



Universidade do Minho

2023/2024

Mestrado em Engenharia e Gestão de Sistemas de Informação

Modelação Dinâmica de Sistemas e Organizações

Docente: José Luís Mota Pereira



Snake-Game desenvolvido em
NetLogo



EQUIPA DE TRABALHO



Gonçalo Mendes Rodrigues

PG53843



João Pedro de Carvalho Mendes

PG53938



Rui Pedro Fernandes Coelho

PG54211



José Carlos Moreira Oliveira

A78739



ÍNDICE

| | |
|---|-----------|
| Equipa de Trabalho..... | 2 |
| Índice | 3 |
| 1. Introdução | 4 |
| 2. Descrição do Jogo | 5 |
| 3. Interface..... | 6 |
| 4. Agentes do Jogo | 10 |
| 5. Variáveis Globais do jogo | 12 |
| 6. Funções | 14 |
| 7. Principais dificuldades | 16 |
| 8. Possíveis melhorias | 17 |
| Conclusão | 18 |



1. INTRODUÇÃO

No âmbito da unidade curricular Modelação Dinâmica de Sistemas e Organizações foi nos proposto a elaboração de um projeto em NetLogo, com o intuito de aplicarmos na prática os conceitos abordados nas aulas.

A equipa optou pelo projeto 8, que diz respeito ao desenvolvimento do jogo Snake Game. Neste jogo, o jogador controla o movimento de uma cobra enquanto esta se move num espaço quadrangular limitado.

O nosso desafio para este projeto consistirá em criar a versão original do jogo Snake Game e implementar algumas melhorias relativas ao jogo base. Estas incluem a implementação de diferentes níveis de dificuldade, cada um caracterizado por velocidades de movimento distintas para a cobra e a introdução de outros obstáculos no terreno.

Adicionalmente, planeamos diversificar os objetos que a cobra pode consumir. Cada tipo de objeto proporcionará benefícios específicos, como a redução do tamanho da cobra, pontos extra, obter imunidade face aos obstáculos, entre outros.

Uma inovação adicional será introduzir possibilidades extra de movimento para a cabeça da cobra, permitindo direções que não se restrinjam aos tradicionais 90 graus de cada vez. Isto contribuirá para uma experiência de jogo mais dinâmica e desafiante.

Além disso desenvolvemos também a possibilidade de jogar em multijogador, o que permite aos jogadores jogarem de forma colaborativa ou até desafiarem-se de modo a obter mais pontos que o seu adversário.



2. DESCRIÇÃO DO JOGO

Como é possível identificar pela introdução do presente relatório, o projeto desenvolvido pelo grupo foi o SnakeGame, um jogo onde o jogador controla o movimento de uma cobra enquanto esta se move num espaço quadrangular limitado. Foram ainda implementadas funcionalidades não existentes no jogo original de modo a complementar o projeto desenvolvido e acrescentar complexidade.

O jogo deve ser iniciado ao premir os botões “setup” e “go”. Após esse passo a interface é apresentada, onde podemos escolher os diferentes modos de jogo, e é aqui que entram as diferenças relativamente ao jogo original.

Colocamos a opção de 3 níveis de dificuldade, onde tanto o ambiente de jogo como a velocidade aumentam conforme a dificuldade.

Há também a opção de multijogador, ativado através da seleção da opção amarela, seguida da seleção do nível de dificuldade.

Aqui o funcionamento do jogo é o mesmo, mas com a possibilidade de dois jogadores interagirem ao mesmo tempo, cada um com uma cobra de cor diferente.

Foi introduzida também a funcionalidade de aparecerem diferentes poderes que a cobra pode “comer”, onde cada um deles terá um efeito diferente no comportamento da cobra e na pontuação do jogo.

Adicionalmente existem campos informativos à direita da interface de jogo, que permitem ver informações sobre a sessão atual do jogo, mas que serão explicadas em maior pormenor na secção da Interface.

É também registado o recorde de pontuação para as partidas simples (1 jogador) e multijogador num ficheiro .txt.

O jogo contém sons para todas as interações realizadas pela cobra.

3. INTERFACE

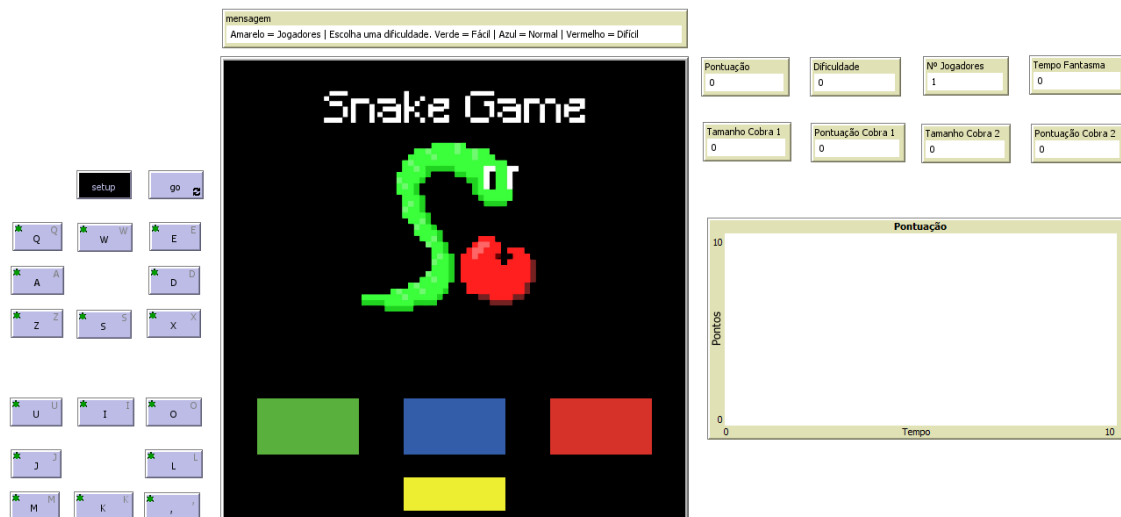


Figura 1 - Interface após pressionar setup

Primeiramente, no lado esquerdo, o utilizador é apresentado com os controlos, sendo os dois primeiros botões, *setup* e *go*, respetivamente, responsáveis por iniciar o menu principal e iniciar o jogo.

Após carregar o menu, o *display* apresenta quatro opções que o utilizador pode escolher, diferenciadas por cores. Os primeiros 3 botões, verde, azul e vermelho, representam as 3 dificuldades que o utilizador pode optar para o jogo. O botão amarelo possibilita a participação de dois jogadores no jogo.

Posteriormente, após pressionado o botão *go*, o *display* modifica para um destes 3 possíveis cenários, dependendo da dificuldade escolhida pelo utilizador. As seguintes imagens, demonstram as opções possíveis, por ordem crescente de dificuldade.

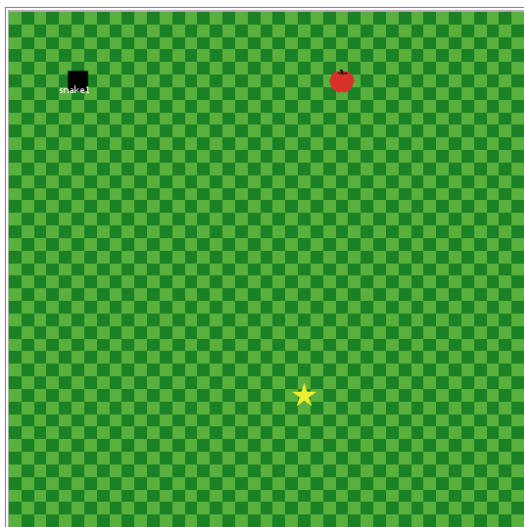


Figura 2 - Dificuldade "Fácil"

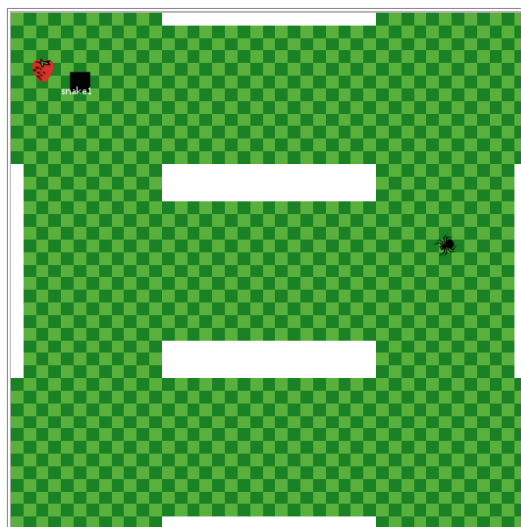


Figura 3 - Dificuldade "Média"

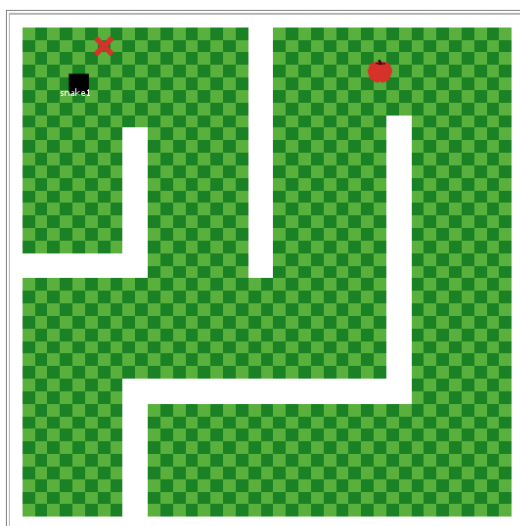


Figura 4 - Dificuldade "Difícil"

Ainda no lado esquerdo, é possível visualizar os controlos para cada um dos jogadores, que possibilitam movimentar as cobras em todas as direções, incluindo na diagonal.

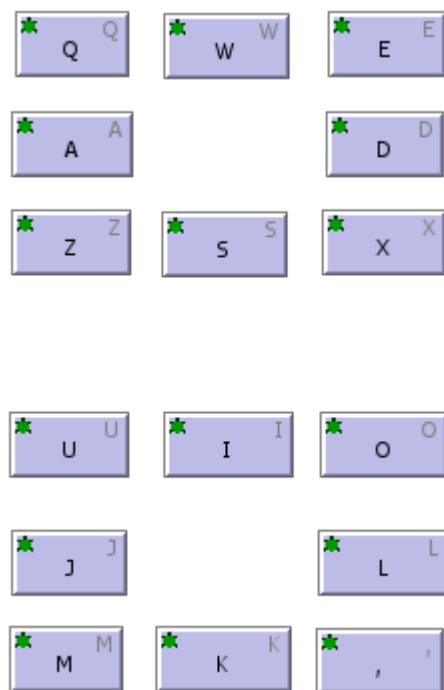


Figura 5 - Teclas de movimentação

O utilizador no lado direito tem à sua disposição vários monitores que permitem visualizar as configurações escolhidas no *setup* do jogo. Estes monitores mostram informações importantes, como a pontuação do jogo, a dificuldade selecionada, o número de jogadores, o tempo em modo fantasma, assim como o tamanho e a pontuação de cada uma das cobras, caso tenham sido escolhidos dois jogadores. Além disso, existe um monitor que permite acompanhar em tempo real a pontuação das cobras durante o jogo.

| | | | |
|----------------------|--------------------------|----------------------|--------------------------|
| Pontuação 300 | Dificuldade Fácil | Nº Jogadores 2 | Tempo Fantasma 0 |
| Tamanho Cobra 1 3 | Pontuação Cobra 1 100 | Tamanho Cobra 2 5 | Pontuação Cobra 2 200 |

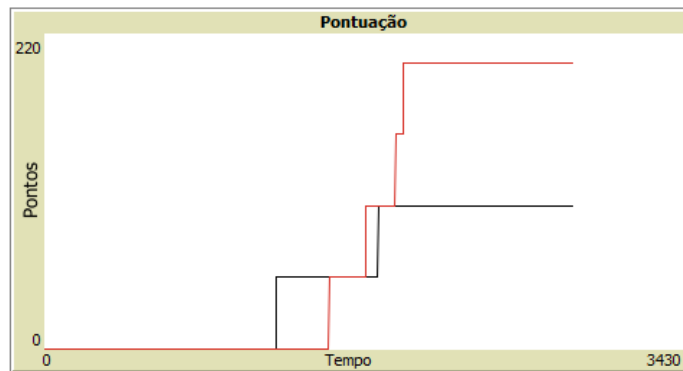


Figura 6 - Monitores e gráfico de informações de jogo



4. AGENTES DO JOGO

Nesta secção do relatório vamos falar em maior detalhe sobre os tipos de agentes existentes no jogo desenvolvido.

No nosso jogo existem 4 tipos de agentes, podendo estes ser as cobras, as frutas e os poderes. Pelo menos dois deles, a cobra e as frutas, são geradas imediatamente na inicialização do jogo, enquanto os poderes vão surgindo ao longo do desenrolar do jogo. Começando pelos agentes principais deste jogo, as **cobras**:


- **Snake1**: agente padrão, a cobra que surge em qualquer modo de dificuldade do jogo individual, surgindo sempre representado com a cor preta. É o agente controlado pelo utilizador, com o qual este interage através dos controlos pré-definidos (teclas W, A, D, S, Q, E, Z, X).
- **Snake2**: agente semelhante ao anterior, **Snake1**, mas que apenas surge quando é utilizado o modo multijogador. Este agente surge com a cor vermelha e é controlado também por controlos de teclado pré-definidos (teclas I, J, L, K, U, O, M).

Sempre que é comida uma fruta, nasce uma nova Snake. Cada cobra do jogo é constituída por um conjunto de Snakes que, a cada tick, ocupam a posição ocupada pela Snake que nasceu antes dela.

Existem também os agentes **Frutas**, que surgem na interface de jogo e cujo objetivo é serem comidas pela cobra, adicionando pontos à pontuação do jogo. Estas podem ser de dois tipos, tendo cada uma delas um valor diferente:

- Forma de **maçã**, que é a forma padrão e acrescenta 50 pontos ao score;
- Forma de **morango**, menos comum, e que acrescenta 100 pontos ao score;

Por último, existem também os **Poderes**, que acrescentam funcionalidades ao jogo, podendo contribuir para um aumento da pontuação, ou pelo contrário, para uma dificuldade acrescida ou até término do jogo. Estes podem ser de diversos tipos:

- 
- Forma de **estrela**, que durante 900 “ticks” permite duplicar a pontuação das frutas comidas pela cobra;
 - Forma de **aranha**, que retira 200 pontos à pontuação atual;
 - Forma de **fantasma**, que muda a cor da cobra para cinzento e permite à cobra ultrapassar paredes durante um determinado período de tempo.
 - Forma de **X**, que reduz o tamanho da cobra.




5. VARIÁVEIS GLOBAIS DO JOGO

Aqui passamos a descrever todas as variáveis globais que utilizamos durante o jogo, para realizar todas as funções expectáveis. Algumas delas são pré-definidas na inicialização do jogo, cujos valores podem variar, e outras são da responsabilidade do jogador, isto é, possuem o valor que este lhes atribuir, através da interação com o jogo.

Detalhe das variáveis do jogo:

- **difficulty**: Armazena a dificuldade escolhida pelo jogador ("Fácil", "Normal" ou "Difícil").
- **difficulty-chosen**: Variável booleana que indica se o jogador já escolheu a dificuldade.
- **mensagem**: Armazena uma mensagem de status ou instrução para o jogador.
- **score**: Armazena a pontuação total do jogo.
- **speed**: Controla a velocidade de movimento das cobras.
- **players**: Indica o número de jogadores (1 ou 2).
- **x** e **y**: São variáveis utilizadas para armazenar as coordenadas x e y de uma snake1 no momento em que uma cobra snake1 come uma fruta.
- **snakenum** e **snakenum2**: funcionam como ID de snake1 e snake2 respetivamente. Variáveis cruciais para movimentar as cobras, visto que cada snake irá ocupar a posição da snake com $ID = ID - 1$.
- **double-score-time**: Armazena o tempo em que o bônus de pontuação dobrada está ativo.
- **time-power**: Armazena o tempo decorrido desde a última criação de um poder.
- **life**: Representa o tempo de vida das aranhas.
- **ticks-at-color-change**: Armazena o número de ticks no momento em que a cor da cobra foi alterada, após comer um fantasma. Utilizado para reverter à cor original após um determinado período.

- 
- **Ghost-Time:** Armazena o tempo restante em que o modo fantasma está ativo, onde as cobras são cinzentas e podem atravessar as paredes do jogo.
 - **head:** direção das snakes2.
 - **sx e sy:** São variáveis utilizadas para armazenar as coordenadas x e y de uma snake2 no momento em que uma cobra snake2 come uma fruta.
 - **whoWas e whoWasPower:** variáveis utilizadas para verificar se foi a snake1 ou snake2 que comeu as frutas e poderes.
 - **score1 e score2:** variáveis que armazenam a pontuação de cada cobra de forma individual.
 - **valor-recorde:** valor correspondente ao recorde atual de pontuação no modo de jogo selecionado.

Estas são as principais variáveis que foram utilizadas pela equipa para desempenhar as funções necessárias no jogo, descritas com algum detalhe, de modo a perceber o funcionamento destas.



6. FUNÇÕES DO JOGO

No ponto anterior do relatório foram descritas as principais variáveis, aqui, no último tópico a abordar, vamos detalhar as funções/ procedimentos que foram utilizados. Devido à complexidade e tamanho do código, vamos apenas definir as principais funções e as principais tarefas desempenhadas por elas.

- Função **setup**:
 - Inicializa as principais variáveis a utilizar no decorrer do jogo, assim como importa a imagem do menu do jogo.
- Função **menu**:
 - Inicia as funções necessárias para que o utilizador tenha as opções para iniciar o jogo, verifica se uma das dificuldades de jogo já foi escolhida (através da função check-mouse-click) e invoca as funções create-map, create-powers, create-fruit e create-snake..
- Função **create-buttons**:
 - Cria os botões do menu de jogo
- Função check-mouse-clcik
 - Verifica se uma das dificuldades de jogo já foi escolhida e o número de jogadores.
- Função **create-map**:
 - Cria o terreno de jogo incluindo os obstáculos.
- Função **create-fruit**:
 - Cria as frutas com maior probabilidade de ser maçã do que morango e invoca a função check-position.
- Função **check-position**:
 - Verifica se a posição onde foram geradas as frutas e os poderes é válida, de modo que não sejam criados em cima de paredes e onde já exista uma cobra.
- Função **create-powers**:



- Cria os vários poderes com diferentes probabilidades para cada um ser gerado e invoca a função `check-position`.
- Função **create-snake**:
 - Cria as cobras no início do jogo.
- Função **move-snake**:
 - Movimenta a cobra com base nos inputs do jogador e invoca a função `move`.
- Função **move**:
 - Função de apoio à função `move-snake`. Verifica se o tempo de fantasma terminou, terminando o poder Fantasma caso isso aconteça. Verifica também se já acabou o tempo de vida da Aranha e também guarda a cada tick a posição das várias cobras.
- Função **go**:
 - Invoca a função `check-collision` e `eat`, como também executa verificações para garantir que a movimentação da cobra é correta (Exemplo: se a cobra se desloca para a esquerda, ela não se pode movimentar diretamente para a direita).
- Função **eat**:
 - Responsável por gerir os scores, tempos e velocidades, bem como o aumento do tamanho ou morte da cobra, com base no tipo de fruta ou poder que é comido.
- Função **check-collision**:
 - Função responsável por verificar se a cobra colidiu com algum outro elemento do jogo, sejam eles paredes (nos níveis de dificuldade médio e alto) ou com a sua cauda.
- Função **game-over**:
 - Função que é chamada quando algumas das condições de término do jogo é atingida, terminando o jogo. Aqui é registado, no ficheiro `Recorde`, o valor da pontuação atingida, caso o score seja recorde, ou seja, mais alto que o maior valor anteriormente atingido.

Em suma, estas são todas as funções necessárias para executar as mais diversas funcionalidades implementadas, permitindo, em conjunto com as variáveis também anteriormente descritas, atingir um funcionamento correto, sem erros e que permita que o utilizador consiga interagir com o jogo da forma que foi planeado pelo grupo.



7. PRINCIPAIS DIFICULDADES

O grupo encontrou duas maiores dificuldades na realização do jogo Snake Game:

- Utilizar os Ticks para controlar alguns ciclos.
- Desenvolver o crescimento da cobra à medida que esta come frutas.

Para alguns dos ciclos e funções que utilizavam Ticks, o grupo inicialmente teve dificuldade em entender a sua utilização, mas, após estudar cuidadosamente a documentação e até fóruns na internet, conseguiu ultrapassar essa barreira.

Além disso, inicialmente o grupo teve dificuldade para que o movimento e crescimento da cobra funciona-se corretamente. Para isso foram testados vários métodos e o objetivo foi concluído com recurso à variável `snakenum1` e `snakenum2` que permitiram identificar de forma correta as várias cobras e consequentemente fazer com que, a cada tick, cada cobra ocupa-se a posição da cobra com o `snakenum` anterior ao seu.



8. POSSÍVEIS MELHORIAS

Para completar ainda mais o Snake-Game poderiam ser adicionados novos poderes, tais como: “Arma” que permitiria à cobra partir paredes durante alguns segundos e “Árvore” que iria aumentar o número de frutas máximo que podem ser geradas ao mesmo tempo no terreno.

Além disso também poderíamos adicionar a possibilidade dos jogadores escolherem se pretendem participar num jogo multijogador colaborativo ou competitivo, onde no modo competitivo poderiam comer a cobra do adversário e venceria a cobra com maior tamanho após o fim do tempo de jogo.



CONCLUSÃO

Concluída a implementação do projeto “Snake-Game”, a equipa sente-se satisfeita com os resultados alcançados. O objetivo principal era desenvolver o famoso “Snake-Game” e explorar temas e funcionalidades abordadas nas aulas. Assim, a equipa optou por focar não apenas na jogabilidade tradicional, mas também em oferecer aos jogadores a possibilidade de personalização, como níveis de dificuldade, e interatividade, como a possibilidade de existir dois jogadores.

Durante o desenvolvimento, foram encontrados alguns desafios, mas o conhecimento adquirido nas aulas facilitou a implementação das funcionalidades desejadas. A comunicação eficiente entre os membros da equipa desempenhou um papel crucial no sucesso do projeto, assegurando que todos os objetivos fossem atingidos conforme planeado.

É relevante destacar que, embora o foco desta UC seja o estudo de sistemas complexos, a criação deste jogo permitiu aprofundar o conhecimento na linguagem NetLogo. Esta experiência prática enriqueceu a compreensão da linguagem e das capacidades que esta tem para modelação de sistemas, e conhecer os principais paradigmas de modelação.

Assim, ao finalizar este projeto, a equipa reconhece a importância do paradigma de modelação dinâmica na análise de sistemas complexos. O resultado obtido está alinhado com as expectativas, respeitando os princípios fundamentais estudados ao longo da UC. Este projeto não apenas consolidou o conhecimento adquirido, mas também aprofundou a familiaridade com a linguagem NetLogo.