

UMA APLICAÇÃO DE PROGRAMAÇÃO GENÉTICA. Carlos Eduardo Marchi e Erwin Doescher. – Exatas – Ciência da Computação - Departamento de Matemática, Estatística e Computação – FCT/UNESP – Campus de Presidente Prudente.

Programação Genética é uma técnica automática de programação que proporciona a evolução de programas de computadores que tendem a resolver problemas. No projeto, para que possamos resolver problemas, criamos uma Máquina Virtual que pudesse interpretar e executar as instruções dos programas (indivíduos). Estas instruções implementam os operadores aritméticos, relacionais, entrada e saída de dados, pilhas, empilha, genético, operações. Veja a figura 1.



Figura 1 - Máquina Virtual (M.V.)

O módulo de operações consiste à base da Máquina Virtual, ele possui estruturas, uniões, enumerações.

Dois tipos de enumerações foram criados: uma chamada de “operacao” contendo operações diversas relacionadas a funções de entrada e saída, operadores aritméticos, operadores relacionais, empilha. A segunda chamada de “tipo” contém nome de tipos de variáveis: “caracter” (char), “inteiro” (int), “real_f” (float) e “real_d” (double).

Tendo apenas uma união com o nome de “dado”, ela possui todos os tipos de variáveis: “caracter” (char), “inteiro” (int), “freal” (float) e “dreal” (double).

E por fim vêm as estruturas, que são apenas três. A primeira estrutura indicada para tipos de variáveis chamadas de “tp_var” contendo duas variáveis: “var” do tipo “tipo” (enumeração de tipos de variáveis) e a segunda variável chamada de “valor” do tipo “dado” (união que armazenará valores de acordo com os tipos de “dados”).

A segunda estrutura para o corpo de instruções, chamada de “instrucao” contém também duas variáveis: uma chamada de “ident” do tipo “operação” (enumeração que contém todas as operações que forma o indivíduo). A segunda variável, chamada de “parametro” do tipo “tp_var” (estrutura de variáveis).

E por ultimo vem à estrutura de aptidão, chamada de “vpopulacao”, contendo variáveis de contadores e registro de posição do indivíduo na população.

O módulo de operadores aritméticos é usado para fazer cálculos aritméticos (soma, subtração, multiplicação, divisão). Dentro deste módulo possui várias funções que auxiliam nos cálculos: soma, soma_r, subtração, subtração_r, multiplicação, multiplicação_r, divisão, divisão_r, divresto, divresto_r. Cada uma destas funções contém dois parâmetros: “P” do tipo “pilha” (utilizada para armazenar valores de operações e programa teste) e “valor_const” do tipo “tp_var” (estrutura contendo o tipo e o valor de uma variável).

Todas essas funções têm execução semelhante: verificam o tipo da variável e o valor contido nela, coleta dois valores da pilha. O primeiro valor está no penúltimo campo da pilha e o segundo no topo da pilha. Isto é válido para funções que apenas coletam os valores e não os retira da pilha. A funções que coletam valores e os retira da pilha contém no final do nome da função o “_r”: soma_r, divresto_r. Portanto, a coleta dos valores muda, passa a ser sempre o topo da pilha. Os valores contidos na pilha podem ser dos tipos: caracter, inteiro, real_f e real_d. Como os valores a serem calculados podem ser de tipos distintos é feito um cast. O cast serve para que tipos distintos de valores se tornem iguais. Depois é feito o calculo de acordo com a função e o resultado armazena sempre no topo da pilha.

São parecidos também com os operadores aritméticos, os operadores relacionais. Os operadores relacionais possuem os mesmos parâmetros e forma de execução. Porém, ao invés de fazer cálculos

aritméticos, fazem comparações dos valores coletados na pilha, são eles: maior, maior_r, menor, menor_r, diferente, diferente_r, maiorigual, maiorigual_r, menor igual, menorigual_r, igual, igual_r, desvio_cond, desvio_inc. Esses dois últimos fazem um desvio. O desvio condicional (função “desvio_cond”) coleta o valor do indivíduo, verifica o tipo de dados e analisa o seu valor. Se este valor for positivo e estiver dentro do tamanho do indivíduo (MAXPROGRAM) retornará esse valor como o desvio. Caso não esteja de acordo com a condição retornará “-1” e não executará o salto. O desvio incondicional (função “desvio_inc”) retornará um valor dentro do tamanho do indivíduo (MAXPROGRAM).

O módulo de entrada e saída possuem os mesmos parâmetros das funções de operadores aritméticos e operadores relacionais. Na função de entrada, é inserido um valor dentro do tipo da variável por um usuário qualquer, que é colocado no topo da pilha. Portanto, existem funções para inserir valores de caracteres, inteiros, floats e doubles. São passados na hora da inserção: o valor da variável e o tipo dela. Exemplo: caso insira um caracter, o valor do caracter é seu código em ASCII e o seu tipo será CHARACTER.

As funções de saída imprimem o valor que está no topo da pilha. Mas, antes é feita uma verificação do tipo de dado a ser impresso e só depois é feita a impressão. Há dois tipos de funções de saídas, a que apenas lê um valor no topo da pilha e não o retira e a outra que lê e o retira da pilha.

Outra função da M.V. parecida com a função de entrada é a função empilha. A única diferença é de quem insere os valores. Na função de entrada quem insere é qualquer usuário, e na função empilha é o programador que insere um valor (constante). Esta função tem por finalidade inserir constantes no programa. É inserido um valor e o seu tipo no topo da pilha.

O próximo módulo da M.V. é a manipulação de pilha. Este módulo contém funções que ajudam a manipular a pilha. A estrutura pilha foi montada a partir de um vetor de MAXPROGRAM itens do tipo “tp_var”. A primeira variável denota o topo da pilha (topo) e a segunda é um vetor (vetor) do tipo “tp_var” de tamanho MAXPROGRAM. A pilha é uma função muito importante, pois todas as operações realizadas na M.V de alguma forma utilizam a pilha. A pilha armazena o programa “teste” que é usado para avaliar todos os indivíduos da população, e também serve para armazenar o resultado de operações aritméticas, relacionais, entrada e saída, empilha.

O mais importante módulo da M.V é chamado de “genético”. Este módulo tem por finalidade aplicar os conceitos mais importantes e fundamentais de algoritmos genéticos. Ela possui funções para criar uma população, avaliar cada indivíduo da população, fazer recombinações e mutações dos melhores indivíduos da população. Possui três vetores: o primeiro estará todos os indivíduos da população, chamado de “vetpopula”, do tipo “vetprograma”. O segundo vetor, chamado de “vetprograma”, do tipo “instrucao” contém instruções. Essas instruções, como foi dito anteriormente, possuem variáveis de qual operação será executada e os parâmetros dessa operação. O conjunto dessas instruções forma um indivíduo (programa). E por fim, o terceiro, vetor chamado de “vet_avaliacao”, do tipo “vpopulacao”. A finalidade deste vetor é, através de um programa “teste”, avaliar cada indivíduo da população e ordená-los de acordo com suas melhores aptidões.

O módulo “executa” pode chamar todos os outros módulos, exceto o módulo genético.

Como dito anteriormente, todos os indivíduos da população são do mesmo tipo e tamanho. Esses indivíduos são do tipo “instrucao”, tendo como parâmetros o tipo de variável (“tp_var”) e a instrução (“operação”). Inicialmente, estas instruções são sorteadas e agrupadas em um vetor de tamanho fixo. O tamanho dos indivíduos (MAXPROGRAM) é inicialmente 200, mas dependendo do resultado obtido poderá haver alterações (Veja a figura 2 (A)).

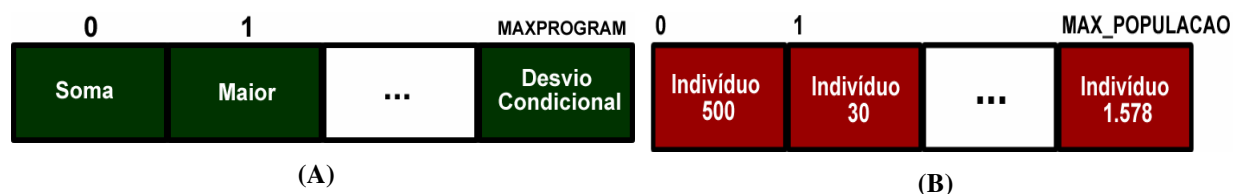


Figura 2: (A) Indivíduo

(B) Avaliação

Cada um desses indivíduos representa uma possível solução para um dado problema. A união destes indivíduos forma a população. Veja figura 3.

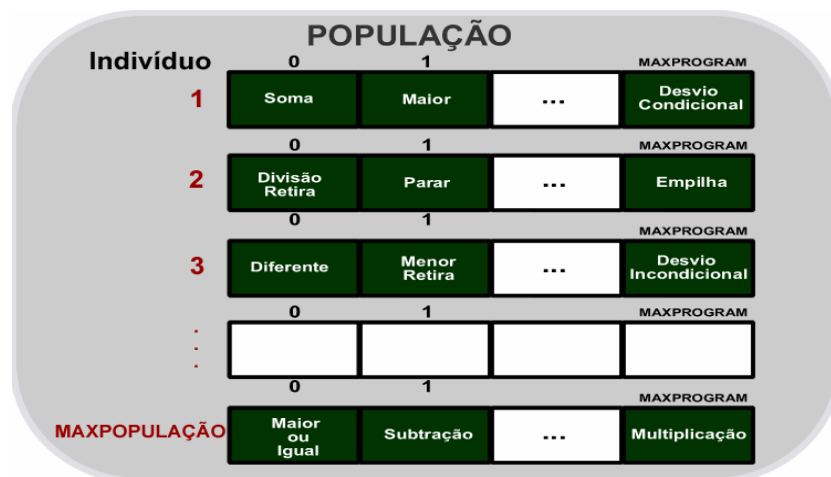


Figura 3 - População

Todos os indivíduos da população são agrupados em um vetor de tamanho fixo. O tamanho inicial da população é 10000 (MAXPOPULACAO), porém poderá haver necessidade de mudança de acordo com o resultado final da solução. O tipo da população é um vetor (vetprograma), cada campo do vetor é um indivíduo.

A máquina virtual permite que cada indivíduo seja executado, produzindo uma resposta que, comparada com a resposta do programa teste, determina um valor de aptidão. Os indivíduos serão ordenados de acordo com as suas aptidões na população. O tamanho do vetor é o mesmo que o da população (MAXPOPULACAO), porém não do mesmo tipo. Veja a figura 2 (B).

O seu tipo é uma estrutura contendo o registro do indivíduo na população e contadores de instruções realizadas, de instruções válidas e inválidas. Dependendo do valor da solução poderá haver necessidade de acrescentar mais contadores para uma melhor avaliação.

Serão realizados cruzamentos (crossing-over) entre os indivíduos com melhores aptidões. Esses indivíduos trocarão trechos de instruções (cromossomos) permitindo que as próximas gerações herdem suas características. A figura 4 contém dois indivíduos (A e B). O indicador de cruzamento é um valor aleatório entre 0 (zero) e MAXPROGRAM, e indica o ponto onde os genes dos dois indivíduos serão trocados. Todos os cruzamentos gerarão dois filhos (programas). Os dois novos indivíduos (AB1 e AB2) possuem um trecho de genes idênticos aos dos pais (A e B). Estes dois indivíduos serão inseridos na população através de sua aptidão.



Figura 4 - Cruzamento (crossing-over)

Para aumentar a diversidade da população, ocorrerão mutações, segundo um fator especificado. Este fator é um valor percentual de indivíduos da população que ocorrerão mutações. O seu valor é de 10% (dez por cento) da população. Porém poderá ser mudado de acordo com o resultado da solução do problema. Veja a figura 5.

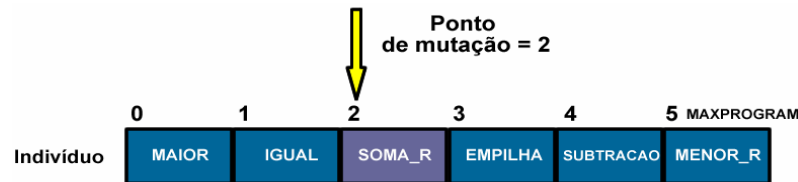


Figura 5 - Mutação

É alterada aleatoriamente uma instrução (gene) de um indivíduo, (a seta na figura 5 indicando o ponto de mutação). Este ponto de mutação é escolhido fazendo um sorteio entre o início 0 (zero) e o tamanho (MAXPROGRAM) do indivíduo. A mutação gerará um novo indivíduo que será avaliado e inserido na população. Por fim, o melhor indivíduo da população estará mais apto a solucionar o problema mesmo não o solucionando corretamente.

A figura 6 estabelece em fluxograma do processo descrito que pode ser resumido como:

- 1) Define um problema (programa teste);
- 2) Geração de uma população;
- 3) Avaliação desta população, a máquina virtual ajuda na avaliação executando cada indivíduo;
- 4) Mutação de acordo com o valor de mutação;
- 5) Resultado da possível solução do problema.

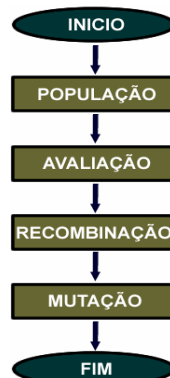


Figura 6 - Fluxograma de um Algoritmo Genético

Fatores que alteram o funcionamento do sistema, permitindo ou não encontrar a solução:

- 1) Tamanho da população (MAXPOPULACAO);
- 2) Tamanho do indivíduo (MAXPROGRAM);
- 3) Forma de cruzamento (os melhores indivíduos);
- 4) Taxa de mutação (TAXA_MUTACAO).

No estágio atual do trabalho, foram produzidos programas (vetores de instruções de tamanho fixo) que receberam uma aptidão baseada nos seguintes fatores: 1) relação entre o número de instruções válidas e inválidas; 2) Aptidão nula para os programas que não se encerrem dentro de um número determinado de instruções executadas. Com apenas estes dois critérios, o processo não foi capaz de encontrar um programa capaz de resolver o problema proposto. Acreditamos que os resultados obtidos serão melhores a partir da introdução do cruzamento e da mutação dos indivíduos.