



ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

Avenida Professor Luciano Gualberto, travessa 3 nº 158 CEP 05508-900 São Paulo SP

Telefone: (011) 818-5583 Fax (011) 818-5294

Departamento de Engenharia de Computação e Sistemas Digitais

PCS 2428 e PCS 2059 – INTELIGÊNCIA ARTIFICIAL ALGORITMO GENÉTICO E PROGRAMAÇÃO GENÉTICA

Autores:

*Luis Gustavo Ludescher
Rodrigo Augusto Azevedo Pelegrini Perez
Tiago Matos*

Data de emissão:

12/10/2008



ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

Avenida Professor Luciano Gualberto, travessa 3 nº 158 CEP 05508-900 São Paulo SP
Telefone: (011) 818-5583 Fax (011) 818-5294

Departamento de Engenharia de Computação e Sistemas Digitais

Conteúdo

Conteúdo	2
Introdução	3
Tipos de algoritmos genéticos.....	3
Elementos de um algoritmo genético evolutivo	4
Representação de variáveis e seleção delas para o algoritmo	5
Função de custo (ou fitness function)	5
A população.....	5
Seleção	5
Método da roleta.....	6
Método do Torneio	6
Método da Amostragem Universal Estocástica	6
Atuadores sobre o algoritmo genético.....	7
Crossover	7
Mutação.....	8
Exemplo dos atuadores no problema das oito rainhas.....	8
Exemplos de utilização de algoritmos genéticos.....	9
PROGRAMACAO GENÉTICA.....	9
Passos iniciais para a programação genética	10
Conjunto de terminais e de funções	10
Medida da aptidão.....	11
Parâmetros.....	11
Critério de parada e designação de resultados	11
Atuadores sobre a programação genética.....	11
Reprodução	11
Crossover.....	11
Mutação	12
Exemplo de utilização da Programação Genética.....	13
Aplicativo de Exemplo criado.....	13
Resultados notados	17
BIBLIOGRAFIA	17



Introdução

A definição formal de algoritmos genéticos foi feita por John Holland na década de 1960. O desenvolvimento dos algoritmos genéticos se deu durante a década de 70 e foi feito por John Holland com a ajuda de estudantes (com grande destaque para David Goldberg) e colaboradores da Universidade de Michigan.

Os algoritmos genéticos são uma extensão da programação evolutiva que usa a biologia evolutiva para a elaboração de algoritmos. A computação evolutiva trata de sistemas para a resolução de problemas utilizando modelos computacionais que se baseiam na Teoria da Evolução. Os algoritmos evolutivos são classificados em três tipos: Algoritmos genéticos, Estratégias de Evolução e Programação genética. Na definição original de algoritmos evolutivos, algoritmos genéticos é uma abstração da biologia evolutiva.

O algoritmo genético elaborado por Holland é um método para transformar uma população de cromossomos (conjunto de strings de zeros e uns, ou bits), sendo que cada indivíduo possui um valor associado *que indica a sua chance de sobrevivência*, em uma nova população. Nessa transformação o algoritmo genético usa os princípios de sobrevivência e reprodução dos indivíduos mais aptos (oriundos da Teoria Evolucionista de Darwin) e os princípios da evolução genética, mais especificamente os seus operadores: crossover, mutação e inversão.

Esse tipo de algoritmo resolve um problema procurando numa população inicial de indivíduos, achando a melhor solução (ou uma solução próxima da ideal) através da reprodução (evolução) dessa população inicial.

Tipos de algoritmos genéticos

Existem duas classificações principais de algoritmos genéticos, que estão relacionadas no escopo da solução do problema que resolvem: o algoritmo genético binário e o algoritmo genético de variáveis contínuas. Esses dois tipos de algoritmo possuem a mesma estrutura básica e possuem os mesmos tipos de atuadores para a recombinação genética e seleção natural. A diferença existente entre esses dois algoritmos se encontra apenas no domínio de suas variáveis.

O algoritmo genético binário representa suas variáveis como uma seqüência binária codificada, ou seja, uma seqüência de zeros e uns, e trabalha com esta para minimizar o custo da função. Um exemplo da utilização desse tipo de



algoritmo está o modelamento do problema das oito rainhas (capítulo 4 de [4]) e o modelamento para achar o ponto mais alto de uma montanha (capítulo 2 de [5]).

No caso do algoritmo genético de variáveis contínuas, o algoritmo trabalha com variáveis contínuas para a minimização do custo. Um exemplo da utilização desse tipo de algoritmo está na resolução de equações matemáticas e todo tipo de problema em que a quantização dos valores prejudica o resultado. Outra utilização desse tipo de algoritmo é na resolução de problemas onde se deseja saber a resolução com a máxima precisão disponível no computador.

Elementos de um algoritmo genético evolutivo

Nesse item são descritos os principais componentes existentes em um algoritmo genético. Mais especificamente serão mostrados exemplos relacionados ao algoritmo genético do tipo binário.

A figura 1 ilustra os principais componentes de um algoritmo genético binário.

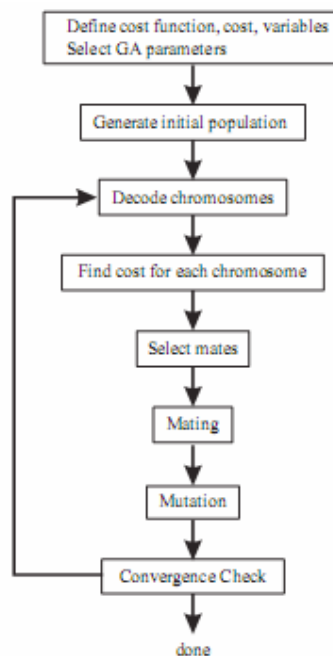


Figura 1: Fluxo de funcionamento de um algoritmo genético binário



Representação de variáveis e seleção delas para o algoritmo

Fazendo uma analogia com a genética, um conjunto de variáveis no algoritmo genético é chamado de cromossomo e um conjunto de cromossomos é chamado de população do algoritmo. A seleção das variáveis (cromossomo) para o algoritmo genético deve ser feita de acordo com o problema com o qual se quer solucionar.

No caso do problema para achar o ponto mais alto de uma montanha as variáveis que seriam escolhidas de acordo com o escopo de problema seriam a latitude e a longitude.

Função de custo (ou fitness function)

A função de custo pode ser representada por uma função matemática, um experimento científico ou um jogo. Essa função recebe como entrada os valores do gene do cromossomo e fornece como resultado a sua aptidão. Esse valor retornado é usado no processo de seleção natural sobre a população de cromossomos.

No caso do problema para achar o ponto mais alto de uma montanha a função de custo seria a elevação do ponto em uma determinada coordenada. Já no caso do jogo das oitos rainhas a função de custo seria o número de rainhas que não estão se atacando mutuamente.

A população

O algoritmo genético é inicializado com um grupo inicial de cromossomos. Esses cromossomos são chamados de população do algoritmo genético. Com o passar do tempo essa população vai sofrer alterações (provocadas pelos atuadores do algoritmo) em seu código e evoluir. A população do Algoritmo Genético não irá aumentar de tamanho, pois o tamanho dessa população é limitado.

Seleção

Esse componente é responsável por selecionar um cromossomo na população para ser utilizado na reprodução. Essa seleção se dá através da análise do valor da aptidão do cromossomo no ambiente (valor este calculado pela função de custo). Existem três métodos principais para selecionar os cromossomos que irão se reproduzir numa população: o método da roleta, o método do torneio e o método da amostragem universal estocástica.



Método da roleta

Método de seleção mais utilizado devido a sua facilidade de implementação. Consiste em realizar a seleção dos cromossomos semelhante ao que ocorre nos jogos de azar.

A figura 2 ilustra o método da roleta.

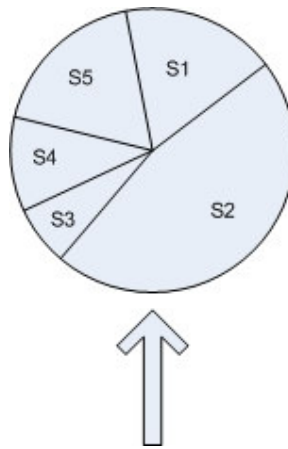


Figura 2: Método de seleção por roleta

Método do Torneio

Consiste em selecionar n indivíduos da população aleatoriamente com a mesma probabilidade. Entre estes n cromossomos aquele com maior aptidão são selecionados para a população intermediária. O processo se repete até a população intermediária ser preenchida.

Método da Amostragem Universal Estocástica

É uma variação do método da roleta em que n elementos são selecionados ao mesmo tempo.

A figura 3 ilustra o método da Amostragem Universal.

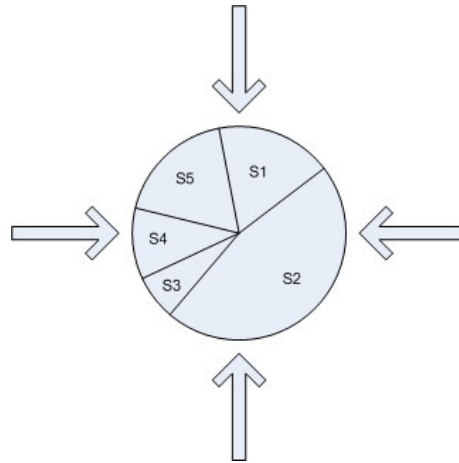


Figura 3: Método da seleção por SUS

Atuadores sobre o algoritmo genético

Existem dois tipos de atuadores sobre a população do algoritmo genético: o crossover e a mutação. A seguir são explicados esses atuadores.

Crossover

Esse atuador é responsável pela escolha de um "locus" de um cromossomo e pela troca dos pedaços de cromossomo entre dois cromossomos vizinhos. Essa troca acarreta a geração de descendentes completamente distintos da geração originária. A figura 4 ilustra a atuação do crossing over.

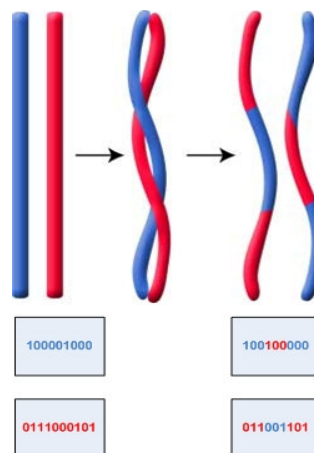


Figura 4: Exemplo do processo de crossing over



Mutação

Esse atuador é responsável pela alteração do valor de um gene (bit) escolhido aleatoriamente. A figura a seguir ilustra um exemplo de mutação em um cromossomo.

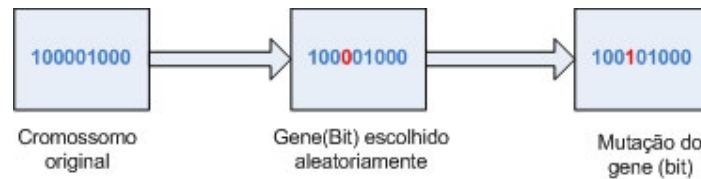


Figura 5: Exemplo de atuação da mutação sobre os genes

Exemplo dos atuadores no problema das oito rainhas

Modela-se o problema das oito rainhas considerando que cada posição do cromossomo corresponde a uma coluna do tabuleiro e o valor em cada gene corresponde à linha que a rainha se encontra. As figuras 6 e 7 exemplificam o modelamento do problema das oito rainhas utilizando algoritmos genéticos e a ação dos atuadores sobre o algoritmo genético.

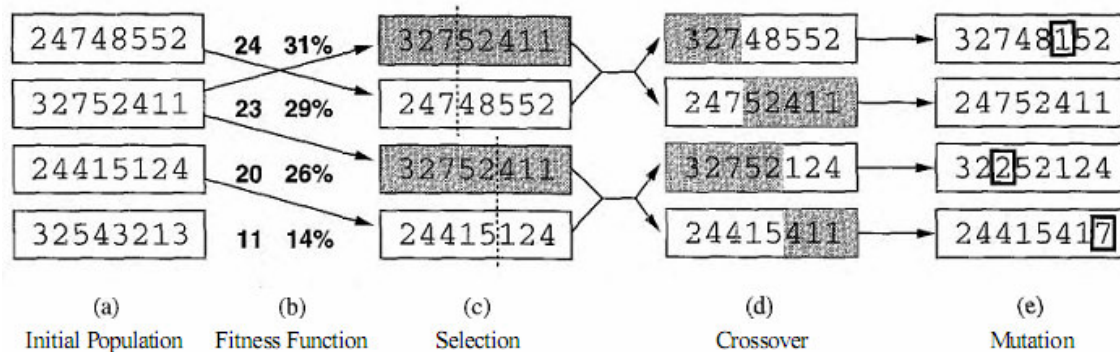


Figura 6: Ação dos atuadores sobre o problema das oito rainhas

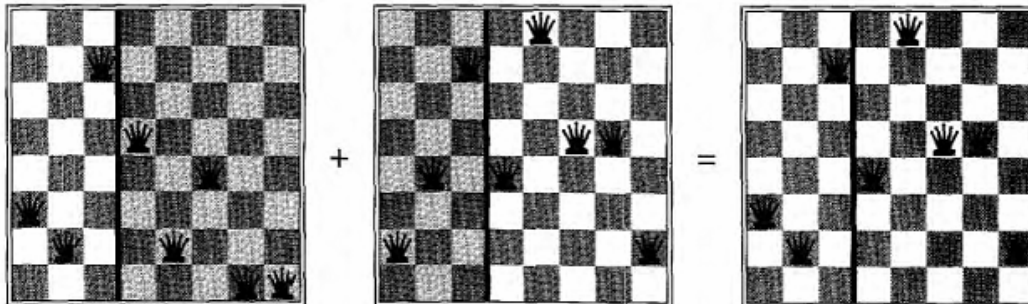


Figura 7: Ilustração do tabuleiro para os dois primeiros itens da letra c e o primeiro da letra e.

Exemplos de utilização de algoritmos genéticos

Entre as aplicações dos algoritmos genéticos para a resolução de problemas estão os seguintes exemplos:

- Achar o ponto mais alto de um mapa
- Resolução do problema das oito rainhas
- "Arte Genética"
- Palavras cruzadas
- Problema do caixeiro viajante
- Planejamento de trajetória de robôs
- Resolução de equações não lineares de alta ordem

PROGRAMACAO GENÉTICA

A programação genética foi originada no início da década de 80 e possui como desafio o desenvolvimento de um método para a criação automática de programas a partir de uma descrição de alto nível do problema a ser resolvido. Ela é uma extensão do algoritmo genético com a diferença que a população genética contém um conjunto de programas. Os programas são armazenados na forma de árvores sintáticas.

No início a programação genética possui um conjunto inicial de programas gerados aleatoriamente. Ela aplica os princípios da evolução e da genética para gerar uma nova (e em algumas vezes melhorada) população de programas.



Esse tipo de técnica tem a vantagem de combinar a expressiva representação simbólica de alto nível dos programas de computador com a eficiência (quase ótima) de busca do algoritmo genético. Um programa de computador que resolve (ou quase resolve) um dado problema geralmente surge desse processo.

Possui como características principais o fato de evitar o problema do máximo local (semelhante aos algoritmos genético e a Têmpera Simulada), ser um algoritmo probabilístico e não possuir uma base de conhecimento prévia.

Passos iniciais para a programação genética

A programação genética precisa de cinco passos preparatórios iniciais para conseguir gerar um programa em sua saída (conforme a figura 8). A seguir são descritos cada um desses cinco passos.

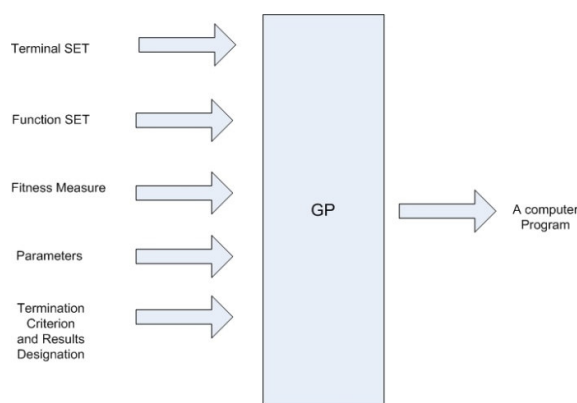


Figura 8: Passos preparatórios para geração de um programa

Conjunto de terminais e de funções

Esses itens são os ingredientes básicos para a geração de programas pela programação genética. Basicamente são formados pelos operadores aritméticos (adição, subtração, divisão e multiplicação), as entradas do algoritmo, as constantes e um operador de ramificação condicional.



Medida da aptidão

Esse passo corresponde ao cálculo da aptidão para o problema. Geralmente a função de cálculo de aptidão é fornecida pelo usuário.

Parâmetros

Esse item corresponde à especificação dos parâmetros de controle de funcionamento do algoritmo.

Critério de parada e designação de resultados

Esse parâmetro especifica o critério de parada do algoritmo e o método de designação dos resultados para o funcionamento.

Atuadores sobre a programação genética

A seguir são descritos os três tipos de atuadores sobre a programação genética: a reprodução, o crossover e a mutação.

Reprodução

A operação de reprodução ocorre em um programa de computador selecionado com uma probabilidade ponderada pelo valor de aptidão do programa (calculada através da função de avaliação). Esse operador funciona basicamente para copiar um determinado programa para a próxima geração.

Crossover

O atuador de crossover age sobre um par selecionado de programas da população selecionados através da função de avaliação e cria uma nova geração utilizando partes de cada um dos pais. A figura 9 exemplifica o processo de crossover sobre as árvores da programação genética.

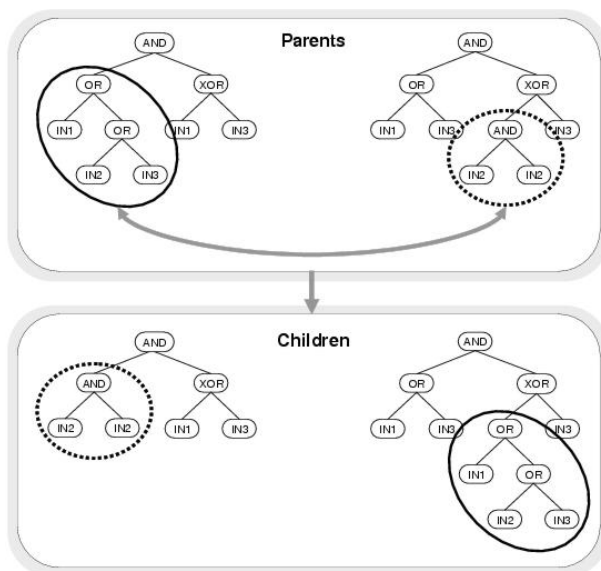


Figura 9: Processo de crossover na criação de uma nova geração

Mutação

Esse atuador é responsável pela alteração de uma sub-árvore escolhida aleatoriamente de uma árvore mãe. Após a alteração essa nova árvore gerada passa a fazer parte da nova geração. A figura 10 ilustra um exemplo de mutação ocorrendo em uma árvore.

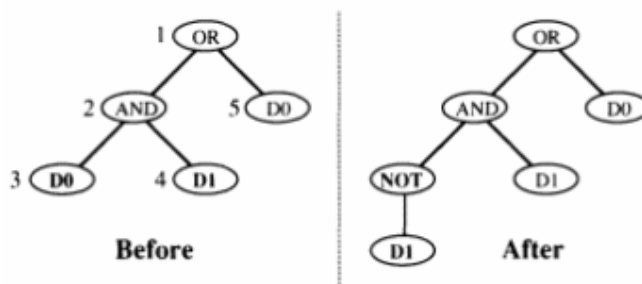


Figura 10: Processo de mutação na criação de uma nova geração



Exemplo de utilização da Programação Genética

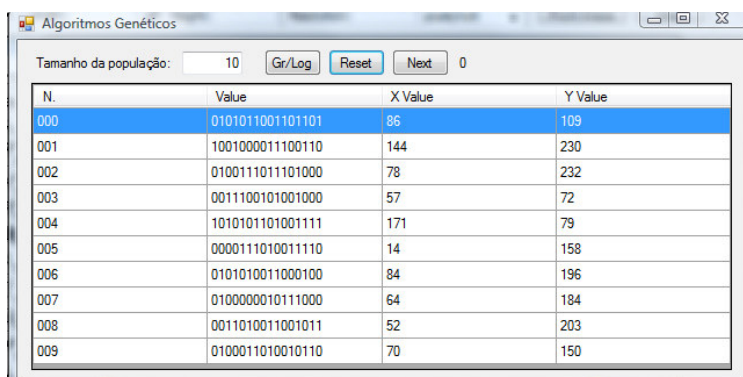
Existem diversos exemplos de aplicações da Programação Genética atualmente. Um dos exemplos de utilização é no controle de agentes jogadores de futebol de robô onde a programação genética é usada no controle automático dos jogadores (disponível em [8]).

Outras aplicações da Programação Genética ocorrem em:

- Criação de quatro diferentes algoritmos para o problema da identificação do segmento da transmembrana para proteínas.
- Síntese de amplificadores de 60 e 96 decibéis.
- Síntese de circuitos computacionais analógicos para raiz quadrada, elevação ao cubo, elevação ao quadrado, raiz cúbica, logaritmo e funções Gaussianas.
- Síntese de um circuito analógico de tempo-real para otimização-temporal de um robô.
- Síntese de um termômetro eletrônico.
- Síntese de um circuito de referência de voltagem.

Aplicativo de Exemplo criado

Para ajudar na apresentação do trabalho de Algoritmos Genéticos, o grupo decidiu desenvolver uma aplicação que mostra passo a passo os cromossomos se evoluindo.



N.	Value	X Value	Y Value
000	0101011001101101	86	109
001	1001000011100110	144	230
002	0100111011101000	78	232
003	0011100101001000	57	72
004	1010101101001111	171	79
005	0000111010011110	14	158
006	0101010011000100	84	196
007	0100000010111000	64	184
008	0011010011001011	52	203
009	0100011010010110	70	150

Figura 11: Tela Inicial do aplicativo criado



A Figura 11 mostra a tela inicial da aplicação. Deve-se num primeiro passo escolher o tamanho da população e apertar o botão "Reset" para criar a população, no exemplo, o tamanho da população escolhido é 10.

A primeira coluna é um número único que mostra o número do cromossomo, serve para identificar se ele ainda está lá quando há uma nova iteração, e serve para se ter uma estimativa de qual geração ele é.

A segunda coluna são dois bytes concatenados, o primeiro byte representa o valor X do cromossomo, o segundo byte representa o valor Y do cromossomo.

A tela de gráfico e de log se abrem ao clicar o botão "Gr/Log". O gráfico para a configuração inicial é:

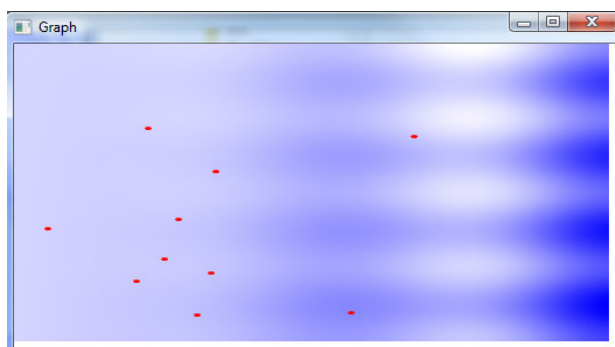


Figura 12: Tela gráfica apresentando os pontos

Os pontos mostram a posição x e y na superfície que representa a função de avaliação. O eixo vertical é invertido. Manchas azuis são as melhores áreas e manchas brancas as piores. Pode-se imaginar que numa superfície quanto mais azul maior é a altitude. Portanto, há uma cordilheira na direita e uma planície na esquerda.

A função de avaliação é uma função matemática comum de duas variáveis x e y. Nas figuras utilizaremos a equação:

$$f(x,y) = xy + x^2 \sin\left(y \frac{4\pi}{255} - 2\pi\right)^2 + \cos\left(x \frac{4\pi}{255} - 2\pi\right) + y$$

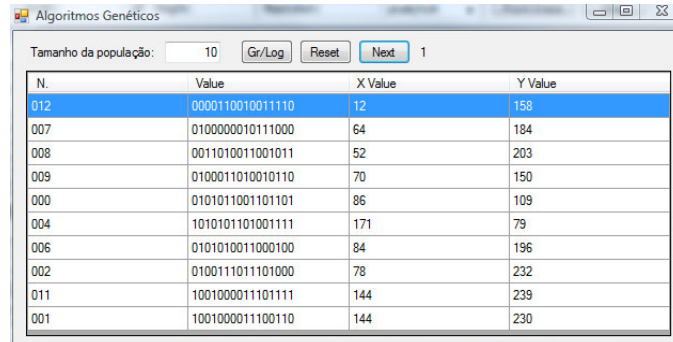


ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

Avenida Professor Luciano Gualberto, travessa 3 nº 158 CEP 05508-900 São Paulo SP
Telefone: (011) 818-5583 Fax (011) 818-5294

Departamento de Engenharia de Computação e Sistemas Digitais

Após a primeira iteração (apertar botão "Next"), a tabela é ordenada quanto a seu valor na função de avaliação. O cromossomo 1 possui o maior valor.



The screenshot shows a window titled 'Algoritmos Genéticos'. At the top, there are controls: 'Tamanho da população:' with a value of 10, and buttons for 'Gr/Log', 'Reset', and 'Next'. The 'Next' button is highlighted, and a small '1' is next to it. Below these controls is a table with four columns: 'N.', 'Value', 'X Value', and 'Y Value'. The table contains 12 rows of data, with the first row (N. 012) highlighted in blue.

N.	Value	X Value	Y Value
012	0000110010011110	12	158
007	0100000010111000	64	184
008	0011010011001011	52	203
009	0100011010010110	70	150
000	0101011001101101	86	109
004	1010101101001111	171	79
006	0101010011000100	84	196
002	0100111011101000	78	232
011	1001000011101111	144	239
001	1001000011100110	144	230

Figura 13: Resultados para a primeira iteração do algoritmo

Segundo o log, houve uma mutação em um dos cromossomos (nesse caso a aplicação o trata com um id novo). O cromossomo de número 5 se tornou o de número 12 por mutação.

O cromossomo 4 e o 1 por crossover, deram origem ao 10 e ao 11. Como 2 filhos chegaram, os dois piores da população 3 e 10 morreram. O cromossomo de número 10 teve uma morte prematura.

Mutation: 0000111010011110 => 0000110010011110

Crossover: 1010101101001111 + 1001000011100110 = 1010101101000110 ,
1001000011101111

Iteração 1

Reset

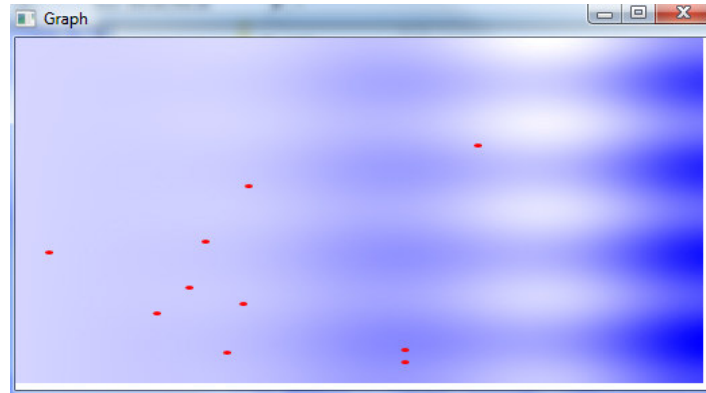


Figura 14: Distribuição dos pontos depois da primeira iteração

Na iteração de número 44 todos os cromossomos ficaram com o mesmo valor:

N.	Value	X Value	Y Value
125	1000110111011100	141	220
126	1000110111011100	141	220
123	1000110111011100	141	220
117	1000110111011100	141	220
114	1000110111011100	141	220
109	1000110111011100	141	220
101	1000110111011100	141	220
097	1000110111011100	141	220
096	1000110111011100	141	220
087	1000110111011100	141	220

Figura 15: Resultados após 44 iterações



Figura 16: Distribuição dos pontos após 44 iterações



O algoritmo, nesse caso, não encontrou o valor máximo possível da função, ele ficou preso em um máximo local. Este resultado é totalmente aceitável, já que é um processo estocástico e, portanto o resultado nem sempre é o mesmo a cada execução.

Resultados notados

Aumentando-se o número da população para 30, as chances de se conseguir o máximo global são bem maiores.

Em poucas iterações os cromossomos já se juntam bastante.

Aumentamos o fator de mutação para dar mais chances dos cromossomos testarem uma área maior da superfície.

Há diversas configurações na taxa de mutação e de crossover que podem melhorar o algoritmo.

BIBLIOGRAFIA

[1] Mitchell, Melanie. An introduction to Genetic Algorithms. MIT Press, 1999. 158p.

An introduction to genetic algorithms.

[3] Rothlauf, Franz. Representations for Genetic and Evolutionary Algorithms. 2.ed. Netherlands: Springer, 2006. 325p.

[4] Russel, Stuart; Norvig, Peter. Artificial Intelligence: a modern approach. 2.ed. New Jersey: Pearson, 2003. 1069p.

[5] Haupt, Randy L. ; Haupt, Sue Ellen. Practical Genetic Algorithms. 2.ed. New Jersey: Willey, 2004. 253p.

[6] Koza, John R.; Bennett III, Forrest H.; Andre, David; Keane, Martin A.. Genetic Programming III: Darwinian Invention and Problem Solving. San Francisco: Morgan Kaufmann, 1999. 1154p.

[7] Rezende, Solange Oliveira. Sistemas Inteligentes: Fundamentos e aplicações. 1.ed. Barueri: Manole, 2005. 525p.

[8] Maia Jr., Luiz Carlos; Bianchi, Reinaldo A. C.. Usando Programação Genética para evoluir agentes jogadores de futebol de robô. Revista FEI, 2001. Disponível em: <http://www.fei.edu.br/~rbianchi/publications/RevistaFEI2001.pdf>. Acessado em: 11/10/2008.

[9] Goldberg, David E. Algorithmes génétiques. 1.ed. France: Addison-Wesley, 1994. 417p.