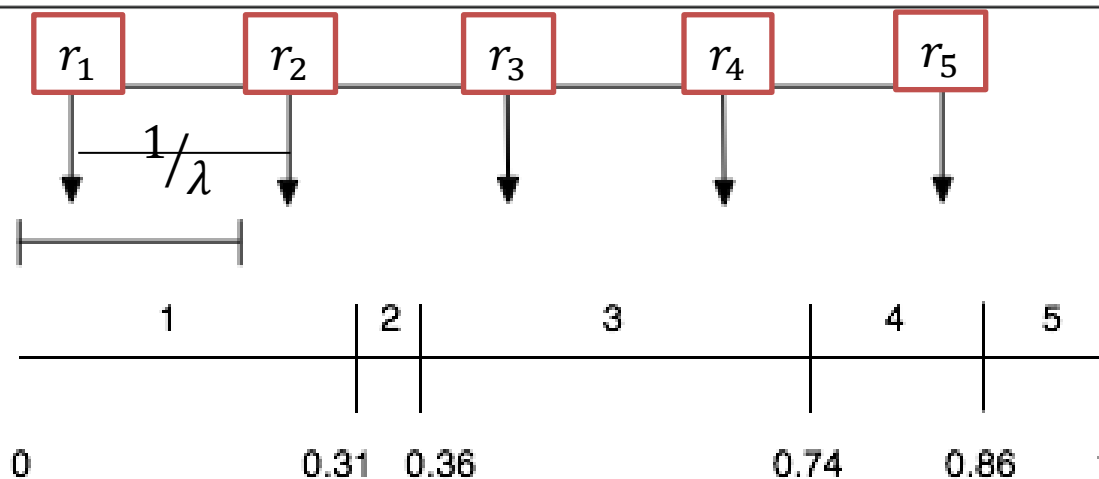
- indivíduos são mapeados para segmentos de reta contíguos no intervalo $[0,1]$;
- o tamanho do segmento de cada indivíduo é proporcional a sua **prob. de seleção**;
- obtenha $a_0 = 0$, $a_i = PS_i + \sum_{k=1}^{i-1} PS_k$ p/ $i = 1 \dots \mu$
- crie N marcas igualmente espaçadas no intervalo $[0,1]$, tal que a primeira marca (ponto de seleção) corresponde a um número aleatório $r_1 \in [0, 1/\lambda]$. As demais marcas são obtidas, tal que :

$$r_i = r_{i-1} + \frac{1}{\lambda}, \text{ para } i = 2 \dots \lambda$$

- selecione λ indivíduos de acordo com as posições das λ marcas.
- Isso é equivalente a rodar a **Roleta** uma única vez com pontos de seleção igualmente espaçados.

Exemplo:

No.	chromosome	fitness	fraction of total
1	0100010001	6.82	0.31
2	1100101001	1.11	0.05
3	1100111001	8.48	0.38
4	0101011111	2.57	0.12
5	1100100100	3.08	0.14
Totals:		22.05	1.00



$a_i =$

```
BEGIN
  set current_member = i = 1;
  Pick a random value r uniformly from  $[0, 1/\mu]$ ;
  WHILE ( current_member  $\leq \lambda$  ) DO
    WHILE ( r  $\leq a[i]$  ) DO
      set mating_pool[current_member] = parents[i];
      set r = r +  $1/\mu$ ;
      set current_member = current_member + 1;
    OD
    set i = i + 1;
  OD
END
```




Fig. 3.21. Pseudocode for the stochastic universal sampling algorithm

- valor esperado para o número de cópias selecionadas do i -ésimo indivíduo:

$$E(n_i) = \lambda \times PS(i)$$

(λ = number to be selected; $PS(i)$ = prob. de seleção)

- **Roleta:**

- nem sempre garante que n_i cópias serão selecionadas;

- **SUS:**

- possui menor variância;
 - é computacionalmente mais eficiente
 - garante $\text{floor}(E(n_i)) \leq n_i \leq \text{ceil}(E(n_i))$

- **FPS** e **Ranking** confiam em informação global da população (μ e f_i para $i = 1 \dots \mu$):
 - isso pode acarretar problemas em implementações paralelizadas;
 - além disso, eles confiam no conhecimento de uma função de *fitness*, a qual nem sempre é conhecida.
 - **exemplo:** evolução de estratégias em um jogo;
- **Seleção por Torneio:**
 - não requer informação global da população;
 - pode controlar o mecanismo de **pressão na seleção** mais facilmente.

- **Seleção por Torneio:**

- escolha k indivíduos aleatoriamente e selecione o melhor deles;
 - k é parâmetro que determina o tamanho do Torneio:
 - Qto maior k , maior a prob. de se selecionar indivíduos com *fitness* acima da média → maior **pressão na seleção**.
- Torneio pode ser determinístico ou probabilístico:
 - **Determinístico** ($p=1$): a cada torneio, sempre o indivíduo mais apto é selecionado;
- Outras variações:
 - **com substituição** (*with replacement*)
 - **sem substituição** (*without replacement*) → menor variância;

```
BEGIN
  set current_member = 1;
  WHILE ( current_member  $\leq \mu$  ) DO
    Pick k individuals randomly, with or without replacement;
    Select the best of these k comparing their fitness values;
    Denote this individual as i;
    set mating_pool[current_member] = i;
    set current_member = current_member + 1;
  OD
END
```

Fig. 3.22. Pseudocode for the tournament selection algorithm

Componentes de um AG

1. Representação;
2. Operadores de Variação:
 1. Recombinação e Mutação;
3. Modelos de População;
4. **Mecanismos de Seleção:**
 1. Seleção dos Pais
 2. **Seleção dos Sobreviventes (substituição);**

Seleção de Sobreviventes

- Reduz da população atual de $\mu + \lambda$ indivíduos para μ indivíduos;
- duas principais abordagens:
 - 1. substituição baseada em Idade:**
 - adotada pelo GGA (Geracional);
 - qdo $\lambda < \mu$ (*steady-state*) pode ser implementada como uma fila F.I.F.O. ou como “deleção aleatória”;
 - 2. substituição baseada em *fitness*:**
 - a seguir;

2. substituição baseada em *fitness*:

- todos os λ descendentes são incluídos;
- o *fitness* é usado para decidir quais dos λ pais serão substituídos;
- a decisão pode ser tomada usando-se os métodos FPS, *Ranking* (+ Roleta ou SUS), Torneio

ou através da

- remoção dos “piores” (*genitor*): os λ piores pais são substituídos;
 - pode levar a **convergência prematura**. Por esse motivo, ela é geralmente aplicada a grandes populações ou com a política de não permitir indivíduos duplicados;

- **Leitura Recomendada:**

- Capítulo 3 do Livro:

A.E. EIBEN, J.E. SMITH, Introduction to Evolutionary Computing (Natural Computing Series), Springer.

