|  |
| --- |
| **RELATÓRIO** PROJETO ESTRUTURA DE DADOS AVANÇADOS |

|  |
| --- |
| **Tiago Barroso Fontes 33222** |

|  |
| --- |
| Licenciatura em Engenharia de Sistemas Informáticos |

Índice

[Resumo 5](#_Toc198587079)

[Introdução 6](#_Toc198587080)

[1. Estruturas de dados 7](#_Toc198587081)

[1.1. Estrutura Adjacência 7](#_Toc198587082)

[1.2. Estrutura Antena 7](#_Toc198587083)

[1.3. Estrutura Grafo 8](#_Toc198587084)

[1.4. Estrutura CaminhoNode 8](#_Toc198587085)

[1.5. Estrutura FilaNode 8](#_Toc198587086)

[2. Algoritmos e Funções Principais 9](#_Toc198587087)

[2.1. Adicionar Adjacência 9](#_Toc198587088)

[2.2. Adicionar Antena 9](#_Toc198587089)

[2.3. Carregar Antenas Do Ficheiro 10](#_Toc198587090)

[2.4. Procura em Profundidade 10](#_Toc198587091)

[2.5. Procura em Largura 11](#_Toc198587092)

[2.6. Imprimir Caminho 11](#_Toc198587093)

[2.7. Encontrar Caminhos 12](#_Toc198587094)

[2.8. Mostrar Interseções 12](#_Toc198587095)

[2.9. Calcular Interseções 13](#_Toc198587096)

[2.10. Exportar Resultados 14](#_Toc198587097)

[2.11. Libertar Grafo 14](#_Toc198587098)

[2.12. Limpar Visitados 15](#_Toc198587099)

[2.13. Função Principal 15](#_Toc198587100)

[3. Testes e Validações 16](#_Toc198587101)

[3.1. Resultados Obtidos 16](#_Toc198587102)

[Conclusão 17](#_Toc198587103)

[GitHub 18](#_Toc198587104)

Índice de Figuras

[Fig. 1 - Representação da Estrutura de Dados Adjacência. 7](#_Toc198587059)

[Fig. 2 - Representação da Estrutura de Dados Antena. 7](#_Toc198587060)

[Fig. 3 - Representação da Estrutura de Dados Grafo 8](#_Toc198587061)

[Fig. 4 - Representação da Estrutura de Dados CaminhoNode 8](#_Toc198587062)

[Fig. 5 - Representação da Estrutura de Dados FilaNode 8](#_Toc198587063)

[Fig. 6 - Código relativo para adicionar Adjacência. 9](#_Toc198587064)

[Fig. 7 - Código relativo para adicionar Antena. 9](#_Toc198587065)

[Fig. 8 - Função de Carregamento de antenas do ficheiro. 10](#_Toc198587066)

[Fig. 9 - Função de Procura em Profundidade. 10](#_Toc198587067)

[Fig. 10 – Função de Procura em Largura. 11](#_Toc198587068)

[Fig. 11 - Função de Imprimir Caminhos 11](#_Toc198587069)

[Fig. 12 – Função de Encontrar Caminhos. 12](#_Toc198587070)

[Fig. 13 - Função para Mostrar Interseções 12](#_Toc198587071)

[Fig. 14 - Função para Calcular Interseções 13](#_Toc198587072)

[Fig. 15 - Função para Exportar Resultados 14](#_Toc198587073)

[Fig. 16 - Função para Libertar Grafo 14](#_Toc198587074)

[Fig. 17 - Função para Libertar Visitados 15](#_Toc198587075)

[Fig. 18 – Diagrama da Função Principal 15](#_Toc198587076)

[Fig. 19 - Matriz utilizada no teste do programa. 16](#_Toc198587077)

[Fig. 20 - Resultado obtido para a matriz de teste. 16](#_Toc198587078)

Resumo

Este relatório apresenta o trabalho desenvolvido na Fase 2 do Projeto da Unidade Curricular de Estruturas de Dados Avançadas (EDA), centrado na aplicação de grafos para modelar e analisar uma rede de antenas com frequências de ressonância específicas. A problemática envolve a deteção de locais com efeito nefasto devido à interação entre antenas sintonizadas na mesma frequência. Foram exploradas técnicas de representação de grafos, implementadas funcionalidades de pesquisa (em profundidade e largura) e análise de caminhos. Este trabalho contribui com uma abordagem programática eficiente em linguagem C para problemas relacionados com propagação de sinais e análise de redes.

Introdução

Este relatório descreve o trabalho realizado na segunda fase do projeto da unidade curricular de Estruturas de Dados Avançadas (EDA), que consistiu na modelação e análise de redes de antenas utilizando grafos.

A problemática envolve a identificação de localizações com efeito nefasto devido à interação entre antenas com a mesma frequência. Para tal, foi desenvolvida uma solução em linguagem C que representa cada antena como um vértice de um grafo, interligando as que partilham a mesma frequência.

O objetivo principal foi aplicar algoritmos de procura e análise de caminhos sobre esta estrutura, permitindo explorar a conectividade entre antenas e a ocorrência de interferências. A metodologia seguiu uma abordagem incremental, desde a modelação do tipo de dados até à implementação e testes com ficheiros de entrada.

# Estruturas de dados

O sistema foi construído com base em cinco estruturas de dados principais, a estrutura Adjacencia, a estrutura Antena, a estrutura Grafo, a estrutura CaminhoNode e a estrutura FilaNode.

## Estrutura Adjacência

A estrutura Adjacência representa uma ligação entre duas antenas, funcionando como um nó de uma lista ligada. Cada elemento armazena o índice da antena de destino e um apontador para o nó seguinte, permitindo registar múltiplas conexões a partir de uma mesma antena de forma dinâmica.

Uma imagem com texto, Tipo de letra, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Representação da Estrutura de Dados Adjacência.

## Estrutura Antena

Cada antena é modelada através da estrutura Antena, a qual contém:

* O carácter que representa a frequência de operação;
* As coordenadas de posição na matriz (linha e coluna);
* Um apontador para a lista de adjacências, que identifica as ligações com outras antenas;
* Um campo booleano utilizado para controlo de visitação nos algoritmos de travessia do grafo.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Representação da Estrutura de Dados Antena.

## Estrutura Grafo

A estrutura Grafo agrega a totalidade do sistema, sendo composta por:

* Um vector estático com capacidade para armazenar até 50 antenas;
* Um contador que mantém o registo do número de antenas inseridas.

Uma imagem com texto, Tipo de letra, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Representação da Estrutura de Dados Grafo

## Estrutura CaminhoNode

Cada caminho é modelado através da estrutura CaminhoNode, a qual contém:

• Um apontador para uma estrutura Antena, representando a antena associada àquele nó do caminho;

• Um apontador para o próximo nó do caminho, permitindo a ligação sequencial entre diferentes antenas;

• A estrutura é utilizada para representar uma lista ligada, onde cada nó corresponde a uma etapa do caminho entre antenas.

Uma imagem com texto, Tipo de letra, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Representação da Estrutura de Dados CaminhoNode

## Estrutura FilaNode

Cada elemento da fila é modelado através da estrutura FilaNode, a qual contém:

• Um apontador para uma estrutura Antena, representando a antena armazenada naquele nó da fila;

• Um apontador para o próximo nó da fila, permitindo a ligação sequencial entre os elementos;

• A estrutura é utilizada para representar uma fila ligada, frequentemente usada em algoritmos de travessia, como a busca em largura (BFS).

Uma imagem com texto, Tipo de letra, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Representação da Estrutura de Dados FilaNode

# Algoritmos e Funções Principais

## Adicionar Adjacência

Cria uma ligação entre duas antenas ao alocar dinamicamente um novo nó de Adjacência, que é inserido no início da lista da antena de origem. Esta função é utilizada durante o carregamento dos dados para representar as arestas do grafo.

Uma imagem com texto, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Código relativo para adicionar Adjacência.

## Adicionar Antena

Adiciona uma nova antena ao alocar dinamicamente uma estrutura Antena, cujos campos são preenchidos com os dados fornecidos (frequência e posição na matriz). A nova antena é inserida no início da lista de antenas do grafo. Esta função é utilizada durante o carregamento dos dados para representar os vértices do grafo.

Uma imagem com texto, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Código relativo para adicionar Antena.

## Carregar Antenas Do Ficheiro

Realiza a leitura do ficheiro de entrada e instância as estruturas de cada antena com base nas suas posições e frequências. Durante este processo, são automaticamente estabelecidas conexões entre antenas que partilham a mesma frequência, através da função Adicionar Adjacência

. Uma imagem com texto, captura de ecrã, software, ecrã

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função de Carregamento de antenas do ficheiro.

## Procura em Profundidade

Implementa o algoritmo de pesquisa em profundidade (Depth-First Search), percorrendo recursivamente a rede a partir de uma antena de origem. A travessia é registada num ficheiro de saída, permitindo a análise dos caminhos explorados.

Uma imagem com texto, captura de ecrã, Tipo de letra, software

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função de Procura em Profundidade.

## Procura em Largura

Executa a pesquisa em largura (Breadth-First Search), utilizando uma fila para explorar todas as antenas adjacentes antes de avançar para os níveis seguintes. Este algoritmo é útil para identificar caminhos mínimos ou obter uma visão em camadas da rede.

Uma imagem com texto, captura de ecrã, software

Os conteúdos gerados por IA poderão estar incorretos.

Fig. – Função de Procura em Largura.

## Imprimir Caminho

Imprime o caminho percorrido de forma recursiva, começando pelo final e seguindo até ao início da lista ligada. Para cada antena no caminho, escreve no ficheiro de saída a frequência e as coordenadas. Esta função é utilizada para apresentar o trajeto completo entre antenas, preservando a ordem correta do percurso.

Uma imagem com captura de ecrã, texto, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função de Imprimir Caminhos

## Encontrar Caminhos

Identifica todos os caminhos possíveis entre duas antenas especificadas. A função segue uma abordagem de retrocesso (backtracking), explorando todas as rotas válidas com base nas conexões existentes.

Uma imagem com texto, captura de ecrã, software, ecrã

Os conteúdos gerados por IA poderão estar incorretos.

Fig. – Função de Encontrar Caminhos.

## Mostrar Interseções

Analisa intersecções físicas entre antenas de diferentes frequências, verificando se se encontram na mesma linha ou coluna da matriz. As intersecções detetadas são registadas e apresentadas como potenciais pontos de interferência cruzada.

Uma imagem com texto, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função para Mostrar Interseções

## Calcular Interseções

Percorre a rede para identificar interferências entre antenas da mesma frequência. A análise considera posições alinhadas horizontal, vertical e diagonalmente, sinalizando os locais onde o efeito nefasto pode ocorrer.

Uma imagem com texto, captura de ecrã, software, ecrã

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função para Calcular Interseções

## Exportar Resultados

Compila todos os dados recolhidos ao longo da execução num relatório final. Este inclui a listagem de antenas, resultados das pesquisas DFS e BFS, caminhos encontrados e pontos críticos de interferência.

Uma imagem com texto, captura de ecrã, software

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função para Exportar Resultados

## Libertar Grafo

Liberta a memória alocada dinamicamente durante a execução. Esta função percorre cada antena e remove a respetiva lista de adjacências, prevenindo fugas de memória.

Uma imagem com texto, captura de ecrã, Tipo de letra, software

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função para Libertar Grafo

## Limpar Visitados

Reposiciona o estado de visitação de todas as antenas para falso. Esta operação é necessária para garantir o correto funcionamento de novas análises consecutivas.

Uma imagem com texto, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Função para Libertar Visitados

## Função Principal

A função main coordena a execução do programa. Inicia com o carregamento dos dados através de CarregarAntenasDoFicheiro, seguido de uma verificação à existência de antenas carregadas. Em caso de sucesso, procede à exportação dos resultados com ExportarResultados. No final, invoca LibertarGrafo para libertação da memória e termina a execução com uma mensagem de conclusão.

Uma imagem com texto, captura de ecrã, software, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Fig. – Diagrama da Função Principal

# Testes e Validações

Como caso de teste, foi utilizada uma matriz 6x8 contendo 6 antenas de diferentes frequências. A disposição das antenas foi definida de forma a explorar diversos cenários de interferência, possibilitando a validação dos algoritmos de comparação.

Uma imagem com captura de ecrã, Tipo de letra, design

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Matriz utilizada no teste do programa.

## Resultados Obtidos

O programa identificou corretamente as antenas, os caminhos entre elas e os pontos de interferência, confirmando o correto funcionamento dos algoritmos.

Uma imagem com texto, captura de ecrã

Os conteúdos gerados por IA poderão estar incorretos.

Fig. - Resultado obtido para a matriz de teste.

Conclusão

A segunda fase do projeto centrou-se na aplicação de grafos como modelo abstrato para representar redes de antenas e respetivas interações. A substituição das listas ligadas por grafos permitiu analisar de forma mais rigorosa as ligações entre antenas com a mesma frequência, bem como identificar caminhos e pontos críticos de interferência.

A implementação dos algoritmos de procura (em profundidade e largura), bem como da deteção de interseções entre frequências distintas, revelou-se eficaz na identificação dos efeitos nefastos. Os resultados obtidos demonstraram que a estrutura proposta permite escalar a análise para redes maiores e mais complexas.

Este trabalho evidenciou a utilidade dos grafos na modelação de problemas reais de telecomunicações, reforçando a importância da abstração algorítmica e da organização modular do código. A abordagem seguida promoveu uma solução robusta, eficiente e alinhada com os objetivos pedagógicos da unidade curricular.

GitHub

O repositório do projeto encontra-se disponível em:

<https://github.com/Fontes9/Projeto-EDA>