SET PACKING: Contribuições Pedagógicas para o Aprendizado no Escopo da Teoria da Computação

SET PACKING: Pedagogical Contributions for Learning within the Scope of the Theory of Computation

Emanuel B. Fonseca[1], Victhor C. Magalhães[1], Daniel Martins da Silva[2] e Tanilson D. dos Santos[1]

[1] *Universidade Federal do Tocantins, Ciência da Computação, Palmas, Brasil*
[2] *Universidade Federal do Norte do Tocantins, Araguaína, Brasil*

Data de recebimento do manuscrito: 30/11/2025
Data de aceitação do manuscrito: 30/01/2026
Data de publicação: 10/02/2026

Resumo—Este trabalho propõe uma abordagem metodológica e didática para a reprodução e elucidação dos conceitos de NP-Completude e da intratabilidade inerente a problemas computacionais, especialmente os combinatórios, utilizando o *Set Packing Problem* (SP) como estudo de caso. A metodologia consiste na exposição detalhada do problema, seguida pela reprodução da demonstração formal do pertencimento do SP à classe NP-Completa, incluindo a construção e análise de seu verificador polinomial e a apresentação passo a passo da técnica de redução polinomial de *CLIQUE* para *SP*. O principal resultado e a contribuição deste artigo é o desenvolvimento de um material pedagógico que visa aprimorar a compreensão integral dos conceitos da Teoria da Computação, auxiliando o público leigo a assimilar de forma eficaz o significado da complexidade computacional.

Palavras-chave—Teoria da Computação, Complexidade Computacional, NP-Completude, Empacotamento de Conjuntos, Redução Polinomial

Abstract—*This work proposes a methodological and didactic approach for the reproduction and elucidation of the concepts of NP-Completeness and the inherent intractability of combinatorial problems, using the Set Packing Problem (SP) as a case study. The methodology consists of a detailed exposition of the problem, followed by the reproduction of the formal demonstration of SP's membership in the NP-Complete class, including the construction and analysis of its polynomial verifier and the step-by-step presentation of the polynomial reduction technique from CLIQUE to SP. The main result and contribution of this article is the development of pedagogical material that aims to enhance the integral understanding of Theory of Computation concepts, assisting the public in effectively assimilating the meaning of computational complexity.*

Keywords—*Theory of Computation, Computational Complexity, NP-Completeness, Set Packing Problem, Polynomial Reduction*

I. Introdução

Teoria da Computação (TC) é a base teórica para a Ciência da Computação e Engenharia da Computação, fornecendo as ferramentas conceituais necessárias para compreender o que pode ser computado e, crucialmente, com qual eficiência. Foi no início da década de 1970 que a disciplina ganhou uma nova dimensão com o surgimento da *Teoria da Complexidade Computacional*, impulsionada pelo trabalho de Stephen Cook (1971) [1] e, notavelmente, por Richard Karp (1972) [2]. Este campo de estudo buscou categorizar problemas com base nos recursos (tempo e espaço) requeridos por seus melhores algoritmos de solução, culminando na definição das classes $\mathcal{P}$ (*Polynomial Time*) e $\mathcal{NP}$ (*Nondeterministic Polynomial Time*) e na formulação do problema $\mathcal{P}$ **versus** $\mathcal{NP}$, definido como um dos Millennium Prize Problems pelo Clay Mathematics Institute [3].

Neste contexto, a complexidade representa o que de fato significa a *intratabilidade* de problemas, ou seja, a prova de que certas questões computacionais não podem ser resolvidas de forma eficiente (em tempo polinomial) por qualquer algoritmo determinístico conhecido, a menos que $\mathcal{P}$=$\mathcal{NP}$. A compreensão da NP-Completude é, portanto, essencial para que o futuro profissional saiba quando buscar soluções exatas ou métodos alternativos como algoritmos aproximados e heurísticos.

Considere por exemplo um conjunto universo U e uma coleção de subconjuntos S. O objetivo é verificar se existe

Dados de contato: Emanuel Badaró Fonseca, badaro.fonseca@uft.edu.br

uma subcoleção dentro de S contendo pelo menos k subconjuntos que sejam mutuamente disjuntos, isto é, que não compartilhem nenhum elemento comum entre si (a interseção entre qualquer subconjunto escolhido é vazia). Este é o Problema do Set Packing (Empacoteamnto de Conjuntos), um dos 21 problemas NP-Completos de Karp [2] e o qual será utilizado como estudo de caso neste artigo.

A complexidade inerente à Teoria da Computação, especificamente no que tange às reduções polinomiais e à classe NP-Completa, representa um desafio pedagógico constante, apesar da relevância teórica de problemas como o Set Packing. Com o objetivo de mitigar essas dificuldades, este trabalho propõe uma construção pedagógica da prova de NP-Completude do Set Packing. A estrutura do artigo segue uma lógica sequencial: a Seção 2 fornece o embasamento teórico e as definições fundamentais. A Seção 3 examina a literatura pertinente e trabalhos correlatos. As Seções 4 e 5 constituem o cerne do trabalho, apresentando a descrição e a demonstração formal do problema, assim como a estratégia de redução. As contribuições e reflexões sobre o aprendizado são debatidas na Seção 6, seguidas pelas conclusões na Seção 7.

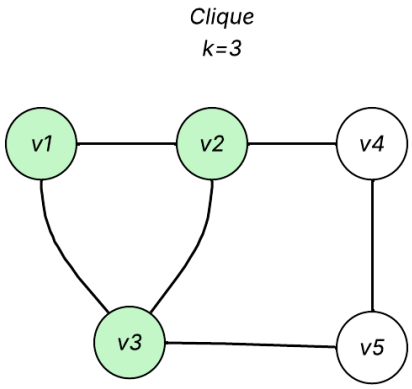

Figura 1: Exemplo de uma clique de tamanho 3. Clique $C = \{v_1, v_2, v_3\}$.

II. PRELIMINARES

Esta seção apresenta os conceitos e ferramentas de análise pertinentes que serão utilizados ao longo de todo este trabalho.

Um *conjunto* é definido como uma coleção não ordenada de elementos distintos. A partir deste conceito, estabelece-se a relação de *subconjunto*: diz-se que um conjunto B é um subconjunto de um conjunto A (denotado por $B \subseteq A$) se todos os elementos presentes em B são também elementos de A. Por exemplo, dado o conjunto $A = \{1,2,3\}$, podemos definir um subconjunto $B = \{1,2\}$. Como ambos os elementos de B estão em A, temos que $B \subseteq A$.

Um *grafo* G é definido formalmente como um par ordenado $G = (V,E)$, composto por um conjunto finito e não vazio de vértices V e um conjunto de arestas E. Neste contexto, os vértices constituem os elementos de V e representam as entidades ou objetos que estão sendo modelados. Já as arestas são definidas como pares não ordenados (u,v) de vértices distintos de V, representando as conexões ou relações estabelecidas entre esses vértices.

Considere o grafo presente na Figura 1 como exemplo, onde o conjunto de vértices é $V = \{v_1, v_2, v_3, v_4, v_5\}$ e o conjunto de arestas é $E = \{(v_1, v_2), (v_2, v_3), (v_1, v_3), (v_4, v_5), (v_3, v_5)\}$. Neste caso, o vértice v_1 está conectado ao vértice v_2 pela aresta (v_1, v_2).

Uma *clique* em um grafo é um subconjunto de vértices $C \subseteq V$ tal que todo par de vértices distintos em C está conectado por uma aresta em E. A Figura 1 mostra um exemplo visual de uma clique de tamanho 3 em um grafo.

Na Teoria da computação, um *problema de decisão* é uma questão que tem uma resposta simples de “sim” ou “não”. É como fazer uma pergunta que pode ser respondida com um "verdadeiro" ou "falso".

Utilizando o grafo da Figura 1 e $k = 3$, a pergunta “Existe uma clique de tamanho 3 neste grafo?” é um problema de decisão cuja resposta é “Sim”.

A *complexidade computacional* é o estudo de quão eficientemente um algoritmo (uma receita para resolver um problema) pode resolver um problema em termos de tempo e recursos (como memória). Medimos o “tempo” pelo número de passos que o algoritmo leva para rodar, especialmente à medida que o tamanho da entrada aumenta.

Para ordenar uma lista de n números, um algoritmo simples pode levar n^2 passos (como o Bubble Sort), enquanto um mais eficiente leva $n \log n$ passos (como o Merge Sort).

Um problema é classificado como pertencente ao Tempo Polinomial $\mathcal{P}$ quando existe um algoritmo capaz de encontrar sua resposta (do tipo sim ou não, para problemas de decisão) em um número de passos que cresce, no máximo, como um polinômio do tamanho da entrada. Por essa razão, problemas em P são geralmente categorizados na literatura como “fáceis” ou “eficientemente solúveis”.

A Classe $\mathcal{NP}$ (Non-deterministic Polynomial time) agrupa problemas de decisão para os quais, dada uma “solução candidata” (denominada certificado), é possível *verificar* sua correção em tempo polinomial. Esta definição não implica necessariamente que a solução possa ser encontrada de forma eficiente, mas sim que podemos confirmar rapidamente a validade de uma proposta apresentada. Vale ressaltar, por fim, que todo problema pertencente à classe $\mathcal{P}$ está também contido em $\mathcal{NP}$.

Por exemplo, no jogo Sudoku, encontrar a solução para um tabuleiro vazio pode ser demorado, mas dada uma grade preenchida (o certificado), verificar se ela segue as regras (não repetir números nas linhas, colunas e quadrantes) é muito rápido (polinomial).

A *Redução Polinomial* ($A \leq_p B$) é uma ferramenta formal utilizada para demonstrar que um problema é “pelo menos tão difícil” quanto outro. O processo consiste na capacidade de transformar eficientemente (em tempo polinomial) qualquer instância de um problema A em uma instância de um problema B, preservando a equivalência das respostas: a instância de A é positiva se, e somente se, a instância correspondente de B também o for.

Como exemplo, suponha que o Problema A seja “encontrar a saída de um labirinto físico de sebes”. Se soubermos

 ISSN: 2675-3588

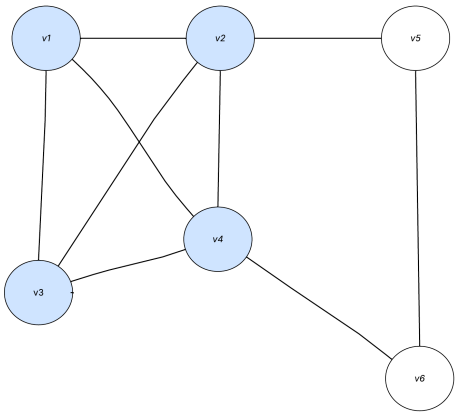

Figura 2: Entrada: o grafo G e $k = 4$. A resposta é "Sim", pois $\{v_1, v_2, v_3, v_4\}$ é uma clique.

transformar (reduzir) este labirinto em um desenho de Grafo (Problema B), onde cada cruzamento é um vértice e cada corredor é uma aresta, podemos usar um algoritmo de computador conhecido para resolver B. Assim, resolver o grafo resolve o labirinto.

O problema da Clique é um problema de decisão NP-Completo.

CLIQUE

Entrada: Um grafo $G = (V, E)$ e um inteiro $k \geq 1$.
Questão: Existe um subconjunto de vértices $V' \subseteq V$ tal que $|V'| \geq k$ e, para todo par de vértices distintos em V', existe uma aresta em E que os conecta?

Considere como exemplo a Figura 2.

O problema do Conjunto Independente também é um problema de decisão NP-Completo.

INDEPENDENT SET

Entrada: Um grafo $G = (V, E)$ e um inteiro $k \geq 1$.
Questão: Existe um subconjunto de vértices $V' \subseteq V$ tal que $|V'| \geq k$ e não há arestas em E conectando quaisquer dois vértices em V'?

Considere como exemplo de Independent Set a Figura 3.

O problema do Empacotamento de Conjuntos é central para este trabalho e pertence à classe dos problemas NP-Completos.

SET PACKING

Entrada: Uma coleção $C = \{S_1, S_2, \ldots, S_m\}$ de subconjuntos de um conjunto universal U, e um inteiro $k \geq 1$.
Questão: Existe uma subcoleção $C' \subseteq C$ tal que $|C'| \geq k$ e todos os conjuntos em C' são mutuamente disjuntos (ou seja, para quaisquer dois conjuntos $S_i, S_j \in C'$ com $i \neq j$, tem-se $S_i \cap S_j = \emptyset$)?

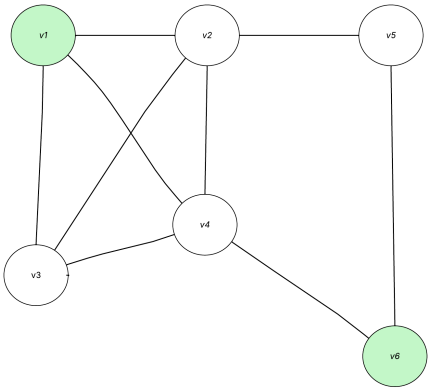

Figura 3: Entrada: Grafo G e $k = 2$. A resposta é "Sim", pois o subconjunto $\{v_1, v_6\}$ tem tamanho 2 e não existe a aresta (v_1, v_6) em $E(G)$.

Exemplo: Entrada: $U = \{1,2,3,4\}$, $C = \{\{1,2\},\{2,3\},\{3,4\}\}$ e $k = 2$. A resposta é "Sim", pois a subcoleção $C' = \{\{1,2\},\{3,4\}\}$ tem tamanho 2 e os conjuntos são disjuntos.

III. Trabalhos Relacionados

A literatura sobre o problema do Set Packing (SP) é extensa, abrangendo desde as provas fundamentais de complexidade até aplicações modernas em teoria dos jogos e otimização. Nesta seção, destacamos trabalhos que fundamentam a teoria, exploram limites de tratabilidade e dialogam com a proposta pedagógica deste artigo.

A referência primária para a classificação do Set Packing é o trabalho clássico de Karp [2], onde demonstrou em suas pesquisas a redutibilidade entre 21 problemas combinatórios, estabelecendo o Set Packing como NP-Completo através de uma cadeia de reduções originada na Satisfiabilidade (SAT). Este trabalho define a posição do problema na hierarquia de complexidade. Complementarmente, Garey e Johnson [4] sistematizaram a metodologia de provas de NP-Completude. O presente artigo adota a estrutura formal de construção e verificação polinomial proposta por eles, adaptando seu rigor técnico a uma abordagem pedagógica.

Avançando para estratégias de resolução exata e modelagem, no trabalho de Delorme, Gandibleux e Rodriguez (2004) [5], podemos notar que eles trabalham com uma nova abordagem de modelagem para a resolução exata. Na pesquisa, os autores abordam o problema de Set Packing propondo uma transformação para o problema de clique máxima e obtêm uma redução significativa na complexidade do espaço de busca. Os resultados são interessantes por demonstrarem como a reformulação do modelo matemático e o uso de limitantes superiores podem acelerar a resolução de instâncias difíceis.

Expandindo o escopo para generalizações com aplicações práticas, no trabalho de Muritiba et al. (2010) [6], podemos notar que eles trabalham com o problema de empacotamento de bins com conflitos (BPPC). Na pesquisa, os autores abordam o problema propondo novos limites inferiores baseados na computação de cliques maximais, novos

limites superiores através de uma abordagem meta-heurística envolvendo busca tabu e um operador de cruzamento, e um algoritmo exato baseado em uma formulação de cobertura de conjuntos, resolvido por meio de geração de colunas e branch-and-price. Os resultados são interessantes por demonstrar a eficácia dos algoritmos propostos em um vasto conjunto de instâncias de referência da literatura, resolvendo 780 de 800 instâncias para a otimalidade e melhorando consistentemente algoritmos anteriores.

Investigando a tratabilidade sob condições específicas, no trabalho de Jia, Zhang e Chen (2004) [7], podemos notar que eles trabalham com a complexidade parametrizada do problema de empacotamento de conjuntos. Na pesquisa, os autores abordam o problema de m-Set Packing (onde o tamanho de cada conjunto é limitado por uma constante m) propondo um algoritmo eficiente de complexidade parametrizada e obtêm a prova de que o problema é tratável por parâmetro fixo (FPT) em relação ao tamanho da solução k. Os resultados são interessantes por demonstrarem que, ao restringir o tamanho dos conjuntos, é possível superar a intratabilidade geral e encontrar soluções exatas de forma eficiente para valores pequenos de k.

Sob a ótica de soluções aproximadas para o caso geral, no trabalho de Fürer e Yu (2014) [8], podemos notar que eles trabalham com a análise teórica de algoritmos de aproximação. Na pesquisa, os autores abordam o problema de k-Set Packing investigando o poder das melhorias locais (local improvements) e obtêm limites de aproximação refinados para o problema. Os resultados são interessantes por aprofundarem o entendimento sobre as limitações e capacidades da busca local, fornecendo garantias mais justas para a qualidade das soluções encontradas por essa classe de algoritmos.

Retornando às inovações em modelagem matemática, no trabalho de Alidaee et al. (2008) [9], podemos notar que eles trabalham com uma nova abordagem baseada em Programação Quadrática Binária Irrestrita (UBQP/QUBO). Na pesquisa, os autores abordam o problema de Set Packing transformando as restrições de disjunção em penalidades na função objetivo quadrática e obtêm soluções de alta qualidade que rivalizam com métodos especializados em termos de tempo e eficiência. Os resultados são interessantes por demonstrarem que uma estrutura unificada e sem restrições pode simplificar a resolução de problemas combinatórios complexos, permitindo o uso de heurísticas genéricas robustas.

Por fim, o trabalho de Lassance e Bianchini [10] investigou o impacto de estratégias didáticas no ensino de Teoria da Computação. O estudo conclui que abordagens descritivas e a participação ativa dos discentes (via seminários) são eficazes para diluir a complexidade das reduções polinomiais. Seguindo esta diretriz pedagógica, nosso trabalho adota uma estratégia de decomposição visual e prova passo a passo, visando contribuir com o entendimento de conceitos abstratos de intratabilidade computacional pelo público em geral.

IV. Descrição do Problema

O problema do Set Packing (Empacotamento de Conjuntos), doravante denominado SP, é um problema fundamental na otimização combinatória e na teoria da complexidade computacional. Formalmente, dado um universo finito U e uma família de subconjuntos $S = \{S_1, S_2, \ldots, S_m\}$, onde $S_i \subseteq U$, um *packing* é uma subcoleção $S' \subseteq S$ tal que todos os conjuntos em S' são mutuamente disjuntos, ou seja, $S_i \cap S_j = \emptyset$ para quaisquer $S_i, S_j \in S'$ distintos [4].

O problema pode ser abordado sob duas perspectivas. Na versão de *otimização*, o objetivo é encontrar a subcoleção S' de cardinalidade máxima. Na versão de *decisão* — a qual utilizamos para a prova de NP-completude — a entrada inclui um inteiro k, e a pergunta é se existe um packing de tamanho pelo menos k ($|S'| \geq k$) [2]. Uma generalização comum é o *Weighted Set Packing*, onde cada conjunto S_i possui um peso w_i, e o objetivo é maximizar a soma dos pesos dos conjuntos disjuntos selecionados.

A relevância do SP decorre de sua capacidade de modelar situações de alocação de recursos onde o compartilhamento é impossível (restrição de exclusividade). A aplicação mais notável ocorre em *Leilões Combinatórios* [11]. Neste cenário, um leiloeiro tem um conjunto de itens distintos (U) e os licitantes oferecem lances por "pacotes" de itens (S_i). Como cada item só pode ser vendido uma vez, o leiloeiro deve selecionar um conjunto de lances vencedores que não disputem o mesmo item, maximizando o lucro total. Outras aplicações críticas incluem o *Airline Crew Scheduling* (escalonamento de tripulações), onde cada voo deve ser coberto por uma única equipe e as equipes (conjuntos de voos) não podem estar em dois lugares ao mesmo tempo [12].

Para fins pedagógicos, o SP pode ser visualizado como o dilema de um organizador de festas que possui uma lista de grupos de amigos que desejam comparecer ao evento, Figura 4. O universo U representa as cadeiras disponíveis na mesa principal. Cada grupo (S_i) exige sentar-se em cadeiras específicas e recusa-se a compartilhar seus assentos com estranhos. Se um grupo deseja a cadeira 3 e outro também deseja a cadeira 3, eles são incompatíveis. O desafio do organizador é aceitar o maior número possível de grupos sem gerar conflitos de assentos.

Embora o SP seja NP-difícil no caso geral, existem subclasses importantes que admitem algoritmos eficientes:

- **Cardinalidade Limitada ($|S_i| \leq 2$):** Se todos os conjuntos na família S tiverem no máximo 2 elementos, o problema torna-se equivalente ao *Maximum Matching* (Emparelhamento Máximo) em grafos. Neste caso, os elementos de U são vértices e os conjuntos S_i são arestas. O problema pode ser resolvido em tempo polinomial $O(E\sqrt{V})$ pelo algoritmo de Edmonds [13].

- **Grafos de Intervalo:** Se os elementos de U puderem ser ordenados linearmente tal que cada S_i forme um intervalo contíguo, o problema equivale ao *Interval Scheduling*, que pode ser resolvido com uma estratégia gulosa em $O(n \log n)$ [14].

Para o caso geral, algoritmos exatos baseados em programação dinâmica ou inclusão-exclusão atingem complexidades da ordem de $O^*(2^n)$, o que é impraticável para instâncias grandes [15]. Essa intratabilidade computacional, contrastando com a eficiência das subclasses restritas apresentadas, sugere que o Set Packing pertence à classe dos problemas mais difíceis da computação. A seção a seguir formaliza essa intuição, provando a NP-Completude

 ISSN: 2675-3588

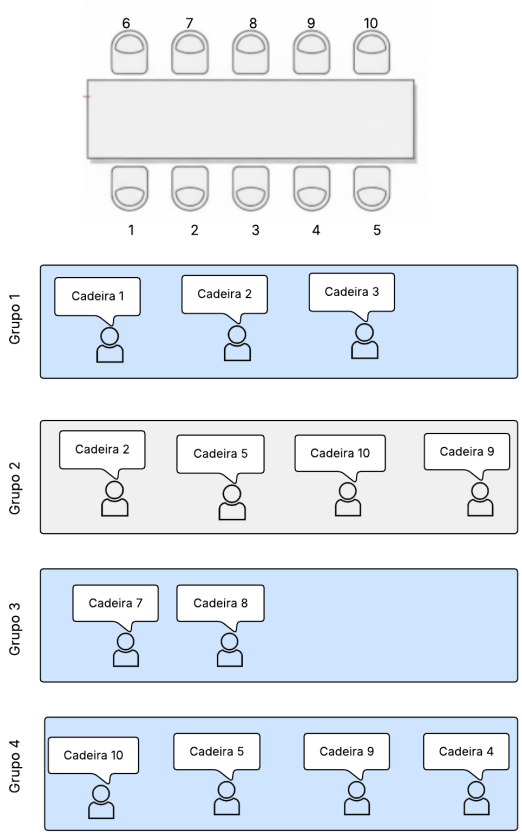

Figura 4: Representação visual de uma instância de Set Packing. O universo $U = \{1,..,10\}$. Os grupos 1, 3 e 4 formam um packing válido (azul). O grupo 2 cria conflito com os grupos 1 e 4.

do problema através de uma redução polinomial a partir do problema da Clique.

V. Demonstração e Contribuições

A seguir, estabelecemos o lema principal deste artigo.

Lema 1. *Set Packing é NP-Completo.*

Proof. Seguindo o esquema clássico, dividimos a prova em duas etapas identificadas: (1) provar que $SP \in \mathcal{NP}$ (NP-pertinência) e (2) provar que SP é NP-difícil por meio de uma redução polinomial apropriada.

Tome (1)

Para mostrar que $SP \in \mathcal{NP}$, basta exibir um verificador polinomial que, dada uma solução candidata S', determine se ela constitui um subconjunto de l conjuntos mutuamente disjuntos.

A instância do problema consiste em um universo U com $n = |U|$ elementos e uma coleção de m conjuntos $S = \{S_1, \ldots, S_m\}$. O certificado é uma subcoleção $S' \subseteq S$ supostamente de tamanho l.

O verificador executa duas tarefas: (i) verificar se $|S'| = l$ e (ii) verificar a disjunção par a par entre os conjuntos de S'.

Temos que o primeiro passo é executado em $O(|S'|)$, que é limitado por $O(m)$, pois $|S'| \leq m$.

Para o segundo passo, o algoritmo percorre cada par distinto de conjuntos em S'. O número de pares é dado pela combinação de $|S'|$ elementos tomados 2 a 2:

$$\binom{|S'|}{2} = \frac{|S'|!}{2!(|S'|-2)!} = \frac{|S'|(|S'|-1)}{2} = \frac{|S'|^2 - |S'|}{2}.$$

Como o termo dominante é quadrático e sabemos que $|S'| \leq m$, conclui-se que o número de verificações é limitado por $O(m^2)$.

Para cada par (S_i, S_j), verifica-se se $S_i \cap S_j = \emptyset$. Se representarmos cada conjunto como uma lista de elementos, o teste de interseção pode ser feito verificando elemento a elemento, exigindo tempo proporcional ao tamanho total do universo. Assim, cada teste leva tempo $O(n)$.

Portanto, o tempo total gasto na verificação das disjunções é:

$$O(m^2) \cdot O(n) = O(m^2 n).$$

Neste termo, o fator quadrático m^2 provém da comparação de todos os pares possíveis de conjuntos, enquanto o fator linear n surge do custo computacional de verificar a interseção de dois conjuntos específicos.

Somando todos os passos, obtemos um verificador com custo máximo

$$O(m) + O(m^2 n) = O(m^2 n),$$

que é polinomial no tamanho da entrada. Assim, concluímos que $SP \in \mathcal{NP}$.

Tome (2)

Para provar a NP-dificuldade, reduziremos o problema da Clique ao Set Packing. O problema da Clique é escolhido como origem pois a relação "dois vértices são conectados" pode ser mapeada inversamente para "dois conjuntos são disjuntos" se construirmos o universo baseados nas arestas que *não* existem. Assim temos a seguinte redução: CLIQUE $\leq_p$ SP.

Seguiremos com a estratégia de construção em que a redução $f(G,k) = (U,S,l)$ transforma uma instância de Clique em uma de Set Packing preservando a propriedade isomórfica

$$\textit{Vértices adjacentes em } G \iff \textit{Conjuntos disjuntos em } S.$$

A construção define o inteiro l com o mesmo tamanho da clique, ou seja, $l = k$. O universo U é composto pelas **não-arestas** de G, de modo que cada par de vértices que *não* está conectado em G vira um elemento em U:

$$U = \{(v_i, v_j) \mid v_i, v_j \in V, i \neq j, (v_i, v_j) \notin E\}$$

O propósito desta construção é que, se dois vértices não têm aresta, eles "compartilham" um conflito (o elemento em U), impedindo que sejam escolhidos juntos. Para a coleção S, para cada vértice $v_i \in V$, criamos um conjunto S_i que contém todas as não-arestas incidentes a v_i:

$$S_i = \{(v_i, v_j) \in U \mid j \neq i\}$$

A Figura 5 ilustra um exemplo visual desta redução, destacando como as não-arestas formam o universo de conflito.

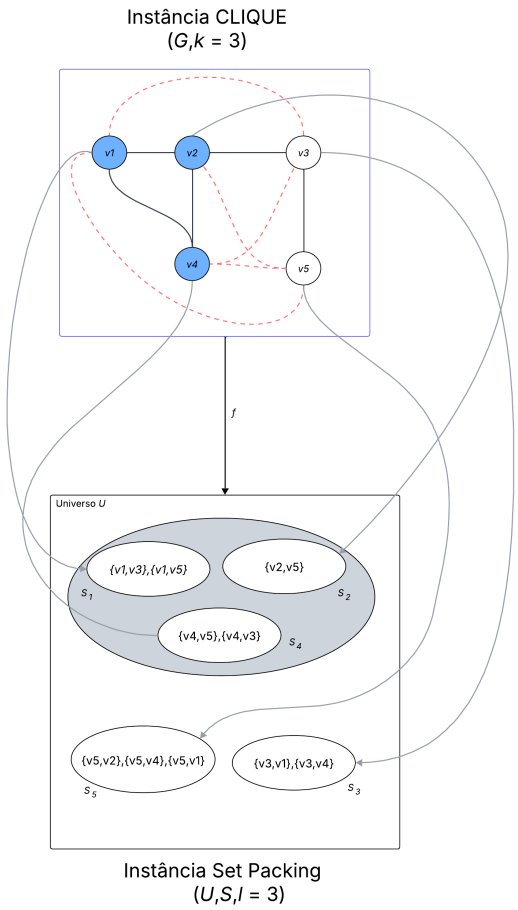

Figura 5: A existência da clique $C = \{v_1, v_2, v_4\}$ (em azul) no grafo implica a existência de uma subcoleção de conjuntos disjuntos $S' = \{S_1, S_2, S_4\}$ na instância de Set Packing.

No exemplo ilustrado na Figura 5, é possível observar a redução f de CLIQUE para Set Packing observa-se que os vértices v_3 e v_5 não são adjacentes a todos os membros da clique, o que gera um conjunto de não-arestas (como (v_1, v_3), (v_3, v_4), (v_2, v_5), (v_4, v_5) e (v_1, v_5)) que passam a constituir o universo U. Pela regra de construção, onde cada S_i contém as não-arestas incidentes a v_i, os conjuntos S_3 e S_5 acabam compartilhando elementos com outros conjuntos, o que representa conflitos. Em contrapartida, como v_1, v_2 e v_4 estão plenamente conectados entre si, não existem não-arestas entre eles no universo, garantindo que seus conjuntos correspondentes S_1, S_2 e S_4 sejam mutuamente disjuntos, satisfazendo a condição de validade do Set Packing.

Contudo, não basta apenas montar a construção da redução. Também precisamos mostrar que (i) esta redução é polinomial e (ii) preserva a simetria entre os problemas.

(i). Devemos provar que a transformação preserva a resposta do problema original, ou seja, (G,k) tem Clique $\iff$ (U,S,l) tem Set Packing.
($\Rightarrow$) *Se G tem clique de tamanho k, então (U,S) tem packing de tamanho $l = k$.*

Proof. Seja C a clique em G. Selecionamos os conjuntos correspondentes $S' = \{S_i \mid v_i \in C\}$. Temos $|S'| = |C| = k = l$. Para quaisquer dois conjuntos distintos $S_i, S_j \in S'$, os vértices correspondentes v_i, v_j estão na clique. Logo, a aresta (v_i, v_j) *existe* em E. Como U contém apenas *não-arestas*, o par $(v_i, v_j) \notin U$. A única interseção possível entre S_i e S_j seria o elemento (v_i, v_j). Como $(v_i, v_j) \notin U$, ele não pode estar nem em S_i nem em S_j. Logo, $S_i \cap S_j = \emptyset$. Assim, S' é um Set Packing válido. □

($\Leftarrow$) *Se (U,S) tem packing de tamanho $l = k$, então G tem clique de tamanho k.*

Proof. Seja S' o packing. Selecionamos os vértices $C = \{v_i \mid S_i \in S'\}$. Temos $|C| = |S'| = l = k$. Suponha, por contradição, que C *não* seja uma clique. Então existem dois vértices $v_i, v_j \in C$ tal que a aresta $(v_i, v_j) \notin E$. Se a aresta não existe, então o par (v_i, v_j) é uma não-aresta, logo $(v_i, v_j) \in U$. Pela construção, S_i contém todas as não-arestas de v_i (incluindo (v_i, v_j)) e S_j contém todas as de v_j (incluindo (v_i, v_j)). Portanto, $(v_i, v_j) \in S_i \cap S_j$, o que implica $S_i \cap S_j \neq \emptyset$. Isso contradiz a hipótese de que S' é um packing (conjuntos disjuntos). Logo, a aresta deve existir para todos os pares, e C é uma clique. □

(ii). A construção do universo U exige iterar sobre todos os pares de vértices, uma operação limitada por $\binom{n}{2} = O(n^2)$. A construção da coleção S exige criar n conjuntos, onde para cada um verificamos $n-1$ pares, totalizando também $O(n^2)$. Portanto, conclui-se que a função de redução f é executada em tempo polinomial: $O(n^2)$.

Assim por (1) e (2), provamos o Lema 1 ao demonstrarmos que o Set Packing é NP-Completo, validando sua pertinência a $\mathcal{NP}$ e em seguinda apresentando uma redução polinomial a partir do CLIQUE. Esta prova ilustra o uso de "conflitos" (neste caso, não-arestas) como elementos básicos de construção para forçar restrições de exclusão mútua. □

VI. Resultados e Reflexões

A elaboração deste trabalho resultou na produção de um material didático autossuficiente para o estudo da NP-Completude do problema *Set Packing* (SP). Diferentemente de abordagens tradicionais que frequentemente omitem passos intermediários das reduções polinomiais, os resultados aqui apresentados focam na explicitação da lógica construtiva. A principal contribuição pedagógica deste estudo é a sistematização visual e analógica da redução $CLIQUE \leq_p SP$. Na literatura clássica, a definição do universo U como o conjunto de "não-arestas" é frequentemente apresentada de forma puramente algébrica, o que dificulta a visualização geométrica por parte do estudante. Para mitigar essa barreira, desenvolvemos a analogia do "Organizador de Festas", uma narrativa lúdica que provou-se eficaz para traduzir a restrição abstrata de "interseção vazia" para uma restrição concreta de "conflito de assentos", facilitando a intuição inicial sobre o problema. Adicionalmente, a diagramação da redução apresentada nas Figuras 5 e 4 permite ao aluno rastrear visualmente como um vértice no grafo se transforma em um conjunto, e como a ausência de uma aresta se materializa em um elemento compartilhado no universo U.

Durante a estruturação da prova de NP-Dificuldade, identificamos que a maior dificuldade cognitiva reside na "inversão lógica" exigida pela redução a partir do problema

 ISSN: 2675-3588

da Clique. Enquanto reduções a partir do *Independent Set* (IS) mapeiam arestas diretamente para elementos do universo (conflito direto), a redução a partir da Clique exige o uso do grafo complementar (ou não-arestas). Essa distinção é sutil e é uma fonte comum de erro. Para superar esse obstáculo, adotamos a estratégia de definir explicitamente o universo U como um conjunto de "conflitos potenciais", reforçando que, para que um *packing* (pacote de vértices) seja válido, os elementos não podem ter conflitos (não-arestas) entre si — o que força a existência das arestas no grafo original.

Em termos de aplicabilidade acadêmica, este artigo foi estruturado para servir como material complementar na disciplina de Teoria da Computação. A Seção 4 (Descrição do Problema) pode ser utilizada como texto introdutório para aulas sobre problemas de empacotamento, enquanto a Seção 5 (Demonstração) serve como guia para listas de exercícios avançados que exigem a formalização de reduções.

Por fim, embora o foco deste trabalho seja a intratabilidade (NP-Completude), é importante refletir sobre o comportamento prático. Em cenários reais, como os leilões combinatórios mencionados, não se busca a prova de inexistência de solução, mas sim a melhor solução possível em tempo hábil. Experimentos simples com algoritmos gulosos (selecionar o menor conjunto disponível iterativamente) demonstram que, embora não garantam a solução ótima (o k máximo), oferecem aproximações rápidas. Esta constatação reforça a importância pedagógica de distinguir entre a *dificuldade do pior caso* (foco da teoria $\mathcal{NP}$) e a *solubilidade prática* via heurísticas.

VII. Considerações Finais

Este trabalho cumpriu seu objetivo principal de provar a NP-Completude do problema *Set Packing* (SP), oferecendo uma abordagem pedagógica que preenche a lacuna entre a definição formal e a intuição geométrica. Através da redução polinomial a partir do problema da Clique ($CLIQUE \leq_p SP$), demonstramos que a dificuldade computacional de encontrar grupos mutuamente exclusivos em uma coleção é equivalente a encontrar subgrafos completos. O principal resultado obtido não foi apenas a reafirmação da complexidade do problema, mas a sistematização de um método de ensino que utiliza analogias lúdicas (o "Organizador de Festas") e diagramas visuais passo a passo para facilitar a assimilação de conceitos abstratos por estudantes de graduação.

É importante ressaltar, contudo, que o escopo deste artigo limitou-se à análise da *complexidade de pior caso* e à versão de *decisão* do problema. Entre as limitações, destaca-se a ausência de implementação de solucionadores exatos (como *branch-and-bound*) ou heurísticos para resolver instâncias do problema, visto que a implementação restringiu-se ao algoritmo verificador polinomial para validação da classe $\mathcal{NP}$, e não para avaliação de desempenho em *benchmarks*. Além disso, optou-se pelo foco em uma redução única via Clique para garantir a profundidade e a clareza didática, em detrimento da abrangência de outras reduções possíveis, como a partir do *Exact Cover* ou *3-SAT*.

A base teórica estabelecida neste artigo abre diversas frentes para investigação acadêmica e desenvolvimento didático em trabalhos futuros. Uma extensão natural seria o estudo comparativo de algoritmos gulosos e meta-heurísticas (como Algoritmos Genéticos ou *Simulated Annealing*) para a versão de otimização do Set Packing, analisando o *gap* de aproximação dessas soluções. Outra perspectiva relevante é a análise em classes de grafos especiais, visto que o Set Packing torna-se solúvel em tempo polinomial quando o universo e os conjuntos podem ser modelados como grafos de intervalo ou grafos de corda. Adicionalmente, sugere-se investigar a complexidade parametrizada (FPT) para verificar a tratabilidade do problema para valores pequenos de k. Por fim, propõe-se a extensão pedagógica através do desenvolvimento de uma ferramenta de software interativa que permita aos alunos desenharem grafos e visualizarem, em tempo real, a transformação dos vértices e arestas nos conjuntos do Set Packing. Conclui-se que o Set Packing, apesar de sua complexidade inerente, é um excelente veículo para o ensino da Teoria da Computação, servindo como porta de entrada para discussões mais amplas sobre otimização combinatória e limites da computabilidade.

Referências

[1] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the third annual ACM symposium on Theory of computing*, 1971, pp. 151–158.

[2] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, 1972, pp. 85–103.

[3] Clay Mathematics Institute, "Millennium prize problems," https://www.claymath.org/millennium-problems/, 2000, acesso em: 30 nov. 2025.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: W. H. Freeman and Company, 1979.

[5] M. Delorme, X. Gandibleux, and J. Rodriguez, "A new approach for modeling and solving set packing problems," *European Journal of Operational Research*, vol. 152, no. 3, pp. 626–645, 2004.

[6] A. E. F. Muritiba, M. Iori, E. Malaguti, and P. Toth, "Algorithms for the bin packing problem with conflicts," *INFORMS Journal on Computing*, vol. 22, no. 3, pp. 401–415, 2010.

[7] W. Jia, C. Zhang, and J. Chen, "An efficient parameterized algorithm for m-set packing," *Journal of Algorithms*, vol. 50, no. 1, pp. 106–117, 2004.

[8] M. Fürer and H. Yu, "Approximating the k-set packing problem by local improvements," in *International Symposium on Combinatorial Optimization (ISCO 2014)*, ser. Lecture Notes in Computer Science, vol. 8596. Springer, 2014, pp. 408–420.

[9] B. Alidaee, G. Kochenberger, K. Lewis, M. Lewis, and H. Wang, "A new approach for modeling and solving set packing problems," *European Journal of Operational Research*, vol. 186, no. 2, pp. 504–512, 2008.

[10] B. Yasser, Lassance e Guilherme, "Reflexões e práticas pedagógicas no escopo da disciplina deteoria da computação," *Academic Journal on Computing, Engineering and Applied Mathematics*, 2025, referência pedagógica da disciplina de Teoria da Computação.

[11] S. DeVries and R. Vohra, *Combinatorial Auctions: A Survey*. Springer, 2003.

[12] K. L. Hoffman and M. Padberg, "Cutting plane algorithms in planning and scheduling," *Annals of Operations Research*, vol. 43, no. 1, pp. 55–85, 1993.

[13] J. Edmonds, "Paths, trees, and flowers," in *Combinatorial Optimization: Proceedings of the Symposium in Applied Mathematics*. American Mathematical Society, 1965, vol. 18, pp. 48–57.

[14] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 5th ed. Berlin: Springer, 2012.

[15] A. Björklund, T. Husfeldt, and M. Koivisto, "Set packing and covering with character," in *Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010, pp. 1090–1099.

ISSN: 2675-3588