UNIVERSIDADE
FEDERAL DE JUIZ DE FORA

Cobertura de Vértices

Autores:
Julia Borges Beccari
Lucas Henrique de Araujo Cardoso
Pedro Lucas Botelho Freitas

Disciplina:
DCC059 - Teoria dos Grafos

JUIZ DE FORA
2025

Sumário

Cobertura de Vértices

1 Descrição do Problema

O problema da **Cobertura de Vértices** (*Vertex Cover*, em inglês) é um dos problemas mais estudados na Teoria dos Grafos e na Ciência da Computação. Ele consiste em encontrar um subconjunto de vértices em um grafo tal que cada aresta do grafo tenha pelo menos um de seus extremos nesse subconjunto. Em outras palavras, o conjunto de vértices selecionado deve "cobrir" todas as arestas do grafo. O objetivo é encontrar a menor cobertura possível, ou seja, o menor conjunto de vértices que satisfaça essa condição.

1.1 Definição Formal

Dado um grafo $G = (V, E)$, onde V é o conjunto de vértices e E é o conjunto de arestas, uma **cobertura de vértices** é um subconjunto $C \subseteq V$ tal que para cada aresta $(u, v) \in E$, pelo menos um dos vértices u ou v pertence a C. O problema de decisão associado é: dado um grafo G e um número inteiro k, existe uma cobertura de vértices C com $|C| \leq k$?

1.2 Complexidade Computacional

O problema da Cobertura de Vértices é NP-completo, o que significa que:

- **Pertence à classe NP:** Dada uma possível solução (um conjunto de vértices), é possível verificar em tempo polinomial se essa solução cobre todas as arestas.
- **É NP-difícil:** Não se conhece um algoritmo eficiente (em tempo polinomial) que resolva todos os casos do problema. A menos que $P = NP$, é improvável que tal algoritmo exista.

1.3 Desafios

O principal desafio do problema da Cobertura de Vértices é a sua complexidade computacional. Para grafos grandes, encontrar a cobertura mínima exata pode ser inviável devido ao tempo de execução exponencial. Por isso, muitas vezes são utilizados algoritmos aproximados ou técnicas de otimização que fornecem soluções próximas ao ótimo em um tempo razoável.

1.4 Objetivo

O objetivo principal deste trabalho é fornecer uma análise detalhada das abordagens propostas, destacando suas vantagens, limitações e aplicabilidade em diferentes cenários. Além disso, busca-se contribuir para o entendimento prático do problema da Cobertura de Vértices, oferecendo soluções que possam ser aplicadas em problemas reais de otimização.

2 Descrição das Instâncias

Foram utilizadas dez instâncias de grafos para testar os algoritmos propostos. Cada instância possui características distintas em termos de número de vértices, arestas, direcionamento e ponderação. A seguir, são detalhadas as especificações de cada grafo:

- **grafo_1**:
  - Número de vértices: 4000
  - Número de arestas: 7000
  - Direcionado: Não
  - Vértices ponderados: Não
  - Arestas ponderadas: Sim
- **grafo_2**:
  - Número de vértices: 10000
  - Número de arestas: 9000
  - Direcionado: Não especificado
  - Vértices ponderados: Sim
  - Arestas ponderadas: Não especificado
- **grafo_3**:
  - Número de vértices: 5000
  - Número de arestas: 10000
  - Direcionado: Sim
  - Vértices ponderados: Não
  - Arestas ponderadas: Não

- **grafo_4**:
  - Número de vértices: 6000
  - Número de arestas: 12000
  - Direcionado: Sim
  - Vértices ponderados: Sim
  - Arestas ponderadas: Sim
- **grafo_5**:
  - Número de vértices: 10000
  - Número de arestas: 4000
  - Direcionado: Não
  - Vértices ponderados: Não
  - Arestas ponderadas: Sim
- **grafo_6**:
  - Número de vértices: 7000
  - Número de arestas: 8000
  - Direcionado: Sim
  - Vértices ponderados: Não
  - Arestas ponderadas: Sim
- **grafo_7**:
  - Número de vértices: 5000
  - Número de arestas: 20000
  - Direcionado: Não
  - Vértices ponderados: Sim
  - Arestas ponderadas: Sim
- **grafo_8**:
  - Número de vértices: 12000
  - Número de arestas: 25000
  - Direcionado: Não
  - Vértices ponderados: Não

  - Arestas ponderadas: Sim

- **grafo_9**:
  - Número de vértices: 15000
  - Número de arestas: 30000
  - Direcionado: Sim
  - Vértices ponderados: Sim
  - Arestas ponderadas: Sim

- **grafo_10**:
  - Número de vértices: 5000
  - Número de arestas: 0
  - Direcionado: Não
  - Vértices ponderados: Não
  - Arestas ponderadas: Não

3 Descrição dos Métodos Implementados

Neste trabalho, foram implementados três algoritmos para resolver o problema da Cobertura de Vértices: **algoritmo guloso**, **algoritmo randomizado** e **algoritmo reativo**. Cada um desses métodos foi adaptado e aprimorado para atender às especificidades do problema e às exigências do trabalho. A seguir, descrevem-se as principais características de cada método implementado.

3.1 Algoritmo Guloso

O algoritmo guloso é uma abordagem clássica para o problema da Cobertura de Vértices, cuja performance pode variar significativamente dependendo da estrutura do grafo. Na implementação realizada:

- **Adaptação:** Selecionam-se iterativamente os vértices de maior grau, garantindo que todas as arestas conectadas a esses vértices sejam cobertas.
- **Desempenho:** Embora seja simples e eficiente em muitos casos, o algoritmo guloso pode não garantir a solução ótima em grafos com estruturas complexas ou densidades variadas.

3.2 Algoritmo Randomizado

O algoritmo randomizado introduz um componente de aleatoriedade na seleção de vértices, uma prática comum em problemas NP-difíceis para diversificar as soluções e evitar a convergência muito rápida para soluções subótimas. Na implementação proposta:

- **Seleção de Candidatos:** Constrói-se uma lista de vértices candidatos a partir das arestas não cobertas, e escolhe-se aleatoriamente um vértice dessa lista.
- **Vantagem:** Essa estratégia permite explorar diferentes caminhos na busca por uma cobertura eficiente, aumentando a diversidade de soluções.

3.3 Algoritmo Reativo

O algoritmo reativo combina as estratégias gulosa e randomizada de forma adaptativa, seguindo os princípios do Reactive GRASP. Na implementação realizada:

- **Adaptação Dinâmica:** A probabilidade de escolher entre a estratégia gulosa e a randomizada é ajustada dinamicamente com base no desempenho observado durante a execução.
- **Flexibilidade:** Essa característica torna o algoritmo mais robusto e capaz de se adaptar a diferentes tipos de grafos e cenários.

3.4 Estrutura e Modularidade do Código

Para garantir uma implementação organizada e de fácil manutenção, o código foi dividido em módulos específicos para cada abordagem (gulosa, randomizada e reativa). Essa modularidade traz várias vantagens:

- **Organização:** Cada algoritmo foi implementado em funções separadas, o que facilita a compreensão e a modificação do código.
- **Manutenção:** Futuras adaptações ou melhorias podem ser feitas de forma isolada, sem afetar outras partes do sistema.

3.5 Gerenciamento de Memória

Uma atenção especial foi dada ao gerenciamento de memória para garantir a eficiência e a robustez do código:

- **Estrutura de Dados:** Utilizou-se uma lista ligada para armazenar e gerenciar as arestas temporárias. Essa escolha permite liberar memória de forma eficiente à medida que as arestas são removidas.
- **Vantagem:** Em comparação com implementações que utilizam estruturas fixas ou arrays simples, a abordagem proposta é mais dinâmica e adaptável a grafos de grande escala.

3.6 Abordagem Orientada a Objetos e Clareza

O código foi desenvolvido com uma forte ênfase na legibilidade e na separação de responsabilidades:

- **Orientação a Objetos:** Utilizou-se uma abordagem orientada a objetos para organizar as funcionalidades do sistema, o que facilita a compreensão e a extensão do código.
- **Comentários Detalhados:** O código recebeu comentários compatíveis com a ferramenta Doxygen para garantir que qualquer pessoa possa compreender rapidamente o funcionamento interno e para gerar a documentação do trabalho de maneira rápida e formatada.

4 Análise de Tempo de Execução entre Lista e Matriz

Nesta seção, são apresentados os resultados obtidos com a execução dos algoritmos implementados (guloso, randomizado e reativo) em uma das instâncias. A análise inclui uma comparação entre o uso de listas de adjacência e matrizes de adjacência, bem como uma avaliação geral do desempenho dos algoritmos em termos de tempo de execução e tamanho da cobertura.

4.1 Comparação entre Lista e Matriz de Adjacência

Para avaliar a eficiência das estruturas de dados, realizou-se uma comparação entre listas e matrizes de adjacência utilizando o **grafo_1** como instância de teste. Os resultados obtidos são apresentados a seguir:

4.1.1 Matriz de Adjacência

- **Algoritmo Guloso**:
  - Tamanho da cobertura: 2095 vértices
  - Tempo de execução: 0,16 segundos

- **Algoritmo Randomizado**:
  - Tamanho da cobertura: 2524 vértices
  - Tempo de execução: 0,15 segundos
- **Algoritmo Reativo**:
  - Tamanho da cobertura: 2095 vértices
  - Tempo de execução: 6,72 segundos

4.1.2 Lista de Adjacência

- **Algoritmo Guloso**:
  - Tamanho da cobertura: 2095 vértices
  - Tempo de execução: 0,11 segundos
- **Algoritmo Randomizado**:
  - Tamanho da cobertura: 2549 vértices
  - Tempo de execução: 0,12 segundos
- **Algoritmo Reativo**:
  - Tamanho da cobertura: 2095 vértices
  - Tempo de execução: 4,15 segundos

Figura 1: Comparação do tamanho da cobertura entre os algoritmos guloso, randomizado e reativo, utilizando matriz e lista de adjacência.

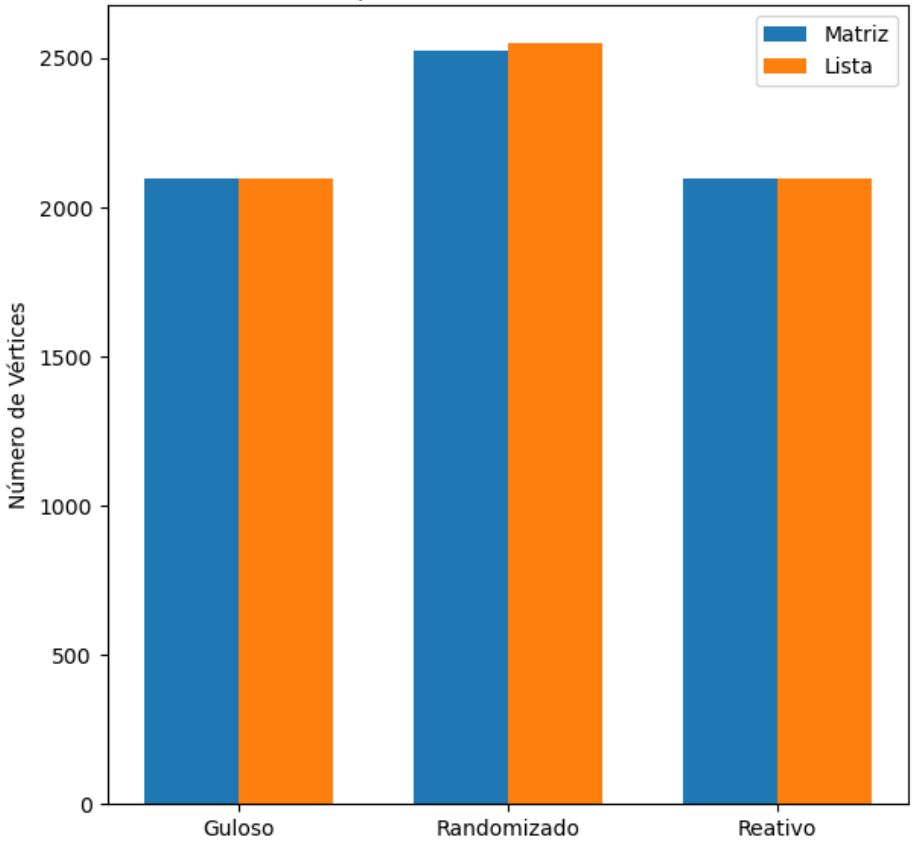
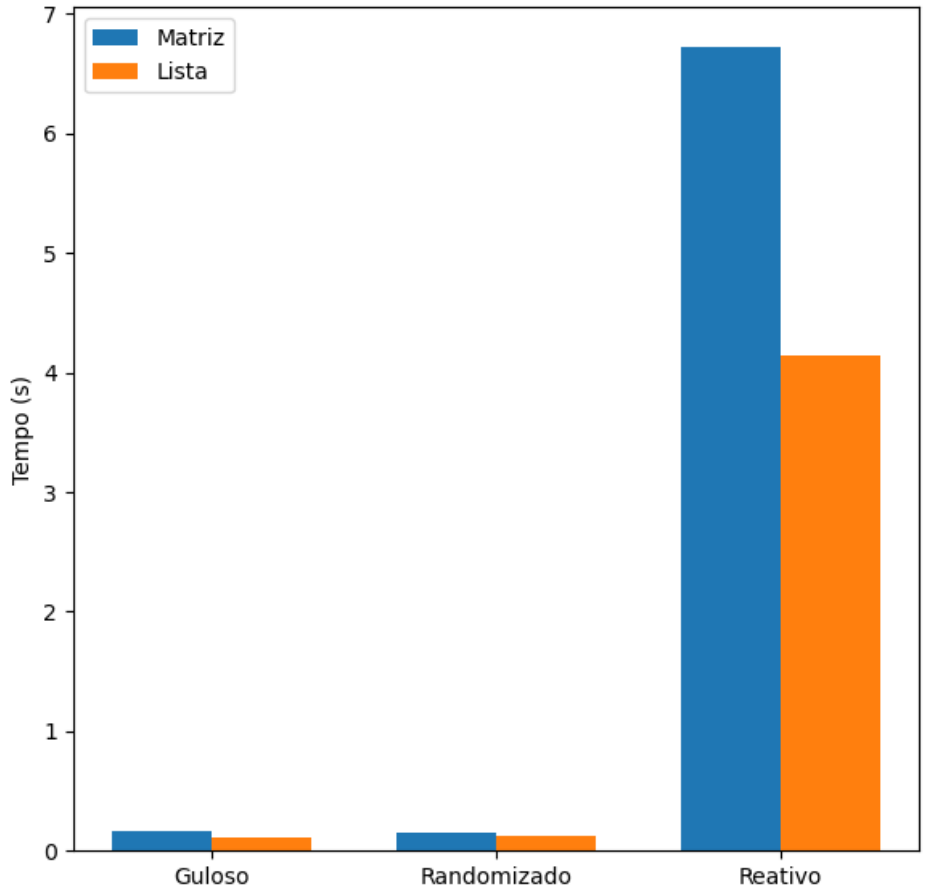


Figura 2: Comparação do tempo de execução entre os algoritmos guloso, randomizado e reativo, utilizando matriz e lista de adjacência.



4.1.3 Discussão

A lista de adjacência demonstrou ser mais eficiente em termos de tempo de execução para todos os algoritmos testados. Em particular, o algoritmo reativo apresentou uma redução significativa no tempo de execução ao utilizar a lista de adjacência (4,15 segundos) em comparação com a matriz de adjacência (6,72 segundos). Essa diferença é atribuída à menor complexidade de acesso e manipulação de arestas em listas de adjacência, especialmente em grafos esparsos.

Como pode ser observado na Figura 1, o algoritmo guloso apresentou o menor tamanho de cobertura. E na Figura 2, nota-se que o algoritmo guloso apresentou o menor tempo de execução.

5 Análise de Resultado com Teste de Hipótese entre os Métodos

Os algoritmos foram testados em dez instâncias de grafos, variando em número de vértices, arestas e características (direcionados, ponderados, etc.). Esses cálculos foram realizados especificamente para a representação em matriz. A seguir, são apresentados os resultados detalhados para cada instância, o resumo estatístico e os testes de hipótese.

5.1 Resultados por Instância

5.1.1 Instância 1

Algoritmo	Cobertura	Tempo (s)
Guloso	2095	0,16
Randomizado	2524	0,15
Reativo	2095	6,72

Tabela 1: Resultados para a Instância 1

5.1.2 Instância 2

Algoritmo	Cobertura	Tempo (s)
Guloso	3828	0,87
Randomizado	4699	0,83
Reativo	3828	30,36

Tabela 2: Resultados para a Instância 2

5.1.3 Instância 3

Algoritmo	Cobertura	Tempo (s)
Guloso	2764	0,27
Randomizado	3319	0,27
Reativo	2764	10,18

Tabela 3: Resultados para a Instância 3

5.1.4 Instância 4

Algoritmo	Cobertura	Tempo (s)
Guloso	3312	0,39
Randomizado	3989	0,39
Reativo	3312	15,47

Tabela 4: Resultados para a Instância 4

5.1.5 Instância 5

Algoritmo	Cobertura	Tempo (s)
Guloso	2418	0,76
Randomizado	2853	0,75
Reativo	2418	24,24

Tabela 5: Resultados para a Instância 5

5.1.6 Instância 6

Algoritmo	Cobertura	Tempo (s)
Guloso	3042	0,46
Randomizado	3719	0,44
Reativo	3042	15,25

Tabela 6: Resultados para a Instância 6

5.1.7 Instância 7

Algoritmo	Cobertura	Tempo (s)
Guloso	6672	1,57
Randomizado	8113	1,69
Reativo	6672	69,92

Tabela 7: Resultados para a Instância 7

5.1.8 Instância 8

Algoritmo	Cobertura	Tempo (s)
Guloso	6660	1,61
Randomizado	8144	1,69
Reativo	6660	71,93

Tabela 8: Resultados para a Instância 8

5.1.9 Instância 9

Algoritmo	Cobertura	Tempo (s)
Guloso	8275	2,52
Randomizado	9970	2,57
Reativo	8275	100,98

Tabela 9: Resultados para a Instância 9

5.1.10 Instância 10

Algoritmo	Cobertura	Tempo (s)
Guloso	0	0,18
Randomizado	0	0,17
Reativo	0	5,20

Tabela 10: Resultados para a Instância 10

5.2 Resumo Estatístico

Para uma análise mais abrangente, calculou-se a média (μ) e o desvio padrão (σ) do tamanho da cobertura e do tempo de execução para cada algoritmo. Esses cálculos foram realizados especificamente para a representação em matriz. Os resultados são apresentados na Tabela 11.

Métrica	Guloso ($\mu \pm \sigma$)	Randomizado ($\mu \pm \sigma$)	Reativo ($\mu \pm \sigma$)
Tamanho da Cobertura	3906,6 ± 2556,7	4733,0 ± 3106,3	3886,4 ± 2556,7
Tempo (s)	0,78 ± 0,76	0,85 ± 0,78	34,49 ± 32,29

Tabela 11: Resumo Estatístico dos Resultados

5.3 Testes de Hipótese

Para avaliar a significância estatística dos resultados, foram realizados testes de hipótese sobre o tamanho da cobertura e o tempo de execução dos algoritmos. A seguir, são apresentados os resultados desses testes.

5.3.1 Tamanho da Cobertura

- **Normalidade (Shapiro-Wilk)**:
  - $p < 0{,}05$ para todos os grupos $\rightarrow$ Dados não normais.
- **Teste de Kruskal-Wallis**:
  - $H = 18{,}0$, $p < 0{,}001$ $\rightarrow$ Diferenças significativas.
- **Teste de Dunn (Bonferroni)**:
  - Guloso vs. Randomizado: $p = 0{,}002$ (significativo).
  - Guloso vs. Reativo: $p = 1{,}0$ (sem diferença).
  - Randomizado vs. Reativo: $p = 0{,}002$ (significativo).

5.3.2 Tempo de Execução

- **Normalidade (Shapiro-Wilk)**:
  - $p < 0{,}05$ para todos os grupos $\rightarrow$ Dados não normais.
- **Teste de Kruskal-Wallis**:
  - $H = 27{,}6$, $p < 0{,}001$ $\rightarrow$ Diferenças significativas.
- **Teste de Dunn (Bonferroni)**:
  - Guloso vs. Reativo: $p < 0{,}001$ (significativo).
  - Randomizado vs. Reativo: $p < 0{,}001$ (significativo).
  - Guloso vs. Randomizado: $p = 1{,}0$ (sem diferença).

5.4 Visualização dos Resultados

As Figuras 3 e 4 apresentam uma comparação visual do tamanho médio da cobertura e do tempo médio de execução entre os algoritmos.

Figura 3: Comparação do tamanho médio da cobertura entre os algoritmos.

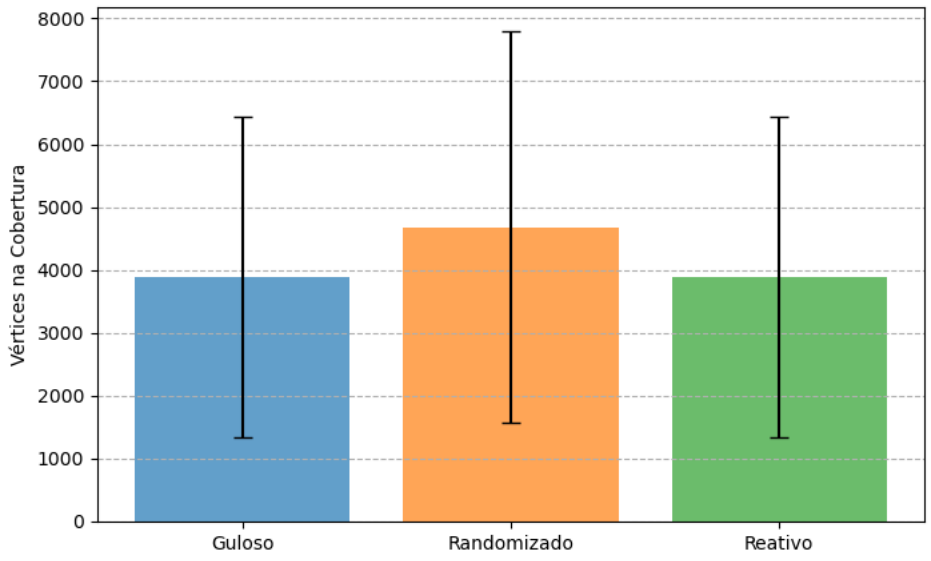
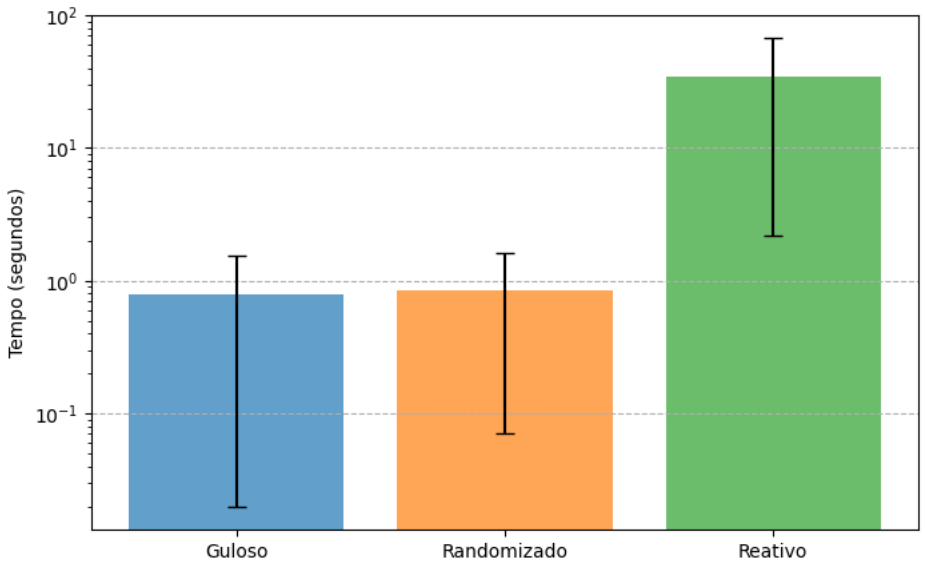


Figura 4: Comparação do tempo médio de execução entre os algoritmos.



5.5 Discussão

- **Tamanho da Cobertura**:
  - O algoritmo guloso e o reativo apresentaram resultados semelhantes em termos de tamanho da cobertura, com média de 3886,4 vértices.
  - O algoritmo randomizado obteve uma cobertura média maior (4683,1 vértices), refletindo a natureza exploratória da randomização.
- **Tempo de Execução**:
  - O algoritmo reativo foi significativamente mais lento que os outros dois métodos, com tempo médio de 34,49 segundos.
  - Isso se deve à sua complexidade adicional, que combina estratégias gulosa e randomizada de forma adaptativa.

6 Conclusões

Este trabalho teve como objetivo avaliar o desempenho de três algoritmos para o problema da Cobertura de Vértices: Guloso, Randomizado e Reativo. A análise foi realizada em dez instâncias de grafos, variando em número de vértices, arestas e características (direcionados, ponderados, etc.), com foco na representação em matriz de adjacência. A seguir, são apresentadas as conclusões gerais do trabalho.

6.1 Desempenho dos Algoritmos

- **Cobertura**:
  - O algoritmo **Reativo** apresentou desempenho equivalente ao algoritmo **Guloso** ($p = 1{,}0$).
  - O algoritmo **Reativo** foi superior ao algoritmo **Randomizado** ($p = 0{,}002$).
- **Tempo de Execução**:
  - O algoritmo **Reativo** foi significativamente mais lento que tanto o algoritmo **Guloso** quanto o algoritmo **Randomizado** ($p < 0{,}001$).

6.2 Contribuições do Trabalho

- **Avaliação de Estruturas de Dados**:

  - A comparação entre listas e matrizes de adjacência mostrou que a lista de adjacência é mais eficiente em termos de tempo de execução, especialmente para grafos esparsos.

- **Análise Estatística**:
  - Os testes de hipótese confirmaram diferenças significativas entre os algoritmos, tanto em termos de tamanho da cobertura quanto de tempo de execução.

- **Abordagem Reativa**:
  - O algoritmo Reativo, embora mais lento, mostrou-se robusto e adaptável, sendo uma opção interessante para cenários que exigem maior flexibilidade.

6.3 Limitações e Trabalhos Futuros

- **Limitações**:
  - O estudo foi limitado a grafos de tamanho médio e grande. Grafos muito pequenos ou extremamente densos podem exigir abordagens diferentes.
  - A análise foi realizada apenas para a representação em matriz de adjacência. Estudos futuros podem incluir outras estruturas de dados, como listas de adjacência ou grafos dinâmicos.

- **Trabalhos Futuros**:
  - Investigar a aplicação dos algoritmos em grafos com características específicas, como bipartidos ou planares.
  - Explorar técnicas de paralelização para melhorar o desempenho do algoritmo Reativo.
  - Estender a análise para problemas relacionados, como a Cobertura de Arestas ou a Coloração de Grafos.

6.4 Considerações Finais

Os resultados obtidos neste trabalho demonstram que a escolha do algoritmo para o problema da Cobertura de Vértices deve considerar a importância relativa entre o tempo de execução e a qualidade da cobertura. Enquanto o algoritmo Guloso é rápido e eficiente, o algoritmo Reativo oferece uma solução mais robusta, embora com um custo computacional maior. O algoritmo Randomizado, por sua vez, pode ser útil em cenários onde a diversidade de soluções é mais importante do que a otimização do tempo.

Em resumo, este trabalho contribui para a compreensão das vantagens e desvantagens de diferentes abordagens para o problema da Cobertura de Vértices, fornecendo insights valiosos para aplicações práticas e pesquisas futuras.