PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
CURSO DE ENGENHARIA DE SOFTWARE

ANDRÉ LUIZ RODRIGUES
BRUNO RAMOS FREITAS
DOUGLAS SOBREIRA GARCIA

Trabalho 1 - Agentes e Algoritmo de Busca

Inteligência Artificial

Turma 031

Introdução

O algoritmo de busca é um método utilizado para procurar por elementos dentro de um conjunto de elementos previamente especificados. Um problema de busca é uma tarefa que pode ser resolvida por meio de algoritmo de busca, que recebe um problema como entrada e retorna a melhor solução encontrada para o problema proposto após resolvê-lo de várias outras formas possíveis. Uma solução é uma sequência de ações de um estado inicial a um estado final (objetivo), a solução ótima é aquela com o menor custo de caminho. Pode ser um livro em uma biblioteca, ou pode ser dados criptografados usados principalmente entre as guerras. Seu formato de linguagem computacional evoluiu com a construção dos primeiros computadores.

Quando falamos sobre o tema de algoritmos de busca, é inevitável não falarmos sobre algoritmos genéticos, fundamentado nas pesquisas de Darwin sobre sistemas naturais e sua capacidade de adaptação, David. Goldberg verificou que tal processo poderia ser incorporado e simulado matematicamente em um algoritmo de busca, que posteriormente foi chamado de Algoritmo Genético. Mesmo com sua simplicidade, fortes aproximações numéricas têm sido encontradas em problemas difíceis de serem resolvidos.

Objetivo

O trabalho consiste na simulação de um jogo, no qual um agente inseto consegue encontrar um caminho que permite recolher/comer toda a comida existente em um labirinto utilizando as melhores combinações possíveis a fim de encontrar o melhor caminho para a solução . O ambiente consiste em uma matriz N × N (Figura 1). A entrada E é sempre na posição (0.0) mas as comidas C podem estar em qualquer lugar. A entrada é sempre conhecida pelo agente, e é a sua célula inicial. Já as células onde estão a comida, ele tem que descobrir. São distribuídos no ambiente N/2 comidas. Foram fornecidos arquivos contendo os labirintos. Na figura, os valores 1 representam as paredes.

E	1	1	1	1	1	1	1		1
				C					1
1	1	1		1	1	1	1	1	
		1		1					
C		1		1	1		1	1	1
				1	C				
	1	1	1	1			1		C
	1							1	1
	1	1	1	1	1	1			1
C									1

Figura 1 – Labirinto 10 x 10

Justificativa

Com base no objetivo do problema apresentado, o grupo optou por utilizar o método de algoritmo genético aplicando inteligência artificial através de sua heurística para sugerir soluções que melhor atendam às condições propostas no problema, a escolha deste método foi de comum acordo entre os membros, por ser um método de fácil implementação, apresentar bons resultados para problemas de otimização e também por ser o método que os membros do grupo possuem maior conhecimento .

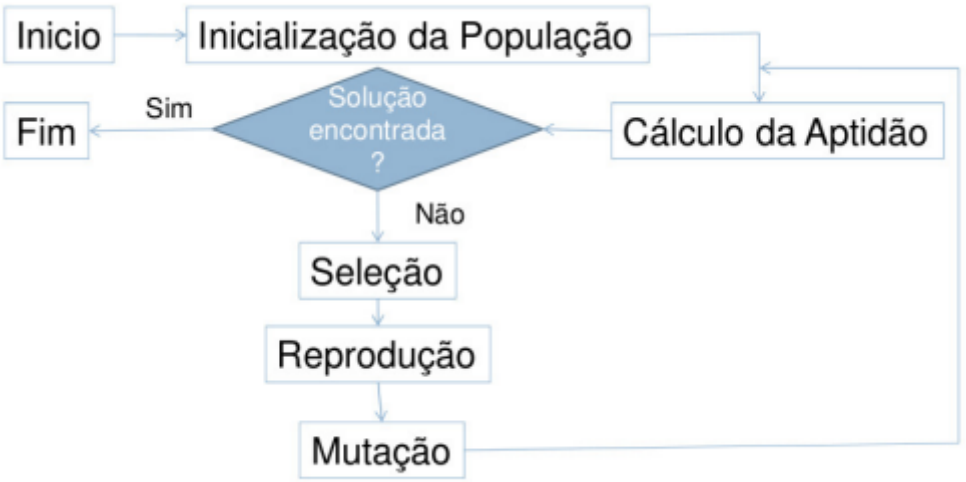

Figura 2 - Ciclo Algoritmo Genético

Desenvolvimento

Para o desenvolvimento do algoritmo, primeiramente pensamos em organizar e estruturar o código para que ficasse de fácil leitura e entendimento do código. Separamos as classes em 3 diretórios, são eles: IA, MODEL e UTILS.

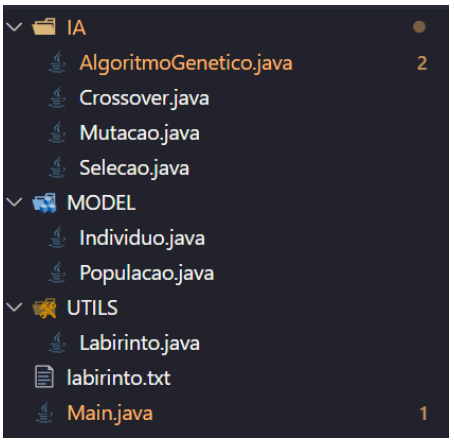

Figura 3 - Arquitetura

O diretório UTILS é onde desenvolvemos a classe Labirinto, ela é a responsável por realizar o carregamento do labirinto, para a exibição do mesmo(Figura 2), utilizamos a biblioteca JPanel do próprio Java

Legenda:

VERMELHO: Posição Inicial{0,0};
PRETO: Barreiras;
BRANCO: Área Livre;
AZUL: Comidas/Objetivos.

Reservamos o diretório MODELS para fazer a modelagem dos objetos do nosso algoritmo, que no caso são os Indivíduos e a Populacao. Para o armazenamento do caminho percorrido(genes) utilizamos *ArrayList* e os movimentos possíveis dentro de um array, no caso os movimentos possíveis são apenas as direções que o indivíduo pode se mover, eles ainda não são calculados e nem verificados. Já a Classe Populacao é a responsável por gerar a população(Geracao), nesta classe também reservamos alguns métodos que serão responsáveis por coletar o melhor indivíduo de cada geração.

Por fim temos o diretório IA, ele será o responsável por realizar toda a parte lógica do nosso algoritmo, contendo as etapas que compõem o algoritmo genético, além de toda a heurística do programa, nele temos as classes: AlgoritmoGenetico, Crossover, Mutacao e Selecao. Antes de explorarmos as classes que compõem o diretório IA, é importante falarmos como elaboramos o cálculo responsável por encontrar as soluções(Cálculo de Aptidão).

O cálculo de aptidão é feito através do conjunto de genes dos cromossomos que cada indivíduo possui, os genes são o caminho que cada indivíduo percorreu dentro do labirinto, exemplo: baixo, direita, cima, esquerda, e diagonais. Para cada movimento que o indivíduo realiza para fora dos limites do labirinto, é descontado de sua aptidão 100 pontos, no caso o objetivo final é que o indivíduo que possua a melhor solução tenha a melhor pontuação. Quando um indivíduo encontra uma barreira, ele também é penalizado com 100 pontos, para estimular a movimentação dos indivíduos dentro do labirinto, não realizamos a penalização sobre a quantidade de movimentos, quando um indivíduo percorre um caminho ainda não visitado, ele pontua 2 pontos, caso o caminho já tenha sido visitado ele acrescenta 1 ponto em sua aptidão, quando o indivíduo encontra um dos objetivos(comida) ele acrescenta 30 pontos.

LEGENDA	
Movimento Fora Labirinto	-100 Pontos
Barreira Encontrada	-100 Pontos
Caminho já visitado	+1 ponto
Caminho nunca visitado	+2 pontos
Comida encontrada	+30 pontos

Visando ter um melhor precisão utilizamos alguns parâmetros para auxiliar neste objetivos, são eles: Quantidade da População igual á 1000 para termos uma grande diversidade de possibilidades de indivíduos para serem possíveis soluções para o nosso problema; quantidade de gerações definimos como 300; e quantidade limite de movimentos, que definimos como 75. Além destes 3 parâmetros, temos também as taxas de crossover e de mutação. Para evitarmos uma convergência, utilizamos uma taxa de mutação relativamente baixa, em torno de 30% e usamos uma taxa de crossover em 70%.

Na classe Selecao temos 2 funções no qual são as responsáveis pelo método de seleção a ser aplicada para gerar a nova população(População Intermediária), o elitismo e o torneio, a seleção por elitismo pega o melhor indivíduo da geração atual e disponibiliza na nova geração(População Intermediária). Já o método de torneio irá pegar 2 indivíduos de forma aleatória e irá verificar qual é o melhor indivíduo. Após a seleção, temos o Crossover, onde utilizamos o método uniponto, ou seja, iremos pegar 2 indivíduos(Pai e Mãe), definimos um ponto de corte em seus genes, e cruzamos seus cromossomos para gerar dois novos indivíduo(Filhos) diferentes. A demonstração do funcionamento do método escolhido está nas tabelas abaixo.

MÃE	6	9	7	3	2	8	6	1	4	7

PAI	2	6	6	8	9	6	3	2	2	3

FILHO1	6	9	7	3	2	6	3	2	2	3
FILHO2	2	6	6	8	9	8	6	1	4	7

Após o processo de crossover, temos o que chamamos de mutação, o qual é o responsável por realizar a troca(mutação) de **n** genes de um indivíduo da população intermediária, o que irá definir a quantidade de genes é a taxa de mutação. Após realizar a mutação é feito um novo cálculo de aptidão para verificarmos se conseguimos encontrar a solução desejada.

GENES ANTES DA MUTAÇÃO

FILHO1	6	9	7	3	2	6	3	2	2	3

GENES APÓS A MUTAÇÃO

FILHO1	6	9	5	3	2	6	3	2	2	3

Após realizado todo o ciclo de desenvolvimento do algoritmo, é possível verificarmos a melhor solução encontrada com base na aptidão do indivíduo. As informações que serão exibidas no terminal serão:
Geração do indivíduo, Total de comidas encontradas, Genes do melhor indivíduo e sua pontuação final (aptidão).