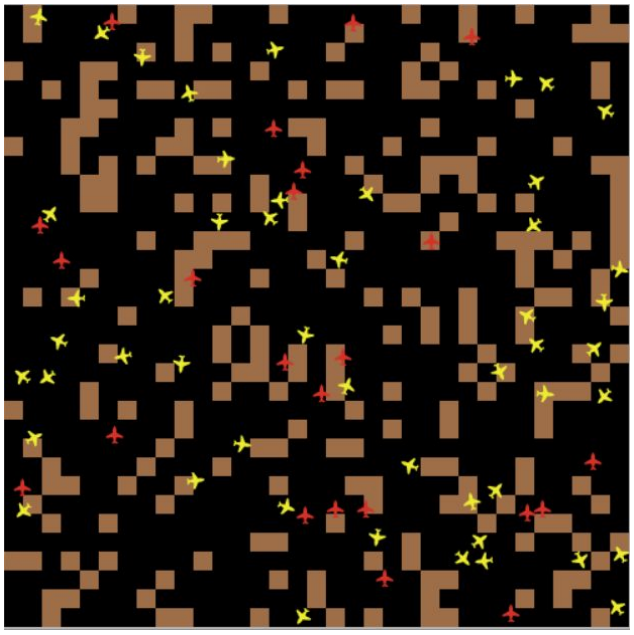
Instituto Superior de Engenharia de Coimbra
Cadeira de Introdução à Inteligência Artificial
Ano Lectivo 2018/2019

TRABALHO PRÁTICO Nº1



Diogo Oliveira Braz Monteiro - 21270484
Ricardo Trindade e Silva Alvim Cardoso - 21270943

Índice

Introdução

Um agente racional é uma entidade, que habita um determinado ambiente, capaz de percecionar o seu meio envolvente e agir sobre este de maneira a que o agente atinja o maior sucesso. Esta racionalidade depende de quatro fatores:

→ conhecimento inicial do ambiente
→ perceção sequencial
→ ações que pode tomar
→ função usada para avaliar o sucesso

Idealmente, o agente para uma determinada sequência, age com o objetivo de ter uma maximização da função de medida do seu sucesso, baseando-se na sua perceção sequencial e conhecimento inicial do ambiente. Neste trabalho, pretende-se implementar todos os fatores de um agente racional de forma a aplicarmos os conhecimentos adquiridos tanto nas aulas teóricas como nas aulas práticas.

Objetivos

Implementação e análise de comportamentos racionais para agentes reativos ao ambiente. Foi-nos fornecido um conceito base, onde os agentes têm comportamentos de acordo com a sua especificação seja ela uma Mosca, MoscaEsteril ou OvoMosca.

O Ambiente

O ambiente consiste numa grelha bidimensional fechada, onde habitam três tipos de agentes:

→ **Moscas**: estes agentes percepcionam as células à sua frente, atrás de si, à esquerda e direita (neighbors4), a ação prioritária destes agentes é a da interação e só depois a da alimentação, possuem memória do seu nível de energia.

→ **MoscaEsteril**: estes agentes percepcionam as células que se encontram à sua volta (neighbors), a ação prioritária destes agentes é a da interação, possuem memória do seu nível de energia.

→ **OvoMosca**: estes agentes não percepcionam nem se movimentam, apenas possuem memória do número de agentes que irá criar.

→ **Alimento**: aumenta o valor de energia do agente que a consumir (valores modificáveis).

Objetivo principal

Efetuar alterações no ambiente e no comportamento dos agentes de modo a poder conter/eliminar uma praga de moscas.

Implementação

Comportamento dos Agentes

Moscas

Os agentes Moscas têm comportamento específico de acordo com a sua percepção do ambiente que os rodeia. Eles priorizam a interação entre outros agentes e só depois a alimentação. Este tipo de agente possui memória onde guarda o valor da energia e da fertilidade. Sempre que um agente Mosca percecionar um outro agente do mesmo gênero ele cria um ovo (um ovo por agente visível).

Fig 1 - Área de perceção dos agentes Moscas

MoscaEsteril

Os agentes MoscaEsteril verificam também um comportamento específico de acordo com a sua percepção do ambiente que os rodeia no entanto a sua priorização continua a ser a interação, este agente não se alimenta. Como os agentes Moscas este também possui memória, mas só guarda o valor da energia. Sempre que existe interação com uma Mosca, baixa a fertilidade da Mosca em 0-10% (configurável). Quando existe interação com um agente do mesmo gênero, o agente com menos energia morre. Por fim quando existe interação com um agente do tipo OvoMosca, o Ovo perde um agente dos que ia criar.

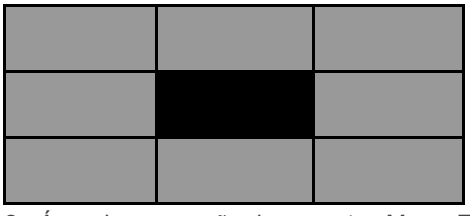


Fig 2 - Área de percepção dos agentes MoscaEsteril

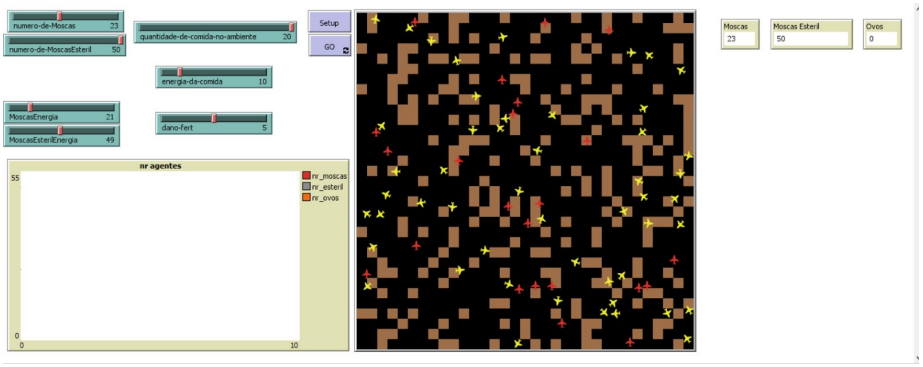
OvoMosca

Os agentes OvoMosca não possuem movimento, não têm percepção e não se alimentam, por outro lado têm memória, onde guardam o número de moscas que irá gerar. Este agente dura um determinado número de ticks, depois de ultrapassar os ticks pré-definidos o agente morre criando o número de moscas que guarda em memória.

Comportamento das células

Células Castanhas - As células castanhas representam a comida em que o seu nascimento é configurável através de um slider na interface da aplicação, com uma percentagem entre 5-20% do total de células do ambiente.
Estas células são responsáveis por manter vivos os agentes que as consomem, através do aumento de energia dos mesmos.

Interface



Sliders:

numero-de-moscas/numero-de-moscasEsteril - Controlam o número de moscas dos dois tipos que nascem no ambiente, entre 0-50.

MoscasEnergia/MoscasEsterilEnergia - São usados para regular a energia dos dois agentes Mosca e MoscaEsteril, no início da simulação, tendo valores entre 0-100.

quantidade-de-energia-no-ambiente - Permite controlar a percentagem de comida que nasce no ambiente, tendo uma percentagem entre 5-20% em relação ao total de células do do ambiente.

energia-da-comida - Controla a quantidade de energia que cada célula de comida vai dar às turtles que a consomem, com valores compreendidos entre 0-50.

dano-fert - Este slider permite controlar a percentagem que cada mosca estéril vai retirar a uma mosca em termos de fertilidade, entre 0-10% do total de fertilidade que a mosca possui.

Botões:
Setup - Permite criar as turtles e preparar o ambiente de simulação.

Go - Permite começar a simulação.

Gráfico:
Permite visualizar o número de agentes existentes em todo o ambiente, sendo estes os Ovos das Moscas, as Moscas e as Moscas Estéreis.

Monitor:
Servem para visualizar o número de agentes ovos, moscas e moscas estéreis existentes no ambiente.

Especificação resumida do código

```
to setup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end

to setup-patches
  set-patch-size 15
  ask patches [
    set pcolor black
    if random 101 < quantidade-de-comida-no-ambiente [
      set pcolor brown

    ]
  ]
   to setup-turtles

     clear-turtles
     create-Moscas numero-de-Moscas [
       set fertilidade random 101
       set shape "airplane"
       set color red
       setxy random-xcor random-ycor
       set heading 0
       set energia MoscasEnergia
     ]

     create-MoscasEsteril numero-de-MoscasEsteril [
       set shape "airplane"
       set color yellow
       set energia MoscasEsterilEnergia
       setxy random-xcor random-ycor
       set fertilidade 0
     ]
   end
```

Estes dois procedimentos servem para fazer o primeiro *setup* do ambiente e das turtles, o setup serve para chamar os outros procedimentos existentes que criam tantos as turtles como os patches.
O procedimentos Setup-patches cria a comida e a define a cor do ambiente.

O procedimento setup-turtles cria os agentes que vão estar no ambiente dando todas as características devidas.

```
to go
  tick
  if ticks >= 1000
  [stop]
  ;if count OvosMosca > 1000
  ; [show "a praga venceu"
  ; stop
  ;]
  if not any? turtles
  [stop]

  move-Moscas
  move-MoscasEsteril

  if count turtles = 0
  [stop]

  ask OvosMosca [
    if criaçao < 1
      [die]

    if birth-tick <= 0[
      hatch-Moscas criaçao
      [
        set fertilidade random 101
        set shape "airplane"
        set color red
        set heading 360
        set energia MoscasEnergia
      ]
      die
    ]
    set birth-tick (birth-tick - 1)
  ]
end
```

Este procedimento faz com que a aplicação inicie, fazendo com que esta não passe dos 1000 ticks, e caso não haja turtles a aplicação para

Chama os procedimentos para mover as moscas

Neste bloco caso haja algum ovo sem número de moscas para criar ele morre

Neste bloco caso algum ultrapasse os ticks definidos ele cria as moscas e morre.

Este procedimento “move-moscas” é o procedimento que origina os movimentos das moscas.

```
to move-Moscas
  ask Moscas [

    set espaçoVazio neighbors4
    ifelse any? espaçoVazio[
      set direçao one-of espaçoVazio
      face direçao
      move-to direçao
    ]
    [
      mexe
    ]

    set FertTemp fertilidade
    if count Moscas-on neighbors4 > 0
    [
      ask Moscas-on neighbors4
      [
        set OutraFert fertilidade
        set criar ((FertTemp + OutraFert) / 20 )

        hatch-OvosMosca 1
        [
          set shape "circle"
          set color white
          set birth-tick 10
          set criaçao criar
        ]
      ]
    ]

    ifelse [pcolor] of patch-here = brown
      [
        ask patch-here [set pcolor black]
        set energia energia-da-comida
    ]
    [
      if any? neighbors4 with [pcolor = brown]
      [comer-comida]
    ]
    set energia energia - 1
    death
  ]
end
```

Este bloco de código faz com que a mosca se direcione para uma patch vazia e se mova para ela.

Nesta parte de código verifica-se se existe moscas nas vizinhanças e cria-se um ovo.

Neste bloco a mosca verifica se existe comida e chama o procedimento comer-comida.

```
to move-MoscasEsteril
  ask MoscasEsteril [

    ifelse count turtles-on neighbors > 0 [
      if count Moscas-on neighbors > 0
      [ask Moscas-on neighbors
        [set fertilidade (fertilidade * (dano-fert / 100))]
      ]
      set aux1 energia
      if count MoscasEsteril-on neighbors > 0 [
        ask MoscasEsteril-on neighbors [
          set aux2 energia
          ifelse aux2 < MoscasEsterilEnergia * 0.9[
            if aux2 > aux1[
              set morre 1
              stop
            ]
            if aux1 > aux2 [
              set aux1 (aux1 + energia)
              die]
            if aux1 = aux2 [
              mexe
            ]
          ]
          [
            mexe

          ]
        ]
      ]
```

Neste bloco de código é retirada a fertilidade das moscas que sejam percepcionadas pela mosca estéril.

Esta parte é usada para que a mosca estéril encontrando outra do mesmo roube a sua energia e mate a que tem maior energia.

```
    set EnergMaior 0
    ifelse count Moscas-on patch-here >= 2[
      set breed Moscas
      set fertilidade random 101
      set color red
      ask moscas-on patch-here[
        if energia > EnergMaior
        [set EnergMaior energia]
      ]
    ]
    [ifelse count MoscasEsteril-on patch-here >= 2
      [
        set breed Moscas
        set fertilidade random 101
        set color red
        set energia 1
      ]
      [if count OvosMosca-on patch-here >= 2
        [
          set breed Moscas
          set fertilidade random 101
          set color red
          set energia 1 ]
      ]
    ]

    set energia (energia - 1)

    death
  ]
end
```

Neste bloco se a mosca estéril percecionar um ovo vai lhe decrementar a criação de moscas em 1 valor.

Neste bloco de código a mosca estéril verifica se existem 2 ou mais agentes a sua volta na mesma patch e caso aconteça cria moscas.

Neste bloco de código decrementa-se a energia em cada iteração e chama-se o procedimento death.

```
to comer-comida
  face one-of neighbors4 with [pcolor = brown]
  fd 1
  if pcolor = brown
  [
    ask patch-here [set pcolor black]
    set energia energia-da-comida
  ]

  ask Moscas[
    death
  ]


end

to death
  if energia < 0
  [die]
end

to mexe
  set varNum (random 2)
  ifelse varNum = 0
  [
    set heading 90 fd 1
  ]
  [
    set heading -90 fd 1
  ]
end
```

Neste procedimento verifica-se as patches vizinhas diretas e caso haja comida nelas a mosca vai direcionar-se para elas e comer.

Este procedimento verifica se a energia é menor que 0 e caso seja o agente morre.

Este procedimento faz com que sejam devolvidos dois valores aleatórios para que as moscas se movimentem aleatoriamente para a esquerda ou para a direita.

Conclusão

Concluímos com este trabalho que não temos um plano certo para a contenção da praga, contudo com as experiências efetuadas podemos afirmar que para um maior número de sucesso teríamos de aumentar mais de dez vezes o número de Moscas Estéreis em relação às Moscas. Outro plano para a contenção seria aumentar o dano de fertilidade das Moscas Estéreis, tendo na mesma de ter um número muito superior de Moscas Estéreis em relação às Moscas.

A quantidade de alimento no ambiente é um problema para a contenção da praga, pois quanto mais alimento houver no campo mais energia as Moscas irão ter, que por sua vez lhes darão mais tempo para encontrarem outro agente para se reproduzirem.

Os planos anteriormente mencionados para a contenção da praga não são certos, porque a percentagem de sucesso não foi 100%.

Para comprovar que o sucesso dos planos de controlo de praga não foram 100% segue em anexo os testes que realizámos.