Desenvolvendo AEs

Conceitos e Aplicações

Sumário

- Algoritmos Genéticos
- Evolutionary Programming
- Evolutionary Strategy
- Genetic Programming

Algoritmos Genéticos - AGs

- Holland introduziu o chamado *simple genetic algorithm* (SGA)
- Principais características:
  - Enfatiza o uso da recombinação.
  - Codificação: Binária
  - Recombinação : n-pontos ou uniforme
  - Mutação: inversão binária
  - Seleção para reprodução: proporcional ao fitness
  - Seleção para sobrevivência: geracional.

Algoritmo Genético

```
1: procedure Algoritmo Genético
2:     t ← 0
3:     Gerar População Inicial P(t)
4:     Avaliar P(t)
5:     while not criteriodeparada do
6:         for i ← 1 to N/2 do
7:             Selecionar dois indivíduos de P(t).
8:             Aplicar crossover aos dois indivíduos com probabilidade p_c.
9:             Mutar os novos indivíduos gerados com probabilidade p_m.
10:            Inserir os novos indivíduos em P(t + 1).
11:        end for
12:        t ← t + 1
13:    end while
14: end procedure
```

Figure: Pseudocódigo para AG.

Evolutionary Programming - EP

- Fogel, Owens e Walsh introduziram o conceito de Evolutionary programming.
- EP tinha como objetivo alcançar inteligência.
- Inteligência ⇔ Comportamento adaptativo.
  - A capacidade de predição do ambiente se torna uma condição para o comportamento adaptativo.
  - Logo, a capacidade para prever seria fundamental para comportamento inteligente.

Evolutionary Programming - EP

- Principais características:
  - Enfatiza a adaptação do efeito da mutação no decorrer do processo evolutivo.
  - Codificação: Real
  - Recombinação : Não há!!!!
  - Mutação: Gaussiana
  - Seleção para reprodução: determinística
  - Seleção para sobrevivência: probabilística (μ+μ)

Evolutionary Programming

1. t=1
2. Iniciar e avaliar população P(t)
3. **Repita**
   i. P'(t) = Mutaçao (P(t))
   ii. Avaliar P'(t)
   iii. P(t+1) = Selecionar(P(t) ∪ P'(t))
   iv. Inserir filho gerado na população
   v. t = t+1
4. **até** Critério de Parada

Evolutionary Programming - EP

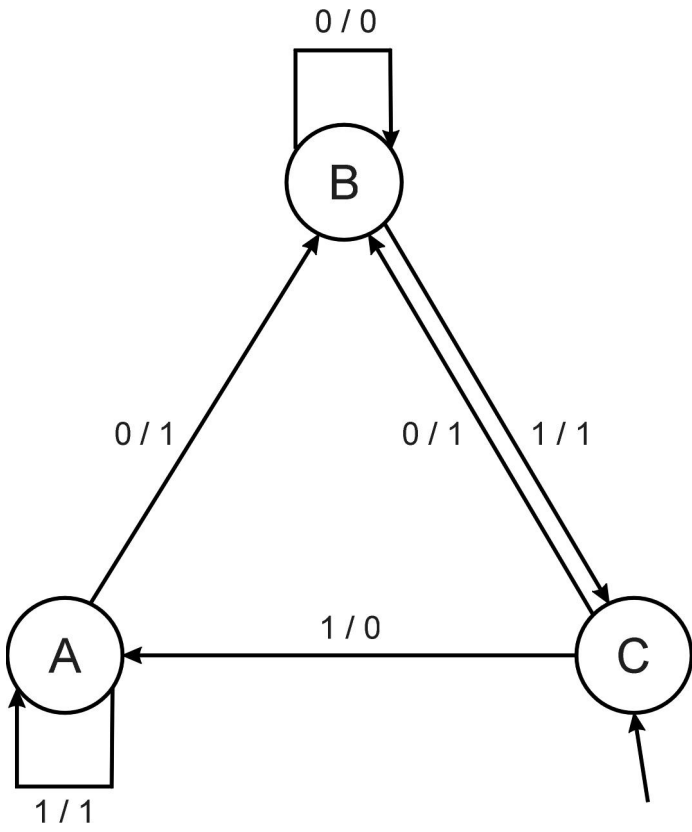
- Autômatos Finitos ou Máquinas de Estados Finitos:
  - Modelo matemático abstrato capaz de representar circuitos lógicos ou mesmo programas de computadores.
  - Essa máquina de estados estará em apenas um de um conjunto finito de estados possíveis.

Evolutionary Programming - EP

- Finite state machine (FSM)
- Componentes:
  - Estados (States): S
  - Inputs: I
  - Outputs: O
  - Função de transição dos estados δ:
    - $S \times I \rightarrow S \times O$
- Usada, por exemplo, para predizer o próximo símbolo de entrada em uma sequência

Exemplo: Prevendo o próximo input

- Input: 011101
- Output: 110111
- Critério de avaliação:
  # In(t+1)= #Out(t)
- Estado Inicial: C
- Resultado:
  - Acerta 3 de 5



Evolutionary Programming - EP

- Evoluindo FSMs
  - Seleção para reprodução:
    - Cada FSM sofre mutação uma única vez.
  - Mutação
    - Troca um símbolo de entrada
    - Troca uma transição de estado, por exemplo, alterando o sentido da aresta.
    - Delete um estado
    - Troca o estado inicial.
  - Seleção para sobrevivência: (μ+μ)

Evolutionary Programming - EP

- Atualmente:
  - Diferentes tipos de representação podem ser utilizadas
  - As representações impactam no tipo de mutação.
  - Auto adaptação dos parâmetros de mutação.

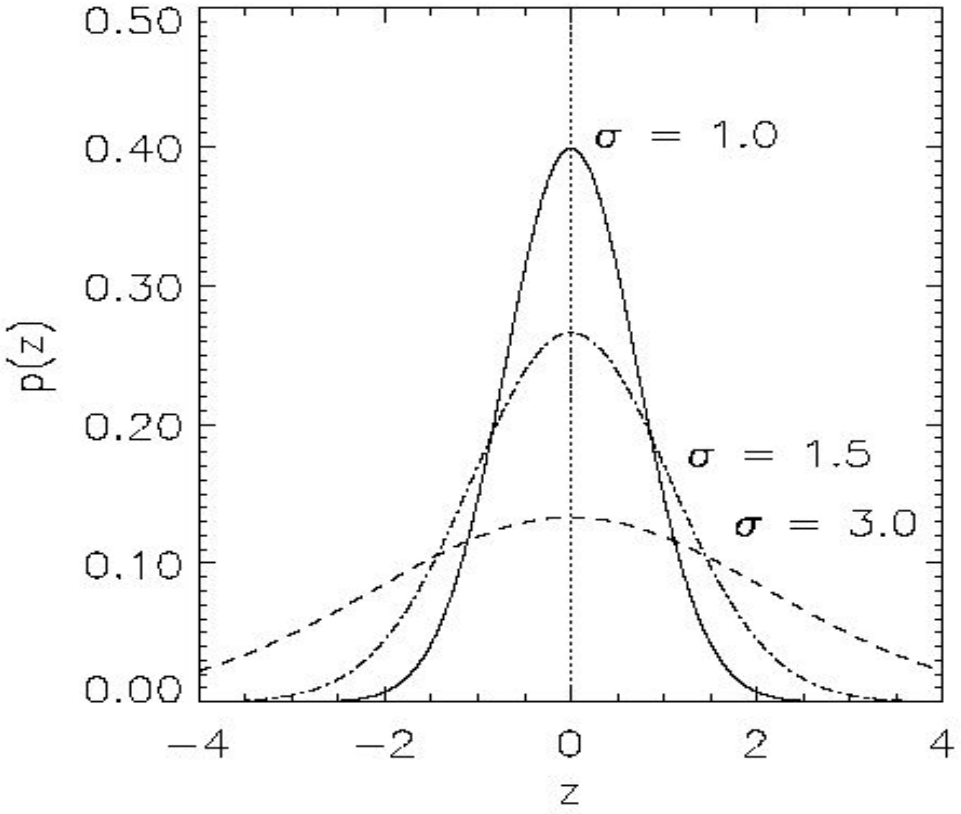
Evolutionary Programming - EP

- Representação:
  - $\langle x1,\ldots,xn, \sigma1,\ldots,\sigma n \rangle$
  - Variáveis: : x1,…,xn
  - Desvio padrão das mutações: σ1,…,σn

Evolutionary Programming - EP

- Mutação:
  - Altera variáveis e tamanho da mutação
    - $(x1,\ldots,xn, \sigma1,\ldots,\sigma n)$
    - $\sigma i' = \sigma i \bullet (1 + \alpha \bullet N(0,1))$
    - $x'i = xi + \sigma i' \bullet Ni(0,1)$
    - Parâmetro $\alpha \approx 0.2$
    - Ni(0,1): distribuição Guassiana centrad (média 0 e desvio 1).
- Outras distribuições podem ser utilizadas, por exemplo, Cauchy.

Evolutionary Programming - EP



Evolutionary Programming - EP

- Não há recombinação.
- Motivo:
  - Cada ponto do espaço de busca representa uma espécie, ou seja, abstraindo um conjunto de indivíduos.
  - Logo, não há recombinação entre espécies diferentes.

Evolutionary Programming - EP

- Seleção para reprodução
  - Mutação é aplicada sobre cada espécie.
  - Não há taxa de mutação
    - Mutação é determinística
  - Fitness não influencia a geração da nova espécie.

Evolutionary Programming - EP

- Seleção para sobrevivência
  - Após a mutação, teremos:
    - P(t): μ espécies
    - P'(t): μ novas espécies
  - Seleção por torneio Round-Robin
    - Seleciona-se cada espécie $x \in P(t) \cup P'(t)$
    - x compete contra q outras espécies selecionadas aleatoriamente de $P(t) \cup P'(t)$.
    - Registra-se o número de vitórias obtidas por x.
    - As μ espécies com maior número de vitórias permanecem na próxima geração.

Evolutionary Strategy - ES

- Rechenberg e Schwefel introduziram ES na década de 70 na Alemanha.
- Aplicações relacionada à otimização numérica.

Evolutionary Strategy - ES

- Principais características:
  - Codificação : Real.
  - Recombinação: discreta ou intermediária.
  - Mutação: Guassiana.
  - Seleção para reprodução: Uniforme.
  - Seleção para sobrevivência: (μ,λ) or (μ+λ)
  - Auto adaptação dos parâmetros de mutação.

Evolutionary Strategy - ES

1. i=1
2. Iniciar e avaliar população
3. **Repita**
    i. **Enquanto** (i ≤ c.λ) **faça**
        a. selecionar k≥ 2 pais aleatoriamente
        b. Recombinar (x,σ, α) gerando filho
        c. Mutar (x,σ, α) no filho gerado
        d. Avaliar filho gerado
        e. Inserir filho gerado na *mating pool*
        f. i = i+1
    ii. **fimEnquanto**
    iii. Selecionar μ melhores indivíduos com (λ , μ) ou ((λ + μ) .
    iv. Inserir os μ melhores indivíduos na população da próxima geração
4. **até** Critério de Parada

Evolutionary Strategy - ES

- Codificação: (x1,…,xn, σ1,…,σn ,α1,…, αk)
  - Variáveis: x1,…,xn.
  - Tamanho da mutação: σ1,…,σnσ
  - Rotação angular: α1,…, αnα

Evolutionary Strategy - ES

- Mutação Caso 1 - $(x1,\ldots,xn, \sigma)$
  - $x'i = xi + N(0,\sigma)$
  - $(x1,\ldots,xn, \sigma) \rightarrow (x'1,\ldots,x'n, \sigma')$
  - Ordem é importante
    i. $\sigma \rightarrow \sigma'$
    ii. $x \rightarrow x' = x + N(0,\sigma')$

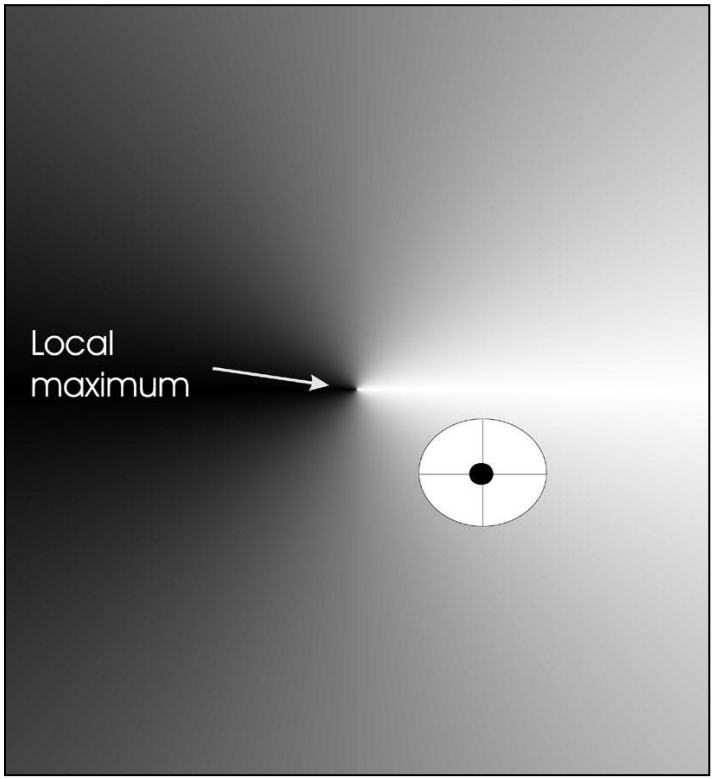
Evolutionary Strategy - ES

- Mutação Caso 1 - (x1,…,xn, σ):
  - Exemplo 1: $\sigma \rightarrow \sigma'$ com "1/5 success rule":
    i. $\sigma = \sigma / c$ se sr > 1/5
       $\sigma = \sigma \bullet c$ se sr < 1/5
       $\sigma = \sigma$ se sr = ⅕
       - sr: porcentagem de mutações bem sucedidas
       - $0.8 \leq c \leq 1$
    ii. $x'i = xi + N(0,\sigma)$

Evolutionary Strategy - ES

- Mutação Caso 1 - (x1,…,xn, σ):
  - Exemplo 2: $\sigma \rightarrow \sigma'$ com $\exp(\tau \bullet N(0,1))$:
    i. $\sigma' = \sigma \bullet \exp(\tau \bullet N(0,1))$
    ii. $x'i = xi + \sigma' \bullet N(0,1)$
      - $\tau \propto 1/\sqrt{n}$

Evolutionary Strategy - ES

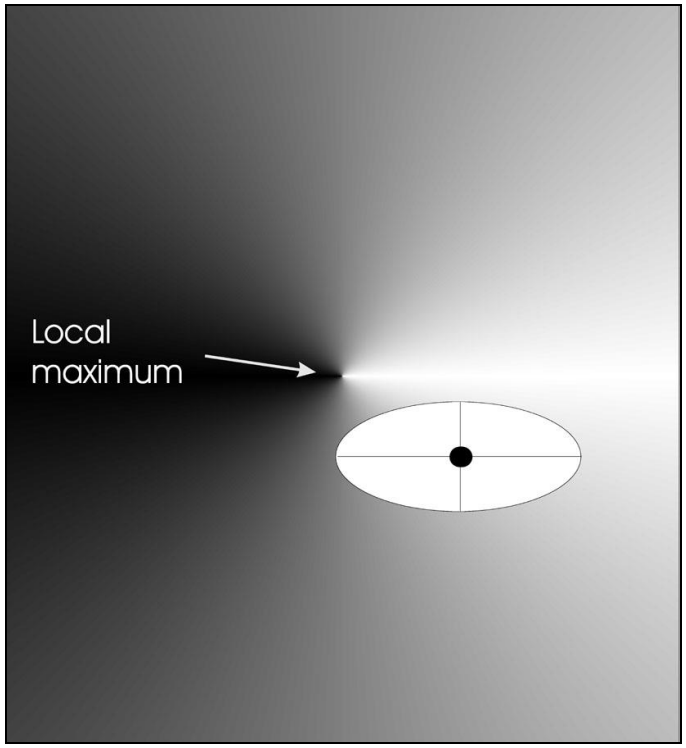


Mesma probabilidade de mutação. Fonte: Eiben (2005)

Evolutionary Strategy - ES

- Mutação Caso 2 - (x1,…,xn, σ1,…, σn):
  i. $\sigma'_i = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1))$
  ii. $x'_i = x_i + \sigma'_i \cdot N_i(0,1)$
    - τ': taxa aprendizado global
    - τ: taxa aprendizado local
    - $\tau \propto 1/(2\,n)^{1/2}$ e $\tau \propto 1/(2\,n^{1/2})^{1/2}$

Evolutionary Strategy - ES

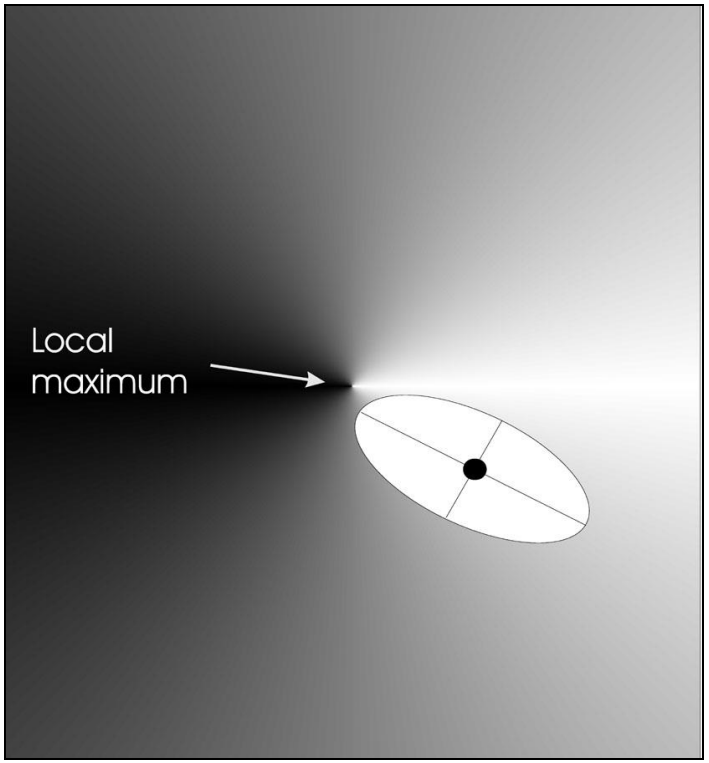


Dispersão elíptica. Fonte: Eiben (2005)

Evolutionary Strategy - ES

- Mutação Caso 3 - $(x1,\ldots,xn, \sigma1,\ldots, \sigma n, \alpha1,\ldots, \alpha k)$:
  - i. $\sigma'i = \sigma i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot Ni(0,1))$
    $\alpha'j = \alpha j + \beta \cdot N(0,1)$
  - ii. $x' = x + N(0,C')$
  - C': Matriz de covariância
    - i. $cii = \sigma i2$
      $cij = 0$ se i e j não estão correlacionados
      $cij = \frac{1}{2} \cdot (\sigma i2 - \sigma j2) \cdot \tan(2 \alpha ij)$, caso contrário.
  - $\tau \propto 1/(2n)^{1/2}$, $\tau \propto 1/(2n^{1/2})^{1/2}$ e $\beta \approx 5°$
  - $\sigma i' < \varepsilon 0 \Rightarrow \sigma i' = \varepsilon 0$
  - $|\alpha'j| > \pi \Rightarrow \alpha'j = \alpha'j - 2\pi \operatorname{sign}(\alpha'j)$

Evolutionary Strategy - ES



Dispersão Elíptica e Angular. Fonte Eiben (2005)

Evolutionary Strategy - ES

- Recombinação:
  - Gera um único novo indivíduo
    - Aplicado para cada gene do cromossomo:
    - Define o gene pelo valor médio dos pais.
  - Define o gene como cópia de um dos genes dos pais.
  - Seleciona dois ou mais pais:
    - Seleciona dois pais para criar o novo indivíduo.
    - Seleciona dois pais para criar cada gene do novo indivíduo.

Evolutionary Strategy - ES

	Two fixed parents	Two parents selected for each i
$z_i = (x_i + y_i)/2$	Local intermediary	Global intermediary
z_i is x_i or y_i chosen randomly	Local discrete	Global discrete

Evolutionary Strategy - ES

- Seleção para reprodução
  - Pais selecionados via distribuição uniforme.
  - Logo, todos os indivíduos têm a mesma chance de serem selecionados para reprodução.
  - Pais na ES fazem parte da população como um todo.

Evolutionary Strategy - ES

- Seleção para sobrevivência
  - Aplicado após a criação der λ novos indivíduos a partir da população μ via crossover e mutação.
  - Elimina de forma determinística:
    - via (μ,λ) - apenas filhos - pode esquecer o melhor.
    - via (μ+λ)- apenas os melhores - elitismo
  - A pressão seletiva na ES costuma ser alta com:

$$\lambda \approx 7 \bullet \mu$$

Genetic Programming- GP

- Koza introduziu GP nos EUA na década de 90.
- Aplicada em aprendizado de máquinas voltada a tarefas como predição e classificação.
- Competia com redes neurais
- Necessidade de populações com tamanho na casa dos milhares.
- Costuma ser lento.

Genetic Programming- GP

- Principais características:
  - Codificação: árvores
  - Recombinação: troca de subárvores.
  - Mutação: mudança aleatória na árvore.
  - Seleção para reprodução: proporcional ao fitness.
  - Seleção para sobrevivência: geracional.

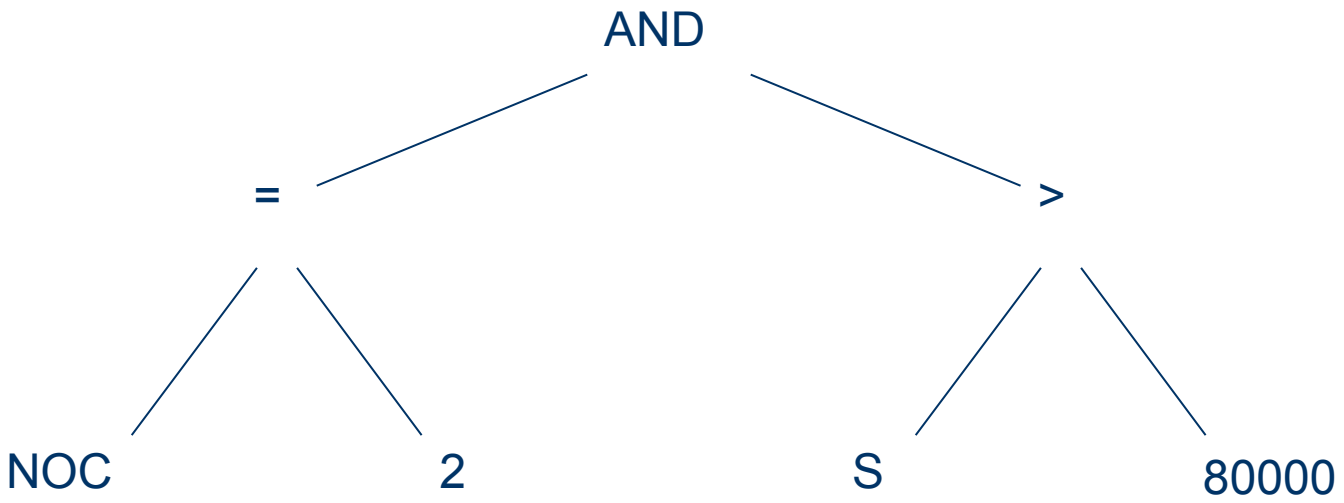
Genetic Programming- GP

- Exemplo:
  - IF (NOC = 2) AND (S > 80000) THEN good ELSE bad
  - Generalizando:

    IF formula THEN good ELSE bad
- Espaço de busca é definido pelo conjunto de fórmulas.
- Fitness: percentual de casos classificados corretamente.

Genetic Programming- GP

IF (NOC = 2) AND (S > 80000) THEN good ELSE bad



Genetic Programming- GP

Mais exemplos:

- Arithmetic formula

$$2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$$

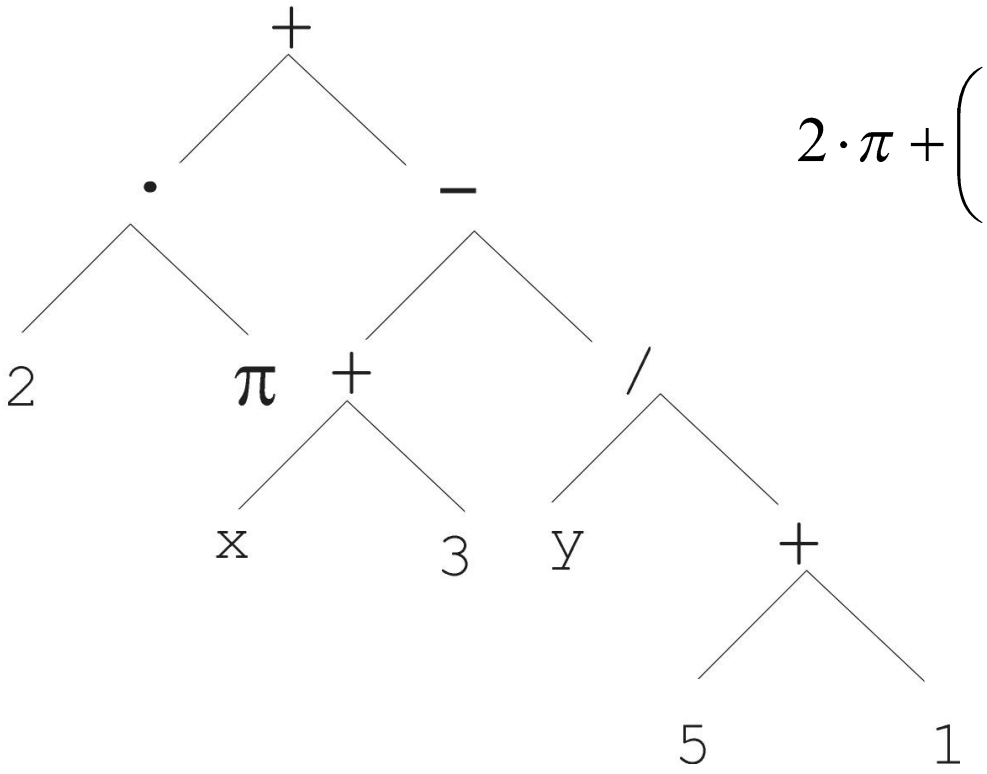
- Logical formula

$$(x \wedge \text{true}) \rightarrow ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$$

- Program

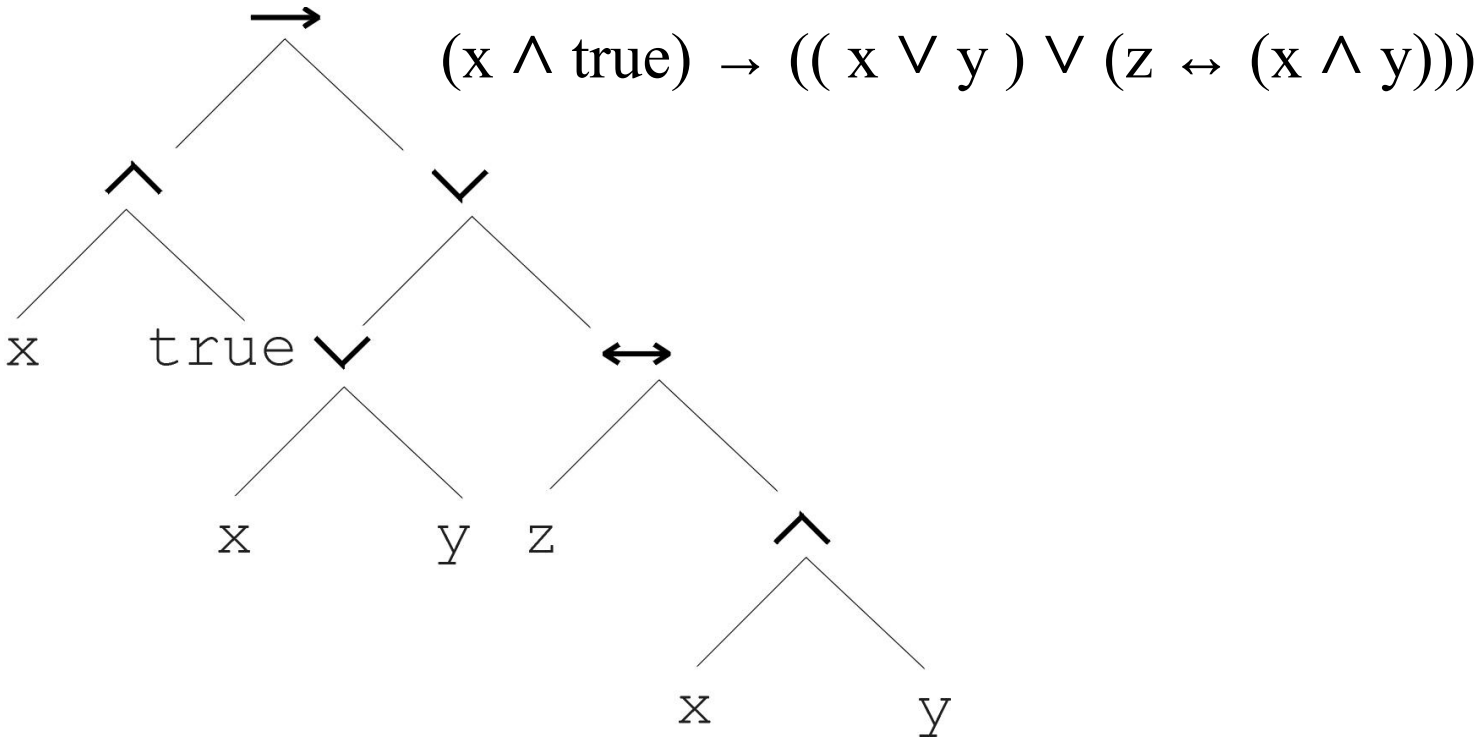
```
i =1;
while (i < 20)
{
    i = i +1
}
```

Genetic Programming- GP

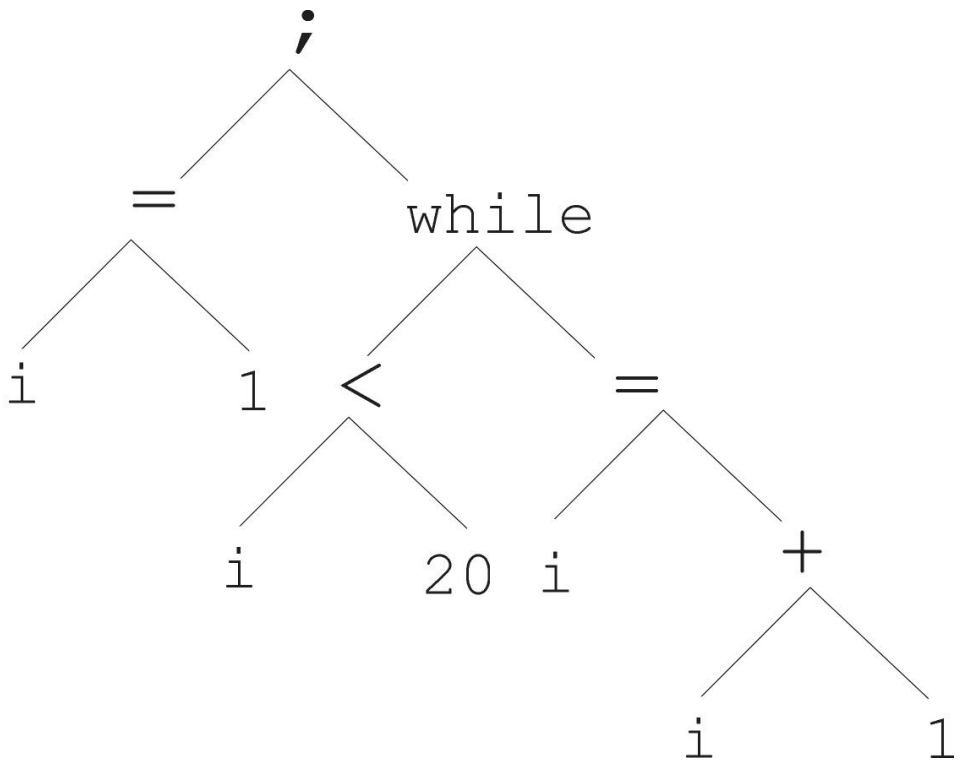


$$2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$$

Genetic Programming- GP



Genetic Programming- GP



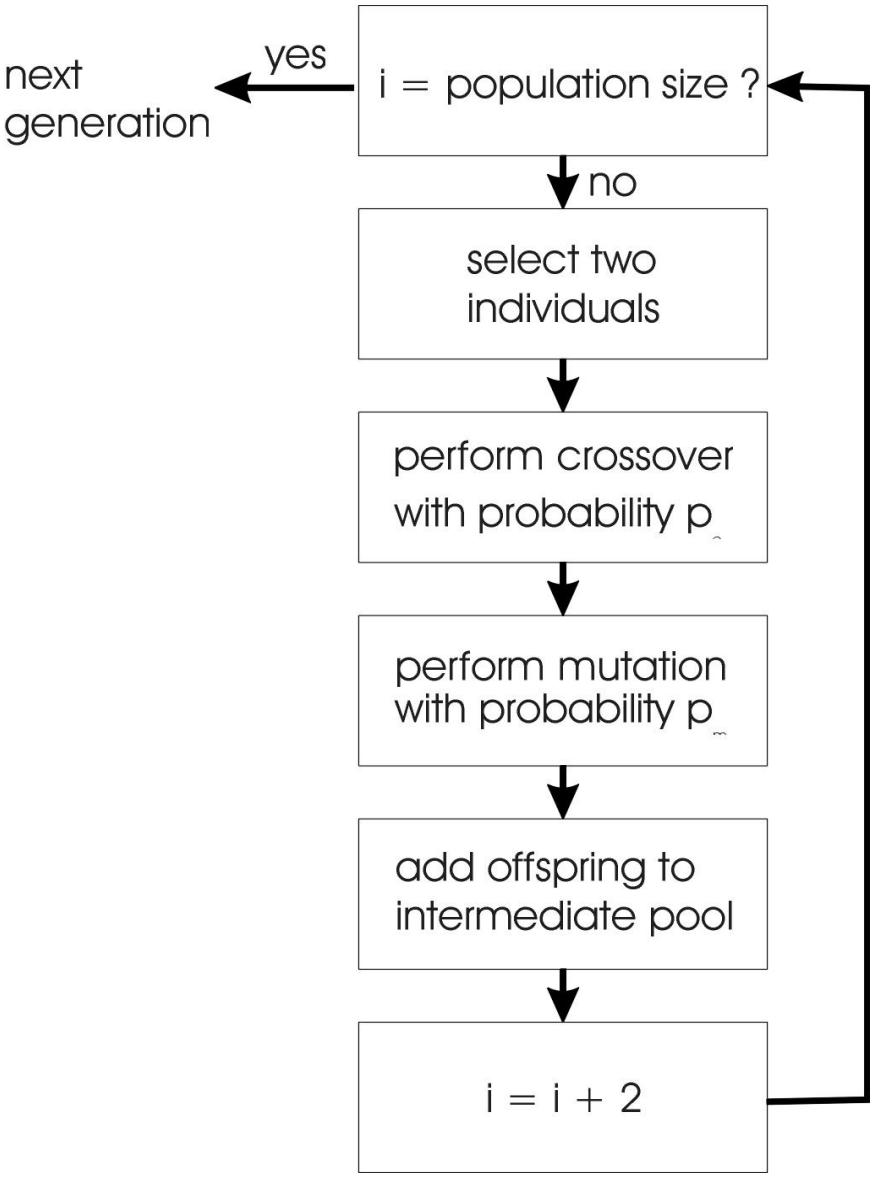

```
i =1;
while (i < 20)
{
    i = i +1
}
```

Genetic Programming- GP

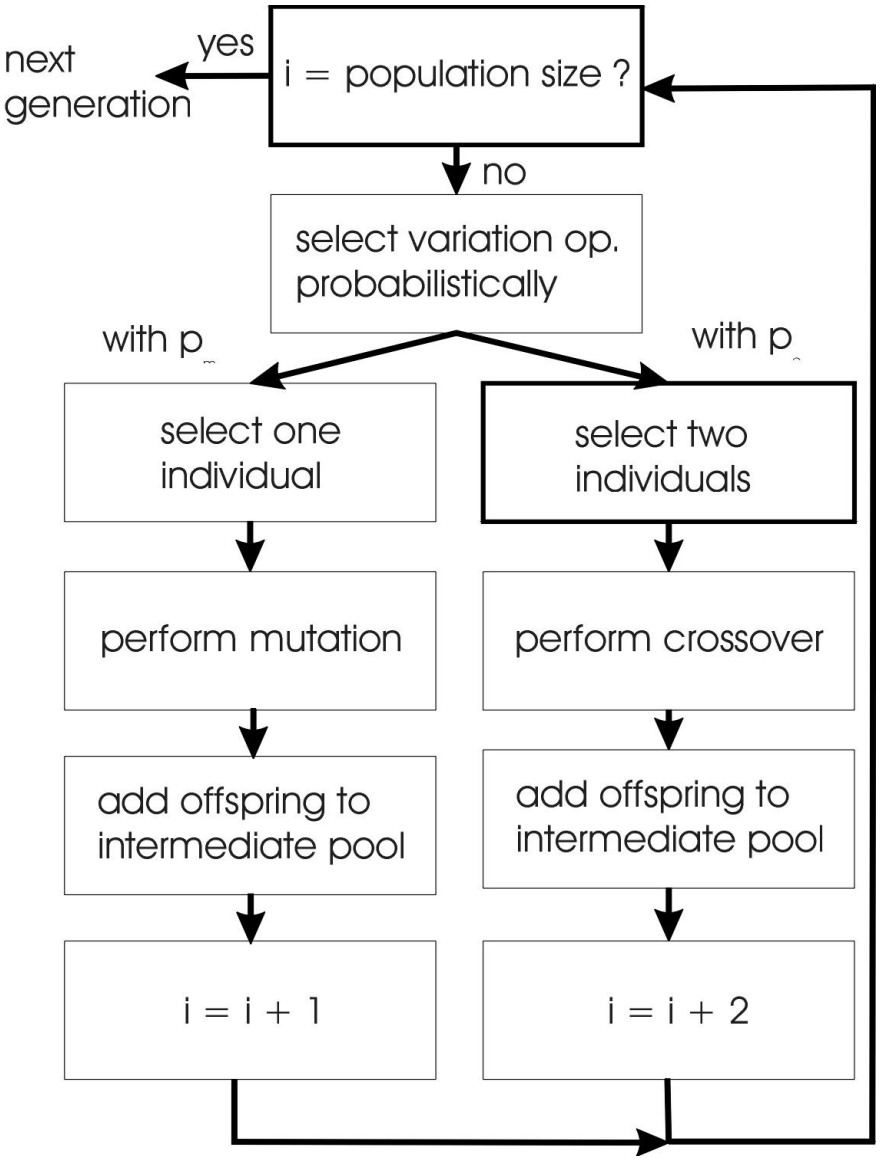
- GA, ES, EP utilizam estruturas lineares para codificar as informações nos cromossomos.
- Estruturas em árvore permitem uma representação não-linear dos cromossomos.
- O tamanho costuma ser fixo nas codificações utilizadas por GA, ES, EP.
- As árvoers na GP podem variar em largura e profundidade.

Genetic Programming- GP

- Codificação em árvore:
  - T: conjunto de nós terminais.
  - F: conjunto de funções
- Processo de geração recursivo:
  1. Cada $t \in T$ é uma expressão correta
  2. $f(e_1, \ldots, e_n)$ é uma expressão correta, se $f \in F$, $arity(f)=n$ e $e_1, \ldots, e_n$ são expressões corretas
  3. There are no other forms of correct expressions
- In general, expressions in GP are not typed (closure property: any $f \in F$ can take any $g \in F$ as argument)



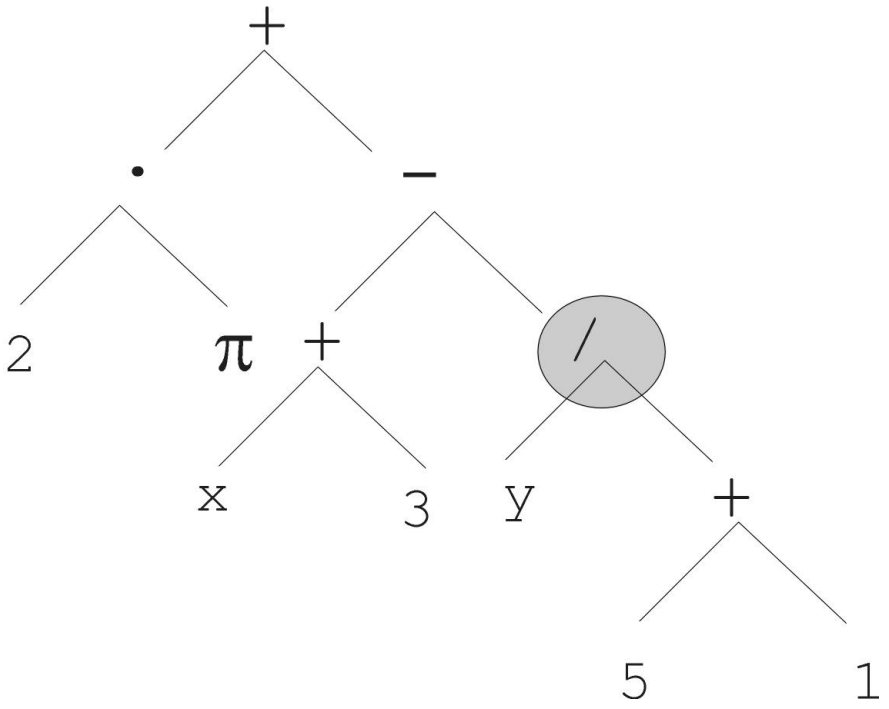
GA flowchart

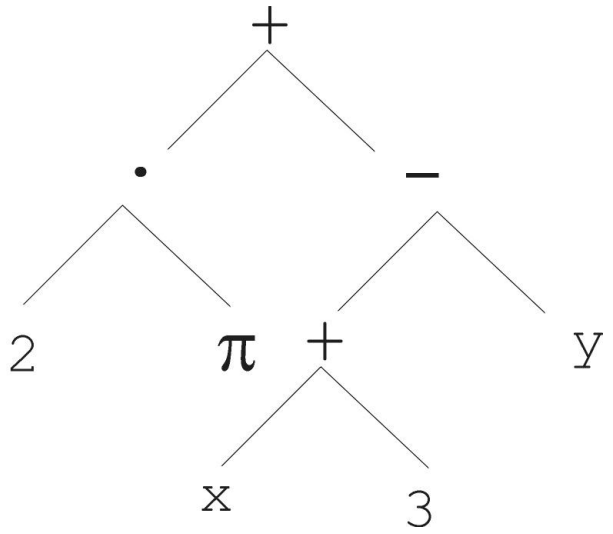


GP flowchart

Genetic Programming- GP

- Mutação - Exemplo



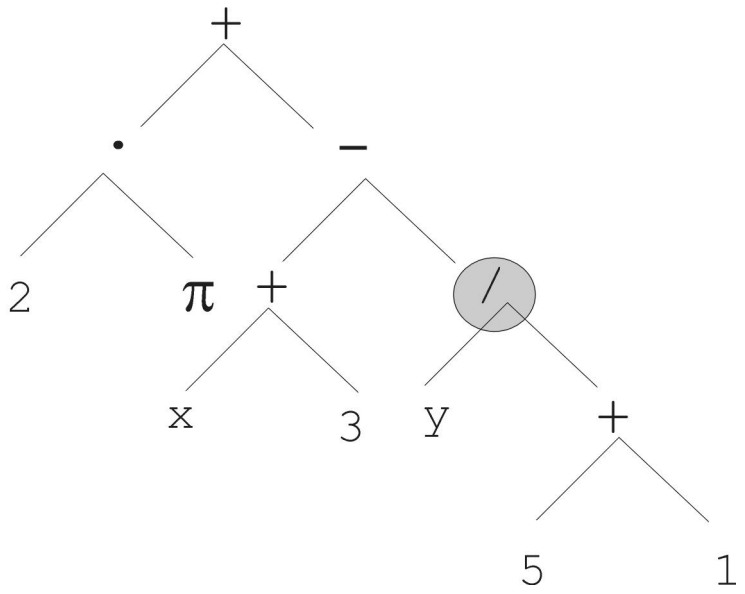


Genetic Programming- GP

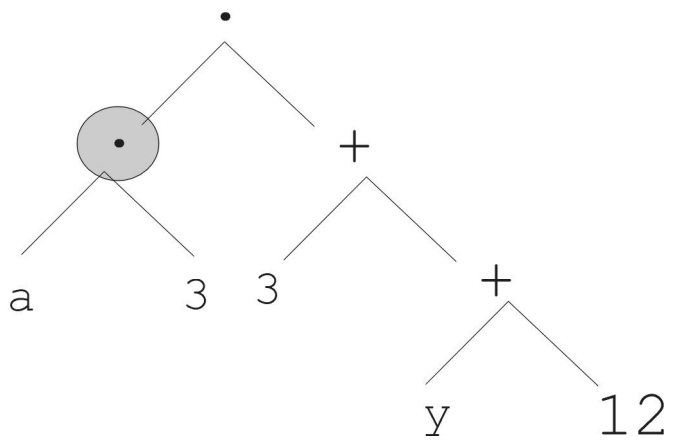
- Mutação
  - Dois parâmetros:
  - p_m: probabilidade de escolher entre mutação e crossover;
  - Probabilidade de escolher um ponto da árover como nó raiz de uma subárvore para ser trocado.

Genetic Programming- GP

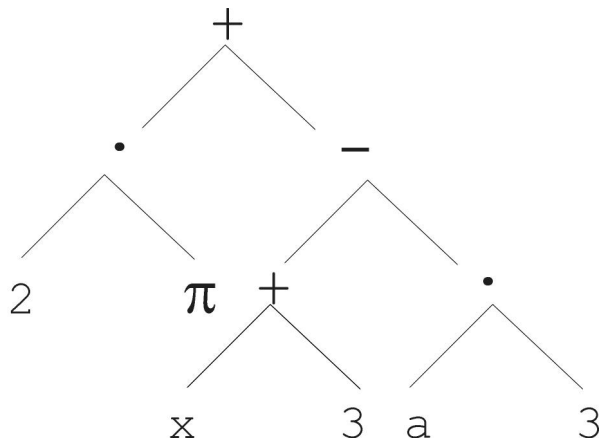
- Recombinação
  - Troca de duas subárovers aleatoriamnte escolhidas a partir dos pais.
- Dois parâmetros para recombinação:
  - p_c : escolha entre recombinação e mutação
  - Probabilidade de escolher um ponto interno dentro de cada pai como ponto de recombinação.



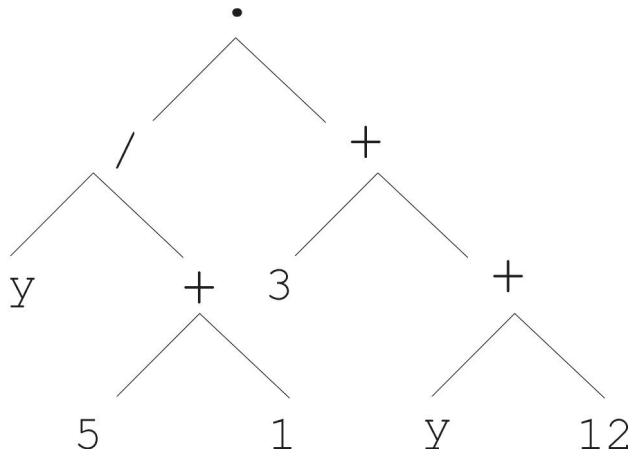
Parent 1



Parent 2



Child 1



Child 2

Genetic Programming- GP

- Seleção dos pais
  - Seleção proporcional ao fitness
    - Para populações muito grandes:
    - Ordenar indivíduos pelo valor do fitness e separar em dois grupos:
      - x% dos melhores indivíduos no grupo 1
      - (100-x)% no grupo 2.
    - 80% são selecionados do grupo 1, 20% do grupo 2
    - Para pop. size = 1000, 2000, 4000, 8000, podemos ter x = 32%, 16%, 8%, 4%

Genetic Programming- GP

- Seleção para sobrevivência:
  - Mais usual: geracional
  - Elitismo tem sido aplicado.

Genetic Programming- GP

- Inicialização:
  - Profundidade máxima da árvore é ajustada: D_{max}
- Método Completo: Toda árvore tem depth d= D_{max}:
  - Para nós em d < D_{max} , escolha aleatoriamente f ∈ F
  - Para nós em d = D_{max} , escolha aleatoriamente t ∈ T
- Método Crescimento: Toda árvore tem depth d ≤ D_{max}:
  - Para nós em d < D_{max} , escolha aleatoriamente x ∈ F ∪ T
  - Para npos em d = D_{max} , escolha aleatoriamente t ∈ T
- Combinação de ambos.

Genetic Programming- GP

- Bloat = “survival of the fattest”
- O tamanho das árvores cresce no decorrer das gerações
  - Proibir operadores de gerarem filhos com tamanho muito grande.
  - Penalizar árvores grandes.

Micro Algoritmos Genéticos - µGA

- Algoritmo genético que evolui uma pequena população de indivíduos.
- A população é reinicializada segundo um critério de convergência.
- A mutação não é aplicada já que diversidade é fornecida pelas reinicializações executadas.

Micro Algoritmos Genéticos - μGA

- Resultados teórico obtidos por Goldberg (1989) indicavam que uma população de três indivíduos era suficiente para se atingir convergência, independentemente do tamanho do indivíduo (cromossomo).

- O termo Micro GA (μGA) foi introduzido por Krishnakumar (1989) quem primeiro implementou o método com uma população de 5 indivíduos.

Micro Algoritmos Genéticos - µGA

- A implementação proposta em Krishnakumar (1989) considerava representação binária e foi aplicada a três funções mono-objetivo do tipo unimodal, multi-modal e não estacionária.

- O µGA convergiu mais rapidamente para a região promissora do espaço de busca do que um AG padrão.

Micro Algoritmos Genéticos - µGA

- Uma implementação de µGA em problemas multi-objetivos foi proposta por de Coello e Pulido (2001).

- O µGA também apresenta uma convergência mais rápida para a fronteira de pareto que um AG padrão.

Micro Algoritmos Genéticos - μGA

- Goldberg (1989) sugere os seguintes passos para se obter sucesso trabalhando com pequenas populações:

Passo1: Aletoriamente gere uma pequena população.

Passo2: Execute os operadores genéticos até obter convergência.

Passo3: Enviar o melhor indivíduo da população que convergiu para compor a população da próxima geração. Os demais indivíduos são gerados aleatoriamente.

Passo4: Volte para o Passo2.

Micro Algoritmos Genéticos - µGA

- O µGA proposto por Krishnakumar (1989) adapta a ideia de Goldberg (1989) através dos seguintes passos:

Passo1: Selecione uma população de tamanho 5 aleatoriamente ou com 4 indivíduos gerados aleatoriamente e 1 bom indivíduo obtido anteriormente.

Passo2: Calcule o fitness dos indivíduos e determine o melhor. O melhor indivíduo é mantido na população da próxima geração.

Passo3: Escolha os demais 4 indivíduos para reprodução, onde o melhor indivíduo também poderá ser escolhido, utilizando a estratégia de seleção por torneio. Evitar selecionar via torneio a partir de dois indivíduos idênticos.

Passo4: Aplicar sempre crossover (taxa de crossover = 100%). A mutação não é executada.

Passo5: Se a população convergiu, vá para Passo 1.

Passo 6: Vá para Passo2.

Resultados - Krishnakumar (1989)

- Função Unimodal
  - $J1 = x^2 + y^2 + z^2$, onde $-5.12 \leq x, y, z \leq 5.12$
- Representação
  - binária com indivíduos de tamanho 30.
- O *simple (standard) genetic algorithm* (SGA) foi executado:
  - utilizando a mesma seleção por torneio,
  - populações com tamanho N = 50 e 200.
- Um total de 25 execuções independents foram realizadas para μGA e SGA.

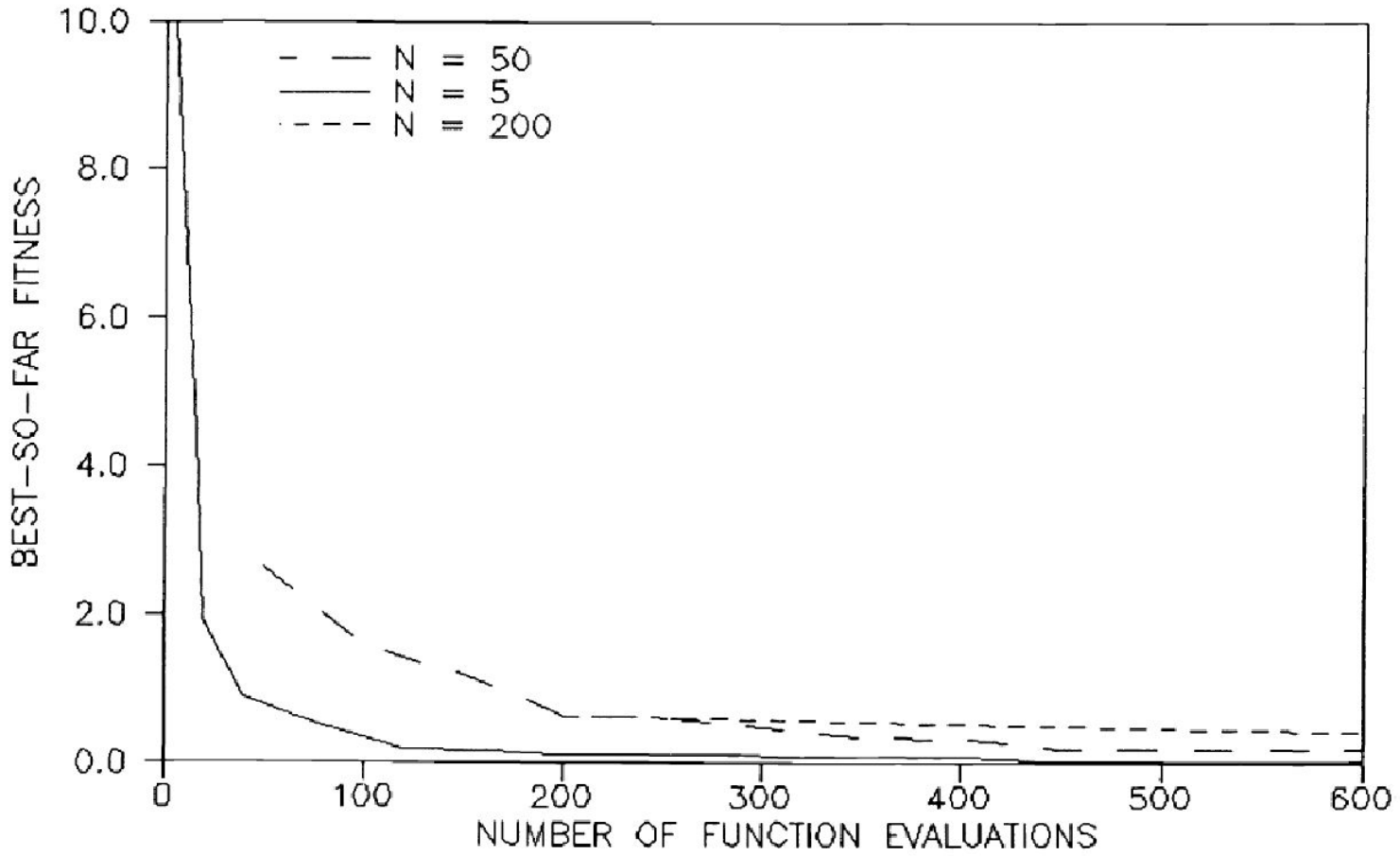


Fig1. Média das melhores soluções obtidas para J1
(adaptada de Krishnakumar, 1989)

Resultados - Krishnakumar (1989)

- Função Multi-modal
  - $J2=\left(0.002+\left(\sum_{j=1}^{25}(f_j(x))^{-1}\right)\right)^{-1}$, $f_j(x)=j+\sum_{i=1}^{2}\left(x_i-a_{ij}\right)^6$, $-65.536 \leq x_i \leq 65.536$

- O *simple (standard) genetic algorithm* (SGA) foi executado agora com populações com tamanho N = 50

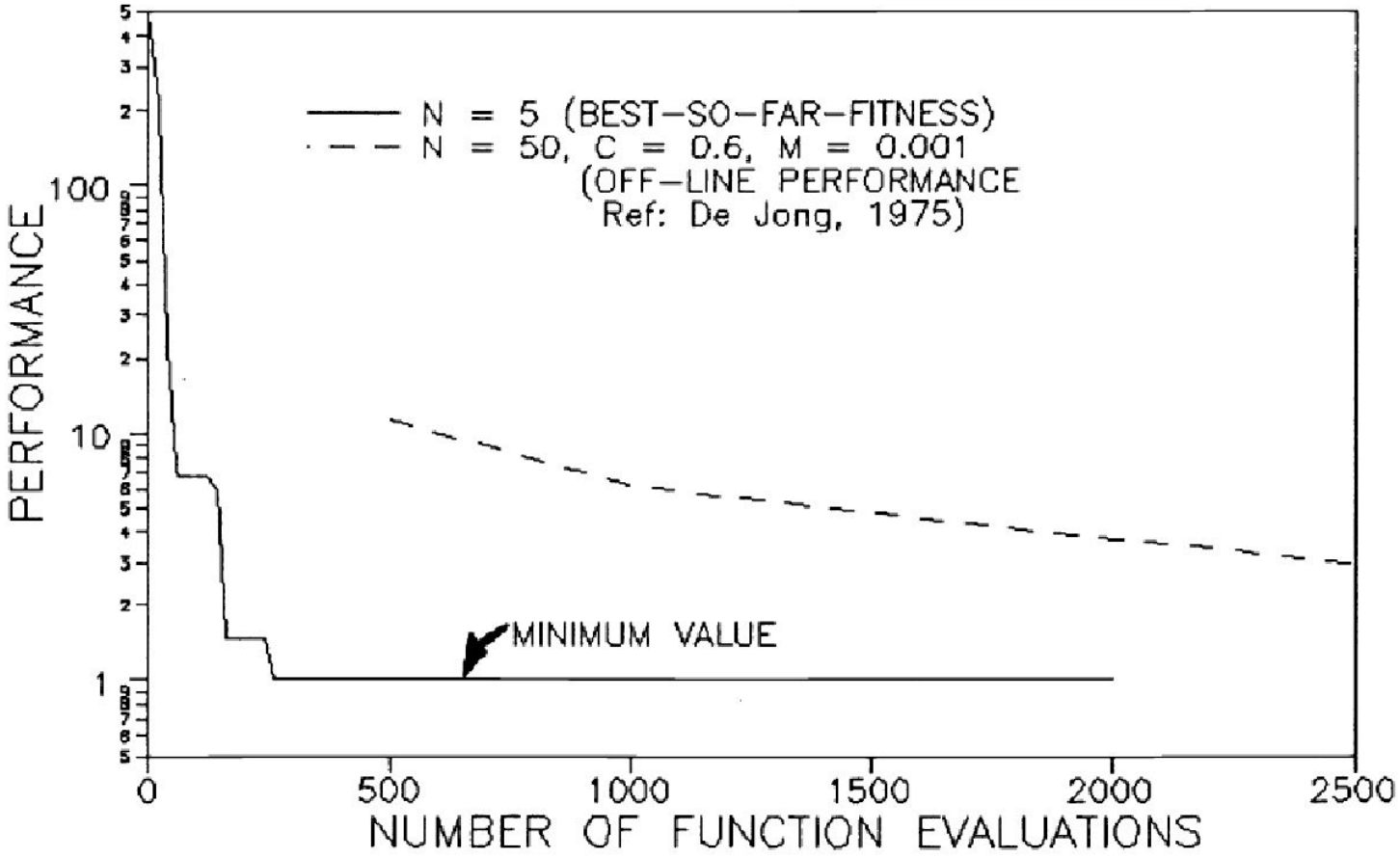


Fig1. Média das melhores soluções obtidas para J2 (adaptada de Krishnakumar, 1989)

Referências

- Coello Coello, C. A., & Pulido, G. T. (2001). A micro-genetic algorithm for multiobjective optimisation. Proceedings of the Genetic and Evolutionary Computation Conference. Springer-Verlag, pp. 126–140.

- Goldberg, David E. (1989). Sizing populations for serial and parallel genetic algorithms. In: Proceedings of the Third International Conference on genetic algorithms, San Mateo, California, Vol. 1196, pp. 70–79.

- Krishnakumar, K. (1989). Micro-genetic algorithms for stationary and nonstationary function optimisation. In: SPIE: Intelligent control and adaptive systems, Philadelphia, PA, Vol. 1196, pp. 289–296.