Heurísticas para Cálculo de Grandes Cliques em Grafos

Leandro de Souza Oliveira
Orientador: Dr. Vagner Pedrotti

[1]Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (UFMS)
Caixa Postal 549 – 79.070-900 – Campo Grande – MS – Brazil

leandro_oliveira@ufms.br

Abstract. *This paper documents an Atividade Orientada de Ensino (AOE), a supervised academic work registered as a non-disciplinary curricular activity within the UFMS program. The study focuses on the implementation and experimental evaluation of heuristics for computing large cliques in graphs and their effect on pruning during Zykov tree branching for graph coloring. Experiments are performed using timetabling instances from FACOM-UFMS, allowing performance measurement and analysis of search reduction in practical scenarios.*

Resumo. *Este artigo documenta uma Atividade Orientada de Ensino (AOE), um trabalho acadêmico supervisionado registrado como componente curricular não disciplinar no contexto da UFMS. O estudo envolve a implementação e avaliação experimental de heurísticas para obtenção de grandes cliques em grafos, analisando seu impacto no processo de poda na árvore de Zykov para coloração de grafos. Experimentos foram feitos com instâncias reais do problema de alocação de disciplinas da FACOM-UFMS para medir o desempenho e redução do espaço de busca em cenários práticos.*

1. Introdução

Esta Atividade complementa o TCC [de Souza Oliveira 2025] ao aprofundar o estudo de heurísticas para detecção de grandes cliques em grafos. A motivação surge das dificuldades observadas no processo de validação de instâncias do problema de alocação de disciplinas da FACOM/UFMS, onde a identificação rápida de uma solução pode auxiliar no processo como um todo.

Enquanto o TCC foca na modelagem e implementação da árvore de Zykov para validação estrutural, esta AOE dedica-se a revisar e analisar heurísticas implementadas que buscam cliques máximais dentro da árvore de Zykov - permitindo avaliar o impacto dessas estratégias em problemas combinatórios reais.

2. Conceitos Fundamentais

Denota-se por $G = (V, E)$ um grafo não dirigido, V representa o conjunto de vértices enquanto E representa os pares de vértices ligados por uma aresta, Ou seja, dados $x, y \in V$ e $xy \in E$ significa que existe uma aresta entre x e y em G.

Uma *clique* C é um subconjunto de V tal que, para quaisquer dois vertices $x, y \in C$, temos $xy \in E$. Diz-se que uma clique qualquer C_a é *maximal* se não existe nenhuma outra clique C_b tal que $C_a \subset C_b$. Já uma *clique máxima* é uma clique de maior cardinalidade entre todas as cliques de G. Uma clique $C_{\max}$ é máxima se, e somente se, $|C_{\max}| \geq |C'|$ para todas as cliques de G.

A árvore de Zykov de um grafo G é uma árvore binária baseada em G onde, para cada par de vértices não adjacentes, gera-se dois ramos, um que adiciona a aresta e outro que os contrai. Em nossa modelagem no TCC, é importante para cada nó encontrar grandes cliques para auxiliar na poda da árvore.

3. Heurísticas para Cálculo de Grandes Cliques

Foram utilizadas duas heurísticas para cálculo de cliques maximais durante a exploração da árvore de Zykov. A primeira é uma abordagem ingênua: seleciona-se um vértice aleatório e, se for compatível com o clique atual, o vértice é incluído; o processo continua iterativamente enquanto o tamanho do clique é aumentado. A vantagem dessa modelagem é seu tempo de execução rápido.

A segunda abordagem é baseada em GRASP (*Greedy Randomized Adaptive Search Procedure*) [Resende and Ribeiro 2014], uma meta-heurística iterativa composta por duas fases: construção gulosa aleatorizada de uma solução inicial e posterior refinamento por busca local. Executam-se múltiplas vezes, mantendo-se o melhor resultado obtido. No contexto deste trabalho, GRASP é utilizado para explorar diferentes combinações de vértices com maior variabilidade, potencialmente produzindo cliques maiores que o método ingênuo. Apresentamos, abaixo, um template básico do método GRASP para cliques maximais:

Algorithm 1: GRASP para construção de clique máximal em um grafo G

Entrada: Grafo $G = (V, E)$, número de iterações $MaxIter$, semente $Seed$
Saída: Maior clique encontrada C^*

```
1 C* ← ∅ ;                                  // Melhor clique encontrada
2 para k = 1 até MaxIter faça
3 |   C ← ∅;
4 |   C ← CliqueGulosaRandômica(G, Seed);
5 |   C ← BuscaLocal(C, G);
6 |   se |C| > |C*| então
7 |   |   C* ← C;
8 retorna C*
```

Em sequência vamos definir e ilustrar algoritmos para as duas fases do GRASP.

3.1. Busca Gulosa Randômica

Algoritmos gulosos randômicos aplicam a mesma construção iterativa de um algoritmo guloso comum. Sua única distinção é a introdução da escolha aleatória, que constroi soluções distintas em cada execução.

A construção aleatória é guiada pela construção da *RCL*, do inglês, *restricted candidate list*, formado pelos melhores elementos encontrados em uma etapa de escolha gulosa. Assim, uma solução parcial é dada a partir da adição de um vértice encontrado na *RCL* de cada iteração.

Algorithm 2: CliqueGulosaRandômica()

Entrada: Grafo $G = (V, E)$, semente $Seed$
Saída: Uma clique máximal $C \in V$

Escolher aleatoriamente $x \in V$;
$C \leftarrow \{x\}$;
$Cand \leftarrow N(x)$; `// apenas vértices que podem crescer a clique`
enquanto $Cand \neq \emptyset$ **faça**
    Avaliar $score(v)$ para todo $v \in Cand$;
    Construir RCL contendo os vértices com maior $score(v)$;
    Selecionar um $s \in RCL$ aleatoriamente;
    $C \leftarrow C \cup \{s\}$;
    Atualizar $Cand \leftarrow \{v \in V : v$ é adjacente a todos os vértices de $C\}$;
retorna C

3.2. Busca Local

A segunda fase busca, iterativamente, aprimorar a solução construida com uma busca em sua vizinhança de cliques maximais, sucessivamente substituindo C por uma clique vizinha C^*. Terminando quando não há clique melhor em sua vizinhança, constituindo uma solução localmente ótima.

Algorithm 3: BuscaLocal()

Entrada: Grafo $G = (V, E)$, clique maximal inicial C
Saída: Clique maximal localmente ótima

enquanto *existe par* (u, v) *que melhora* C **faça**
    Escolher um par (u, v) em $V \setminus C$ que seja adjacente a todos os vértices de C, exceto um único vértice $z \in C$ e tal que $uv \in E$.
    $C \leftarrow (C \setminus \{z\}) \cup \{u, v\}$;
    $C \leftarrow$ MaximizarClique(G, C) ; `// A clique vizinha pode não ser maximal`
retorna C

Com isso, formalizamos um modelo para criação de qualquer heurística baseada em GRASP para cliques maximais.

4. Discussão

No contexto da validação estrutural via árvore de Zykov: Como o processo de poda por invalidade em nosso modelo depende da validação de uma clique C de G, encontrar uma clique C de grande cardinalidade torna-se um componente crítico, uma vez que cada clique representa um subconjunto de vértices potencialmente inviável sob as restrições do problema de alocação. Heurísticas eficientes reduzem o custo exploratório da árvore ao identificar cliques maiores, aumentando a poda da árvore de Zykov.

A expectativa inicial é que o GRASP apresente desempenho superior à escolha aleatória, validaremos essa hipotese experimentalmente na próxima seção.

Para a implementação de uma heurística baseada em GRASP, precisamos solidificar os parâmetros que generalizamos anteriormente na descrição de um GRASP para cliques maximais. Tais parâmetros e nossas escolhas para o trabalho foram:

- **Número de iterações** Cada iteração constrói uma clique, guardamos a clique de maior cardinalidade encontrada, o valor escolhido para esse parâmetro foi cinco.
- **Função gulosa de avaliação**: $score(v) = grau(v)$, supondo que vértices com maior grau tem maior probabilidade de gerar cliques grandes.
- **Lista restrita de candidatos (RCL)** São os candidatos (vértices) da escolha randômica. Guardamos os vértices que tenham um $score$ com, no mínimo, 80% do maior valor observado.

Com esses parâmetros definidos, conduzimos uma série de execuções utilizando as duas abordagens. Analisamos os resultados obtidos na próxima seção.

5. Resultados

Com as duas heurísticas previamente discutidas, realizamos três execuções de comparação: uma com a heurística aleatória, o GRASP completo (com construção e busca local) e o GRASP incompleto (apenas construção, sem fase de busca local). O objetivo é observar o tamanhos das cliques geradas e os ganhos de cada fase do GRASP para as instâncias abordadas na árvore de Zykov.

5.1. Instância avaliada

Os testes foram realizados sobre uma mesma instância de um grafo derivado do modelo de alocação de disciplinas. Essa instância foi construida a partir de dados reais da FACOM-UFMS, e foi amplamente testada dentro do trabalho principal. Assim, a análise aqui apresentada reflete o comportamento real das heurísticas dentro do contexto da validação de instâncias para alocação de disciplinas com a árvore de Zykov. A instância em questão corresponde ao semestre letivo de 2025-1.

5.2. Análise dos testes

A Figura 1 apresenta a distribuição acumulada dos tamanhos de clique obtidos para as três abordagens testadas. Nota-se que ambas as variações do GRASP atingem majoritariamente cliques de tamanho entre 29 e 30, enquanto a heurística aleatória distribui-se entre valores de 8 até 31, servindo como apoio empírico e sustentando a hipotese levantada de que o GRASP performaria melhor.

Tabela 1. Quantidade total de cliques geradas durante a execução da árvore de Zykov

Heurística	Total de cliques geradas
GRASP completo	2566
GRASP incompleto (sem busca local)	2646
Heurística aleatória	2766

As linhas da Figura 1 acumulam todos os tamanhos de clique obtidos pelos testes de cada heurística. Assim, cada curva mostra, para cada tamanho de clique, a proporção de execuções que obtiveram valores menores ou iguais a ele: as heurísticas GRASP concentram rapidamente a maior parte das execuções em cliques de 29 a 30, enquanto a heurística aleatória se espalha uniformemente.

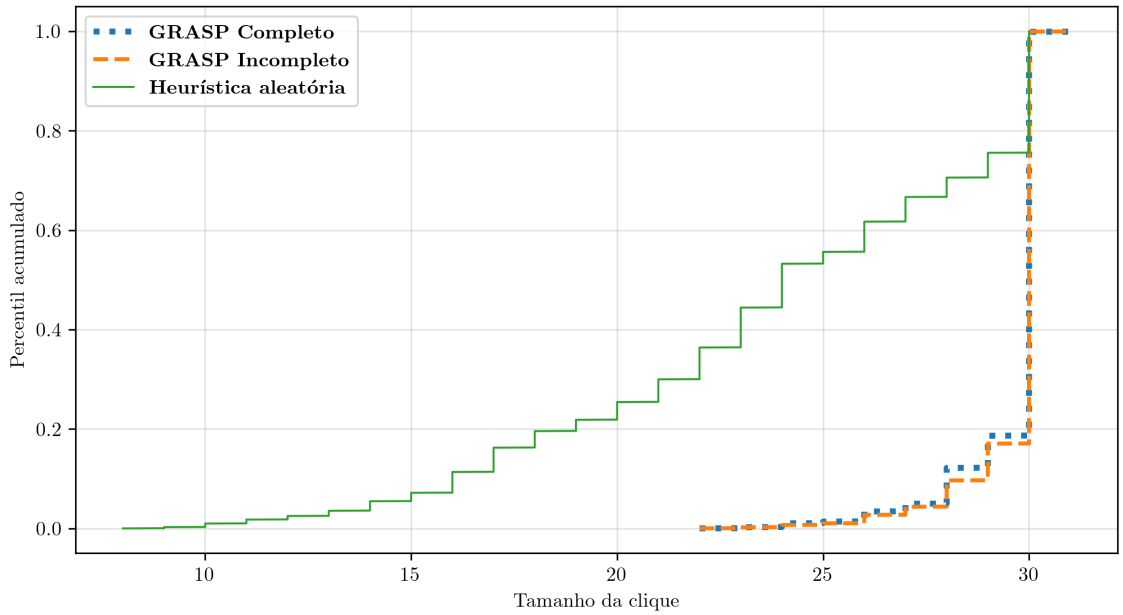

Figura 1. Distribuição acumulada dos tamanhos de clique encontrados por cada heurística.

Curiosamente, para o caso desta instância, não há ganho observado com a adição da etapa de busca local. Teorizamos, então, que para as instâncias construídas para a árvore de Zykov, a etapa de construção do GRASP sozinha já é adequada para o caso de uso.

Isso porque, dentro da árvore de Zykov, nos limitamos a 35 cores que correspondem a um semestre letivo da UFMS. Logo, nenhuma clique pode exceder 35 vértices. Assim, a fase construtiva do GRASP parece suficientemente eficaz para a construção de cliques deste tamanho.

6. Conclusão

Este trabalho, desenvolvido como AOE e em paralelo ao Trabalho de Conclusão de Curso, permitiu consolidar conhecimento prático no uso e implementação de heurísticas para

busca de grandes cliques em grafos. Verificamos empiricamente o ganho de performance de uma heurística GRASP e levantamos um resultado relevante sobre a ausência da etapa de busca local não comprometer a qualidade final para as instâncias relevantes da árvore de Zykov de alocação de disciplinas.

Referências

de Souza Oliveira, L. (2025). Validação de instâncias de coloração de vértices aplicado à alocação de disciplinas.

Resende, M. G. and Ribeiro, C. C. (2014). Greedy randomized adaptive search procedures: Advances and applications. In Burke, E. K. and Kendall, G., editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pages 287–312. Springer, 2 edition.