UNIVERSIDADE FEDERAL DO MARANHÃO

BACHARELADO INTERDISCIPLINAR EM CIÊNCIA E TECNOLOGIA

EECP0008 - INTELIGENCIA ARTIFICIAL

EMANUELLE DA SILVA LAUNE

HENRIQUE ABREU MACEDO

JOSE NUNES DE SOUSA NETO

VINICIUS ANDRE ALMEIDA PEREIRA

TRABALHO DE AGENTES

SÃO LUÍS - MA

DEZEMBRO/2024

EMANUELLE DA SILVA LAUNE (2022043170)

HENRIQUE ABREU MACEDO (2019012019)

JOSE NUNES DE SOUSA NETO (2022003263)

VINICIUS ANDRE ALMEIDA PEREIRA (2022029662)

TRABALHO DE AGENTES

Documento apresentado como requisito parcial de avaliação da disciplina Inteligência Artificial - Turma 02, no curso Bacharelado Interdisciplinar em Ciência e Tecnologia da Universidade Federal do Maranhão.

Orientador: Prof. Dr. Thales Levi Azevedo Valente.

SÃO LUÍS - MA

DEZEMBRO/2024

SUMÁRIO

1 INTRODUÇÃO

Durante as aulas de inteligência artificial, foi abordado o conceito de Agentes, aquele que toma atitudes autônomas sobre o ambiente usando de atuadores para alcançar um objetivo de acordo com informações recolhidas por sensores.

Uma das principais características do agente é a autonomia, pois cabe a ele tomar decisões em tempo real, independentes e de maneira complexa para escolher o melhor caminho, sem exatamente uma pessoa participando do processo de escolha de ações. O aperfeiçoamento do agente ao longo do tempo ocorre por meio de feedbacks, podendo ser resposta obtida por um crítico ou pelo próprio ambiente, permitindo que sejam feitas novas estratégias com o passar do tempo.

Os agentes podem ser classificados nas seguintes categorias:

- Reflexivo simples: Não apresentam níveis mais aprofundados, se limitam apenas a seguir uma linha de regras e comandos.
- Baseado em metas/objetivos: Planeje suas ações de acordo com o objetivo a longo prazo.
- Baseado em utilidade: São chamados a partir de outros agentes para executar uma ação específica.
- De aprendizagem: Buscam aprimoramento das funcionalidades com o passar do tempo, com a obtenção de novos dados e criação de novas táticas para resolução de tarefas.

Já os modelos baseados em agentes (ABM) representam ferramentas computacionais usadas para entender, explorar e interpretar de maneira abstrata comportamentos de sistemas complexos. Esses modelos fazem simulação de comportamentos de múltiplos agentes individuais, capazes de tomar decisões e interagir com o meio e entre si, e a partir dessas interações é possível notar padrões de comportamento biológico, social e econômico.

Um exemplo da implementação para compreensão prática é o modelo de segregação de Schelling, com o foco em analisar padrões de segregação espacial utilizando de base indivíduos ou famílias habitando um espaço e adquirindo preferências de vizinhos e mudando de espaço caso a preferência não seja atendida, estudo elaborado para demonstrar como um agente é uma unidade autônoma cujas decisões e baseiam em regras simples, e que a interação com outros agentes e o próprio ambiente o modifica.

No caso do trabalho apresentado, com uso do NetLogo, utilizaremos esses métodos para estudar de forma mais prática os fenômenos existentes em um ambiente de formiga e as possibilidades de ações entre elas mesmas e com o ambiente de acordo com alguns obstáculos. As formigas são consideradas como agentes individuais, de comportamento autônomo, que dividem o espaço com outras formigas da mesma característica.

Sobre as características dos agentes, foram definidos os seguintes parâmetros sobre o trabalho:

- Ambiente parcialmente observável: A formiga não possui uma visão completa do ambiente
- Determinismo probabilístico: Suas ações culminam em resultados que podem variar de diferentes modos
- Não episódico: Cada rodagem da simulação será independente da antiga.
- Multiagente: Vários agentes interagem ao mesmo tempo no ambiente.

NetLogo é uma linguagem de programação simples no formato de multi-agente que permite ao usuário modelar uma gama de fenômenos naturais ou sociais, ideal para desenvolver modelos de simulação baseado em agentes e estudar a reação de causa e efeito no ambiente. O sistema multiagente, pertencente à inteligência artificial distribuída, estuda a individualização dos agentes em um ambiente multiagente, nesse contexto, significa a autonomia de cada agente, e escolhas próprias mesmo em um espaço com outros agentes, desse modo, mesmo havendo vários em lugar, cada um terá seu próprio objetivo, e traçará métodos diferentes para alcançá-lo.

É possível aplicar à simulação certas regras e normas para um ou dezenas de agentes durante a operação, com o objetivo de estudar a conexão entre os indivíduos e a evolução de seus comportamentos.

Como linguagem, NetLogo pertence à família Lisp, permitindo a funcionalidade de agentes e simultaneidade e programação orientada a agente, fatores essenciais para estudar sistemas complexos.

Em NetLogo, existem quatro tipos de agentes: tartarugas (turtles), patches, links e o observador (observer).

1. Turtles (tartarugas): são agentes móveis que se deslocam pelo mundo.

2. Patches: representam o mundo, que é bidimensional e dividido em uma grade de pequenos quadrados que formam o "chão" onde as tartarugas se movem. Dará condição de funcionamento dos objetos que estão na simulação.
3. Links: são agentes que conectam duas tartarugas, formando ligações entre elas.
4. Observer (observador): não tem uma localização específica – você pode imaginá-lo como um "olhar" que observa o mundo de tartarugas e patches.

2 OBJETIVOS

O objetivo principal deste trabalho é aplicar na prática os aprendizados da sala de aula, especialmente no tema de agentes, usando a linguagem de programação NetLogo, fazendo modificações no código original com a aplicação de novas funções, elementos, comportamentos e tendências e analisar como essas adições transformaram o ambiente de simulação das formigas.

3 DESENVOLVIMENTO

Apesar do objetivo da configuração permanecer o mesmo, isto é testar o comportamento das formigas ao procurar comida com diferentes obstáculos, foram adicionadas algumas funcionalidades ao código.

3.1 FUNCIONALIDADES INCREMENTADAS

3.1.1 Variáveis globais

```
globals [
  raining?            ;estado global:indica chuva ou nao
  rain-intensity      ;intensidade da chuva
  sunny?              ;estado global do sol
  sun-intensity       ;intensidade do sol
  climate-duration    ;duracao do clima
```

As condições definidas para a condição climática inicialmente é de que o clima estará neutro, nem chovendo, nem ensolarado.

As operárias se movem mais devagar na chuva e no sol há evaporação mais rápida de feromônios.

3.1.2 Condições do clima

```
to go
  switch-climate       ; Alterna entre sol e chuva
  adjust-sun-visuals   ; Ajusta visualmente o sol
  sunny-effects        ; Aplica efeitos do sol
  toggle-rain
  create-rain
  move-rain
  evaporate-rain
```

Em que:

- switch-climate: Cria a alternância de climas, “Sunny” e “raining”, quando o climate-duration se iguala a 0. Caso seja ensolarado (sunny? = true), o sol desativa e muda para chuva (raining? = true).
- A duração é um valor aleatório entre 50 a 100 ticks, a intensidade é um valor aleatório entre 50 e 100, e quando há chuva, a intensidade do sol é 0.
- sunny-effects: Efeitos do sol, isto é patches azuis que indicariam a chuva saem aos poucos, e com isso a mudança de escala de cor.
- create-rain: Criação de gotas de chuva “raindrop” no mapa. Se a condição (raining?) for verdadeira, serão geradas gotas de chuva, que irão em direção para baixo, e a quantidade de gotas é atrelada a intensidade da chuva.
- move-rain: movimentação das gotas de chuva, simulando-a. Elas se movem para baixo
- evaporate-rain: Evaporação das gotas, o espaço voltando ao estado seco.
- toggle-rain: Indicar alternância na possibilidade de chover.
- adjust-sun-visuals: Ajustes na configuração do sol, de acordo com o clima estar ensolarado ou chuvoso.

3.1.3 Adição de predadores

```
to create-predators
  create-turtles 1                    ; Cria um predador
  [set size 11                        ; tamanho
   set color brown                    ; cor
   set label "tamandua"               ; nome do predador
   setxy -1 21                        ; posição do predador
   set life 60                       ; vida inicial predador(Tamanduá)
  ]
  ;;Cria o Sapo
  create-turtles 1                    ; Cria um predador
  [set size  4                        ; tamanho
   set color green                    ;cor
   set shape "frog top"               ;nome do predador
   setxy 20 -22                       ;posição do predador
   set life 12                        ;vida inicial predador(Sapo)
  ]
  ; Cria a aranha
  create-turtles 1                   ; Cria um predador
  [set size 5                        ; tamanho
   set color magenta                    ; cor
   set shape "spider"                ; formato
   setxy -20 0                       ; posição do predador
   set life 10                       ; vida inicial(aranha)
  ]
end
```

A função principal dos predadores é serem uma ameaça às formigas, o que as obriga a evitá-los, mudando o seguimento original e dificultando a busca por comida, tornando o tempo de busca por alimento maior. Isso torna a dinâmica mais realista, mais próxima de um ambiente natural, e o elemento de risco no ambiente obriga a formiga a traçar estratégias de segurança na coleta e sobrevivência.

A implementação de feromônios no ambiente estimula a sensação de perigo e cria um estado de alerta nas formigas.

```
to predator-attack                      ;ataque do predador
  ask turtles with [label = "tamandua"] [ ; Verifica se e tamandua
    let prey one-of turtles in-radius 1 with [color = red or color = orange] ; verifica se há uma formiga no raio 1
    if prey != nobody [     ; se for diferente de ninguem
     ask prey [
       set life life - 7                ;diminui a sua vida em -7
        if life <= 0 [die]              ;verifica a morte ou nao da formiga
      ]
     set predator-alert true            ;alerta do predador
    ]
  ]
   ask turtles with [shape = "frog top"] [ ; Verifica se e sapo
    let prey one-of turtles in-radius 1 with [color = red or color = orange] ; verifica se há uma formiga no raio 1
    if prey != nobody [     ; se for diferente de ninguem
     ask prey [
       set life life - 4                ;diminui a sua vida em -4
        if life <= 0 [die]              ;verifica a morte ou nao da formiga
      ]
     set predator-alert true            ;alerta do predador
    ]
  ]
   ask turtles with [shape = "spider"] [ ; Verifica se e aranha
    let prey one-of turtles in-radius 1 with [color = red or color = orange] ; verifica se há uma formiga no raio 1
    if prey != nobody [     ; se for diferente de ninguem
     ask prey [
       set life life - 3                ;diminui a sua vida em -3
        if life <= 0 [die]              ;verifica a morte ou nao da formiga
      ]
     set predator-alert true            ;alerta do predador
    ]
  ]
end
```

Caso não veja uma formiga, o predador vai se mover aleatoriamente pelo ambiente.

```
to predator-move
  ask turtles with [label = "tamandua"] [ ; se for tamandua
    let prey one-of turtles with [ant-type = "operaria" or ant-type = "guerreira"]
    ;; Define prey como uma tartaruga aleatória do conjunto de tartarugas com o tipo de formiga "operaria" ou "guerreira"
    ifelse prey != nobody [             ; Se prey não for nobody (há uma presa):
      face prey                         ; Predador olha na direção da presa
      fd 0.5                              ; Move-se em direção à presa
    ] [                                 ; se nao tiver presa
      rt random 4                      ; Caso não haja presa, move-se aleatoriamente de 0 a 3 graus para a direita
      lt random 4                      ; Gira um ângulo aleatório de 0 a 3 graus para a esquerda
      fd 0.2                           ; Move-se uma pequena distância
    ]
    if not can-move? 1 [ rt 180 ]       ; Se não puder se mover, gira 180 graus
  ]
  ask turtles with [shape = "frog top"] [ ; se for sapo
    let prey one-of turtles with [ant-type = "operaria" or ant-type = "guerreira"]
    ;  Define prey como uma tartaruga aleatória do conjunto de tartarugas com o tipo de formiga "operaria" ou "guerreira"
    ifelse prey != nobody [              ; Se prey não for nobody (há uma presa):
      face prey                         ; Predador olha na direção da presa
      fd 3                              ; Move-se em direção à presa
    ] [                                 ; se não tiver presa
      rt random 10                     ; Caso não haja presa, move-se aleatoriamente de 0 a 10 graus para a direita
      lt random 10                     ;  Gira um ângulo aleatório de 0 a 10 graus para a esquerda
      fd 1.2                           ; ; Move-se uma pequena distância
    ]
    if not can-move? 1 [ rt 180 ]       ; Se não puder se mover, gira 180 graus
  ]
   ask turtles with [shape = "spider"] [ ; se for aranha
    let prey one-of turtles with [ant-type = "operaria" or ant-type = "guerreira"]
    ;  Define prey como uma tartaruga aleatória do conjunto de tartarugas com o tipo de formiga "operaria" ou "guerreira"
    ifelse prey != nobody [              ; Se prey não for nobody (há uma presa):
      face prey                         ; Predador olha na direção da presa
      fd 2                              ; Move-se em direção à presa
    ] [                                 ; se não tiver presa
      rt random 7                     ; Caso não haja presa, move-se aleatoriamente de 0 a 10 graus para a direita
      lt random 7                     ;  Gira um ângulo aleatório de 0 a 10 graus para a esquerda
      fd 0.8                           ; ; Move-se uma pequena distância
    ]
    if not can-move? 1 [ rt 180 ]       ; Se não puder se mover, gira 180 graus
  ]

end
```

3.1.4 Comportamento das formigas

Há dois tipos de formigas:

- Guerreira, com função de defender o ninho, patrulhar a área e atacar predadores em um raio maior.(Possuem uma vida e tamanho maior, mas velocidade reduzida);
- Operária, com função de defender o ninho, se defender de predadores em um raio menor e coletar comida.(Possuem vida e tamanho menores, mas velocidade aumentada).

```
create-turtles population [          ; cria formigas com base no valor do slider 'population
  ifelse random 100 < 75 [           ; 75% operarios e 25% guerreiras
    set ant-type "operaria"          ; operaria
    set size 1                       ; tamanho
    set color red                    ; cor
    set shape "ant"                  ; formato da formiga
    set life 3                       ; vida da formiga operaria
  ] [
    set ant-type "guerreira"         ; guerreira
    set size 2                       ; tamanho
    set color orange                 ; cor
    set shape "ant 2"                ; formato
    set life 7                       ; vida
  ]
```

```
to defend-nest ; defesa do ninho
  let predator one-of turtles in-radius 2 with [label = "tamandua" or shape = "frog top" or shape = "spider"]
  ; se tiver um tamandua ou sapo ou aranha em um raio de 2
  if predator != nobody [                ; se predador for diferente d eninguem
    ask predator [
      set life life - 2 ; diminuem a vida -2 em -2
      if life <= 0 [ die ]  ; verifica a morte do predador
    ]
  ]
end


to ant-defense                           ;defesa/ataque das formigas
  ask turtles with [ant-type = "guerreira"] [        ; verifica se a formiga é guerreira
    let predator one-of turtles in-radius 3 with [label = "tamandua" or shape = "frog top" or shape = "spider"]
    ; verifica se o predador esta no raio 3
    if predator != nobody [                          ; se tiver preador
      ask predator [
        set life life - 2                ;diminui a vida de -2 em -2
        if life <= 0 [die]               ;verifica a morte ou nao do predaor
      ]
      set predator-alert true            ;alerta de predador
    ]
  ]
 ;; Operárias fogem ou defendem com base na proximidade do ninho
  ask turtles with [ant-type = "operaria"] [ ;Tipo operária
    let predator one-of turtles in-radius 2 with [label = "tamandua" or shape = "frog top" or shape = "spider"]
    ;Se estiver em raio 2 esses predadores

    ;; Se estiver longe do ninho, prioriza a fuga
    if not nest? and predator != nobody [
      rt random 180 ; Vira em uma direção aleatória
      fd 2 ; Afasta-se rapidamente
    ]

    ;; Se estiver no ninho ou próximo, tenta defender
    if nest? and predator != nobody [
      ask predator [
        set life life - 0.5 ; Reduz a vida do predador
        if life <= 0 [ die ] ; Remove o predador se a vida chegar a zero
      ]
    ]

    ;; Se o predador estiver próximo, ativa alerta e tenta fugir após defender
    if predator != nobody [
      set predator-alert true ; Ativa o alerta para guerreiras
      rt random 180 ; Vira em uma direção aleatória para fugir
      fd 2 ; Afasta-se rapidamente
    ]
  ]
end
```

```
to patrol ; patrulhar a área
  if random 100 < 10 [ rt random 360 ]
;Movimento aleatório ocasional(probabilidade de 10%)/Se a condição for satisfeita, a tartaruga gira para a direita (rt)
;por um ângulo aleatório entre 0 e 359 graus (random 360).
end
```

```
  ask turtles with [ant-type = "operaria"] [  ; se for operaria
    ifelse color = red [                      ; se a cor for vermelha
      look-for-food                            ; procura pela comida
    ] [
      return-to-nest                           ; se não for retorna pro ninho
    ]
    wiggle
; O procedimento wiggle adiciona uma variação aleatória no movimento da formiga.
    fd 2.5
 ; Move a formiga 2.5 unidades à frente, continuando sua busca ou retorno após ajustar sua direção com wiggle.
  ]
  ask turtles with [ant-type = "guerreira"] [ ; se for guerreira
  if predator-alert [                          ; se alerta de predador for verdadeiro
      defend-nest                             ; defendem o ninho
    ]
    patrol                                    ; patrulha
    fd 1                                      ; velocidade
    ]
```

Incrementa a idade a cada tick,e são removidas caso sejam mortas por um predador ou atinjam o limite de vida.

```
; Lógica de mortalidade baseada na idade
ask turtles [
  if ant-type = "operaria" [ ; Se for operaria
    set max-ant-age 60 + random 10 ; Operárias podem viver um pouco mais de 60 anos e variando ate 70
  ]
  if ant-type = "guerreira" [ ; Se for guerreira
    set max-ant-age 45 + random 5 ; Guerreiras têm vida mais curta devido ao esforço físico de 45 variando até 50
  ]
  if age >= max-ant-age [ die ] ; Remove formigas que atingiram a idade máxima
]
```

As formigas operárias possuem uma idade máxima maior que as guerreiras e ambas possuem uma variação podendo aumentar a sua idade máxima

3.1.5 Alimentos

Há 3 tipos de alimentos disponíveis:

```
to setup-food  ; procedimento dos patches
  ; Configura três fontes de alimento em posições específicas
  if (distancexy (0.6 * max-pxcor) 0) < 5 [
 ; Verifica se a distância do patch ao ponto (0.6 * max-pxcor, 0) é menor que 5.
 ;Este ponto está deslocado 60% para a direita no eixo X do ambiente.
    set food-source-number 1
 ; Define food-source-number como 1, identificando que o patch faz parte da primeira fonte de alimento.
    set food 3                          ; Alta quantidade de comida
    set food-value 3                    ; alta nutrição
  ]
  if (distancexy (-0.6 * max-pxcor) (-0.6 * max-pycor)) < 5 [
 ;Verifica se a distância do patch ao ponto (-0.6 * max-pxcor, -0.6 * max-pycor) é menor que 5.
 ;Este ponto está deslocado 60% para a esquerda no eixo X e 60% para baixo no eixo Y.
    set food-source-number 2
 ; Define food-source-number como 2, identificando que o patch faz parte da segunda fonte de alimento.
    set food 2;                          ; Média quantidade de comida
    set food-value 2                     ; Nutrição Média
  ]
  if (distancexy (-0.8 * max-pxcor) (0.8 * max-pycor)) < 5 [
 ; Verifica se a distância do patch ao ponto (-0.8 * max-pxcor, 0.8 * max-pycor) é menor que 5.
 ;Este ponto está deslocado 80% para a esquerda no eixo X e 80% para cima no eixo Y.
    set food-source-number 3
 ; Define food-source-number como 3, identificando que o patch faz parte da terceira fonte de alimento.
    set food 1                          ; Baixa quantidade de comida
    set food-value 1                    ; Nutrição Baixa
  ]
  ; Se o patch faz parte de uma fonte de alimento, atribui uma quantidade de comida (1 ou 2)
  if food-source-number > 0 [ ;Verifica se o patch faz parte de uma fonte de alimento (food-source-number > 0).
    set food one-of [1 2 3] ; Define a quantidade de comida (food) como 1 2 ou 3, escolhida aleatoriamente.
  ]
end
```

Nos alimentos, as formigas irão priorizar os mais nutritivos e com maior quantidade de comida.

4 RESULTADOS E DISCUSSÃO

As adições ao código aprofundaram a sua funcionalidade, como podemos ver:

- Condições climáticas: As alterações relacionadas à chuva e sol causaram a movimentação mais devagar das formigas na chuva, e evaporação de feromônios mais rápido no sol.
- Formigas: Se dividiram em duas, existindo as operárias, de tamanho menor, que procuram alimentos para levar ao ninho, e se durante o processo tiver um predador, atacam. Enquanto as guerreiras têm a função de patrulhar o ninho, e procurar predadores para atacar. A diferença principal é que a guerreira possui dano maior, quantidade de vida maior, e velocidade menor. As formigas deixam uma trilha de feromônios, que serve para guiar outras formigas.
- Predadores: Tornaram a ação das formigas mais trabalhosa, pois têm principal função de atacá-las caso estejam no mesmo radar,indo na mesma direção, ao ponto de matá-las caso a quantidade de vida zere. São três tipos: sapos, tamanduás e aranhas.

- Tamanduá: Cor marrom, com mais vida e menor velocidade, ataca e mata a formiga instantaneamente.
- Sapo: Cor verde, ataca a formiga também, mas com menos dano que o tamanduá e mais velocidade que ele
- Aranha: Cor magenta, ataca a formiga também, mas com menos vida, dano e velocidade do sapo.
- Alimento: Foi implementado agora tipos de alimentos no ambiente, do mais rico ao mais pobre nutricionalmente, essa classificação influencia diretamente no valor nutricional da comida que a formiga come, os alimentos estão espalhados no ambiente e precisam que as formigas vão até elas para serem coletadas. A comida de valor maior tem tom azul cyan, o de valor médio azul sky e valor baixo lime..
- Interface: Os patches que pertencem ao ninho têm cor violeta, e as formigas retornam a ele deixando o rastro (feromônio). Os predadores também deixam feromônio, mas de cor amarela, que indica à formiga que há presença de predador perto. No sol, os feromônios evaporam mais rápido, o que afeta a capacidade das formigas de encontrar o caminho para o ninho ou alimento.

4.1 COMPORTAMENTO EMERGENTE

O comportamento emergente está relacionado aos padrões e comportamentos coletivos que surgem a partir das interações no ambiente entre indivíduos. No caso das formigas, esse comportamento é demonstrado quando o feromônio cria trilhas que indicam o caminho percorrido até o ninho ou uma fonte de alimento. Esse padrão aparece sem que as formigas tenham conhecimento global do ambiente. Além disso, a interação entre as formigas e predadores evidencia um comportamento coletivo de defesa ou fuga, que emerge para garantir a sobrevivência da colônia.

4.2 AUTO-ORGANIZAÇÃO

A auto-organização é um processo que surge espontaneamente no sistema, sem a necessidade de um agente externo controlador. No modelo, as formigas se distribuem pelo ambiente de maneira organizada: algumas assumem o papel de patrulhar e atacar predadores (as guerreiras), enquanto outras focam na busca por alimentos (as operárias). Essa organização ocorre com base em regras simples, como o uso de trilhas de feromônio, que facilitam a coordenação entre os indivíduos.

4.3 ADAPTAÇÃO

A adaptação refere-se à capacidade de ajustar o comportamento em resposta a mudanças no ambiente. No modelo, isso é demonstrado, por exemplo, pela interação dinâmica entre predadores e formigas. A presença de predadores exige que as formigas alterem sua estratégia, seja para defender o ninho ou para evitar o confronto direto. Além disso, as condições climáticas, como a chuva ou o calor que afeta a evaporação do feromônio, influenciam diretamente no comportamento adaptativo das formigas, mostrando como elas respondem a novos desafios para garantir a sobrevivência e eficiência da colônia.

5 CONCLUSÃO

O trabalho apresentado alcançou com sucesso o objetivo de aplicar e explicar conceitos de inteligência artificial e modelagem baseada em agentes para simular comportamentos autônomos e interações em um ambiente dinâmico. Através do NetLogo, foi possível realizar implementações e alterações significativas no modelo básico de formigas apresentando, ampliando sua complexidade e permitindo análises mais detalhadas e profundas sobre as interações entre agentes e o ambiente.

As modificações introduzidas, como a divisão das formigas em operárias e guerreiras, a inclusão de predadores, a variação de alimentos por valor nutricional e a implementação de condições climáticas, enriqueceram o modelo apresentado anteriormente de maneira substancial.

O impacto das condições climáticas, como chuva e sol, destacou a importância de fatores externos na dinâmica do ambiente. A chuva diminuiu a velocidade das formigas e aumentou o tempo necessário para encontrar alimentos, enquanto o sol provocou uma aceleração na evaporação de feromônios, dificultando a navegação das formigas. Por outro lado, a inclusão de predadores adicionou um elemento de risco, exigindo que as formigas adaptem suas estratégias de sobrevivência para garantir a sobrevivência e a coleta de alimentos.

Em suma, o trabalho demonstrou como simulações baseadas em agentes podem ser utilizadas para estudar sistemas complexos de forma prática e didática. As interações entre agentes autônomos e o ambiente, bem como os fenômenos emergentes observados, fornecem uma capacidade de entender interações e resultados valiosos sobre como regras simples podem levar a comportamentos adaptativos e organizados, refletindo a dinâmica de sistemas

reais. O modelo desenvolvido não só reforça os conceitos teóricos abordados em sala de aula, mas também oferece uma base sólida para futuras explorações e aprimoramentos.

6 REFERÊNCIAS

LIMA, T. F. M.; FARIA, S. D.; FILHO, B. S. S.; CARNEIRO, T. G. S. **Modelagem de sistemas baseada em agentes: alguns conceitos e ferramentas.** XIV Simpósio Brasileiro de Sensoriamento Remoto, Natal, Brasil, 25-30 abril 2009, INPE, p. 5279-5286.

LOBÃO, E. C.; PORTO, A. J. V. **Evolução das técnicas de simulação.** Escola de engenharia de São Carlos- EESC. São Paulo, Brasil, 1999.

NIGEL G. **Agent-based models.**University of Surrey, Guildford, UK, 2011.

NIGEL, G., Troitzsch, K.G. **Simulation for the social scientist.** University of Surrey, Guildford, UK, 1999. NIGEL, G. Agent-based social simulation: dealing with complexity. University of Surrey, Guildford, UK, 2004.

PANTAROLO, E.; AZEVEDO, L. L.; MENEZES, C. S.; MAGDALENA B. C. **Exploração do ambiente orientado a agente NetLogo.** XVI Simpósio Brasileiro de informática na educação. Juiz de Fora, Minas Gerais, Brasil, 2005.

RUSSEL, S.; NORVIG, P. **Inteligência artificial.** Editora Campus, São Paulo, Brasil, 2003.

PAPERT, Seymour. **Logo: Computadores e educação.** São Paulo: Brasiliense, 1985.