UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
FACULDADE DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE SISTEMAS DE INFORMAÇÃO

Trabalho Prático II

Discentes: Adriano Soares de Oliveira
Liliane Lopes Vieira
Renato Alves Gonçalves

Professor: Leonardo Lana de Carvalho
Disciplina: Inteligência Artificial

Diamantina
22 de Setembro de 2017

Introdução:

Os macacos Vervets desenvolveram um meio eficiente para aumentar a proteção dos membros do grupo de seus predadores, esses animais usam alarmes para informar a presença de seus predadores, a águia, o tigre, a cobra entre outros.
O alarme tem uma função adaptativa, pois:

- Se um tigre se aproxima, o comportamento de fuga é o de ir para os galhos mais finos da árvore.
- Se uma águia se aproxima, o comportamento de fuga é o de ir para a copa da árvore.
- Se uma cobra se aproxima é preciso ficar atento a relva e a folhagem.

Cada alarme leva a um comportamento de fuga que é eficiente para um tipo de predador, mas pode não ser para outro.
Todavia os macacos não nascem com este processo sígnico compartilhado pelos outros.
Para que a linguagem traga vantagens adaptativas ao vervets, eles precisam compartilhar a dupla Alarme-Predador.

A Modelagem:

- Matriz Ambiente : Matriz Amb[50][50]**.**
- Vetor de agentes (Macacos) do tamanho M**:** Macacos[M].
- Vetor Predadores de tamanho P: Predador[P].
- Vetor Símbolos tamanho 10: S[10].
- Vetor Alarmes tamanho 10 : Alarmes[10].

Cada macaco possui um vetor com 10 símbolos, sendo cada um associado a um tipo de predador, como por exemplo, o vetor S de posição 1 (S[1]) está associado ao predador do tipo Cobra. Cada predador possui um peso F gerado de forma aleatória.

Etapa 1: Criar e inicializar o ambiente(Matriz 50x50).
Etapa 2 :Criar as espécies de predadores.
Etapa 3: Gerar agentes Vervet.
Etapa 3: Gerar predadores.
Etapa 4: Gerar Matriz de Conhecimento.
Etapa 5: Popular o ambiente com Agentes e Predadores.
Etapa 6: Deslocar os agentes aleatoriamente e soar o alarme ao avistar um predador.
Etapa 7: Gerar planilha de conhecimento.

Regras:

1- Da Visão: Por definição, os agentes macacos possuem um alcance de visão(V) de um raio de 5 casas para frente, para trás, para direita e duas casas para a esquerda. Ao avistar um predador dentro de seu campo de visão, este agente irá consultar sua tabela de Símbolos e verificar de qual predador se trata aquele que está no seu campo de visão, ao identificar o predador, o agente irá soar o alarme que do de acordo com as coordenadas (Símbolo,Predador).

2- O Alarme: Um macaco ao perceber que o alarme S[x] está sendo tocado e um predador P[X] está em seu campo de visão irá combinar as coordenadas (número do alarme,número do predador) e aumentar o valor presente nesta coordenada em 0,1.

O que o algoritmo não irá fazer:

1- Matar macacos: Após um encontro entre um macaco e predador, o algoritmo registra os dados necessário e volta ao sua execução normal sem remover um predador ou macaco do ambiente.

2- Criar comportamentos: O Algoritmo não irá expressar ações como por exemplo : ir para galhos mais finos da árvore ao perceber um tigre se aproximando, ele apenas irá adicionar o peso desta ação ao vetor de conhecimento correspondente.

Modelagem gráfica:

		Peso de alarme por predador							
	Tipo S1	P0	P1	P2	P3	P4	P5	P6	P50
S1	Águia	0,1	0,2	0,3	0,4	0,5	0,6	0,7	
S2	Tigre	1,1	1,2	1,3	1,4	1,5	1,6	1,7	
S3	Leão	2,1	2,2	2,3	2,4	2,5	2,6	2,7	
S4	Cobra	3,1	3,2	3,3	3,4	3,5	3,6	3,7	
S5	onça	4,1	4,2	4,3	4,4	4,5	4,6	4,7	
S6	jacaré	5,1	5,2	5,3	5,4	5,5	5,6	5,7	
S7	Humano	6,1	6,2	6,3	6,4	6,5	6,6	6,7	
S8	jaguatirica	7,1	7,2	7,3	7,4	7,5	7,6	7,7	
S9	Suçuarana	8,1	8,2	8,3	8,4	8,5	8,6	8,7	
S10	gavião	9,1	9,2	9,3	9,4	9,5	9,6	9,7	

Pesos gerados aleatoriamente para cada predador

Onde S é o vetor de Símbolos e Px são colunas para cada predador, ou seja: Cada linha da matriz corresponde a uma espécie de predador e cada coluna corresponde a um predador específico, por exemplo : M[1][3] Espécie= Águia, ID do Predador: Águia3 valor=0,4.
Acime é exibido um exemplo representativo do ambiente e da disposição de cada agente e predador, cada agente possui um de 5 casas para direita,esquerda ,acima e abaixo e um radio auditivo de 15 casas para qualquer direção.

Explicação das Funções

Mapa do ambiente

Ambiente de tamanho 50x50

		Agente		
	Agente		**Predador**	**Agente**
Agente		**Predador**		
Predador		**Agente**		**Predador**

A codificação do algoritmo foi realizada na linguagem de programação java utilizando a IDE de desenvolvimento NetBeans 8.1.

Para a construção do projeto foi criado as classes Agente, Predador e Símbolo, onde cada uma com os seguintes atributos:

Agente:

- Código
- Nome
- Espécie
- Vetor de Símbolos

Predador

- Código
- Nome
- Espécie

Símbolos

- Código

- Símbolo

Funções:

IncializaEspecies()

```
private void IncializaEspecies() {

    Especie.add("Tigre");//0
    Especie.add("Cobra");//1
    Especie.add("Aguia");//2
    Especie.add("Jaguatirica");//3
    Especie.add("Onca");//4
    Especie.add("Suçuarana");//5
    Especie.add("Gavião");//6
    Especie.add("Jacaré");//7
    Especie.add("Humano");//8
    Especie.add("Leão");//9

}
```

InicializaSimbolos()

```
private void InicializaSimbolos() {
    for (int i = 0; i < 10; i++) {
        //Cria uma matriz do tipo:
        //                        Codigo Simbolo
        //                          0     Tigre
        //                          1     Cobra
        Simbolo simbolo = new Simbolo(i, Especie.get(i));
        Simbolos.add(simbolo);
    }

}
```

InicializaAmbiente()

```
private void InicializaAmbiente() {
    for (int i = 0; i < 50; i++) {
        for (int j = 0; j < 50; j++) {
            Ambiente[i][j] = "0";
        }
    }
}
```

GeraAgente()

```
private void GeraAgentes() {
    Random GeraAleatorio = new Random();
    quantidadeDeAgentes = GeraAleatorio.nextInt(quantMaxAgentes);
    jLabel4.setText(String.valueOf(quantidadeDeAgentes));
    for (int i = 0; i < quantidadeDeAgentes; i++) {
        Agente vervet = new Agente();
        vervet.setCodigoAgente(i, "Macaco");
        Agentes.add(vervet);
    }

}
```

GeraPredadores()

```
private void GeraPredadores() {
    Random GeraAleatorio = new Random();
    quantidadeDePredadores = GeraAleatorio.nextInt((quantMaxPredadores));
    jLabel6.setText(String.valueOf(quantidadeDePredadores));
    int tipoPredador = 0;
    for (int i = 0; i < quantidadeDePredadores; i++) {
        Predador predador = new Predador();
        tipoPredador = GeraAleatorio.nextInt(Especie.size());
        predador.setCodigoPredador(i, Especie.get(tipoPredador));
        Predadores.add(predador);
    }
}
```

GeraMatrizDeConhecimento

```
private void GeraMatrizDeConhecimento() {
    //A combinação do vetor Simbolos , com o vetor de predadores gera uma matriz que sera adicionada a cada vervet
    ArrayList<ArrayList<Double>> S = new ArrayList<>();//Matriz de adjascencia Pesos
    ArrayList<Double> PesosPredadores = new ArrayList<>();
    Random GeraAleatorio = new Random();
    double peso = 0;
    for (int i = 0;i< Simbolos.size();i++ ) {
        for (int j = 0; j < Predadores.size(); j++) {
            peso=GeraAleatorio.nextDouble();//Gerar valores entre 0 e 1
            PesosPredadores.add(peso);
        }
        S.add(PesosPredadores);
    }
    //Adiciona a matriz de Conhecimento S para cada Vervet
    for (int i = 0; i < Agentes.size(); i++) {
        Agentes.get(i).setSimbolos(S);
    }
    for (int i=0;i<Agentes.size();i++) AgentesIni.add(Agentes.get(i));
}
```

JornadaDosVervets()

```
private void JornadaDosVervets() throws InterruptedException {
    Random GeraAleatorio = new Random();
    int x = 0, y = 0;//Coordenadas
    Jornada = true;

    int cont = 0;
    int Contador=0;
    //while (Jornada) {
    while (Contador<3000) {
        jLabel2.setText("Número de Iterações: " + cont++);
        x = GeraAleatorio.nextInt(50);//gera uma Cooerdenada X aleatoria entre 0 e 50
        y = GeraAleatorio.nextInt(50);//gera uma Cooerdenada Y  aleatoria entre 0 e 50
        if (!"0".equals(Ambiente[x][y])) {//Se a celula não estiver vazia
            char letra = Ambiente[x][y].charAt(0);
            if (letra == 'M') {
                System.out.println("Visão verver "+x +","+y);
               // jTextArea1.setText("o Vervet "+Ambiente[x][y]+"está se sentindo ameçado! \n"+jTextArea1.getText());

                VisaoVervet(x, y, Ambiente[x][y]);
                MovimentaAgente(x,y);
            }

        }
        Contador++;
    }
}
```

VisaoVervet()

```
private void VisaoVervet(int x, int y, String Vervet) {
    //Essa função verifica o se existe algum predador no campo de visão de um vervet
    Agente agente = new Agente();
    int codigoAgente = RetornaCodigoDoAgente(Vervet);
    agente = Agentes.get(codigoAgente);//copia o objeto Agente
    int esquerda = x - TamanhoDaVisaoAgente;
    int direita = x + TamanhoDaVisaoAgente;
    int acima = y + TamanhoDaVisaoAgente;
    int abaixo = y - TamanhoDaVisaoAgente;
    Double  Alarme[] = new Double[2];//Primeira posição é o alarme, a segundo o peso
    Alarme[0] = -1.0;//Especie do predador
    Alarme[1] = -1.0;//Codigo do predador
    //impedir que receba uma coordenada fora do ambiente
    if (esquerda <0) {
        esquerda = 0;
    }
    if (direita > 50) {
        direita = 50;
    }
    if (acima > 50) {
        acima = 50;
    }
    if (abaixo < 0) {
        abaixo = 0;
    }
    //Verificar se ha predador no raio de visão do vervet
    for (int i = esquerda; i < direita; i++) {
        for (int j = abaixo; j < acima; j++) {
            char letra = Ambiente[i][j].charAt(0);
            if (letra == 'P') {
                int codPred = RetornaCodigoDoAgente(Ambiente[i][j]);
                for(int k=0;k<agente.Simbolos.size();k++){
                    if(agente.Simbolos.get(k).get(codPred)>Alarme[1]){
                        Alarme[0]=Double.valueOf( k);//Codigo do alarme
                        //Alarme[1]=agente.Simbolos.get(k).get(codPred);//codigo do predador
                        Alarme[1]=Double.valueOf(codPred);//codigo do predador
                        if(Alarme[1]>(-1)){
                         //  jTextAreal.setText("O vervet "+Ambiente[i][j]+" Acaba de avistar um predador"+"\n"+jTextAreal.getText());
                        TocaAlarme(Alarme,i,j);//Passa como parâmetro a tipo de alarme e a posição do predador
                    }
```

TocaAlarme();

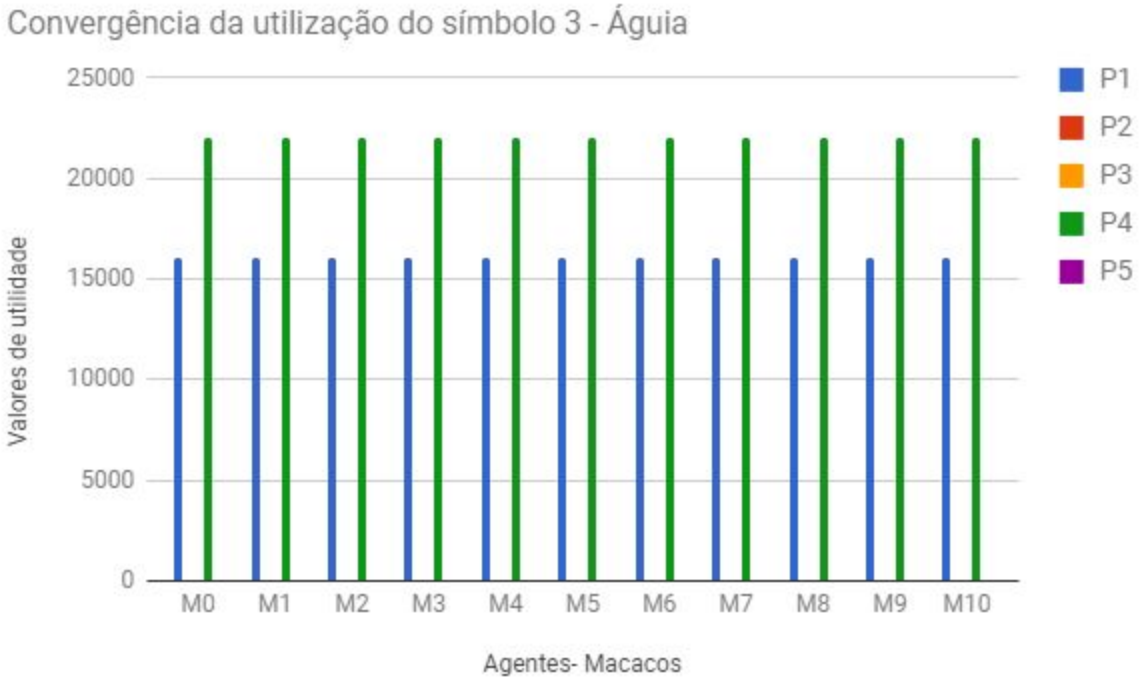
```
private void TocaAlarme(Double[] Alarme, int x, int y) {
   //Por definição um vercet tem um alcance de visão de 15 quadros para cada lado
   int CodSimbolo=Alarme[0].intValue();
   int CodPredador=Alarme[1].intValue();
   int esquerda = x - TamanhoDaVisaoAgente;
    int direita = x + TamanhoDaVisaoAgente;
    int acima = y + TamanhoDaVisaoAgente;
    int abaixo = y - TamanhoDaVisaoAgente;
    if (esquerda < 0) {
        esquerda = 0;
    }
    if (direita > 50) {
        direita = 50;
    }
    if (acima > 50) {
        acima = 50;
    }
    if (abaixo < 0) {
        abaixo = 0;
    }
    //Verifica na redondeza do alarme se existe algum Agente macaco e se esse agente avistou o  predador
    for (int i = esquerda; i < direita; i++) {
        for (int j = abaixo; j < acima; j++) {
            char letra = Ambiente[i][j].charAt(0);
            if (letra == 'M') {
                int codigoAgente = RetornaCodigoDoAgente(Ambiente[i][j]);
               // jTextAreal.setText("O vervet "+Ambiente[i][j]+" Acaba der soar o Alarme"+"\n"+jTextAreal.getText());
                AgenteVerificaPredadorNaRedondeza(codigoAgente,CodSimbolo,CodPredador,i,j);

            }}
```

Análises de resultados

Visando estruturar a análise de resultados, foram elaborados gráficos de iterações x agentes para cada um dos símbolos possíveis. Dessa forma, procedeu-se à análise do índice de convergência da utilização do símbolo para um dado predador.


Convergência da utilização do símbolo 1 - Cobra
20000
15000
10000
5000
0
Valores de utilidade
M0
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
Agentes - Macacos
P1
P2
P3
P4
P5


Convergência da utilização do símbolo 3 - Águia
25000
20000
15000
10000
5000
0
Valores de utilidade
M0
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
Agentes- Macacos
P1
P2
P3
P4
P5

Conclusão

Como sugestão de melhoria do código implementado sugere-se a inclusão de aprendizagem também para os predadores, que poderiam responder as ações dos macacos e estes aprenderem com as consequências sofridas no ambiente, tipo a morte de outros macacos.

A maior dificuldade encontrada para a realização deste trabalho foi relacionada a modelagem do problema. O grupo conclui que o sucesso da implementação depende basicamente do entendimento do problema e modelagem, sendo esta o pilar de todo o desenvolvimento da solução.

A inteligência artificial disponibiliza diversas ferramentas para a solução de problemas do dia a dia. O aprendizado por reforço se mostrou eficiente na criação de uma semântica para o mundo dos macacos. Com isso, vislumbra-se a aplicação dessa técnica no ensino a distância, em jogos, interação de robôs , dentre outras aplicações.