

Inteligência Artificial – ACH2016

Aula 05 – Algoritmos Genéticos e Programação Genética

Norton Trevisan Roman
(norton@usp.br)

14 de março de 2019

Algoritmos Genéticos

Voltemos ao algoritmo...

Função *GENÉTICO*(*população*, *FITNESS*): **indivíduo**

repita

nova_população $\leftarrow \emptyset$

para *i* = 1 até *TAMANHO*(*população*) **faça**

x \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

y \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

filho \leftarrow CRUZA(*x*, *y*)

se *CHANCE_MUTAÇÃO*() **então**

filho \leftarrow MUTA(*filho*)

 Adicione *filho* a *nova_população*

população \leftarrow *nova_população*

até algum indivíduo ser apto o suficiente, ou iterações suficientes terem passado

retorna o melhor indivíduo na população, de acordo com *FITNESS*

Algoritmos Genéticos

Voltemos ao algoritmo...

Função *GENÉTICO*(*população*, *FITNESS*): **indivíduo**

repita

nova_população $\leftarrow \emptyset$

para *i* = 1 até *TAMANHO*(*população*) **faça**

x \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

y \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

filho \leftarrow CRUZA(*x*, *y*)

se *CHANCE_MUTAÇÃO*() **então**

filho \leftarrow MUTA(*filho*)

 Adicione *filho* a *nova_população*

população \leftarrow *nova_população*

até algum indivíduo ser apto o suficiente, ou iterações suficientes terem passado

retorna o melhor indivíduo na população, de acordo com *FITNESS*

Já vimos a Seleção
e o Cruzamento



Algoritmos Genéticos

Voltemos ao algoritmo...

Função *GENÉTICO*(*população*, *FITNESS*): **indivíduo**

repita

nova_população $\leftarrow \emptyset$

para *i* = 1 até *TAMANHO*(*população*) **faça**

x \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

y \leftarrow SELEÇÃO_ALEATÓRIA(*população*, *FITNESS*)

filho \leftarrow CRUZA(*x*, *y*)

se *CHANCE_MUTAÇÃO()* **então**

filho \leftarrow *MUTA*(*filho*)

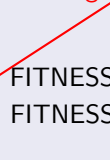
 Adicione *filho* a *nova_população*

população \leftarrow *nova_população*

até algum indivíduo ser apto o suficiente, ou iterações suficientes terem passado

retorna o melhor indivíduo na população, de acordo com *FITNESS*

Falta agora a Mutação



Mutação: Algoritmo

- Assim como o cruzamento, também a mutação ocorre com uma certa probabilidade

Mutação: Algoritmo

- Assim como o cruzamento, também a mutação ocorre com uma certa probabilidade
- Em geral muito menor que a de cruzamento, refletindo a baixa probabilidade de mutação na natureza

Mutação: Algoritmo

- Assim como o cruzamento, também a mutação ocorre com uma certa probabilidade
 - Em geral muito menor que a de cruzamento, refletindo a baixa probabilidade de mutação na natureza
 - Tipicamente entre 0.001 e 0.01

Algoritmos Genéticos – Operadores

Mutação: Algoritmo

- Assim como o cruzamento, também a mutação ocorre com uma certa probabilidade
- Em geral muito menor que a de cruzamento, refletindo a baixa probabilidade de mutação na natureza
- Tipicamente entre 0.001 e 0.01

Função *CHANCE_MUTAÇÃO()*: **boolean**

$m \leftarrow$ probabilidade independente pequena, pré-definida, de haver mutação

retorna *Valor escolhido aleatoriamente, dentre $\{V, F\}$, onde V tem $m\%$ de chance de ser escolhido*

Algoritmos Genéticos – Operadores

Mutação: Algoritmo

Função *MUTA*(*x*): **void**

posição \leftarrow Escolha aleatoriamente uma posição do arranjo *x*
 /* Modifica o valor escolhido */

se *x*[*posição*] *for binário* **então**

 /* inverte o bit */

x[*posição*] $\leftarrow \sim x[\textit{posição}]$

senão

x[*posição*] \leftarrow escolha aleatoriamente um novo valor (dentre os possíveis)

Algoritmos Genéticos – Operadores

Mutação

- Como vimos, a mutação produz uma prole a partir de um único pai

111010001000 \longrightarrow 11101011000
Fonte: ML. Mitchell.

Algoritmos Genéticos – Operadores

Mutação

- Como vimos, a mutação produz uma prole a partir de um único pai

111010001000 \longrightarrow 11101011000
Fonte: ML. Mitchell.

- Produz mudanças aleatórias pequenas

Mutação

- Como vimos, a mutação produz uma prole a partir de um único pai

111010001000 \longrightarrow 11101011000
Fonte: ML. Mitchell.

- Produz mudanças aleatórias pequenas
 - Escolhe aleatoriamente um ou mais genes, e então muda seu(s) valor(es)

Mutação

- Como vimos, a mutação produz uma prole a partir de um único pai

111010001000 \longrightarrow 11101011000
Fonte: ML. Mitchell.

- Produz mudanças aleatórias pequenas
 - Escolhe aleatoriamente um ou mais genes, e então muda seu(s) valor(es)
 - Seu papel é evitar que o algoritmo fique preso em algum ótimo local

Algoritmos Genéticos – Representação

Como representar um cromossomo?

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
- A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
- A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim
- As mais comuns são:

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
 - A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim
- As mais comuns são:
 - Binária

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
 - A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim
- As mais comuns são:
 - Binária
 - De valores inteiros

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
 - A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim
- As mais comuns são:
 - Binária
 - De valores inteiros
 - De valores reais

Algoritmos Genéticos – Representação

Como representar um cromossomo?

- Trata-se de associar uma cadeia de símbolos a cada solução possível representada pelo cromossomo
 - A representação de um cromossomo é bastante específica ao problema, e representações ruins podem levar a um desempenho ruim
- As mais comuns são:
 - Binária
 - De valores inteiros
 - De valores reais
 - Baseada na ordem

Algoritmos Genéticos – Representação

Representação Binária

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações
- O cromossomo consiste de uma sequência de bits

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações
- O cromossomo consiste de uma sequência de bits

1	0	0	1	0	1	1	0
---	---	---	---	---	---	---	---

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações
- O cromossomo consiste de uma sequência de bits

1	0	0	1	0	1	1	0
---	---	---	---	---	---	---	---

- Ideal para representar problemas cujo espaço de soluções é composto por variáveis booleanas

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações
- O cromossomo consiste de uma sequência de bits

1	0	0	1	0	1	1	0
---	---	---	---	---	---	---	---

- Ideal para representar problemas cujo espaço de soluções é composto por variáveis booleanas
- Pode também ser usada para representar qualquer valor numérico também

Algoritmos Genéticos – Representação

Representação Binária

- A mais simples e mais comum das representações
- O cromossomo consiste de uma sequência de bits

1	0	0	1	0	1	1	0
---	---	---	---	---	---	---	---

- Ideal para representar problemas cujo espaço de soluções é composto por variáveis booleanas
- Pode também ser usada para representar qualquer valor numérico também
 - Usando sua representação binária

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo?

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo? Com 4 bits

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo? Com 4 bits
 - Todo valor entre 0 e 15 pode ser escrito assim

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo? Com 4 bits
 - Todo valor entre 0 e 15 pode ser escrito assim

Ex:

0	1	1	0
---	---	---	---

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo? Com 4 bits
 - Todo valor entre 0 e 15 pode ser escrito assim

Ex:

0	1	1	0
---	---	---	---

- E a função objetivo?

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
- Como representar o cromossomo? Com 4 bits
 - Todo valor entre 0 e 15 pode ser escrito assim
Ex:

0	1	1	0
---	---	---	---
- E a função objetivo?
 - Será a própria $f(x) = 15x - x^2$ (queremos seu máximo)

Representação Binária: Exemplo

- Queremos o valor máximo de $f(x) = 15x - x^2$, onde x é inteiro e $0 \leq x \leq 15$
 - Como representar o cromossomo? Com 4 bits
 - Todo valor entre 0 e 15 pode ser escrito assim
- Ex:

0	1	1	0
---	---	---	---
- E a função objetivo?
 - Será a própria $f(x) = 15x - x^2$ (queremos seu máximo)
 - Seguimos então o procedimento já visto...

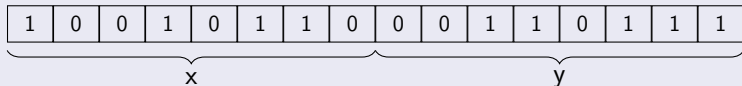
Representação Binária: Exemplo

- E se agora quisermos o máximo de $f(x, y) = (1 - x)^2 e^{-x^2 - (y+1)^2}$, com $-3 \leq x, y \leq 3$?

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

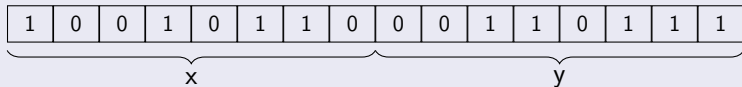
- E se agora quisermos o máximo de $f(x, y) = (1 - x)^2 e^{-x^2 - (y+1)^2}$, com $-3 \leq x, y \leq 3$?
- Concatenamos as representações (8 bits) de x e y



Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se agora quisermos o máximo de $f(x, y) = (1 - x)^2 e^{-x^2 - (y+1)^2}$, com $-3 \leq x, y \leq 3$?
- Concatenamos as representações (8 bits) de x e y

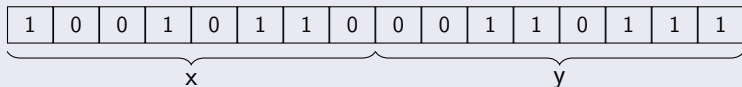


- E a função objetivo?

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se agora quisermos o máximo de $f(x, y) = (1 - x)^2 e^{-x^2 - (y+1)^2}$, com $-3 \leq x, y \leq 3$?
- Concatenamos as representações (8 bits) de x e y

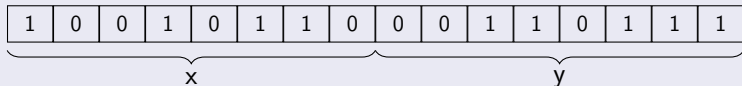


- E a função objetivo?
 - Dado um cromossomo, primeiro decodificamos x e y

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se agora quisermos o máximo de $f(x, y) = (1 - x)^2 e^{-x^2 - (y+1)^2}$, com $-3 \leq x, y \leq 3$?
- Concatenamos as representações (8 bits) de x e y



- E a função objetivo?
 - Dado um cromossomo, primeiro decodificamos x e y
 - E então os usamos em $f(x, y)$

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se x e y forem números reais?

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se x e y forem números reais?
- Mapeamos a esses 8 bits:

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- E se x e y forem números reais?
- Mapeamos a esses 8 bits:
 - Com 8 bits representamos inteiros de 0 a 255

Representação Binária: Exemplo

- E se x e y forem números reais?
- Mapeamos a esses 8 bits:
 - Com 8 bits representamos inteiros de 0 a 255
 - A menor fração do intervalo $[-3, 3]$ representado por esses bits é $\frac{3-(-3)}{255-0} = \frac{6}{255} = 0.0235294$

Representação Binária: Exemplo

- E se x e y forem números reais?
- Mapeamos a esses 8 bits:
 - Com 8 bits representamos inteiros de 0 a 255
 - A menor fração do intervalo $[-3, 3]$ representado por esses bits é $\frac{3-(-3)}{255-0} = \frac{6}{255} = 0.0235294$
 - O inteiro contido nos 8 bits é então multiplicado por 0.0235294 (caindo no intervalo $[0, 6]$)

Representação Binária: Exemplo

- E se x e y forem números reais?
- Mapeamos a esses 8 bits:
 - Com 8 bits representamos inteiros de 0 a 255
 - A menor fração do intervalo $[-3, 3]$ representado por esses bits é $\frac{3 - (-3)}{255 - 0} = \frac{6}{255} = 0.0235294$
 - O inteiro contido nos 8 bits é então multiplicado por 0.0235294 (caindo no intervalo $[0, 6]$)
 - Subtraímos então 3 do resultado, deslocando o intervalo para $[-3, 3]$

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

- Fazemos então $138 \times 0.0235294 = 3.2470572$

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

- Fazemos então $138 \times 0.0235294 = 3.2470572$

- E o valor representado será $3.2470572 - 3 = 0.2470572$

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

- Fazemos então $138 \times 0.0235294 = 3.2470572$

- E o valor representado será $3.2470572 - 3 = 0.2470572$

- 0.2470572 é o valor a ser usado na função objetivo

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

- Fazemos então $138 \times 0.0235294 = 3.2470572$

- E o valor representado será $3.2470572 - 3 = 0.2470572$

- 0.2470572 é o valor a ser usado na função objetivo

- Mas booleanos, inteiros e reais não são as únicas coisas que podemos representar com binários

Algoritmos Genéticos – Representação

Representação Binária: Exemplo

- Ex:

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|

- O inteiro correspondente será 138

- Fazemos então $138 \times 0.0235294 = 3.2470572$

- E o valor representado será $3.2470572 - 3 = 0.2470572$

- 0.2470572 é o valor a ser usado na função objetivo

- Mas booleanos, inteiros e reais não são as únicas coisas que podemos representar com binários

- Restrições e Regras também

Representação Binária: Restrições e Regras

- Restrições servem para limitar os possíveis valores de uma variável

Representação Binária: Restrições e Regras

- Restrições servem para limitar os possíveis valores de uma variável
 - Ex: $-3 \leq x, y \leq 3$

Representação Binária: Restrições e Regras

- Restrições servem para limitar os possíveis valores de uma variável
 - Ex: $-3 \leq x, y \leq 3$
- Na representação binária, podemos fazer com que cada restrição corresponda a uma cadeia específica

Representação Binária: Restrições e Regras

- Restrições servem para limitar os possíveis valores de uma variável
 - Ex: $-3 \leq x, y \leq 3$
- Na representação binária, podemos fazer com que cada restrição corresponda a uma cadeia específica
- Ex:
 - Atributo: tempo

Representado com 3 bits:	100	tempo = ensolarado
	010	tempo = nublado
	001	tempo = chuvoso

Representação Binária: Restrições e Regras

- Podemos combinar restrições com OU (\vee):

101	tempo = ensolarado \vee chuvoso
011	tempo = nublado \vee chuvoso
111	tempo = qualquer tempo

Representação Binária: Restrições e Regras

- Podemos combinar restrições com OU (\vee):

101		tempo = ensolarado \vee chuvoso
011		tempo = nublado \vee chuvoso
111		tempo = qualquer tempo

- Atributo: vento

Representado com 2 bits:	10		vento = forte
	01		vento = fraco

Representação Binária: Restrições e Regras

- Podemos combinar restrições com OU (\vee):

101		tempo = ensolarado \vee chuvoso
011		tempo = nublado \vee chuvoso
111		tempo = qualquer tempo

- Atributo: vento

Representado com 2 bits:

10		vento = forte
01		vento = fraco

- Conjunções de restrições em atributos múltiplos pode ser representadas concatenando-se as cadeias de bits:

01110 | (tempo = nublado \vee chuvoso) \wedge (vento = forte)

Representação Binária: Restrições e Regras

- Regras também podem ser representadas assim

Representação Binária: Restrições e Regras

- Regras também podem ser representadas assim
 - Se vento = forte, então viagem = sim

1111010 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim)

Representação Binária: Restrições e Regras

- Regras também podem ser representadas assim

- Se vento = forte, então viagem = sim

$$1111010 \mid \left(\begin{array}{l} \text{tempo} = \text{ensolarado} \vee \text{nublado} \vee \text{chuvoso} \\ \wedge (\text{vento} = \text{forte}) \Rightarrow (\text{viagem} = \text{sim}) \end{array} \right)$$

- Note que a regra contém uma sub-cadeia para cada atributo, mesmo que um atributo não seja restringido por ela

$$1111010 \mid \left(\begin{array}{l} \text{tempo} = \text{não importa} \\ \wedge (\text{vento} = \text{forte}) \Rightarrow \\ (\text{viagem} = \text{sim}) \end{array} \right)$$

Representação Binária: Restrições e Regras

- Isso faz com que a regra tenha um tamanho fixo em bits

Representação Binária: Restrições e Regras

- Isso faz com que a regra tenha um tamanho fixo em bits
- Sendo que sub-cadeias em posições específicas restringem atributos específicos

1111010 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim)

Representação Binária: Restrições e Regras

- Isso faz com que a regra tenha um tamanho fixo em bits
- Sendo que sub-cadeias em posições específicas restringem atributos específicos

1111010 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim)

- Regras mais complexas são representadas de maneira similar, pela concatenação das representações de cada regra que as compõem

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$

Se $b = V$ então $d = F$ (não importa o valor de a)

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$

Se $b = V$ então $d = F$ (não importa o valor de a)

- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$

Se $b = V$ então $d = F$ (não importa o valor de a)

- Representação: 100111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100
 - Note que os consequentes têm um único bit

Representando regras complexas – Exemplo

- Se $a = V$ e $b = F$ então $c = V$
Se $b = V$ então $d = F$ (não importa o valor de a)
- Representação: 1001111100
 - Note que os consequentes têm um único bit
 - Se a cadeia de bits for de tamanho variável, como neste exemplo, onde ao aumentarmos o número de regras aumentaremos a cadeia, a função de cruzamento deve ser alterada para acomodar esse fato

Problemas com essa representação

- Considere a seguinte regra

1111011 |

Problemas com essa representação

- Considere a seguinte regra

$$\begin{array}{c|l} 1111011 & (\text{tempo} = \text{ensolarado} \vee \text{nublado} \vee \text{chuvoso}) \\ & \wedge (\text{vento} = \text{forte}) \Rightarrow (\text{viagem} = \text{sim} \vee \text{n\~ao}) \end{array}$$

Problemas com essa representação

- Considere a seguinte regra

$$\begin{array}{c|l} 1111011 & (\text{tempo} = \text{ensolarado} \vee \text{nublado} \vee \text{chuvoso}) \\ & \wedge (\text{vento} = \text{forte}) \Rightarrow (\text{viagem} = \text{sim} \vee \text{não}) \end{array}$$

- O 11 final torna a regra inútil (se o vento estiver forte, então a viagem ocorrerá ou não)

Problemas com essa representação

- Considere a seguinte regra

$$\begin{array}{c|l} 1111011 & (\text{tempo} = \text{ensolarado} \vee \text{nublado} \vee \text{chuvoso}) \\ & \wedge (\text{vento} = \text{forte}) \Rightarrow (\text{viagem} = \text{sim} \vee \text{não}) \end{array}$$

- O 11 final torna a regra inútil (se o vento estiver forte, então a viagem ocorrerá ou não)
- As regras devem ser bem pensadas, para evitar a representação de hipóteses indesejadas

Algoritmos Genéticos

Possíveis soluções

1111011 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim \vee não)

Possíveis soluções

1111011 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim \vee não)

- Alterar a representação da condição posterior

Possíveis soluções

1111011 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim \vee não)

- Alterar a representação da condição posterior
 - Ex: aloque um único bit para “viagem”

Possíveis soluções

1111011 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim \vee não)

- Alterar a representação da condição posterior
 - Ex: alocue um único bit para “viagem”
- Alterar os operadores genéticos para evitar essas cadeias

Possíveis soluções

1111011 | (tempo = ensolarado \vee nublado \vee chuvoso)
 \wedge (vento = forte) \Rightarrow (viagem = sim \vee não)

- Alterar a representação da condição posterior
 - Ex: aloque um único bit para “viagem”
- Alterar os operadores genéticos para evitar essas cadeias
- Dar a essa cadeia um valor de *fitness* muito pequeno

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
- Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
 - Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)
- Com valores discretos (inteiros ou simbólicos), o que fazer?

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
 - Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)
- Com valores discretos (inteiros ou simbólicos), o que fazer? Representá-los como inteiros

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
 - Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)
- Com valores discretos (inteiros ou simbólicos), o que fazer? Representá-los como inteiros

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 4 | 7 | 4 | 8 | 5 | 5 | 2 |
|---|---|---|---|---|---|---|---|

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
 - Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)
- Com valores discretos (inteiros ou simbólicos), o que fazer? Representá-los como inteiros

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 4 | 7 | 4 | 8 | 5 | 5 | 2 |
|---|---|---|---|---|---|---|---|

- | | | | | | | | |
|---|---|---|---|---|---|---|---|
| C | B | E | E | D | C | D | E |
|---|---|---|---|---|---|---|---|

Representação Inteira

- A representação binária, contudo, pode gerar cromossomos grandes
 - Especialmente quando tratamos de valores inteiros, reais ou simbólicos (ex: 'A', ' σ ' etc)
- Com valores discretos (inteiros ou simbólicos), o que fazer? Representá-los como inteiros
 - | | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 4 | 7 | 4 | 8 | 5 | 5 | 2 |
|---|---|---|---|---|---|---|---|
 - | | | | | | | | |
|---|---|---|---|---|---|---|---|
| C | B | E | E | D | C | D | E |
|---|---|---|---|---|---|---|---|
- Trata-se da representação que usamos até agora...

Representação Real

- Útil quando queremos definir os genes usando variáveis contínuas

Representação Real

- Útil quando queremos definir os genes usando variáveis contínuas
 - Os cromossomos são então cadeias de valores em um dado intervalo

Representação Real

- Útil quando queremos definir os genes usando variáveis contínuas
 - Os cromossomos são então cadeias de valores em um dado intervalo
- Operador: Cruzamento por média (de 2 para 1)

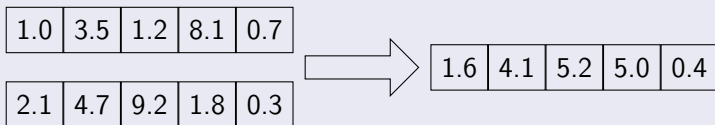
Representação Real

- Útil quando queremos definir os genes usando variáveis contínuas
 - Os cromossomos são então cadeias de valores em um dado intervalo
- Operador: Cruzamento por média (de 2 para 1)
 - Os genes do filho são resultado da média dos genes dos pais (em cada posição)

Algoritmos Genéticos

Representação Real

- Útil quando queremos definir os genes usando variáveis contínuas
 - Os cromossomos são então cadeias de valores em um dado intervalo
- Operador: Cruzamento por média (de 2 para 1)
 - Os genes do filho são resultado da média dos genes dos pais (em cada posição)



Representação Real

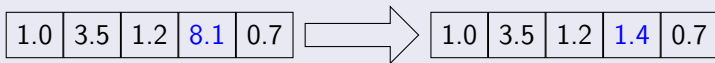
- Operador: Mutação aleatória

Representação Real

- Operador: Mutação aleatória
 - Um gene é substituído por um valor aleatório (num intervalo pré-definido)

Representação Real

- Operador: Mutação aleatória
 - Um gene é substituído por um valor aleatório (num intervalo pré-definido)



Representação Real

- Operador: Mutação aleatória
 - Um gene é substituído por um valor aleatório (num intervalo pré-definido)



- Operador: Mutação por incremento

Representação Real

- Operador: Mutação aleatória
 - Um gene é substituído por um valor aleatório (num intervalo pré-definido)



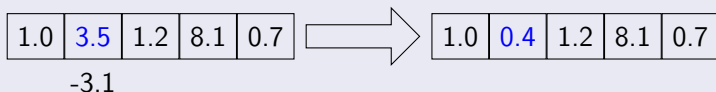
- Operador: Mutação por incremento
 - O gene é somado a um incremento aleatório

Representação Real

- Operador: Mutação aleatória
 - Um gene é substituído por um valor aleatório (num intervalo pré-definido)



- Operador: Mutação por incremento
 - O gene é somado a um incremento aleatório



Representação Real

- Outros tipos de operadores existem

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica
 - Mutação com outras distribuições de probabilidade

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica
 - Mutação com outras distribuições de probabilidade
 - etc

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica
 - Mutação com outras distribuições de probabilidade
 - etc
- A representação real também pode ser usada com inteiros

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica
 - Mutação com outras distribuições de probabilidade
 - etc
- A representação real também pode ser usada com inteiros
 - Nesse caso, a diferença está em como é feito o cruzamento e (potencialmente) a mutação

Representação Real

- Outros tipos de operadores existem
 - Cruzamento por média geométrica
 - Mutação com outras distribuições de probabilidade
 - etc
- A representação real também pode ser usada com inteiros
 - Nesse caso, a diferença está em como é feito o cruzamento e (potencialmente) a mutação
 - Quando usar qual representação dependerá do problema modelado

Algoritmos Genéticos

Representação Baseada na Ordem

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa
 - Ex: problema do caixeiro viajante

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa
 - Ex: problema do caixeiro viajante
- Operador: Cruzamento

5	2	1	3	6	4
---	---	---	---	---	---

1	4	5	6	2	3
---	---	---	---	---	---

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa
 - Ex: problema do caixeiro viajante
- Operador: Cruzamento

5	2	1	3	6	4
---	---	---	---	---	---

1	4	5	6	2	3
---	---	---	---	---	---

Note que, por ser um problema combinatório, os **valores** nos genes não mudam, apenas sua **ordem**

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa
 - Ex: problema do caixeiro viajante

- Operador: Cruzamento

5	2	1	3	6	4
---	---	---	---	---	---

1	4	5	6	2	3
---	---	---	---	---	---

- Inicialmente criamos uma máscara que diz quais genes mantemos e quais terão a ordem alterada (amarelo)

Representação Baseada na Ordem

- Usada para problemas de origem combinatória, onde a ordem importa
- Ex: problema do caixeiro viajante

- Operador: Cruzamento

5	2	1	3	6	4
---	---	---	---	---	---

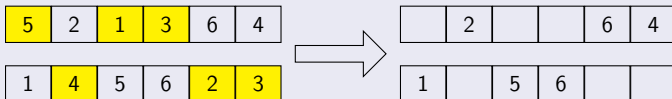
1	4	5	6	2	3
---	---	---	---	---	---

Note também que os genes mantidos num cromossomo são alterados no outro e vice-versa

- Inicialmente criamos uma máscara que diz quais genes mantemos e quais terão a ordem alterada (amarelo)

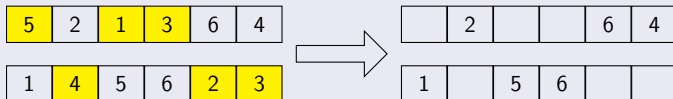
Algoritmos Genéticos

Representação Baseada na Ordem



Algoritmos Genéticos

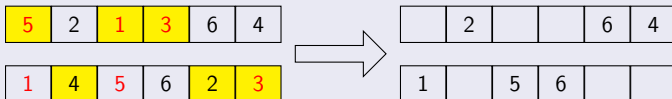
Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo

Algoritmos Genéticos

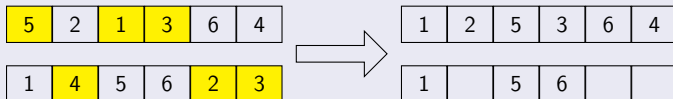
Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo
- Os genes 5, 1 e 3 do 1º cromossomo aparecem na ordem 1, 5, 3 no 2º.

Algoritmos Genéticos

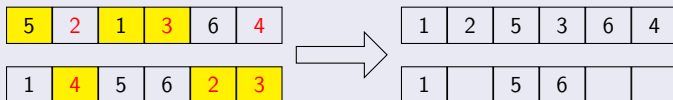
Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo
- Os genes 5, 1 e 3 do 1º cromossomo aparecem na ordem 1, 5, 3 no 2º. Então o colocamos nessa ordem no 1º cromossomo

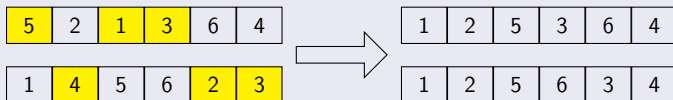
Algoritmos Genéticos

Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo
- Os genes 5, 1 e 3 do 1º cromossomo aparecem na ordem 1, 5, 3 no 2º. Então o colocamos nessa ordem no 1º cromossomo
- Já os genes 4, 2, 3 do 2º cromossomo aparecem na ordem 2, 3, 4 no 1º.

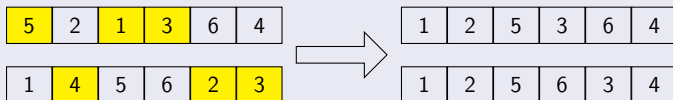
Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo
- Os genes 5, 1 e 3 do 1º cromossomo aparecem na ordem 1, 5, 3 no 2º. Então o colocamos nessa ordem no 1º cromossomo
- Já os genes 4, 2, 3 do 2º cromossomo aparecem na ordem 2, 3, 4 no 1º. Então o colocamos nessa ordem no 2º cromossomo

Algoritmos Genéticos

Representação Baseada na Ordem



- Os genes a serem alterados são permutados de forma a ficarem na ordem em que aparecem no outro cromossomo
- Os genes 5, 1 e 3 do 1º cromossomo aparecem na ordem 1, 5, 3 no 2º. Então o colocamos nessa ordem no 1º cromossomo
- Já os genes 4, 2, 3 do 2º cromossomo aparecem na ordem 2, 3, 4 no 1º. Então o colocamos nessa ordem no 2º cromossomo
- Note, mais uma vez, que só permutamos, não mudamos, valores

Representação Baseada na Ordem

- Operador: Mutação

5	2	1	3	6	4
---	---	---	---	---	---

Representação Baseada na Ordem

- Operador: Mutação

5	2	1	3	6	4
---	---	---	---	---	---

- Escolhemos aleatoriamente posições (e o número de posições) a sofrerem mutação

Representação Baseada na Ordem

- Operador: Mutação



- Escolhemos aleatoriamente posições (e o número de posições) a sofrerem mutação
- Embaralhamos (aleatoriamente) então as posições escolhidas

Algoritmos Genéticos

Paralelizando Algoritmos Genéticos

Algoritmos Genéticos

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos

Algoritmos Genéticos

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos
 - Cada sub-grupo evolui separadamente

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos
 - Cada sub-grupo evolui separadamente
- Cruzamento e comunicação entre grupos são menos freqüentes que intra-grupo

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos
 - Cada sub-grupo evolui separadamente
- Cruzamento e comunicação entre grupos são menos freqüentes que intra-grupo
- Transferências entre grupos ocorrem via **migração**:

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos
 - Cada sub-grupo evolui separadamente
- Cruzamento e comunicação entre grupos são menos freqüentes que intra-grupo
- Transferências entre grupos ocorrem via **migração**:
 - Indivíduos de um grupo são copiados ou transferidos a outro, de tempos em tempos

Algoritmos Genéticos

Paralelizando Algoritmos Genéticos

- Subdivide a população em subgrupos
 - Cada sub-grupo evolui separadamente
- Cruzamento e comunicação entre grupos são menos freqüentes que intra-grupo
- Transferências entre grupos ocorrem via **migração**:
 - Indivíduos de um grupo são copiados ou transferidos a outro, de tempos em tempos
- Reduz *crowding* (veremos mais adiante...)

Algoritmos Genéticos

Em suma

Em suma

- Algoritmos genéticos geram hipóteses pela mutação repetitiva e recombinação de partes das melhores hipóteses

Em suma

- Algoritmos genéticos geram hipóteses pela mutação repetitiva e recombinação de partes das melhores hipóteses
- A cada passo, a população é atualizada, trocando-se parte dela pela prole gerada

Algoritmos Genéticos

Vantagens

Vantagens

- Habilidade de combinar grandes porções de genes que evoluíram independentemente

Vantagens

- Habilidade de combinar grandes porções de genes que evoluíram independentemente
- Podemos abordar problemas de difícil modelagem

Vantagens

- Habilidade de combinar grandes porções de genes que evoluíram independentemente
- Podemos abordar problemas de difícil modelagem
- Facilmente tornados paralelos

Vantagens

- Habilidade de combinar grandes porções de genes que evoluíram independentemente
- Podemos abordar problemas de difícil modelagem
- Facilmente tornados paralelos
- Dificilmente caem em mínimos locais, por darem pulos aleatórios no espaço de hipóteses

Desvantagens

Desvantagens

- Seu sucesso requer um trabalho cuidadoso da representação do problema

Desvantagens

- Seu sucesso requer um trabalho cuidadoso da representação do problema
- *Crowding*

Desvantagens

- Seu sucesso requer um trabalho cuidadoso da representação do problema
- *Crowding*
 - Quando um indivíduo com fitness maior que os outros rapidamente se reproduz

Desvantagens

- Seu sucesso requer um trabalho cuidadoso da representação do problema
- *Crowding*
 - Quando um indivíduo com fitness maior que os outros rapidamente se reproduz
 - Cópias dele e de indivíduos muito similares formam uma grande fração da população

Desvantagens

- Seu sucesso requer um trabalho cuidadoso da representação do problema
- *Crowding*
 - Quando um indivíduo com fitness maior que os outros rapidamente se reproduz
 - Cópias dele e de indivíduos muito similares formam uma grande fração da população
 - Reduz a diversidade da população, deixando o progresso do algoritmo mais lento

Algoritmos Genéticos

Estratégias contra *crowding*

Estratégias contra *crowding*

- *Fitness sharing*:

Estratégias contra *crowding*

- *Fitness sharing*:
 - Reduzir o valor de adaptação do indivíduo pela presença de outros similares na população

Estratégias contra *crowding*

- *Fitness sharing*:
 - Reduzir o valor de adaptação do indivíduo pela presença de outros similares na população
- Restringir os tipos de indivíduos que podem se recombinar para formar a prole

Estratégias contra *crowding*

- *Fitness sharing*:
 - Reduzir o valor de adaptação do indivíduo pela presença de outros similares na população
- Restringir os tipos de indivíduos que podem se recombinar para formar a prole
 - Fazer com que somente os indivíduos mais similares se recombinem, formando grupos de indivíduos similares (múltiplas sub-espécies da população)

Estratégias contra *crowding*

- *Fitness sharing*:
 - Reduzir o valor de adaptação do indivíduo pela presença de outros similares na população
- Restringir os tipos de indivíduos que podem se recombinar para formar a prole
 - Fazer com que somente os indivíduos mais similares se recombinem, formando grupos de indivíduos similares (múltiplas sub-espécies da população)
 - Distribuir espacialmente os indivíduos e permitir que somente os que estiverem mais próximos se recombinem.

Estratégias contra *crowding*

- Mudar a função de seleção para usar posição em lista, e não o mais adaptado

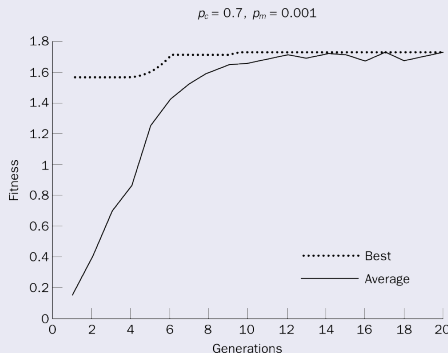
Estratégias contra *crowding*

- Mudar a função de seleção para usar posição em lista, e não o mais adaptado
- Embora os melhores ainda tenham maior probabilidade, esta é melhor distribuída, por depender da posição, e não do valor de *fitness*

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração

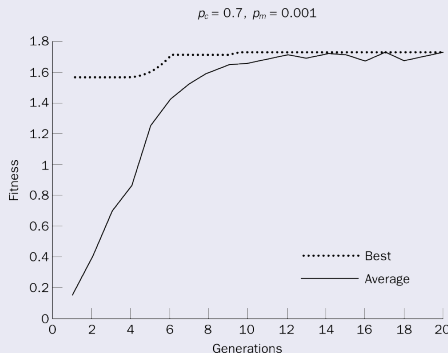


Fonte: Al. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população

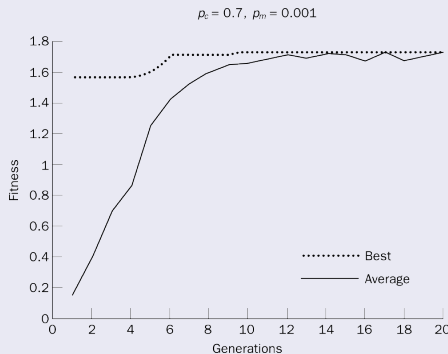


Fonte: AI. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo

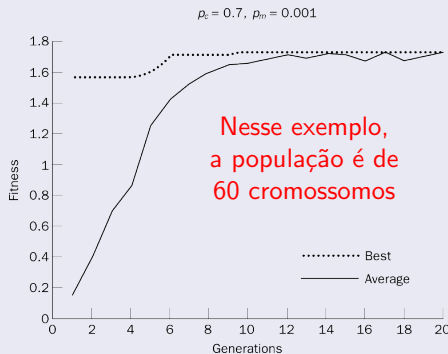


Fonte: A.I. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo

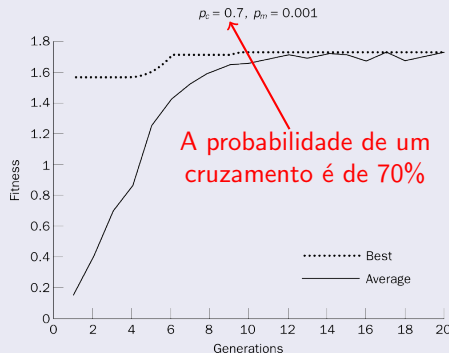


Fonte: A.I. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo

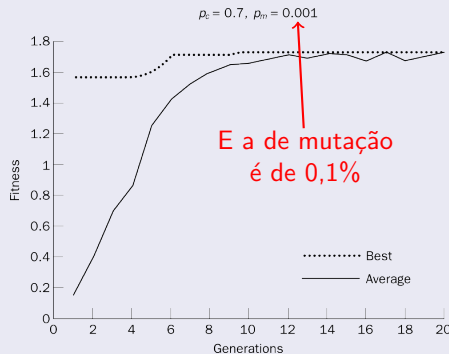


Fonte: A.I. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo

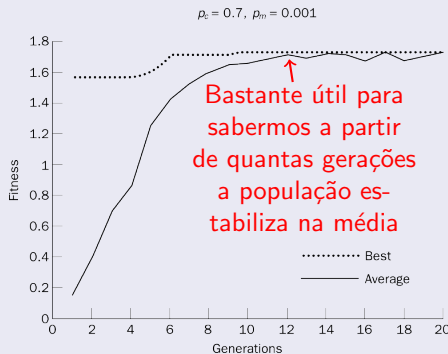


Fonte: AI. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo

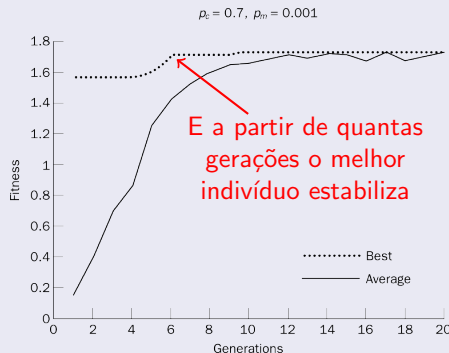


Fonte: AI. Negnevitsky.

Algoritmos Genéticos

Gráfico de Desempenho

- Curva que mostra o desempenho médio da população a cada geração
- Acompanhada de uma mostrando o desempenho do melhor indivíduo da população
- Medidos conforme a função objetivo



Fonte: AI. Negnevitsky.

Programação Genética

Programação Genética

- Técnica de computação evolutiva em que a população é composta de programas ou expressões, em vez de valores

Programação Genética

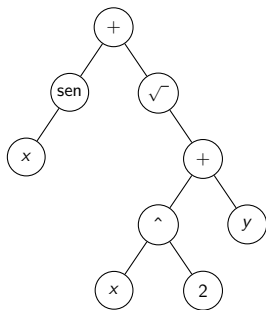
- Técnica de computação evolutiva em que a população é composta de programas ou expressões, em vez de valores
 - Podem evoluir programas completos

Programação Genética

- Técnica de computação evolutiva em que a população é composta de programas ou expressões, em vez de valores
 - Podem evoluir programas completos
- Representa as expressões ou programas por sua árvore sintática

Programação Genética

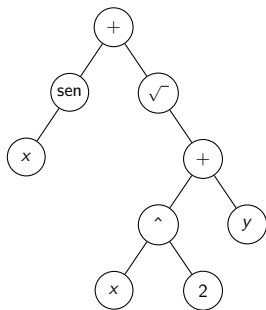
- Técnica de computação evolutiva em que a população é composta de programas ou expressões, em vez de valores
- Podem evoluir programas completos
- Representa as expressões ou programas por sua árvore sintática
 - Ex: $\text{sen}(x) + \sqrt{x^2 + y}$



Fonte: ML. Mitchell.

Programação Genética

- Técnica de computação evolutiva em que a população é composta de programas ou expressões, em vez de valores
 - Podem evoluir programas completos
- Representa as expressões ou programas por sua árvore sintática
 - Ex: $\text{sen}(x) + \sqrt{x^2 + y}$
- Bastante usada em projetos de circuitos



Fonte: ML. Mitchell.

Programação Genética

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas

Programação Genética

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc

Programação Genética

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc
- Definimos então os símbolos terminais

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc
- Definimos então os símbolos terminais
 - Ex: x , y , 3, 2 etc

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc
- Definimos então os símbolos terminais
 - Ex: x , y , 3, 2 etc
- E a função de adaptação (fitness)

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc
- Definimos então os símbolos terminais
 - Ex: x , y , 3, 2 etc
- É a função de adaptação (fitness)
 - É ela que determina quão bem o algoritmo é capaz de resolver o problema

Programação Genética

Uso como Aprendizagem de Máquina

- Começamos definindo as funções primitivas
 - Ex: Seno, raiz etc
- Definimos então os símbolos terminais
 - Ex: x , y , 3, 2 etc
- É a função de adaptação (fitness)
 - É ela que determina quão bem o algoritmo é capaz de resolver o problema
 - Tipicamente obtida pela execução do algoritmo em um conjunto de dados de treino

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino
 - O tempo de treinamento depende das primitivas escolhidas, dos exemplos de treino, e da função de adaptação

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino
 - O tempo de treinamento depende das primitivas escolhidas, dos exemplos de treino, e da função de adaptação
- E o algoritmo evolutivo a ser usado
 - Pode ser o mesmo usado em algoritmos genéticos

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino
 - O tempo de treinamento depende das primitivas escolhidas, dos exemplos de treino, e da função de adaptação
- E o algoritmo evolutivo a ser usado
 - Pode ser o mesmo usado em algoritmos genéticos
 - Mesmo modo de calcular a probabilidade

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino
 - O tempo de treinamento depende das primitivas escolhidas, dos exemplos de treino, e da função de adaptação
- E o algoritmo evolutivo a ser usado
 - Pode ser o mesmo usado em algoritmos genéticos
 - Mesmo modo de calcular a probabilidade
 - Mesmo esquema de cruzamento

Uso como Aprendizagem de Máquina

- Definimos os exemplos de treino
 - O tempo de treinamento depende das primitivas escolhidas, dos exemplos de treino, e da função de adaptação
- E o algoritmo evolutivo a ser usado
 - Pode ser o mesmo usado em algoritmos genéticos
 - Mesmo modo de calcular a probabilidade
 - Mesmo esquema de cruzamento
 - Pode ou não haver mutações

Programação Genética

Algoritmo

população \leftarrow População inicial de árvores sintáticas aleatórias

repita

para cada árvore $A_i \in$ *população* **faça**

$fitness_i \leftarrow$ valor indicando quão bem o algoritmo representado por A_i
 resolveu o problema

população \leftarrow GERA_POP(*população*, $\{fitness_i\}$)

até até algum A_i atingir um erro aceitável

retorna A_i escolhido

Programação Genética

Algoritmo

população \leftarrow População inicial de árvores sintáticas aleatórias

repita

para cada árvore $A_i \in$ *população* **faça**

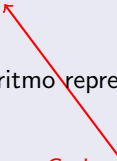
$fitness_i \leftarrow$ valor indicando quão bem o algoritmo representado por A_i
 resolveu o problema

população \leftarrow GERA_POP(*população*, $\{fitness_i\}$)

até algum A_i atingir um erro aceitável

retorna A_i escolhido

Cada árvore sintática
é uma composição
aleatória de funções
e símbolos terminais



Programação Genética

Algoritmo

população \leftarrow População inicial de árvores sintáticas aleatórias

repita

para cada *árvore* $A_i \in$ *população* **faça**

fitness_i \leftarrow valor indicando quão bem o algoritmo representado por A_i
 resolveu o problema

população \leftarrow GERA_POP(*população*, {*fitness_i*})

até *até* algum A_i atingir um erro aceitável

retorna A_i escolhido

Executa cada algoritmo da população e associa a ele um valor de adaptação (*fitness*), indicando quão bem ele foi nos dados de treino

Programação Genética

Algoritmo

população \leftarrow População inicial de árvores sintáticas aleatórias

repita

para cada árvore $A_i \in$ *população* **faça**

$fitness_i \leftarrow$ valor indicando quão bem o algoritmo representado por A_i
 resolveu o problema

população \leftarrow GERA_POP(*população*, { $fitness_i$ })

até algum A_i atingir um erro aceitável

retorna A_i escolhido

← Cria uma nova população (nova geração)

Função GERA_POP(*população*, { $fitness_i$ }): **população**

 Copie os melhores programas da geração atual (clonagem)

 Selecione indivíduos para o cruzamento

 Crie novos programas pelo cruzamento dos programas selecionados

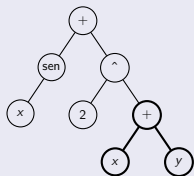
 Aplique a mutação sobre algumas funções escolhidas aleatoriamente

Programação Genética

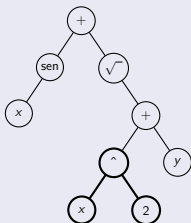
Cruzamentos

- Um filho é criado selecionando-se aleatoriamente um ramo de cada pai

Pais



$$\sin(x) + 2^{x+y}$$



$$\sin(x) + \sqrt{x^2 + y}$$

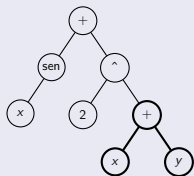
Fonte: ML. Mitchell.

Programação Genética

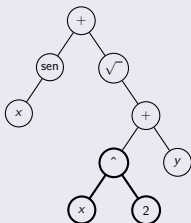
Cruzamentos

- Um filho é criado selecionando-se aleatoriamente um ramo de cada pai, e então fazendo-se o intercâmbio dos ramos selecionados

Pais

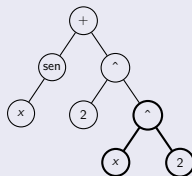


$$\text{sen}(x) + 2^{x+y}$$

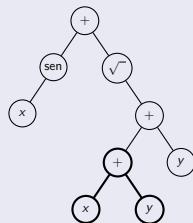


$$\sin(x) + \sqrt{x^2 + y}$$

Filhos



$$\text{sen}(x) + 2^{x^2}$$



$$\text{sen}(x) + \sqrt{x + y + y}$$

Fonte: ML. Mitchell.

Cruzamentos

- E se os pais forem iguais?

Cruzamentos

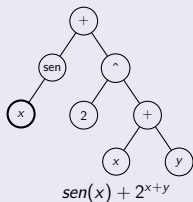
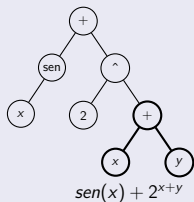
- E se os pais forem iguais?
 - É possível que surja uma prole diferente

Programação Genética

Cruzamentos

- E se os pais forem iguais?
- É possível que surja uma prole diferente

Pais



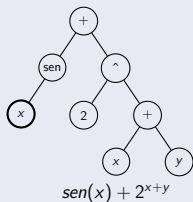
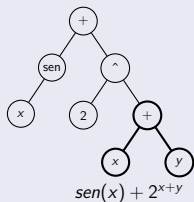
Fonte: Adaptado de ML. Mitchell.

Programação Genética

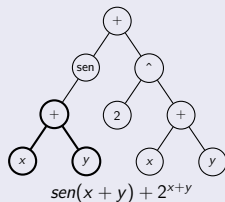
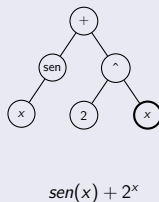
Cruzamentos

- E se os pais forem iguais?
- É possível que surja uma prole diferente

Pais



Filhos

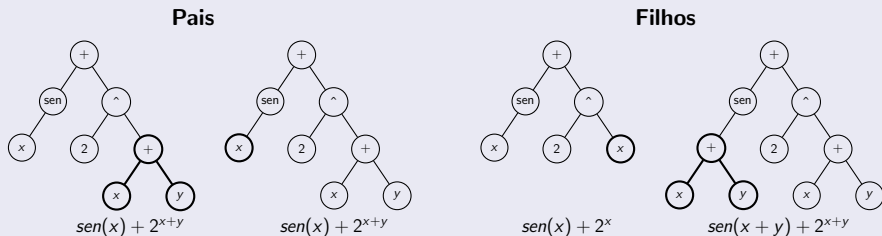


Fonte: Adaptado de ML. Mitchell.

Programação Genética

Cruzamentos

- Em algoritmos genéticos, pais iguais levam a filhos idênticos

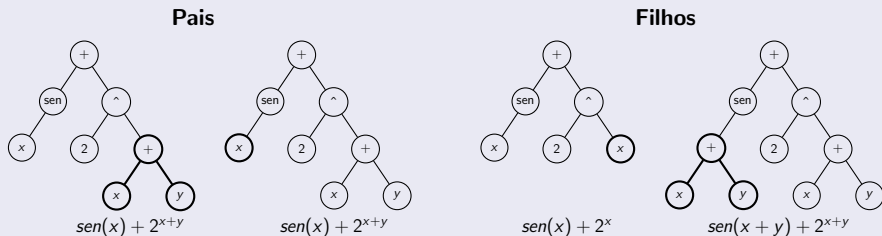


Fonte: Adaptado de ML. Mitchell.

Programação Genética

Cruzamentos

- Em algoritmos genéticos, pais iguais levam a filhos idênticos
- Essa é uma das principais vantagens da programação genética em relação aos algoritmos genéticos



Fonte: Adaptado de ML. Mitchell.

Programação Genética

Mutações

Mutações

- Mutação no nó:

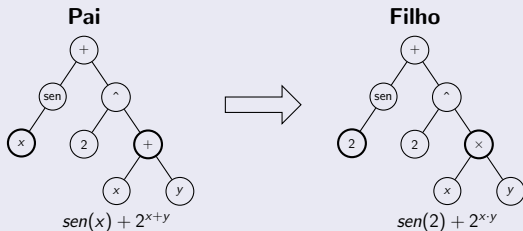
Mutações

- Mutação no nó:
 - Escolha aleatoriamente um ou mais nós, e substitua seu(s) valor(es) por outro(s) valor(es) aleatório(s)

Programação Genética

Mutações

- Mutação no nó:
 - Escolha aleatoriamente um ou mais nós, e substitua seu(s) valor(es) valor(es) por outro(s) valor(es) aleatório(s)

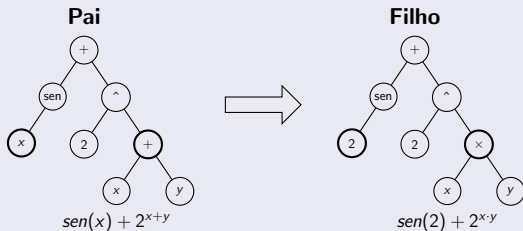


Fonte: Adaptado de ML. Mitchell.

Programação Genética

Mutações

- Mutações no nó:
 - Escolha aleatoriamente um ou mais nós, e substitua seu(s) valor(es) por outro(s) valor(es) aleatório(s)
 - Substitua função por função, operador por operador, e terminal por terminal



Fonte: Adaptado de ML. Mitchell.

Programação Genética

Mutações

- Mutação no ramo:

Programação Genética

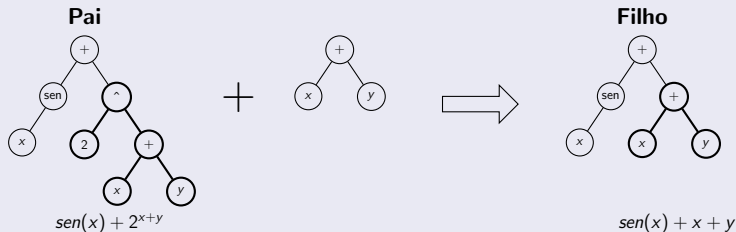
Mutações

- Mutação no ramo:
 - Um **ramo** da árvore é escolhido aleatoriamente e substituído por outro gerado aleatoriamente pelo programa

Programação Genética

Mutações

- Mutaç o no ramo:
 - Um **ramo** da  rvore   escolhido aleatoriamente e substituído por outro gerado aleatoriamente pelo programa

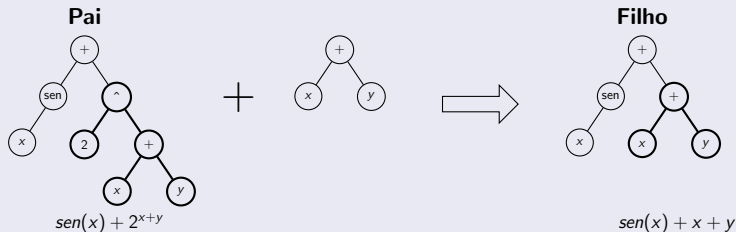


Fonte: Adaptado de ML. Mitchell.

Programação Genética

Mutações

- **Mutação no ramo:**
 - Um **ramo** da árvore é escolhido aleatoriamente e substituído por outro gerado aleatoriamente pelo programa
 - Modifica a estrutura da árvore do programa, pois o novo ramo não precisa ter o mesmo tamanho do anterior



Fonte: Adaptado de ML. Mitchell.

Hiperparâmetros

- Mutações podem ser misturadas

Hiperparâmetros

- Mutações podem ser misturadas
 - Podemos fazer quantas mutações de cada tipo que quisermos

Hiperparâmetros

- Mutações podem ser misturadas
 - Podemos fazer quantas mutações de cada tipo que quisermos
- São então hiperparâmetros (ou metaparâmetros) do modelo

Hiperparâmetros

- Mutações podem ser misturadas
 - Podemos fazer quantas mutações de cada tipo que quisermos
- São então hiperparâmetros (ou metaparâmetros) do modelo
 - Parâmetros definidos de antemão e que não serão modificados quando ajustamos (treinamos) o modelo

Hiperparâmetros

- Exemplo:

Hiperparâmetros

- Exemplo:
 - A cada geração, modificamos valores de nós e a estrutura das árvores

Hiperparâmetros

- Exemplo:
 - A cada geração, modificamos valores de nós e a estrutura das árvores
 - Essas mudanças são guiadas pelos dados, via função de *fitness* → são **treinadas**

Hiperparâmetros

- Exemplo:
 - A cada geração, modificamos valores de nós e a estrutura das árvores
 - Essas mudanças são guiadas pelos dados, via função de *fitness* → são **treinadas**
 - Não modificamos, contudo, quantas mutações fazemos a cada geração

Hiperparâmetros

- Exemplo:
 - A cada geração, modificamos valores de nós e a estrutura das árvores
 - Essas mudanças são guiadas pelos dados, via função de *fitness* → são **treinadas**
 - Não modificamos, contudo, quantas mutações fazemos a cada geração
 - Esse não é um parâmetro refinado a partir dos dados → é um hiper-parâmetro

Referências

- Russell, S.; Norvig P. (2010): Artificial Intelligence: A Modern Approach. Prentice Hall. 3a ed.
- Mitchell, T. (1997): Machine Learning. McGraw-Hil.
- Negnevitsky, M. (2005): Artificial Intelligence: A Guide to Intelligent Systems. Addison-Wesley. 2a ed.
- Goldberb, D.A. (1989): Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley.
- Rothlauf, F. (2006): Representations for Genetic and Evolutionary Algorithms. Springer. 2a ed.
- Coley, D.A. (1999): An Introduction to Genetic Algorithms for Scientists and Engineers. World Scientific

Referências

- https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_genotype_representation.htm
- <https://towardsdatascience.com/parallel-and-distributed-genetic-algorithms-1ed2e76866e3>