CCT/UFCA
Algoritmos em Grafos
1º semestre de 2025

Prof. Carlos Vinicius G. C. Lima
Projeto: Labirinto de Creta
04/09/2025

Informações:

i. A linguagem de programação utilizada pode ser a de sua preferência.

ii. Os algoritmos implementados devem seguir aqueles com a **menor complexidade possível** em termos assintóticos.

iii. Somente serão considerados para avaliação as operações que foram solicitadas explicitamente.

iv. A clareza e concisão das implementações também são objetos de avaliação.

v. Comentários de ajuda no código também serão levados em consideração.

vi. O projeto pode ser implementado em grupos de até 3 pessoas.

vii. **O prazo máximo para entrega é o dia 13/10/2025.**

viii. **A nota do projeto corresponderá a 20% da média final.**

ix. **As notas no projeto serão individuais.**

x. A entrega do projeto consinstirá de forma online (github, por exemplo) contendo: os arquivos de código; um arquivo com as instruções para a execução do seu código; **links** para vídeos contendo a explicação da implementação, um para cada participante da equipe; no vídeo deverá ser explicado por qual parte você foi responsável e como foram implementadas as soluções desta parte. **Não envie arquivos de vídeo.**

Introdução

Na antiguidade, após uma guerra entre Creta e Atenas, Atenas foi obrigada a enviar periodicamente jovens como sacrifício para um labirinto, em sinal de submissão. Dentro dele habita o terrível Minotauro, uma criatura faminta, dotada de força descomunal e sentidos aguçados. O labirinto possui uma única entrada e uma saída única. A missão é simular a travessia de um entrante no labirinto e o comportamento do Minotauro, que caça o intruso.

Cada um que entra no labirinto pode levar consigo uma espada e alguns kits de alimentos que demoram a estragar e são leves, como forma de dar falsa esperança aqueles que entram. Neste caso, se há o encontro com o minotauro, então há uma chance pequena de apenas 1% de chance de vitória do intruso. Mesmo em caso de morte do minotauro, a comida levada dura apenas um tempo limitado, onde é necessário escapar do labirinto antes de um certo número de movimentos nele.

Considere o labirinto representado como um grafo $G = (V, E)$, onde vértices representam posições no labirinto e arestas representam conexões entre duas posições. Dois pontos podem ser acessados por diferentes rotas diretamente, ou seja, através de diferentes arestas. Cada aresta possui um peso associado representando a dificuldade e gasto de tempo para percorrê-la.

Por seu tamanho, o minotauro não consegue utilizar a entrada e saída do labirinto para escapar, mas possui uma força enorme e é mantido alimentado, a menos quando está próximo da época do sacrifício, passando este a ficar faminto e com raiva. As posições do minotauro e prisioneiro no labirinto são atualizadas a cada passo, onde os movimentos se dão de um vértice a outro adjacente. Se o minotauro encontra alguém, ele o ataca com fúria.

Considere também que o minotauro possui ótima audição e olfato, podendo perceber a posição do entrante caso este esteja a uma distância menor ou igual a um certo parâmetro d passado como entrada. Neste caso o minotauro passa a perseguir o entrante com uma velocidade maior, ou seja, passa a se mover dois vértices por vez em direção ao intruso, enquanto o entrante apenas um por vez.

O minotauro conhece todo o labirinto, incluindo as distâncias entre cada ponto, enquanto o entrante desconhece totalmente o labirinto. Porém, considere que o prisioneiro possui uma ótima memória e um novelo de lã, que pode utilizar como guia, caso queira retornar a algum ponto do labirinto e consegue identificar por onde já passou. Considere que as arestas podem ser percorridas em qualquer sentido.

Requisitos do Projeto

Deve-se implementar, em linguagem de programação de sua escolha (Python, C++, Java, etc.), um sistema que:

1. Representação do Labirinto:
   - O labirinto deve ser representado como um **grafo ponderado** $G = (V, E)$.
   - A entrada do programa será um arquivo de texto descrevendo:
     - Número de vértices $|V| = n$;
     - Número de arestas $|E| = m$;
     - Lista de arestas no formato: $u \quad v \quad w(u, v)$ (significa aresta entre vértices u e v com peso $w(u, v)$);
     - Vértice de entrada;
     - Vértice de saída: diferente do de entrada;
     - Posição inicial do Minotauro: qualquer diferente da entrada e saída;
     - Parâmetro de percepção do Minotauro: valor de distância máxima $p(G)$ para que ele detecte o intruso;
     - Tempo máximo $\tau(G)$ de duração do alimento levado com o entrante.
2. **Objetivo:** Executar as dinâmicas de movimentações do prisioneiro e minotauro no labirinto. O prisioneiro terá de escapar em no máximo o tempo $\tau(G)$ pré-determinado como entrada de duração de seu alimento e do período sem se alimentar que o prisioneiro ainda conseguiria sobreviver.

Movimentação

- **Entrante**:
  - Move-se um vértice por vez;
  - Não conhece o labirinto, mas pode utilizar uma estratégia de exploração, utilizando um novelo de lã para retornar por caminhos já percorridos;
  - Se encontrar a saída em qualquer momento, o programa deve indicar que o prisioneiro foi salvo;
  - Se o Minotauro encontrar o prisioneiro em qualquer vértice, então há uma batalha com 1% de chance de sobrevivência.
- **Minotauro**:
  - Move-se normalmente um vértice por vez ao longo do labirinto;
  - Caso detecte o entrante a distância em soma de pesos de arestas menor ou igual ao parâmetro de percepção ($p(G)$), passa a persegui-lo;
  - Durante a perseguição, move-se dois vértices por rodada pelo caminho mínimo em direção ao prisioneiro;
  - Se alcançar o mesmo vértice do entrante antes deste encontrar a saída, o elimina imediatamente com chance de 99%.

Saída

- Relatório contendo:
  - A informação de que o prisioneiro escapou ou como ele morreu;
  - Tempo restante até acabar a comida (considere apenas números inteiros);
  - Sequência de vértices visitados pelo prisioneiro;
  - Se ocorreu, o momento da detecção do prisioneiro pelo minotauro e o momento em que o alcançou;
  - Caminho percorrido pelo Minotauro durante a perseguição, se ocorreu;

Critérios de Avaliação

Os projetos serão avaliados com base nos seguintes critérios:

1. **Corretude da implementação** (40%)
    - O sistema simula corretamente a dinâmica desejada;
    - Os algoritmos foram implementados corretamente.
2. **Eficiência dos algoritmos** (20%)
    - Uso apropriado de estruturas de dados para otimizar tempos de execuções;
    - Implementação de algoritmos adequados para resolver os problemas existentes da melhor forma.
3. **Qualidade do Código** (20%)
    - Código bem estruturado e documentado;
    - Uso adequado de modularização e boas práticas de programação.
4. **Análise e Discussão dos Resultados** (20%)
    - Relatório com os dados solicitados como saída;
    - Discussão sobre a eficiência do método proposto e calibragem dos parâmetros de entrada.

Entrega

Os alunos deverão entregar:

- Código-fonte do sistema.
- **Links** para vídeos explicando a parte implementada por cada membro da equipe.
- As notas serão individuais e levam em conta a proporção e dificuldade do que foi implementado por cada membro em relação a todo o trabalho.
- **Observação:** pode-se fazer uso de bibliotecas para manipulação de estruturas de dados clássicas, como listas, pilhas, filas, heaps, etc. Porém, os algoritmos vistos em sala e dinâmica devem ser implementados sem o uso de bibliotecas disponíveis pelas linguagens utilizadas.