Instituto Superior de Engenharia de Coimbra

Licenciatura Engenharia Informática Pós-Laboral



Trabalho Prático 2

Introdução à Inteligência Artificial

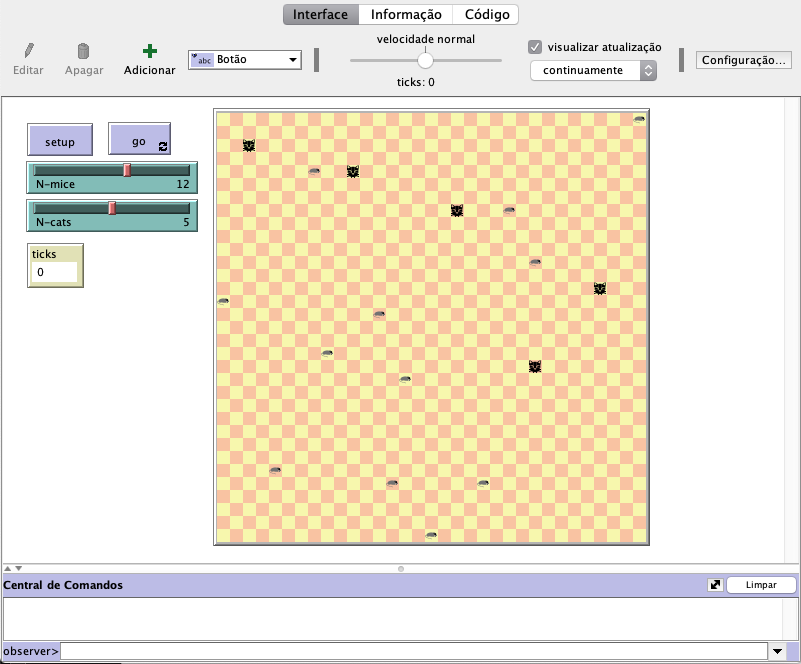
Índice

1. Introdução .........................................................3
2. Algoritmos utilizadas ........................................4
3. Hipóteses estudadas ..........................................5
4. Anexos .............................................................. 7
5. **Introdução**

No trabalho, a seguir apresentado, é nosso intuito simular, utilizando o software “NetLogo”, diversas possibilidades de interação entre ratos e gatos, passando pela utilização de diversas estratégias entre os mesmos, conforme se descreve mais adiante.

Passaremos pela utilização de “spreadsheets” e tabelas resultantes da utilização da funcionalidade “BehaviorSpace” do referido software de simulação, para a análise dos resultados obtidos pelas diferentes experiências realizadas no decorrer do processo.

1. **Estratégias utilizadas**

Uma imagem com texto

Descrição gerada automaticamenteIniciámos o estudo a partir da simulação facultada pelo docente, Fig. 1 e 2, averiguando o comportamento dos agentes no ambiente definido. Após o registo dos dados obtidos e após alterações realizadas no processo de simulação, nomeadamente, acrescentando a possibilidade de alteração dos movimentos dos agentes em causa, ratos e gatos, aumentando ou diminuindo os valores de movimento dos mesmos, criando a possibilidade de resguardo dos ratos em esconderijos criados, aumentando o risco de sobrevivência dos ratos com a colocação de armadilhas, procurando aumentar a dificuldade de sobrevivência dos ratos no ambiente, uma vez que a passagem nestas *patches* provoca a eliminação instantânea dos agentes (ratos) ao contrário dos esconderijos onde estes, uma vez nessa *patch* e durante 3 segundos, se encontram salvaguardados dos “gatos” não podendo por isso ser caçados. Foi também criada a possibilidade de alimentação e reprodução dos agentes. O código utilizado na realização das restantes “soluções” encontradas encontra-se em anexo, bem como a tabela de resultados obtidos.

Fig. 2

Fig. 1

**3.** **Hipóteses utilizadas**

No primeiro caso de estudo, fig. 1, proposta facultada pelo docente, onde apenas existe a possibilidade de movimentação dos agentes sem restrições em qualquer das partes, possibilitando aos gatos eliminar os ratos sem que estes tenham a hipótese de fuga, o fim do processo, conforme indicado em “spreadsheet” anexa, termina rapidamente, ao fim de aproximadamente 120,6 *ticks*, em média, durante a realização de 30 execuções.

Nas hipóteses estudadas a seguir, todas as simulações para além de possuírem 30 execuções de amostra, as mesmas são realizadas alternando os movimentos de fuga dos *agentes* (ratos) ao percecionarem a presença de *agentes* de caça (gatos) entre 1 *patch*, 3 e 5 de distância.

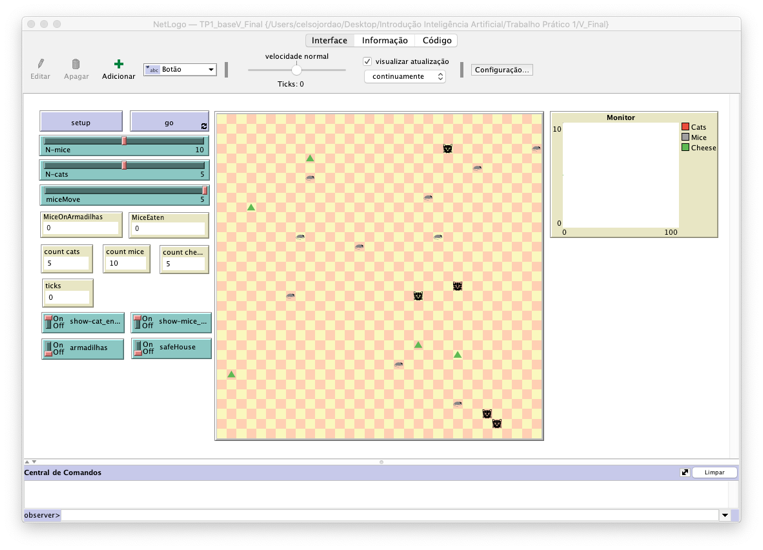
****

Fig. 3

Na segunda hipótese estudada, ao adicionar pontos de alimentação, *agentes* de forma triangular verde, permitindo a reprodução dos *agentes* de fuga (ratos), fig. 3, a possibilidade de sobrevivência dos mesmos aumenta para os 168,7 *ticks*, o que diminui a eficácia de caça por parte dos *agentes* caçadores (gatos).

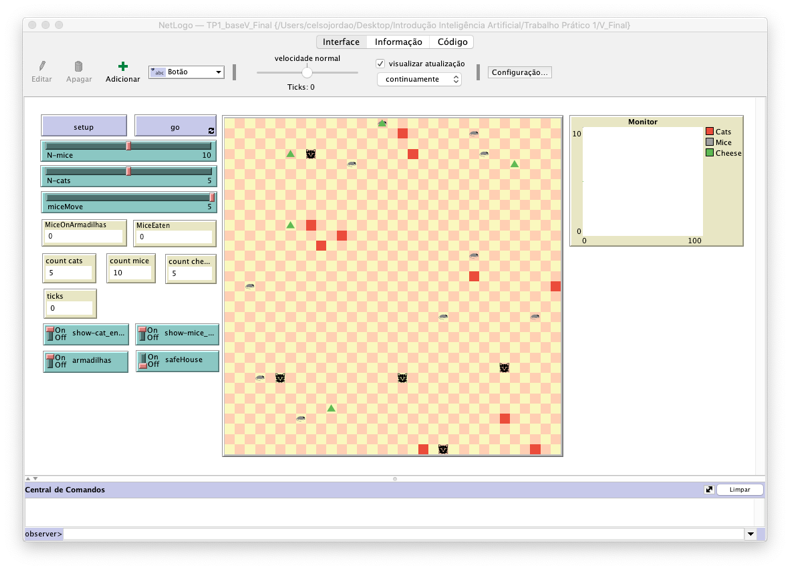


Fig. 4

Ao adicionarmos à simulação pontos de eliminação de *agentes* de fuga, fig. 4, armadilhas em que ao passar por essas *patches* de cor vermelha, onde apenas estes agentes, uma vez que os *agentes* de caça têm a perceção das mesmas evitando-as, são eliminados. Ao fim das 30 execuções de simulação verifica-se que o número médio de duração de cada uma diminuí para os 107,7 *ticks*.

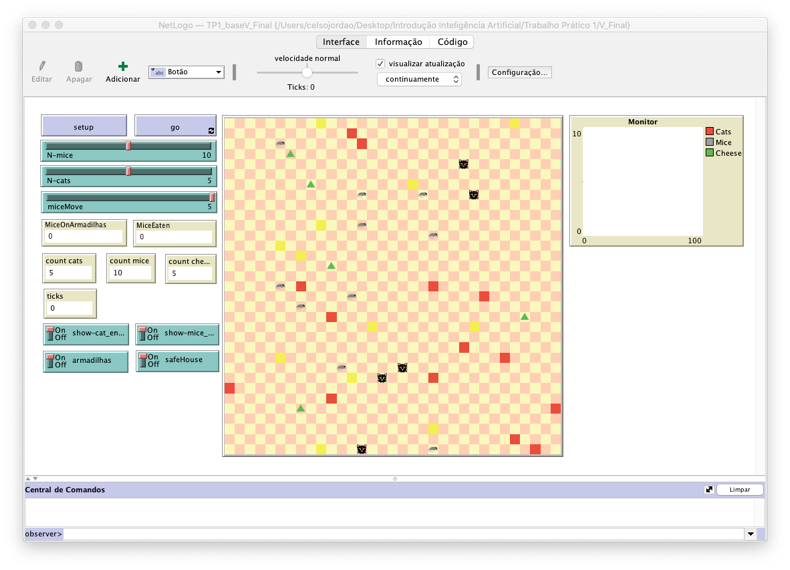
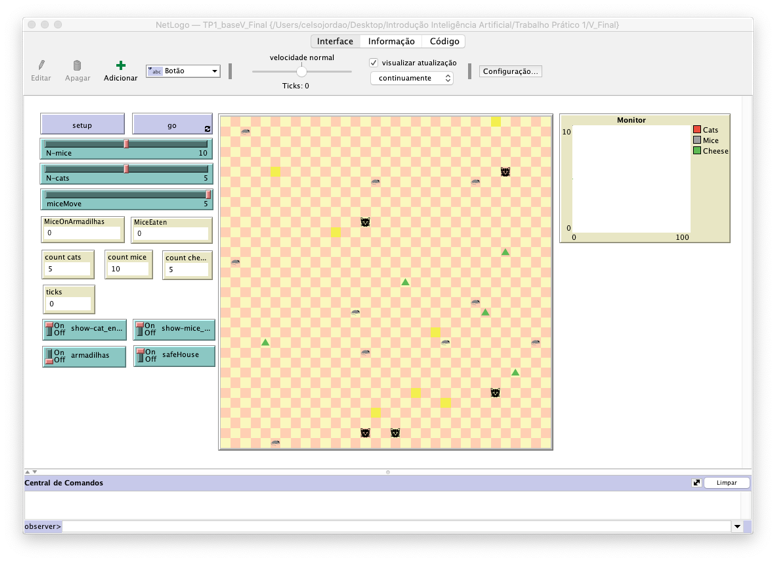


Fig. 5

Adicionando ao ambiente de simulação a existência de pontos de fuga *patches* amarelas, (safeHouses), e mantendo as armadilhas (armadilhas), fig. 5, pontos esses onde os *agentes* de fuga têm a possibilidade de proteção, evitando deste modo, durante três (3) segundos, os *agentes* de caça, não se verifica um grande aumento de sobrevivência dos *agentes* de fuga, 108,2 *ticks*, uma vez que os *agentes* de caça percecionam a existências desses pontos.



Por fim, removendo da simulação anterior as *patches* de armadilha, verifica-se um aumento significativo de sobrevivência, 173,2 *ticks*.

1. **Anexos**

Código utilizado nas simulações:

breed[cats cat]

breed[mice mouse]

breed[triangles triangle]

globals [a b c d w z N\_micedie N\_miceEat]

mice-own [mice\_energy]

cats-own [cat\_energy]

to setup

ca

setup-patches

setup-agents

reset-ticks

end

to setup-patches

ask patches[

let x 28

let y 48

if pycor mod 2 = 0

[set x 48 set y 28]

ifelse pxcor mod 2 = 0

[set pcolor x]

[set pcolor y]

]

ask patches

[

if armadilhas

[

if random 100 < 1

[set pcolor red]

]

]

ask patches

[

if safeHouse

[

if random 100 < 1

[set pcolor yellow]

]

]

end

to setup-agents

create-mice N-mice

[

set shape "mouse side"

set color 4

let x one-of patches with [not any? mice-here and not any? mice-on neighbors and not any? cats-here and not any? triangles-on neighbors and not any? triangles-here]

setxy [pxcor] of x [pycor] of x

set heading one-of [0 90 180 270]

if pcolor = red [setxy random-pxcor random-pycor]

if pcolor = yellow [setxy random-pxcor random-pycor]

]

create-cats N-cats

[

set shape "cat"

set color black

let x one-of patches with [not any? mice-here and not any? mice-on neighbors and not any? cats-here and not any? triangles-on neighbors and not any? triangles-here]

setxy [pxcor] of x [pycor] of x

set heading one-of [0 90 180 270]

if any? mice-here [setxy random-pxcor random-pycor]

if any? triangles-here [setxy random-pxcor random-pycor]

if pcolor = red [setxy random-pxcor random-pycor]

if pcolor = yellow [setxy random-pxcor random-pycor]

]

create-triangles 5

[

set shape "triangle"

set color 55

let x one-of patches with [not any? mice-here and not any? mice-on neighbors and not any? cats-here and not any? cats-on neighbors]

if not any? mice-here [setxy random-pxcor random-pycor]

if not any? cats-here [setxy random-pxcor random-pycor]

if pcolor = red [setxy random-pxcor random-pycor]

if pcolor = yellow [setxy random-pxcor random-pycor]

]

end

to go

move-mice

move-cats

fugir

lunch-time

mice\_lunch

feed

cat\_repro

mice\_repro

mice\_die

tick

if count mice = 0 [stop]

end

to move-mice

ask mice[

fugir

let x one-of neighbors

move-to x

ifelse show-mice\_energy

[ set label mice\_energy ]

[ set label "" ]

]

end

to move-cats

ask cats[

if patch-ahead 1 != nobody [set a patch-ahead 1]

if patch-ahead 2 != nobody [set b patch-ahead 2]

if patch-right-and-ahead 90 1 != nobody [set c patch-right-and-ahead 90 1]

if patch-right-and-ahead -90 1 != nobody [set d patch-right-and-ahead -90 1]

if patch-right-and-ahead 45 1 != nobody [set w patch-right-and-ahead 45 1]

if patch-right-and-ahead -45 1 != nobody [set z patch-right-and-ahead -45 1]

if patch-ahead 2 = red [bk 2]

if patch-ahead 2 = yellow [bk 2]

if patch-right-and-ahead 90 1 = red [set c patch-right-and-ahead 90 1]

if patch-right-and-ahead -90 1 = red [set d patch-right-and-ahead -90 1]

if patch-right-and-ahead 45 1 = red [set w patch-right-and-ahead 45 1]

if patch-right-and-ahead -45 1 = red [set z patch-right-and-ahead -45 1]

if patch-right-and-ahead 90 1 = yellow [set c patch-right-and-ahead 90 1]

if patch-right-and-ahead -90 1 = yellow [set d patch-right-and-ahead -90 1]

if patch-right-and-ahead 45 1 = yellow [set w patch-right-and-ahead 45 1]

if patch-right-and-ahead -45 1 = yellow [set z patch-right-and-ahead -45 1]

let y (patch-set a b c d w z)

let x one-of y

move-to x

if random 100 < 25

[set x [0 90 180 270]]

ifelse show-cat\_energy

[ set label cat\_energy ]

[ set label "" ]

]

end

to fugir

ask mice[

if any? cats-on patch-ahead 2 [left 180 fd miceMove]

if any? cats-on patch-ahead 2 [left 180 fd miceMove]

if any? cats-on patch-right-and-ahead 0 2 [left 180 fd miceMove]

if any? cats-on patch-left-and-ahead 0 2 [right 180 fd miceMove]

if any? cats-on patch-right-and-ahead 45 3 [left 180 fd miceMove]

if any? cats-on patch-left-and-ahead 45 3 [right 180 fd miceMove]

if any? cats-on patch-right-and-ahead 90 2 [left 180 fd miceMove]

if any? cats-on patch-left-and-ahead 90 2 [right 180 fd miceMove]

if any? cats-on patch-right-and-ahead 90 2 [left 90 fd miceMove]

if any? cats-on patch-left-and-ahead 90 2 [right 90 fd miceMove]

if any? cats-on patch-right-and-ahead 135 2 [fd miceMove]

if any? cats-on patch-left-and-ahead 135 2 [fd miceMove]

if any? cats-on patch-right-and-ahead 180 3 [left 90 fd miceMove]

if any? cats-on patch-left-and-ahead 180 3 [right 90 fd miceMove]

if any? cats-on patch-right-and-ahead 225.5 2 [left 90 fd miceMove]

if any? cats-on patch-left-and-ahead 225.5 2 [right 90 fd miceMove]

if any? cats-on patch-right-and-ahead 180 2 [fd miceMove]

if patch-here = yellow [wait 5]

]

end

to lunch-time

ask cats[

if any? mice-on neighbors[

set cat\_energy cat\_energy + 1

ask mice[

if any? cats-on neighbors[set N\_miceEat N\_miceEat + 1]

if any? cats-on neighbors[die]

]

]

]

end

to mice\_lunch

ask mice[

if any? triangles-on patch-here [

set mice\_energy mice\_energy + 1

]

]

end

to feed

ask triangles[

if any? mice-on patch-here [die]

]

end

to cat\_repro

ask cats[

if cat\_energy = 10[

set cat\_energy cat\_energy - 9

hatch 1[set cat\_energy 1]

]

]

end

to mice\_repro

ask mice [

if mice\_energy >= 5 [

set mice\_energy mice\_energy - 4

hatch random 6 [ set mice\_energy 1 ]

]

]

end

to mice\_die

ask mice[

if pcolor = red [set N\_micedie N\_micedie + 1]

if pcolor = red [die]

]

end