UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ALGORITMOS GENÉTICOS

Metaheurísticas de Buscas

ALGORITMOS GENÉTICOS (AG)

Popularizados por **John Holland** podem ser considerados os primeiros modelos algorítmicos desenvolvidos para inspirados nos sistemas genéticos.

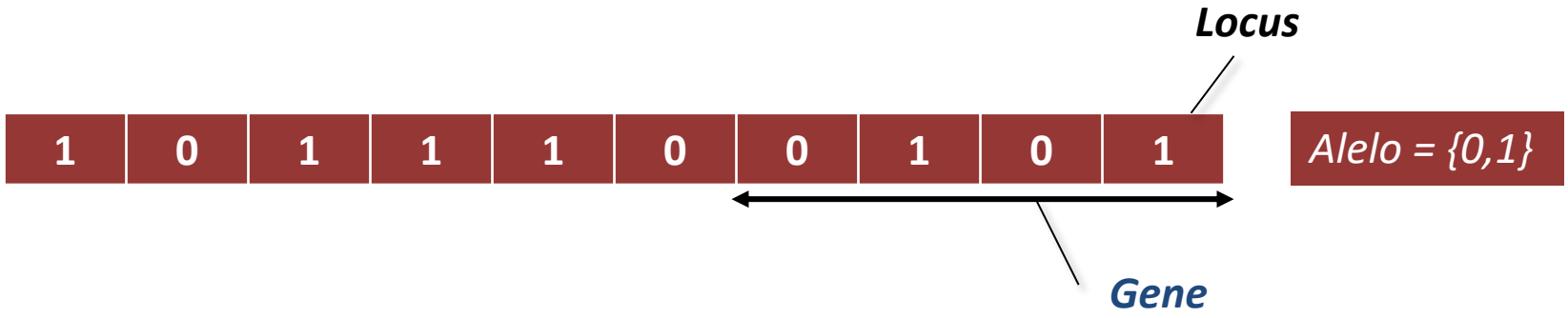
Publicou em 1975 o livro seminal de AG:
Adaptation in Natural and Artificial Systems

ALGORITMOS GENÉTICOS (AG)

- **CROMOSSOMO (ou indivíduo)**: **formado** por genes; normalmente um cromossomo representa uma solução completa a um problema.

- **GENE**: representa uma **faceta/característica** independente das demais.

- **ALELO**: é o **valor** armazenado em um locus.



ALGORITMOS GENÉTICOS (AG)

- **POPULAÇÃO**: conjunto de cromossomos ou de indivíduos.

- **SELEÇÃO**: Sobrevivência do melhor; manter os indivíduos de maior adequabilidade (fitness) da população.

- **REPRODUÇÃO**: realizada pelos operadores abaixo

  Crossover: cruzamento entre pares de cromossomos

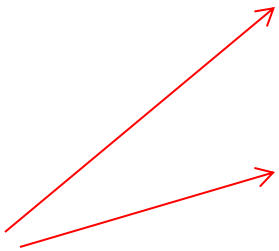
  Mutação: modificação de genes de um cromossomo

AG no contexto de Busca em IA

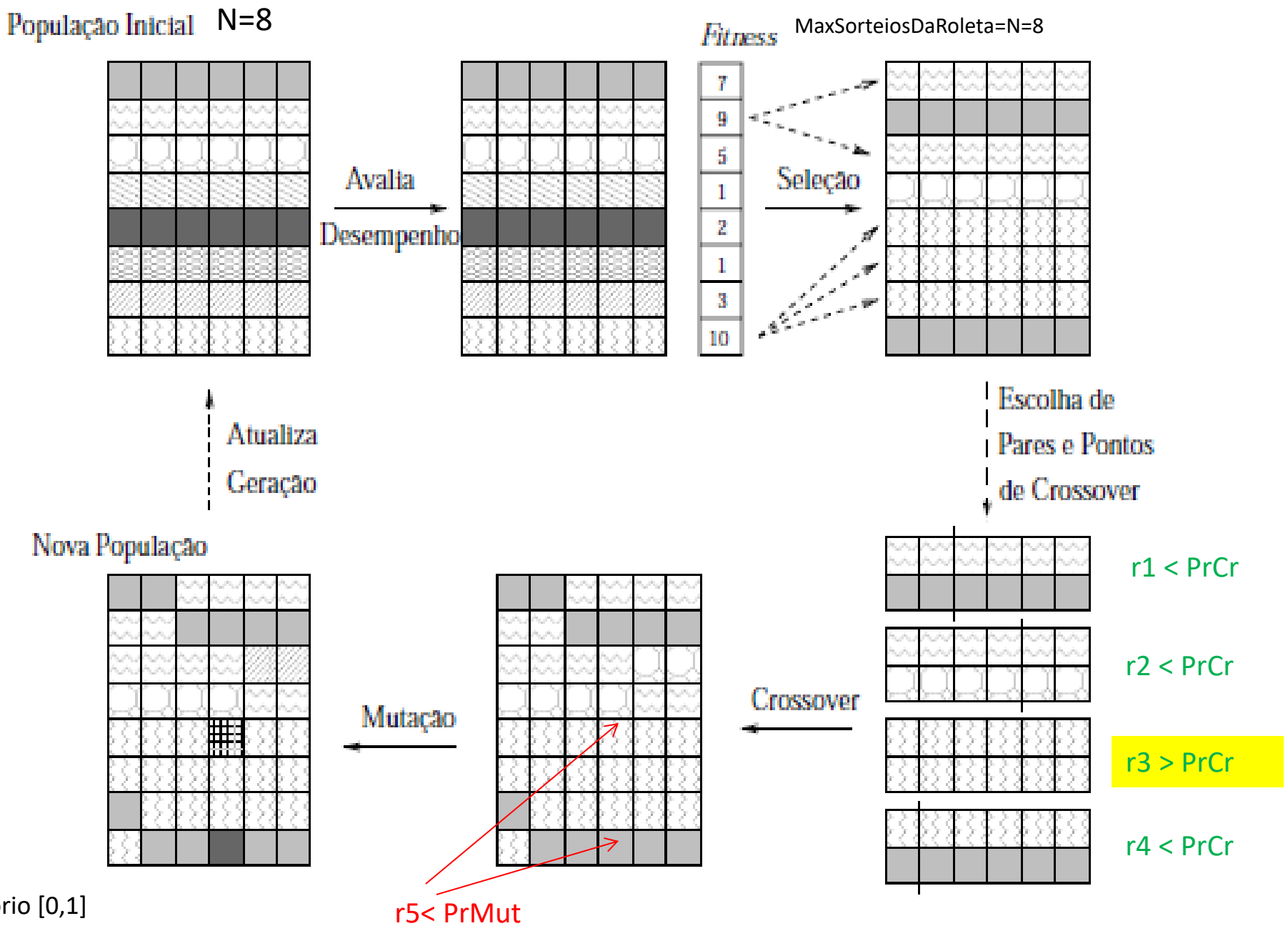
CARACTERÍSTICAS

- **Busca informada**: utiliza função de *fitness* que guia o algoritmo
- **Busca local:** não guarda a informação do caminho para chegar à solução
- **Estados sucessores** gerados pela combinação de dois estados pais
- **Função de adequação ou fitness:** f(n) = h(n)
  f(indivíduo) = função de adequação ou fitness (indivíduo)
- **Útil quando** o espaço de estados é muito grande ou muito complexo para tratamento analítico

ESQUEMA GERAL DE EVOLUÇÃO DE UM ALGORITMO GENÉTICO



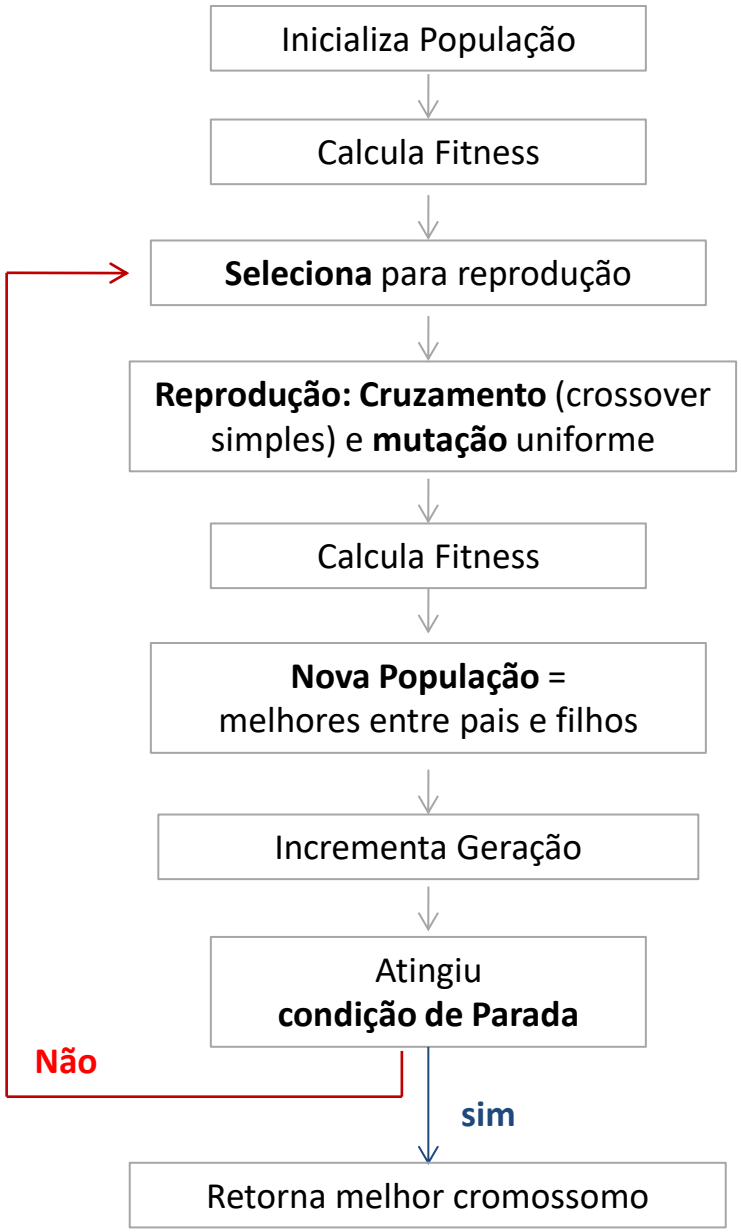
Esquema Geral de um Agoritmo Genético



leatório [0,1]

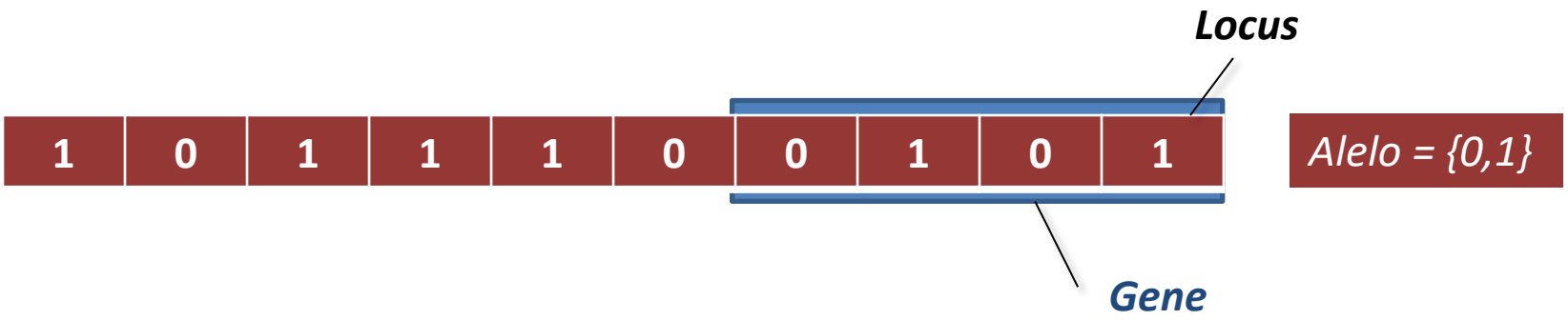
AG Canônico

- Codificação Binária
- **Seleção**
  - Reprodução (método da roleta)
- **Reprodução**
  - Crossover simples (pCROSS)
  - Mutação Uniforme (pMUT)
- **Sobrevivência**
  - Melhores (entre pais e filhos) irão compor a nova população
- **Condição de parada**
  - Geração = MaxGer ou
  - Fitness = máximo atingido ou
  - Estagnação do melhor fitness



AG Canônico: **Codificação binária**

Cromossomo no AG: cadeia de bits de tamanho L



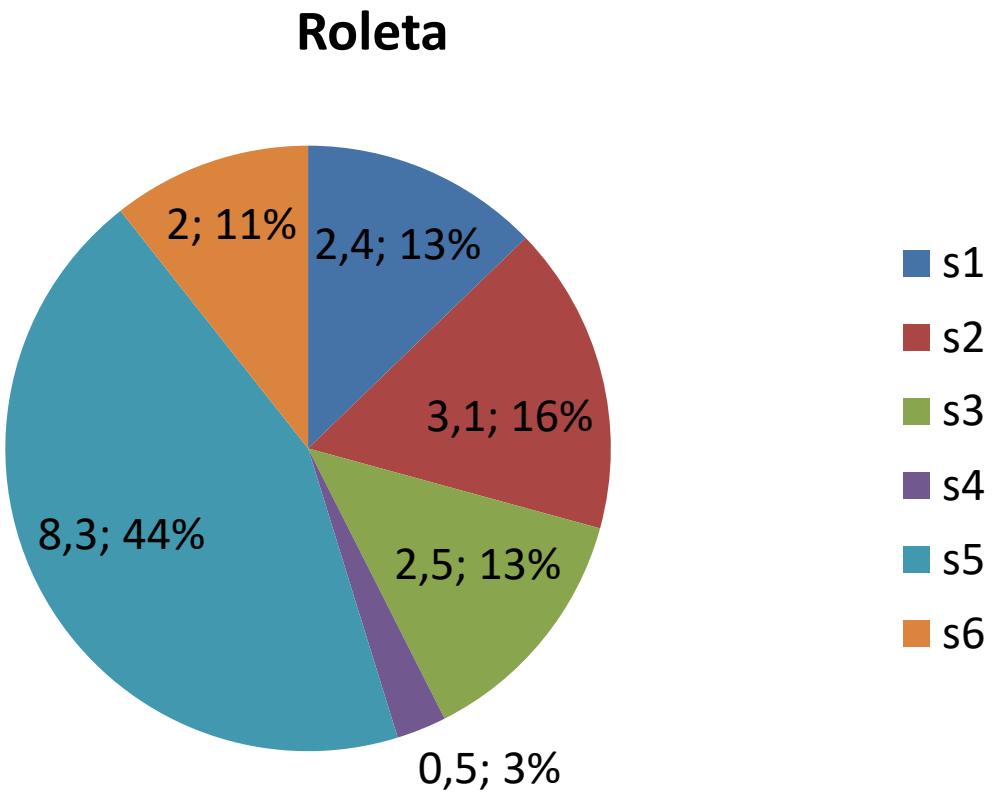
Cada posição (locus) no cromossomo assume um dos dois possíveis alelos, 0 ou 1.

Exemplo: problema da mochila; cada gene indica se um determinado item está ou não na mochila

AG Canônico: **seleção por roleta**

Método de seleção de cromossomos/indivíduos que seleciona K indivíduos , sendo que os de maior probabilidade são os que apresentam maior *fitness*.

Indivíduo	Fitness	Norm.
S1	2,4	13%
S2	3,1	16%
S3	2,5	13%
S4	0,5	3%
S5	8,3	44%
S6	2,0	11%



AG Canônico: **Seleção por roleta**

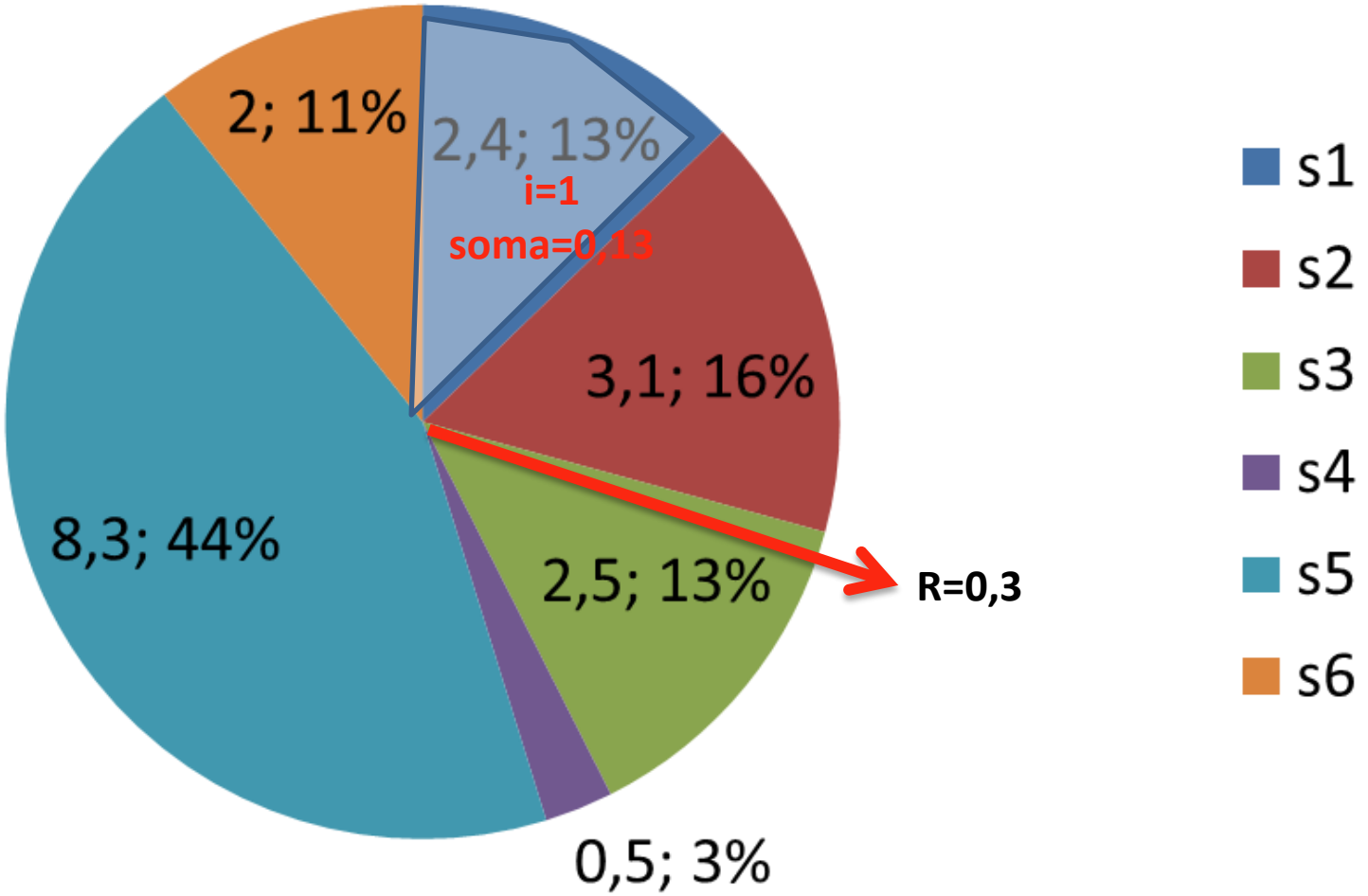
Calcula $p(s_i)$ para i=1,…,N: $$p(s_i) = \frac{fitness(s_i)}{\sum_{k=1}^{N} fitness(s_k)}$$

Algoritmo de seleção por roleta: sortei um indivíduo por chamada

```
i=1;
soma = p(s_i)
Sorteia r ∈ [0, 1]
enquanto soma < r
    i=(i+1)
    soma = soma + p(s_i)
fim enquanto
Retorna s_i
```

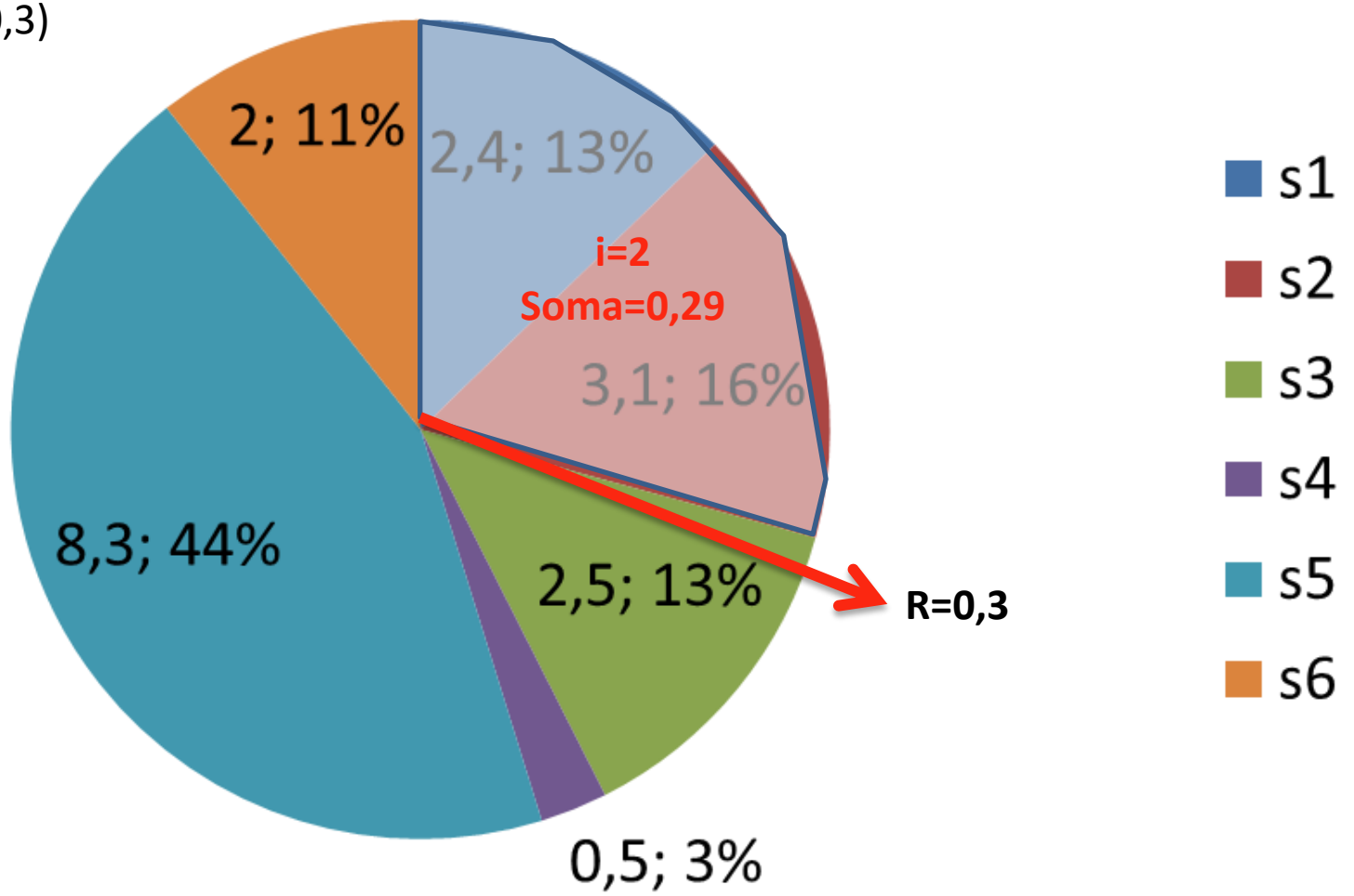
INICIALIZAÇÃO

soma=p(s1)=0,13 < r (0,3)



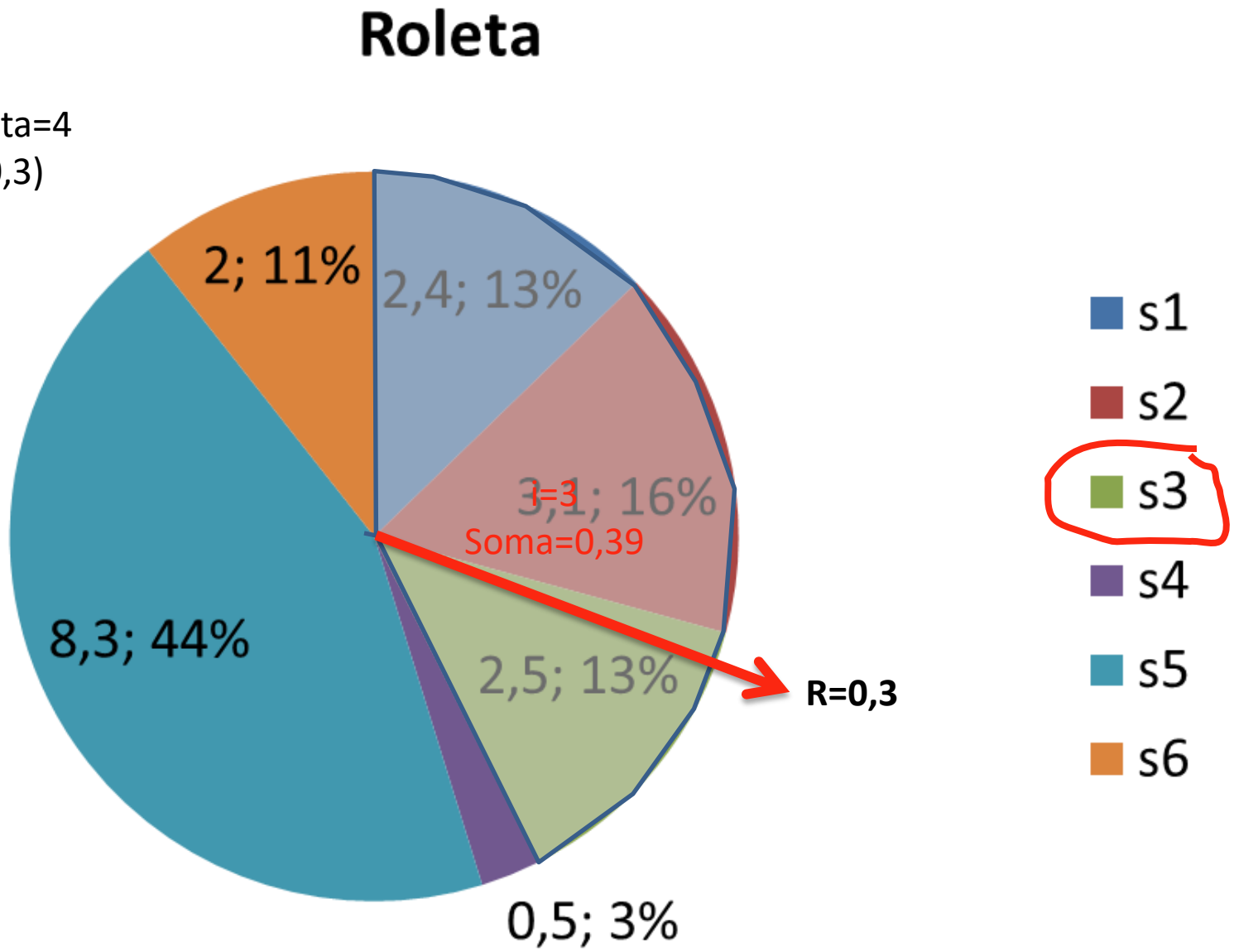
SORTEIO

MaxSorteiosRoleta=4
Soma=0,29 < r (0,3)



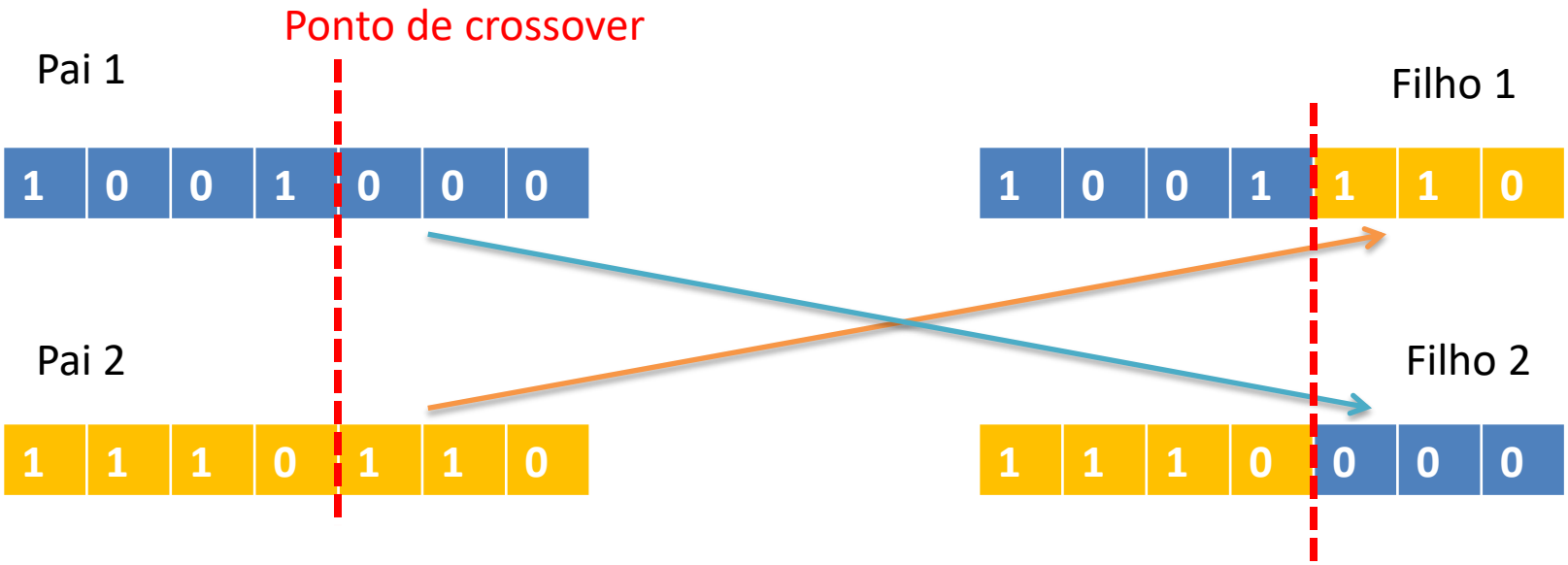
SORTEIO

MaxSorteiosRoleta=4
Soma=0,39 > r (0,3)
Sorteado=[s3]

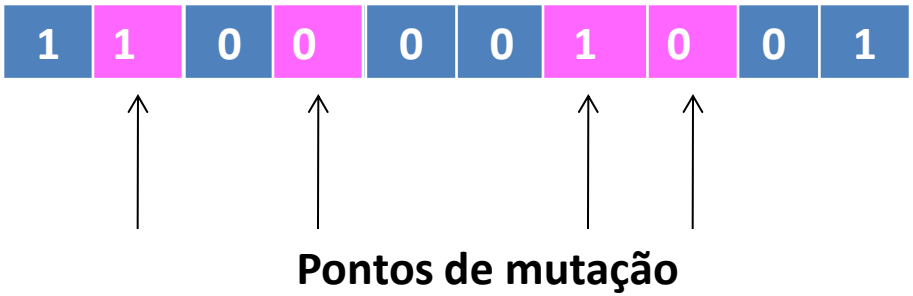


AG Canônico: **Reprodução e crossover**

Crossover Simples (1 ponto): posição de cruzamento escolhida aleatoriamente.



AG Canônico: **Reprodução e Mutação**

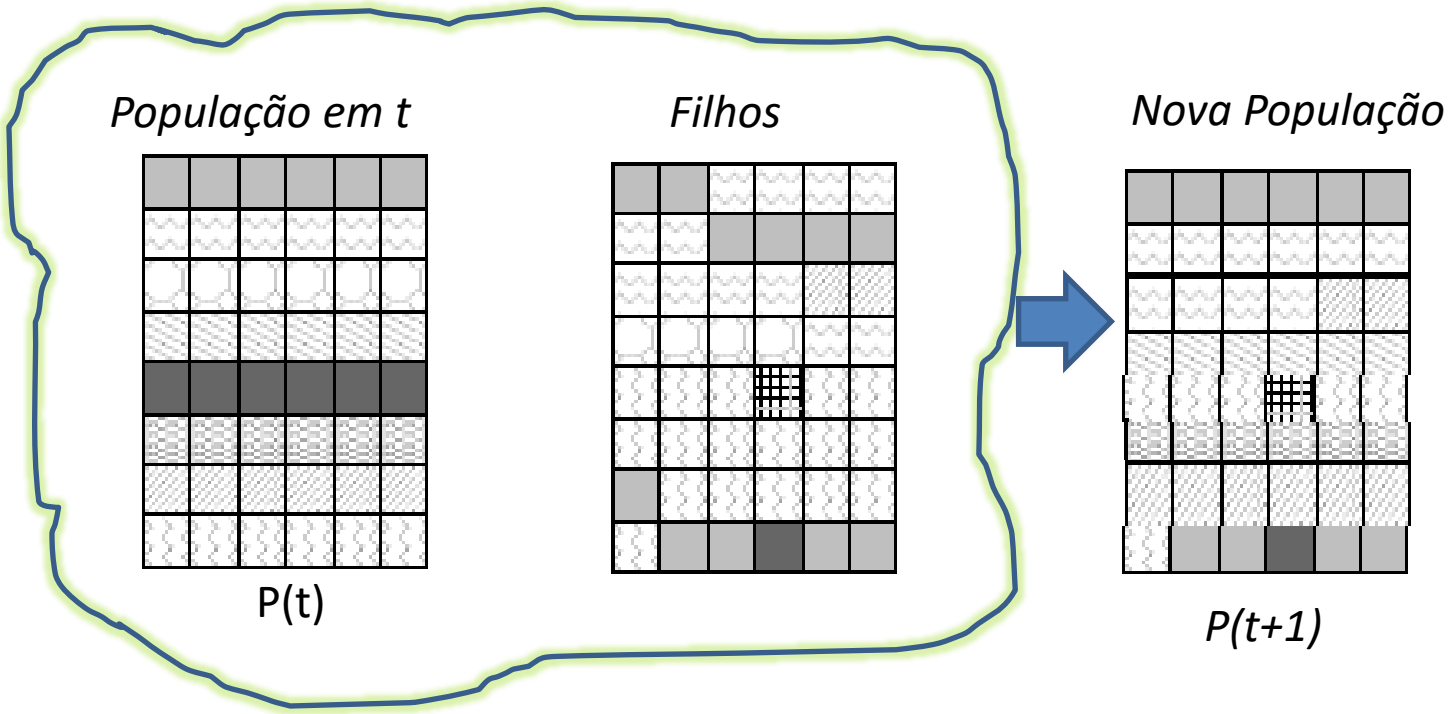


Posições são escolhidas aleatoriamente.

Mutação realizada com probabilidade baixa (tipicamente 0.05)

AG Canônico: **sobrevivência**

Sobrevivência dos Melhores : selecionar entre **pais e descendentes (filhos)** para compor a nova população



P(t+1)= Melhores(P(t) U Filhos)

AG Canônico: **pseudo-código**

```
C:={c1, ..., cn}   população inicial de tamanho N
R tamanho da descendência ou reprodução (normalmente K = N)
pCROSS: probabilidade de fazer crossover (valor típico 0.8)
pMUT: Probabilidade de fazer mutação nos alelos (valor típico 0.05)

AGCanônico(C, N, R, pCROSS, pMUT) {
   Para todo ci de C, calcular fitness(ci);
   geracao:=0;
   do {
      // D = Descendentes = nova geração, calculada a partir de C
      D := selecionar R cromossomos de C pelo método da roleta;
      D':= cruzamento(D); // geração de dois filhos por par (d1,d2),(d3,d4),...,(dk-1, dk)
          de D fazendo crossover com probabilidade pCROSS
      D":= para todo cromossomo di de D', para cada alelo aj de di, mutar aj
          com probabilidade pMUT;
      Para todo di de D", calcular fitness(di);
      // Selecionar melhores entre pais e filhos
      C := selecionar n melhores cromossomos de C ⊔ D";
      geracao++;
   } while (geracao<MAX_GERACOES and !objetivo-alcançado and
            !melhor-fitness estagnado);

   retornar ci de C com melhor fitness;
}
```

AG Canônico: resumo

- Representação do indivíduo: codificação binária
- Seleção
  - Roleta
- Reprodução
  - Crossover simples com seleção aleatória de pares (pais)
  - Mutação Uniforme
- Sobrevivência
  - Melhores entre pais e filhos para compor a nova população

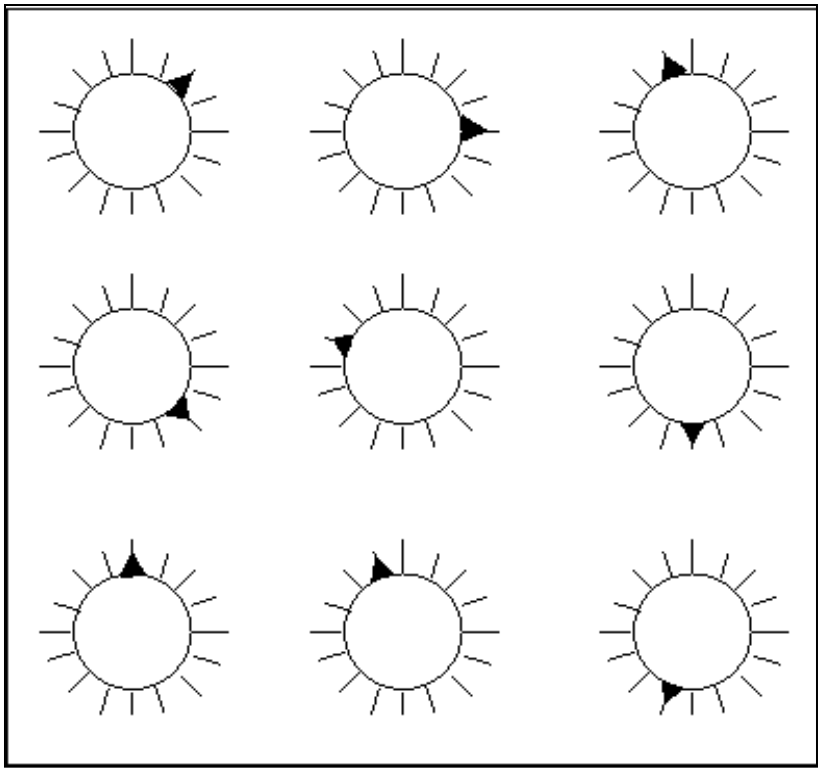
EXEMPLOS

PROBLEMA DOS BOTÕES

Sabendo que cada botão pode ser colocado em 16 posições distintas, encontre a melhor combinação de posições para os 9 botões disponíveis na superfície da caixa preta de modo que o sinal de saída assuma o valor máximo.

Codif. binária ou inteira

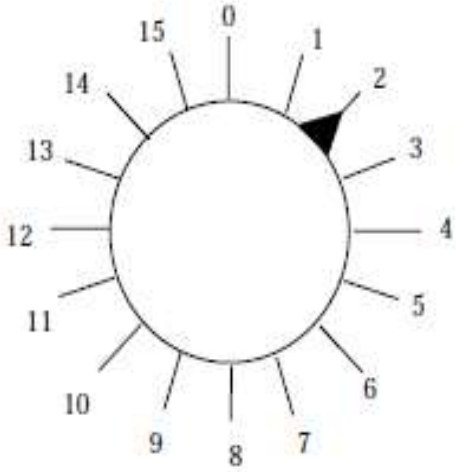
2
4
15
6
13
8
0
15
9



12.6

PROBLEMA DOS BOTÕES

- **Codificação**: Existem 16 posições possíveis para cada um dos 9 botões. Na codificação binária, 4 bits são suficientes para representar cada uma das 16 posições



Posição Atual: 0010

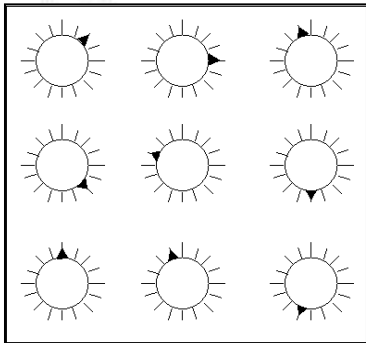
Posição	Representação	Posição	Representação
0	0000	8	1000
1	0001	9	1001
2	0010		
3	0011		
4	0100		
5	0101		
6	0110		
7	0111		

PROBLEMA DOS BOTÕES

Baseado neste tipo de codificação, cada cromossomo associado à solução candidata do problema definido anteriormente é dado por uma seqüência de 36 bits $(b_1, \cdots, b_{36})$, na qual o número de possíveis configurações de botões (soluções candidatas) é $2^{36} \cong 68.72$ bilhões. Neste caso, a solução candidata, mostrada na figura seria codificada por um cromossomo na forma:

$$0010\,0100\,1111\,0110\,1101\,1000\,0000\,1111\,1001\,.$$

Função de Fitness



PROBLEMA DOS BOTÕES

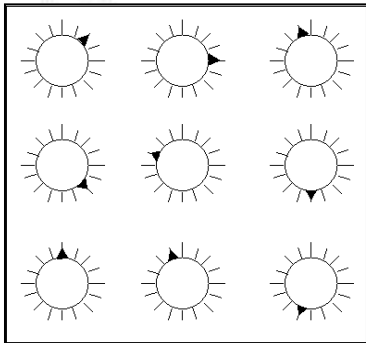
Baseado neste tipo de codificação, cada cromossomo associado à solução candidata do problema definido anteriormente é dado por uma seqüência de 36 bits $(b_1, \cdots, b_{36})$, na qual o número de possíveis configurações de botões (soluções candidatas) é $2^{36} \cong 68.72$ bilhões. Neste caso, a solução candidata, mostrada na figura seria codificada por um cromossomo na forma:

$$0010\,0100\,1111\,0110\,1101\,1000\,0000\,1111\,1001\,.$$

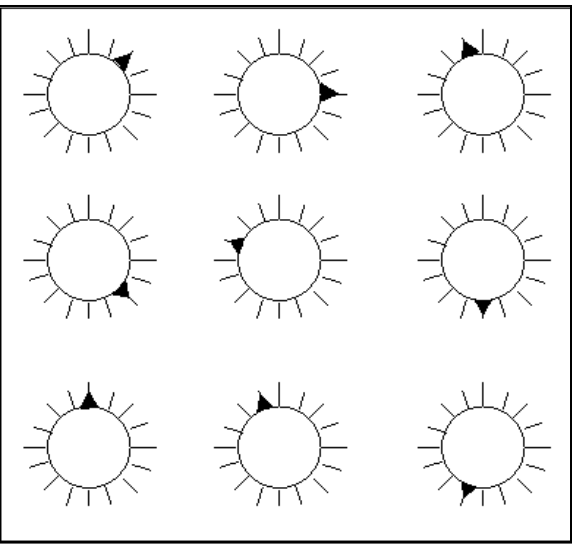
O mapeamento, suposto desconhecido, entre as 2^{36} posições possíveis dos botões e o sinal de saída é dado por:

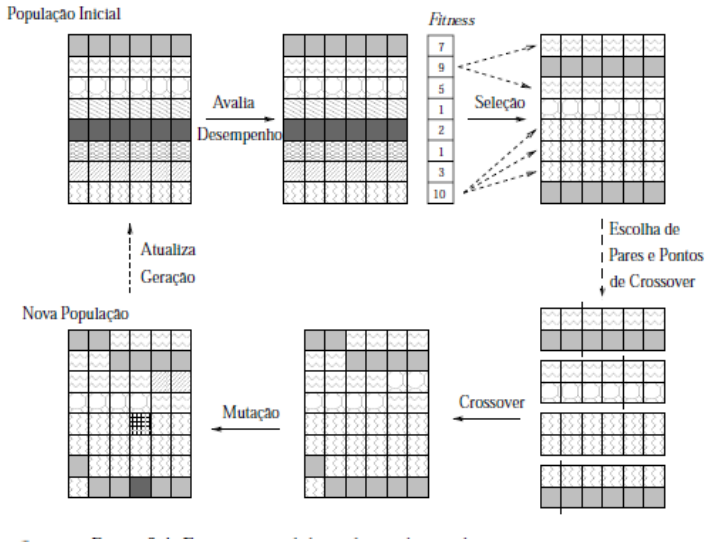
$$\begin{aligned}\text{sinal de saída} = {} & 9 + b_2b_5 - b_{23}b_{14} + b_{24}b_4 - b_{21}b_{10} + b_{36}b_{15} - b_{11}b_{26} + b_{16}b_{17} + b_3b_{33} \\ & + b_{28}b_{19} + b_{12}b_{34} - b_{31}b_{32} - b_{22}b_{25} + b_{35}b_{27} - b_{29}b_7 + b_8b_{13} - b_6b_9 + b_{18}b_{20} - b_1b_{30} \\ & + b_{23}b_4 + b_{21}b_{15} + b_{26}b_{16} + b_{31}b_{12} + b_{25}b_{19} + b_7b_8 + b_9b_{18} + b_1b_{33}\ ,\end{aligned}$$

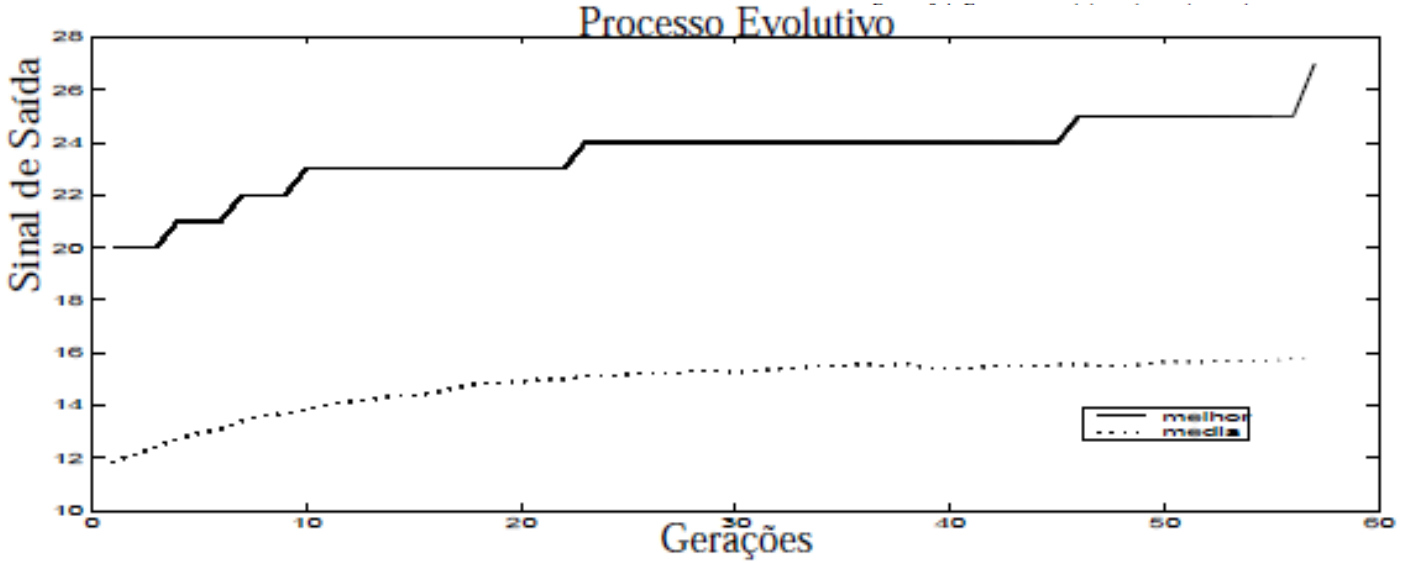
Função de Fitness



AG Canônico no problema dos botões



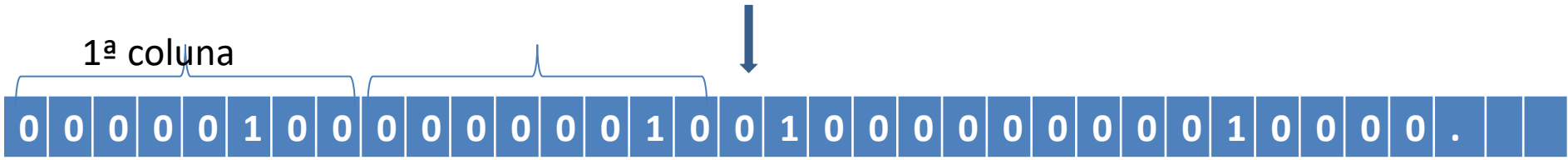




PROBLEMA DAS 8 RAINHAS

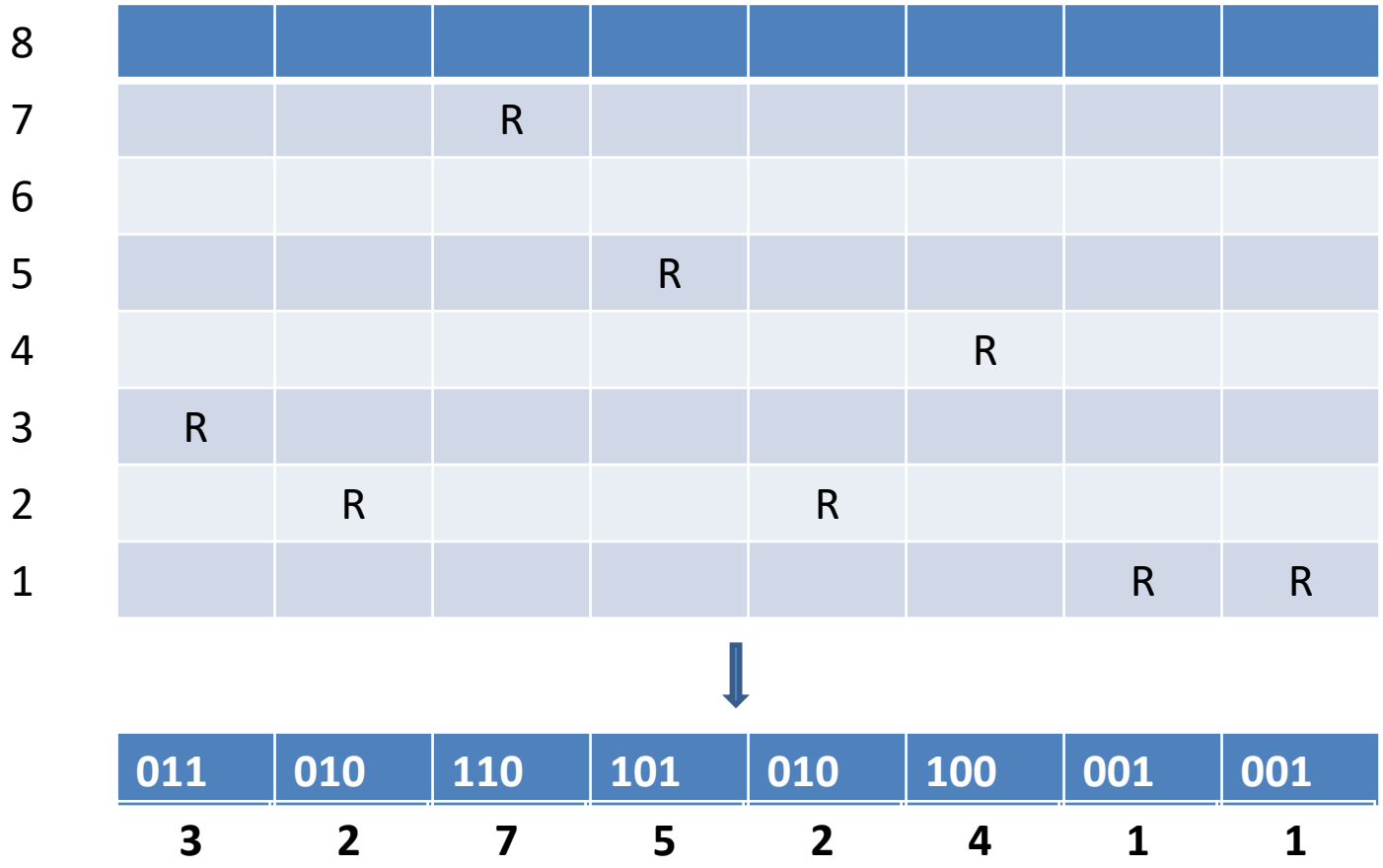
Solução por Codificação binária (matriz binária) -> vetor de bits

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0
0	0	0	0	0	0	1	1



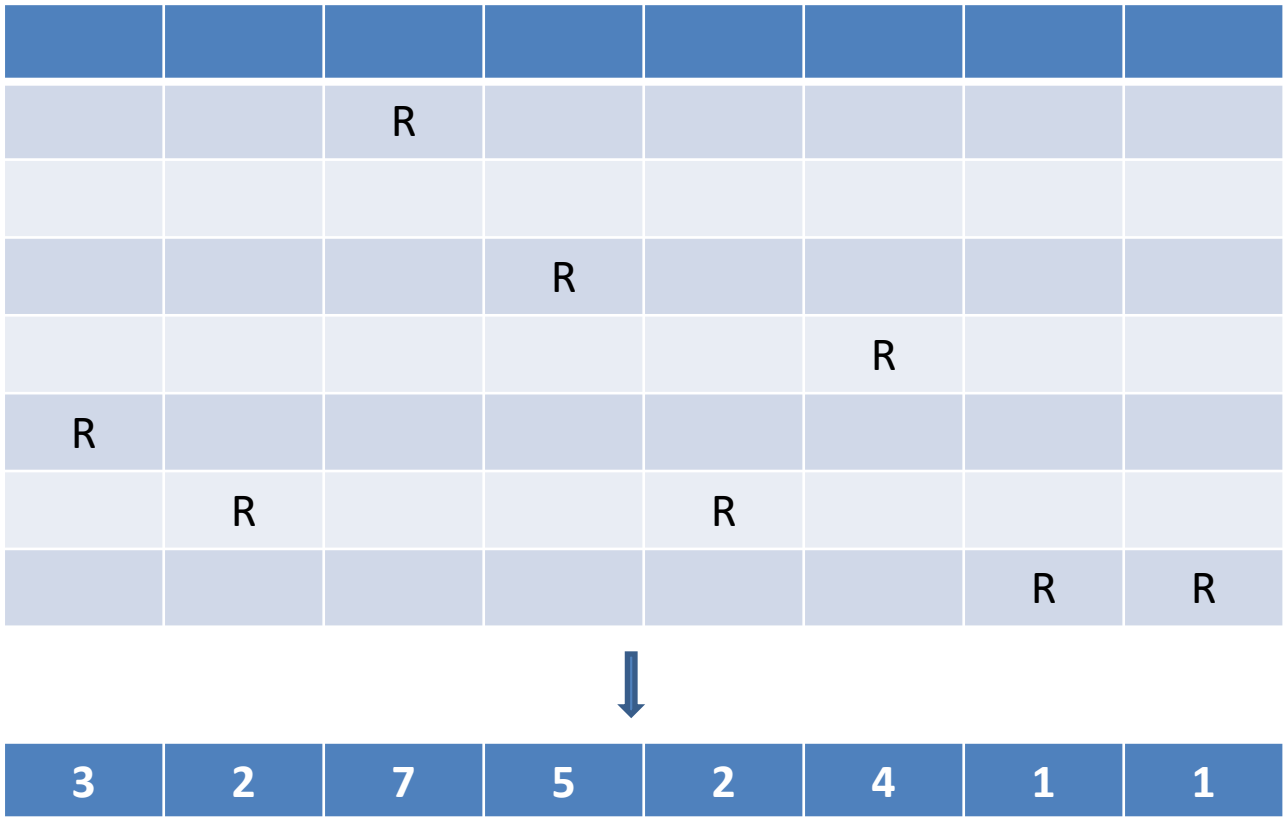
PROBLEMA DAS 8 RAINHAS

Solução por codificação binária (obtida por conversão inteiro-> binário)



PROBLEMA DAS 8 RAINHAS

Solução por Codificação inteira



AG EM PROBLEMAS DE PERMUTAÇÃO

Problemas de permutação

- **Solução** = determina a ordem de uma sequência de eventos

- **Exemplos**
  - **caixeiro viajante***: determinar a ordem de visita das cidades minimizando a distância percorrida (*problema de otimização de rotas)
  - **scheduling:** *ordem de operações para minimizar tempo de produção de peças)*

- *Que codificação utilizar?*
- *Quais operadores utilizar?*

Problemas de Permutação

Normalmente utiliza-se

Codificação numérica que indica a ordem do elemento

Caixeiro viajante

cidade	A	B	C	D	E	F	G	H	I	
ordem visita	**5**	**2**	**9**	**4**	**7**	**6**	**1**	**3**	**8**	= **cromossomo**

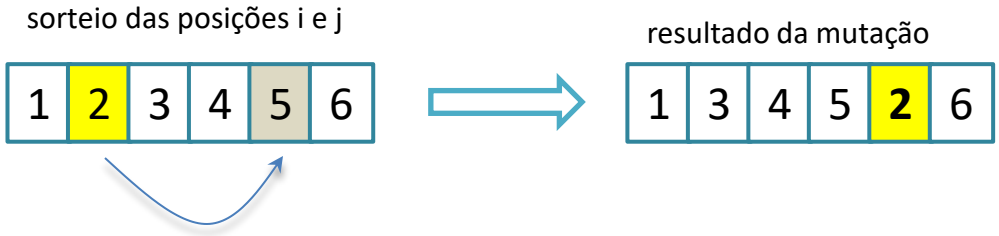
Codificação posicional: a posição do elemento (cidade) no cromossomo indica a ordem (de visita)

Caixeiro viajante

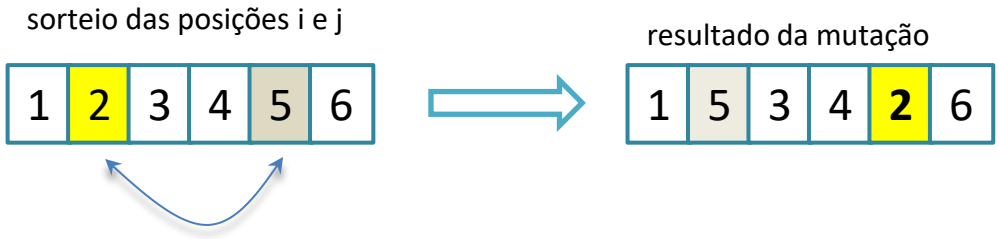
cidade	**G**	**B**	**I**	**D**	**A**	**F**	**E**	**I**	**C**	= **cromossomo**
ordem visita	1	2	3	4	5	6	7	8	9	

AG em Problemas de Permutação: tipos de mutação

Mutação baseada em posição:
retira alelo da posição i e insere em j



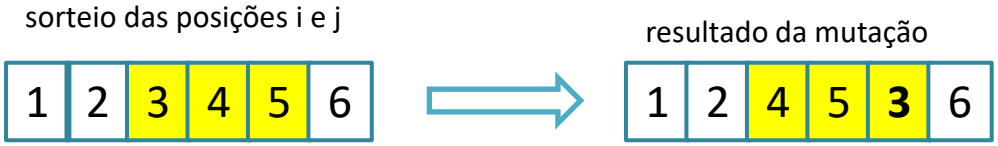
Mutação baseada em ordem:
troca elemento da posição i com o da j



AG em Problemas de Permutação: tipos de mutação

Mutação por embaralhamento

Selecionar aleatoriamente uma sublista e embaralhá-la.



AG em Problemas de Permutação: tipos de crossover

Crossover de Mapeamento Parcial (PMX)