

MAX-2SAT: Contribuições Pedagógicas para o Aprendizado no Escopo da Teoria da Computação

MAX-2SAT: Reflections and Pedagogical Practices within the Scope of the Theory of Computation Course

Raphael Sales de Souza¹, Thiago Gonzaga dos Santos¹, Daniel Martins da Silva¹ e Tanilson Dias dos Santos¹

sales.raphael@mail.uft.edu.br thiago.gonzaga@mail.uft.edu.br danielmartins@mail.uft.edu.br tanilson.dias@mail.uft.edu.br

¹ Universidade Federal do Tocantins, Curso de Ciência da Computação, Tocantins, Brasil

Data de recebimento do manuscrito: 28/11/2025

Data de aceitação do manuscrito: 27/01/2026

Data de publicação: 10/02/2026

Resumo—Este artigo apresenta uma demonstração formal e didática da \mathcal{NP} -completude do problema Maximum 2-Satisfiability (Max-2SAT) por meio de redução polinomial a partir do problema Clique. O Max-2SAT, variante de maximização do problema SAT em que cada cláusula contém no máximo dois literais, questiona se existe uma valoração booleana capaz de satisfazer pelo menos k cláusulas de uma fórmula em forma normal conjuntiva. Embora o problema 2-SAT seja resolvido em tempo polinomial, sua versão de maximização é \mathcal{NP} -completa. A demonstração utiliza uma construção com variável auxiliar que mapeia estruturas de grafos em fórmulas booleanas, estabelecendo correspondência biunívoca entre cliques e valorações satisfatórias. Como contribuições pedagógicas, o trabalho apresenta: (i) prova formal detalhada de \mathcal{NP} -pertinência e \mathcal{NP} -dificuldade; (ii) construção explícita da redução Clique \leq_p Max-2SAT com figuras ilustrativas; (iii) exemplo completo comentado passo a passo; (iv) pseudocódigo do verificador polinomial; e (v) discussões sobre armadilhas comuns e estratégias de compreensão. O material produzido visa facilitar o aprendizado de reduções polinomiais e fortalecer a compreensão sobre a fronteira entre tratabilidade e intratabilidade computacional.

Palavras-chave—Max-2SAT, \mathcal{NP} -completude, Redução Polinomial, Clique, Teoria da Complexidade, Satisfatibilidade Booleana

Abstract—This paper presents a formal and pedagogical demonstration of the \mathcal{NP} -completeness of the Maximum 2-Satisfiability (Max-2SAT) problem through polynomial reduction from the Clique problem. Max-2SAT, a maximization variant of the SAT problem where each clause contains at most two literals, asks whether there exists a Boolean assignment capable of satisfying at least k clauses of a formula in conjunctive normal form. Although the 2-SAT problem is solvable in polynomial time, its maximization version is \mathcal{NP} -complete. The demonstration employs a construction with an auxiliary variable that maps graph structures into Boolean formulas, establishing a bijective correspondence between cliques and satisfying assignments. As pedagogical contributions, this work presents: (i) detailed formal proof of \mathcal{NP} -membership and \mathcal{NP} -hardness; (ii) explicit construction of the Clique \leq_p Max-2SAT reduction with illustrative figures; (iii) complete step-by-step annotated example; (iv) pseudocode for the polynomial verifier; and (v) discussions about common pitfalls and comprehension strategies. The material produced aims to facilitate the learning of polynomial reductions and strengthen understanding of the boundary between tractability and computational intractability.

Keywords—Max-2SAT, NP-Completeness, Polynomial Reduction, Clique, Complexity Theory, Boolean Satisfiability

I. INTRODUÇÃO

A Teoria da Computação estabelece os fundamentos matemáticos para compreender os limites da computação, classificando problemas segundo sua complexidade computacional. Entre as classes de complexidade, a classe

\mathcal{NP} (Nondeterministic Polynomial Time) e, em especial, os problemas \mathcal{NP} -completos ocupam papel central, tanto do ponto de vista teórico quanto prático. Um problema é \mathcal{NP} -completo se pertence à classe \mathcal{NP} e todo problema em \mathcal{NP} pode ser reduzido a ele em tempo polinomial, caracterizando-o como um dos mais representativos quanto à dificuldade computacional.

O conceito de \mathcal{NP} -completude foi introduzido por Stephen Cook em 1971 [1], por meio do Teorema de Cook-Levin, que estabeleceu o problema SAT (Satisfiability)

Dados de contato: Raphael Sales de Souza, sales.raphael@mail.uft.edu.br

como o primeiro problema \mathcal{NP} -completo. Desde então, milhares de problemas foram classificados como \mathcal{NP} -completos através de reduções polinomiais, compondo uma ampla rede de equivalências que fundamenta a noção moderna de intratabilidade algorítmica.

Dentre os problemas derivados de SAT, o *Maximum 2-Satisfiability* (Max-2SAT) ocupa posição de destaque. Trata-se de uma variante de maximização em que cada cláusula contém no máximo dois literais, e o objetivo é determinar uma valoração que satisfaça o maior número possível de cláusulas. Na versão de decisão, dada uma fórmula em forma normal conjuntiva e um inteiro k , pergunta-se se existe uma atribuição que satisfaça pelo menos k cláusulas. Embora o problema 2-SAT seja resolvido em tempo polinomial [2], sua versão de maximização (Max-2SAT) é \mathcal{NP} -completa, conforme demonstrado por Garey, Johnson e Stockmeyer [3].

O Max-2SAT possui aplicações práticas relevantes, como otimização de circuitos eletrônicos, análise de dependências em sistemas de software, depuração de hardware, modelagem de redes biológicas e problemas de agendamento com restrições binárias. Além disso, algoritmos de aproximação e heurísticas para Max-2SAT são estudados de forma ampla na literatura, reforçando sua importância tanto teórica quanto aplicada.

O objetivo deste artigo é apresentar, de maneira didática, a demonstração da \mathcal{NP} -completude do Max-2SAT utilizando a redução polinomial *Clique* \leq_p Max-2SAT. Essa redução não foi explorada em sala de aula e permite discutir a relação entre problemas de grafos e problemas de lógica proposicional.

É importante ressaltar que a principal contribuição deste manuscrito é de natureza *pedagógica*, consistindo na sistematização detalhada e acessível da demonstração da \mathcal{NP} -completude do Max-2SAT. Não são propostos novos resultados teóricos ou algorítmicos; a redução *Clique* \leq_p Max-2SAT aqui apresentada é conhecida na literatura [4]. O valor do trabalho reside na exposição didática estruturada, com exemplos comentados, figuras ilustrativas e discussões sobre armadilhas conceituais, voltada para estudantes e docentes de disciplinas de Teoria da Computação.

As principais contribuições deste trabalho incluem a demonstração formal da \mathcal{NP} -pertinência e da \mathcal{NP} -dificuldade do Max-2SAT, a construção explícita e detalhada da redução polinomial *Clique* \leq_p Max-2SAT, figuras ilustrativas destacando como cada parte do grafo é traduzida para cláusulas 2-SAT, um exemplo completo e comentado exibindo todas as etapas da transformação, o pseudocódigo do verificador polinomial para a versão de decisão do Max-2SAT, e discussões pedagógicas sobre armadilhas comuns e estratégias para compreender reduções entre problemas de grafos e fórmulas booleanas.

O restante deste artigo está organizado da seguinte forma: a Seção 2 – *Preliminares* – apresenta as preliminares necessárias, incluindo definições formais de classes de complexidade, reduções polinomiais e os problemas *Clique* e Max-2SAT; a Seção 3 – *Trabalhos Relacionados* – revisa trabalhos relacionados; a Seção 4 – *Descrição do Problema* – descreve em detalhes o problema Max-2SAT; a Seção 5 – *Demonstração e Contribuições* – apresenta a prova de \mathcal{NP} -completude por meio da redução *Clique* \leq_p Max-2SAT;

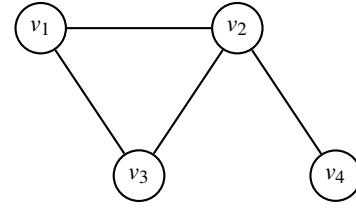


Figura 1: Grafo G1

a Seção 6 – *Resultados e Reflexões* – discute resultados e reflexões.

II. PRELIMINARES

São definidos a seguir os conceitos fundamentais que embasam o restante deste trabalho, abrangendo classes de complexidade, reduções polinomiais, fórmulas em FNC e as especificações formais dos problemas *Clique* e *Max-2SAT*, que constituem, respectivamente, o problema Atacado e o problema Alvo da redução apresentada na Seção 5.

A seguir são apresentados os conceitos fundamentais de teoria dos grafos, que serão essenciais para compreender a redução *Clique* \leq_p Max-2SAT. Um grafo é uma estrutura matemática que modela relações entre objetos. Formalmente, um grafo G é definido por um par ordenado $G = (V(G), E(G))$, onde $V(G)$ representa um conjunto finito e não vazio de vértices (também chamados de nós) e $E(G)$ representa um conjunto de arestas, sendo cada aresta um par de vértices (u, v) com $u, v \in V(G)$ e $u \neq v$. Quando existe uma aresta $(u, v) \in E(G)$, dizemos que os vértices u e v são adjacentes ou vizinhos.

O grau de um vértice $v \in V(G)$, denotado por $\deg(v)$, corresponde ao número de arestas incidentes a ele, ou equivalentemente, ao número de vizinhos que v possui no grafo. Um subgrafo de G é um grafo $G' = (V(G'), E(G'))$ tal que $V(G') \subseteq V(G)$ e $E(G') \subseteq E(G)$, onde todas as arestas de $E(G')$ conectam apenas vértices pertencentes a $V(G')$.

Para ilustrar esses conceitos, considera-se o grafo G1 da Figura 1 com quatro vértices $V(G) = \{v_1, v_2, v_3, v_4\}$ e quatro arestas $E(G) = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4)\}$. Neste grafo, o vértice v_2 possui grau 3, pois está conectado a três outros vértices (v_1 , v_3 e v_4); os vértices v_1 e v_3 possuem grau 2, cada um conectado a dois vizinhos; e o vértice v_4 possui grau 1, estando conectado apenas a v_2 .

Uma clique é um subconjunto de vértices $C \subseteq V(G)$ tal que todo par de vértices distintos em C é adjacente. Formalmente, para quaisquer $u, v \in C$ com $u \neq v$, temos $(u, v) \in E(G)$. O tamanho de uma clique é o número de vértices que ela contém.

A Figura 2 ilustra o conceito de clique de forma detalhada. Nessa figura, o grafo possui quatro vértices $\{v_1, v_2, v_3, v_4\}$ e as arestas $\{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4)\}$. O retângulo tracejado destaca o subconjunto $\{v_1, v_2, v_3\}$, que forma uma clique de tamanho 3. Para verificar que esse conjunto é de fato uma clique, observa-se que existem arestas conectando todos os pares possíveis dentro dele: a aresta (v_1, v_2) conecta v_1 a v_2 , a aresta (v_1, v_3) conecta v_1 a v_3 , e a aresta (v_2, v_3) conecta v_2 a v_3 . Como cada par de vértices do conjunto está conectado por uma aresta, a condição de clique é satisfeita.

O vértice v_4 , representado em cinza mais escuro na figura, não faz parte dessa clique. Embora v_4 esteja conectado a

Clique de tamanho 3

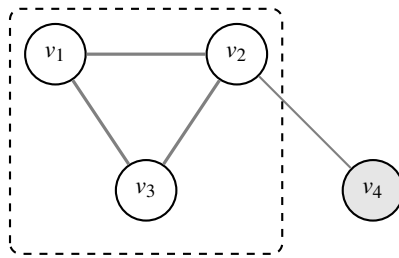


Figura 2: Grafo com clique de tamanho 3 formada por $\{v_1; v_2; v_3\}$.

v_2 pela aresta (v_2, v_4) , ele não possui arestas com v_1 nem com v_3 . Portanto, se fosse incluído v_4 no conjunto, os pares (v_1, v_4) e (v_3, v_4) não seriam adjacentes, violando a definição de clique. Esse exemplo ilustra por que uma clique exige conectividade total entre todos os seus membros, e não apenas conexões parciais.

A seguir são apresentados os conceitos de complexidade computacional, um problema de decisão é um problema cuja resposta é “sim” ou “não”.

Uma Máquina de Turing é um modelo matemático de computação que consiste em uma fita infinita dividida em células, um cabeçote de leitura/escrita que pode mover-se sobre a fita, um conjunto finito de estados, e uma função de transição que determina o comportamento da máquina. A cada passo, a máquina lê o símbolo da célula atual, escreve um novo símbolo (ou mantém o anterior), move o cabeçote para a esquerda ou direita, e muda de estado. Uma Máquina de Turing é *determinística* quando, para cada combinação de estado e símbolo lido, existe no máximo uma ação possível definida pela função de transição. Uma Máquina de Turing é *não determinística* quando podem existir múltiplas ações possíveis para uma mesma configuração, permitindo que a máquina “escolha” entre diferentes caminhos de computação. Esse modelo, proposto por Alan Turing em 1936, captura formalmente a noção intuitiva de algoritmo e constitui a base teórica para a definição de classes de complexidade computacional [5].

Um certificado para um problema de decisão é uma estrutura de dados que, quando fornecida junto com uma instância do problema, permite verificar em tempo polinomial se a resposta para aquela instância é “sim”. Formalmente, um problema L pertence à classe \mathcal{NP} se existe um verificador polinomial V e uma constante c tal que, para toda instância x : $x \in L \iff \exists$ certificado y com $|y| \leq |x|^c$ tal que $V(x, y) = \text{“aceita”}$. O certificado funciona como uma “prova” de que a instância tem resposta positiva. Por exemplo, para o problema Clique, um certificado seria um subconjunto específico de vértices; para um problema de satisfatibilidade booleana, seria uma valoração das variáveis. A existência de certificados verificáveis em tempo polinomial caracteriza a classe \mathcal{NP} e distingue-a de outras classes de complexidade.

A classe de complexidade \mathcal{P} consiste no conjunto de problemas decidíveis por uma Máquina de Turing determinística em tempo polinomial [5]. Formalmente,

$$\mathcal{P} = \{L \mid L \text{ é decidível por uma MT determinística em tempo polinomial}\}.$$

Por sua vez, a classe \mathcal{NP} é definida de forma análoga, substituindo a Máquina de Turing determinística por uma não determinística:

$$\mathcal{NP} = \{L \mid L \text{ é decidível por uma MT não determinística em tempo polinomial}\}.$$

De modo equivalente, \mathcal{NP} reúne os problemas cujas soluções podem ser verificadas em tempo polinomial mediante um certificado apropriado.

Um problema L é \mathcal{NP} -difícil quando todo problema em \mathcal{NP} se reduz a L em tempo polinomial, e é \mathcal{NP} -completo quando, além disso, pertence à própria classe \mathcal{NP} .

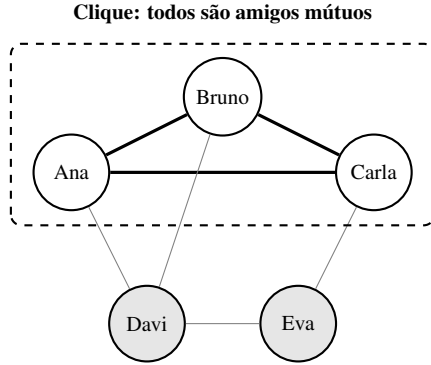
Reduções polinomiais são o principal mecanismo para comparar a dificuldade de problemas. Diz-se que um problema de decisão A reduz-se polinomialmente a outro problema de decisão B , denotado pela notação $A \leq_p B$, quando existe uma função computável em tempo polinomial f tal que, para toda instância x , temos $x \in A \iff f(x) \in B$. O símbolo \leq_p denota a relação de redução em tempo polinomial, indicando que o problema A não é mais difícil que o problema B do ponto de vista computacional. Nessa notação, A é denominado *problema atacado*, pois é o problema cuja complexidade já conhecemos, e B é denominado *problema alvo*, para o qual desejamos provar a complexidade. O uso dessas reduções permite demonstrar \mathcal{NP} -dificuldade e, quando combinado com a pertinência a \mathcal{NP} , também \mathcal{NP} -completude.

No contexto de reduções polinomiais, um *gadget* é uma construção auxiliar padronizada que traduz elementos estruturais do problema Alvo para o problema Atacado, preservando as propriedades essenciais da instância original. A técnica de construção por gadgets permite modularizar a redução, facilitando tanto a verificação de corretude quanto a análise de complexidade. Por exemplo, na redução Clique \leq_p Max-2SAT apresentada neste trabalho, as cláusulas de seleção $(x_i \vee z)$ e $(x_i \vee \neg z)$ funcionam como gadgets que distinguem vértices selecionados de vértices não selecionados, enquanto as cláusulas de incompatibilidade $(\neg x_i \vee \neg x_j)$ atuam como gadgets que impedem a seleção simultânea de vértices não adjacentes.

No contexto de fórmulas booleanas, são definidos formalmente os conceitos fundamentais na ordem lógica de construção. Uma variável booleana é um símbolo que pode assumir um dentre dois valores possíveis, pertencentes ao conjunto $\{0, 1\}$, onde 0 representa falso e 1 representa verdadeiro. Formalmente, dada uma variável x , uma valoração σ associa a x um valor em $\{0, 1\}$, denotado por $\sigma(x)$. A partir de variáveis, construímos literais mediante a seguinte definição: um literal é uma variável x ou sua negação $\neg x$. Se $\sigma(x) = 1$, então o literal x é verdadeiro e o literal $\neg x$ é falso sob σ ; de modo análogo, se $\sigma(x) = 0$, então o literal x é falso e o literal $\neg x$ é verdadeiro sob σ . Combinando literais mediante a operação lógica de disjunção, formamos cláusulas conforme a definição a seguir: uma cláusula é uma disjunção de literais, podendo ser representada formalmente como $C = (l_1 \vee l_2 \vee \dots \vee l_r)$, onde cada l_i é um literal. Uma valoração σ satisfaz a cláusula C quando ao menos um de seus literais é verdadeiro sob σ , isto é, quando existe $i \in \{1, 2, \dots, r\}$ tal que l_i é verdadeiro segundo σ . Finalmente, uma fórmula está em Forma Normal

TABELA 1: DEFINIÇÃO FORMAL DO PROBLEMA CLIQUE.

Clique
Entrada: Um grafo $G = (V(G), E(G))$ e um inteiro positivo k .
Questão: Existe um subconjunto $S \subseteq V(G)$ com $ S \geq k$ tal que todo par de vértices de S é adjacente, isto é, para quaisquer $v_i, v_j \in S$ com $i \neq j$, tem-se $(v_i, v_j) \in E(G)$?

**Figura 3:** Rede social com clique {Ana, Bruno, Carla}.

Conjuntiva (FNC) quando é expressa como uma conjunção de cláusulas, isto é, $F = C_1 \wedge C_2 \wedge \dots \wedge C_m$, onde cada C_i é uma cláusula. Uma valoração σ satisfaz a fórmula F quando satisfaz simultaneamente todas as cláusulas C_i que compõem F .

A seguir são apresentados os dois problemas centrais desta redução: *Clique*, que atua como *problema atacado* (\mathcal{NP} -completo), e *Max-2SAT*, o *problema alvo* cuja \mathcal{NP} -dificuldade será estabelecida.

O problema Clique foi demonstrado como \mathcal{NP} -completo por Karp em 1972 [6]. Para tornar mais clara a estrutura do problema e facilitar a compreensão da redução apresentada posteriormente, é apresentado um exemplo cotidiano que ilustra o conceito de clique, conforme representado na Figura 3. Em uma rede social, cada pessoa é representada por um vértice do grafo, e uma aresta conecta duas pessoas que são amigas entre si. O problema Clique corresponde a encontrar um grupo de pelo menos k pessoas onde todos são mutuamente amigos, isto é, cada par de pessoas dentro desse grupo possui uma relação de amizade direta. Por exemplo, imagine que você deseja organizar um jantar e precisa convidar pelo menos três pessoas, mas com a restrição de que todos os convidados já se conheçam mutuamente para garantir um ambiente confortável e integrado. Determinar se tal grupo existe na sua rede de amigos é exatamente uma instância do problema Clique.

A Figura 3 ilustra esse cenário: Ana, Bruno e Carla formam uma clique de tamanho 3, pois cada par dentro desse grupo possui amizade direta (representada pelas arestas que conectam todos os três entre si). Davi e Eva, embora conectados a alguns membros do grupo, não fazem parte dessa clique pois não são amigos de todos os outros membros simultaneamente — por exemplo, Davi não possui aresta com Carla, e Eva não possui aresta com Ana nem com Bruno.

É importante observar a diferença fundamental entre o

TABELA 2: DEFINIÇÃO FORMAL DO PROBLEMA MAX-2SAT.

Max-2SAT
Entrada: Uma fórmula booleana F em forma normal conjuntiva (FNC), na qual cada cláusula contém no máximo dois literais, e um inteiro positivo k .
Questão: Existe uma valoração booleana das variáveis de F que satisfaça pelo menos k cláusulas?

problema 2-SAT e sua versão de maximização. O problema 2-SAT, que pergunta se existe uma valoração que satisfaz todas as cláusulas de uma fórmula onde cada cláusula tem no máximo dois literais, pode ser resolvido em tempo linear $O(n + m)$, onde n é o número de variáveis e m é o número de cláusulas, através de algoritmos baseados em grafos de implicação e componentes fortemente conexas [2]. Em contraste, a versão de maximização Max-2SAT foi demonstrada como \mathcal{NP} -completa por Garey, Johnson e Stockmeyer [3], que provou sua intratabilidade através de uma redução polinomial a partir do problema Vertex Cover. Essa diferença ilustra como alterações pequenas na especificação de um problema podem resultar em mudanças drásticas em sua complexidade computacional.

Demonstrar que Max-2SAT é \mathcal{NP} -completo requer exibir dois componentes: primeiro, um certificado verificável em tempo polinomial que comprove a pertinência à classe \mathcal{NP} ; segundo, uma redução polinomial a partir de um problema já conhecido como \mathcal{NP} -completo. Neste trabalho, o problema Clique é utilizado como ponto de partida para a redução, estabelecendo uma correspondência entre estruturas altamente conectadas em grafos e fórmulas booleanas com alto grau de satisfatibilidade. Essa redução será desenvolvida em detalhes na Seção 5.

III. TRABALHOS RELACIONADOS

As referências clássicas sobre teoria da complexidade computacional e \mathcal{NP} -completude, como Cook [1], Karp [6], Garey e Johnson [4], Papadimitriou [7] e Sipser [5], constituem obras fundamentais para a compreensão geral de reduções polinomiais, classes de complexidade e taxonomia de problemas intratáveis, fornecendo o alicerce conceitual sobre o qual se desenvolvem estudos mais específicos.

No contexto específico de Max-2SAT e problemas de satisfatibilidade booleana, Goemans e Williamson [8] apresentam um algoritmo de aproximação baseado em programação semidefinida que alcança fator de aproximação de 0.878 para o problema Max-2SAT. O trabalho estabelece um marco importante ao conectar técnicas de otimização contínua com problemas combinatórios discretos, demonstrando que relaxações semidefinidas podem fornecer soluções de alta qualidade mesmo quando a solução ótima é computacionalmente intratável. Os autores utilizam arredondamento aleatório de variáveis baseado em vetores unitários, técnica que se tornou fundamental para o desenvolvimento de algoritmos de aproximação em problemas de satisfatibilidade.

Trevisan et al. [9] investigam a construção sistemática

de *gadgets* para reduções entre variantes de problemas de satisfação de restrições booleanas. O trabalho caracteriza quais propriedades estruturais devem ser preservadas ao transformar instâncias de um problema em outro, estabelecendo condições necessárias e suficientes para que uma redução mantenha a equivalência entre soluções ótimas. Os autores demonstram como gadgets bem projetados permitem controlar precisamente o número de cláusulas satisfeitas na fórmula resultante, técnica essencial para reduções que envolvem problemas de maximização como o Max-2SAT. Essa abordagem sistemática influenciou significativamente o desenvolvimento de novas reduções e a compreensão de limites de aproximabilidade.

Khanna et al. [10] estabelecem uma taxonomia completa de aproximabilidade para problemas de satisfação de restrições booleanas, incluindo Max-2SAT. O trabalho caracteriza formalmente quais variantes desses problemas admitem esquemas de aproximação em tempo polinomial (PTAS) e quais são \mathcal{APX} -completos, isto é, não admitem aproximação arbitrariamente boa sob a hipótese $\mathcal{P} \neq \mathcal{NP}$. Os autores demonstram que Max-2SAT pertence à classe \mathcal{APX} -completa, indicando que, embora existam algoritmos de aproximação com garantias constantes, não é possível obter esquemas que aproximem a solução ótima com erro arbitrariamente pequeno em tempo polinomial. Essa caracterização delimita precisamente as fronteiras entre o que é computacionalmente viável e o que permanece intratável mesmo sob relaxações de otimalidade.

No contexto pedagógico e didático, Lassance, Bianchini e Santos [11] apresentam um estudo fundamentado na experiência da disciplina de Teoria da Computação da Universidade Federal do Tocantins, evidenciando a importância de metodologias ativas baseadas em seminários acadêmicos para a aprendizagem de conceitos abstratos como decidibilidade, complexidade e \mathcal{NP} -completude. Os autores argumentam que a exposição pública, a análise crítica de demonstrações formais e a elaboração de apresentações estruturadas contribuem significativamente para o desenvolvimento de autonomia intelectual e domínio técnico por parte dos estudantes. A discussão mostra como abordagens dialogadas favorecem a consolidação de técnicas de redução polinomial e de formalização rigorosa, aspectos essenciais tanto para a compreensão de problemas intratáveis quanto para a construção de demonstrações corretas. Esse trabalho relaciona-se diretamente com a proposta pedagógica do presente artigo, que busca apresentar a demonstração de \mathcal{NP} -completude do Max-2SAT de forma didática e acessível a estudantes de graduação.

IV. DESCRIÇÃO DO PROBLEMA

O problema *Maximum 2-Satisfiability* (Max-2SAT) é uma variante de maximização do problema clássico SAT, na qual cada cláusula contém no máximo dois literais. O objetivo é determinar uma valoração booleana que satisfaça o maior número possível de cláusulas. Na versão de decisão, investigada neste trabalho, pergunta-se se existe uma valoração capaz de satisfazer pelo menos k cláusulas de uma fórmula em forma normal conjuntiva (FNC).

Para tornar o problema Max-2SAT mais acessível, considere o seguinte cenário: um organizador de eventos

precisa alocar n palestras em dois horários disponíveis, manhã e tarde. Cada palestra deve ocorrer exatamente em um dos dois períodos. Diversos pares de palestrantes expressam preferências conjuntas sobre os horários, representadas por restrições do tipo "pelo menos um de nós deve estar na manhã" ou "pelo menos um de nós deve estar na tarde".

Formalmente, cada palestra pode ser modelada i por uma variável booleana x_i , onde $x_i = 1$ significa que a palestra i está alocada no período da manhã, e $x_i = 0$ indica alocação no período da tarde. Uma preferência expressa por dois palestrantes i e j pode ser representada por uma cláusula booleana como $(x_i \vee x_j)$, que é satisfeita quando pelo menos uma das duas palestras ocorre na manhã, ou $(\neg x_i \vee \neg x_j)$, indicando que se a palestra i for na manhã, então j também deve ser na manhã.

Em muitas situações práticas, as preferências dos palestrantes entram em conflito, tornando impossível satisfazer todas simultaneamente. Por exemplo, se três palestrantes A , B e C expressam as preferências "A ou B na manhã", "B ou C na tarde" e "A ou C em horários opostos", pode ser impossível atender todas ao mesmo tempo. Nesse contexto, o objetivo torna-se maximizar o número total de preferências atendidas, escolhendo uma alocação que satisfaça o maior número possível de restrições.

Esse cenário captura a essência do Max-2SAT: lidar com um sistema de restrições booleanas parcialmente conflitantes e buscar uma solução que maximize a consistência global, mesmo quando a satisfação total é inviável.

Formalmente, uma instância do Max-2SAT consiste em uma fórmula booleana F em forma normal conjuntiva (FNC), composta por cláusulas C_1, C_2, \dots, C_m , cada uma contendo um ou dois literais, e por um inteiro positivo k . O problema consiste em determinar se existe uma valoração σ que satisfaça pelo menos k cláusulas de F , conforme definido na Tabela 2.

Embora o problema 2-SAT seja solucionável em tempo linear, sua versão de maximização (Max-2SAT) apresenta complexidade substancialmente maior. Essa diferença ilustra como pequenas alterações na formulação podem transformar um problema tratável em intratável. O Max-2SAT possui aplicações em gerenciamento de dependências de software, depuração de hardware, análise de redes biológicas e problemas de agendamento com restrições binárias.

Para ilustrar o comportamento do problema, considere a fórmula:

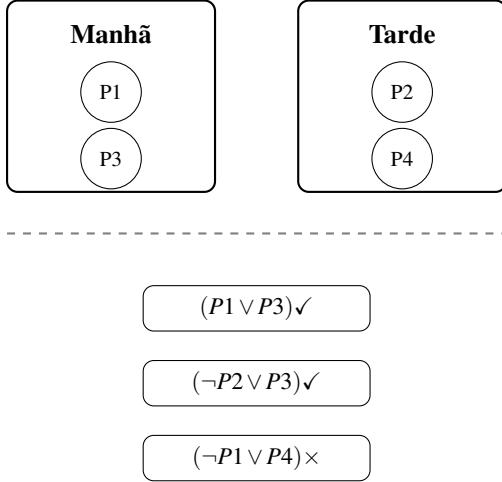
$$F = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee \neg x_3).$$

Nenhuma valoração satisfaz simultaneamente as três cláusulas. Isso ocorre porque as duas últimas impõem condições opostas sobre a variável x_3 : a cláusula $(\neg x_1 \vee x_3)$ força $x_3 = 1$ sempre que $x_1 = 1$, enquanto a cláusula $(\neg x_2 \vee \neg x_3)$ força $x_3 = 0$ sempre que $x_2 = 1$. Como a primeira cláusula $(x_1 \vee x_2)$ exige que pelo menos uma das duas variáveis seja verdadeira, inevitavelmente surge uma contradição. Se $x_1 = 1$, então x_3 deve ser 1, mas isso tende a violar a terceira cláusula. Se $x_2 = 1$, então x_3 deve ser 0, mas isso tende a violar a segunda cláusula. Assim, qualquer tentativa de satisfazer todas as três cláusulas força a violação de pelo menos uma delas.

Apesar disso, é possível satisfazer duas cláusulas. Por

TABELA 3: AVALIAÇÃO EXAUSTIVA DAS VALORAÇÕES PARA A FÓRMULA F .

x_1	x_2	C_1	C_2	C_3	C_4	Total
0	0	0	1	1	1	3
0	1	1	1	0	1	3
1	0	1	1	1	0	3
1	1	1	0	1	1	3

Alocação de Palestras**Figura 4:** 2 de 3 preferências atendidas.

exemplo, a valoração $x_1 = 1$, $x_2 = 1$ e $x_3 = 0$ satisfaz a primeira e a terceira cláusulas, mas viola a segunda.

Esse comportamento evidencia o caráter de otimização do Max-2SAT: quando a estrutura das restrições contém conflitos inevitáveis, o objetivo deixa de ser satisfazer todas as cláusulas e passa a ser maximizar o número de cláusulas satisfeitas.

Para ilustrar de forma pedagógica situações em que não é possível satisfazer todas as cláusulas simultaneamente, considera-se a seguinte fórmula:

$$F = (x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2).$$

Esta fórmula envolve duas variáveis booleanas, x_1 e x_2 , de modo que existem exatamente quatro valorações possíveis. A Tabela 3 apresenta a avaliação sistemática de cada valoração, demonstrando que nenhuma satisfaz simultaneamente as quatro cláusulas.

Como evidenciado na tabela, cada valoração satisfaz exatamente três das quatro cláusulas, caracterizando um caso típico em que a formulação assume natureza de problema de maximização. Este exemplo evidencia a essência do Max-2SAT: quando a estrutura das restrições contém conflitos inevitáveis, o objetivo deixa de ser satisfazer todas as cláusulas e passa a ser maximizar o número de cláusulas satisfeitas.

Retornando à interpretação lúdica apresentada anteriormente, podemos visualizar o problema através do cenário de alocação de palestras. A Figura 4 ilustra de forma resumida a estrutura conceitual desse cenário, destacando os elementos centrais da formalização que será empregada na redução apresentada posteriormente.

Essa analogia capta a essência do Max-2SAT: resolver um

sistema de restrições parcialmente conflitantes e maximizar sua consistência.

Do ponto de vista didático, o Max-2SAT é especialmente valioso por evidenciar de forma clara a diferença entre problemas de satisfação total e problemas de maximização. A análise desse tipo de fórmula permite ao estudante perceber como a impossibilidade de satisfazer todas as cláusulas conduz naturalmente a questões de otimização. O problema também mostra que a estrutura das cláusulas exerce influência direta sobre a complexidade computacional, deixando evidente que restrições simples como limitar cada cláusula a dois literais não garantem a existência de algoritmos polinomiais. Além disso, o Max-2SAT estabelece conexões importantes entre problemas booleanos e problemas em grafos, permitindo interpretar propriedades combinatórias por meio de fórmulas proposicionais. A construção de reduções por *gadgets*, como a utilizada na transformação Clique \leq_p Max-2SAT apresentada na Seção 5, reforça técnicas fundamentais para demonstrações de \mathcal{NP} -completude. Uma compreensão precisa desse comportamento é essencial para acompanhar com rigor a prova apresentada.

V. DEMONSTRAÇÃO E CONTRIBUIÇÕES

Nesta seção é estabelecida a \mathcal{NP} -completude do problema Max-2SAT. A redução utilizada parte do problema Clique e emprega uma construção baseada em gadgets de seleção e incompatibilidade [4], detalhada passo a passo com a variável auxiliar z , as cláusulas de seleção e as cláusulas de exclusão para não-arestas.

Para provar que Max-2SAT é \mathcal{NP} -completo, é necessário demonstrar duas propriedades:

- (i) Max-2SAT $\in \mathcal{NP}$;
- (ii) existe um problema π sabidamente \mathcal{NP} -completo tal que $\pi \leq_p$ Max-2SAT.

Tome (I): Max-2SAT $\in \mathcal{NP}$. Para mostrar que Max-2SAT pertence à classe \mathcal{NP} , é suficiente exibir um certificado de tamanho polinomial e um algoritmo verificador que, dado esse certificado, decide em tempo polinomial se ele constitui uma solução válida para a instância. No caso do Max-2SAT, o certificado é uma valoração booleana $\sigma: \{x_1, x_2, \dots, x_n\} \rightarrow \{0, 1\}$ que atribui valores verdadeiro ou falso a todas as variáveis da fórmula F . Dado esse certificado, o Algoritmo 1 percorre cada cláusula da fórmula, avalia se ela é satisfeita pela valoração fornecida e conta o número total de cláusulas satisfeitas, verificando se esse total atinge pelo menos o limiar k especificado na instância.

Para verificar que Max-2SAT pertence à classe \mathcal{NP} , analisamos a complexidade do algoritmo verificador. O algoritmo percorre cada uma das m cláusulas uma única vez. Como cada cláusula contém no máximo dois literais, a avaliação de C_j sob σ é feita em tempo $O(1)$. Portanto, o tempo total de execução é $O(m)$, que é polinomial no tamanho da entrada. Logo, Max-2SAT $\in \mathcal{NP}$.

Tome (II): Max-2SAT é \mathcal{NP} -difícil via Clique \leq_p Max-2SAT. Para demonstrar que Max-2SAT é \mathcal{NP} -difícil, seleciona-se o problema Clique, definido formalmente na Tabela 1 e conhecido por ser \mathcal{NP} -completo desde o trabalho

Algorithm 1 Verificador Polinomial para Max-2SAT

Entrada: Fórmula $F = C_1 \wedge C_2 \wedge \dots \wedge C_m$ em 2-FNC, inteiro k , certificado σ

Saída: ACEITA se σ satisfaz pelo menos k cláusulas;
REJEITA caso contrário

```

1: contador  $\leftarrow 0$ 
2: for cada cláusula  $C_j \in F$  do
3:   Avalie  $C_j$  sob a valoração  $\sigma$ 
4:   if  $C_j$  é satisfeita por  $\sigma$  then
5:     contador  $\leftarrow$  contador + 1
6:   end if
7: end for
8: if contador  $\geq k$  then
9:   return ACEITA
10: else
11:   return REJEITA
12: end if
```

seminal de Karp [6], e constrói-se uma redução polinomial Clique \leq_p Max-2SAT. Seguindo a terminologia adotada neste trabalho, Clique atua como *problema atacado* (o problema de partida, cuja \mathcal{NP} -completude já é conhecida) e Max-2SAT é o *problema alvo*, para o qual desejamos transferir a dificuldade computacional.

A escolha do problema Clique como ponto de partida para a redução é estratégica por diversas razões. Primeiro, a estrutura de Clique envolve a seleção de um subconjunto de vértices com propriedades específicas (adjacência mútua), o que mapeia naturalmente para variáveis booleanas indicando inclusão ou exclusão de elementos. Segundo, a condição de que todos os pares devem ser adjacentes traduz-se diretamente em cláusulas de incompatibilidade para pares não adjacentes. Terceiro, o parâmetro k (tamanho da clique) pode ser codificado no número de cláusulas satisfeitas, permitindo equivalência precisa entre os problemas. Por fim, a redução Clique \leq_p Max-2SAT ilustra de forma didática a conexão entre problemas de grafos e problemas de lógica proposicional, tema central deste trabalho.

A estratégia geral consiste em criar uma variável booleana x_i para cada vértice, uma variável auxiliar z , e construir cláusulas que incentivam a seleção de vértices enquanto punem escolhas de pares não adjacentes. O parâmetro K' é ajustado para que a satisfação de exatamente K' cláusulas corresponda a uma clique de tamanho k .

A construção formal da fórmula procede da seguinte maneira. Dada uma instância (G, k) do problema Clique, onde $G = (V(G), E(G))$ é um grafo com conjunto de vértices $V(G)$ e conjunto de arestas $E(G)$, e k é um inteiro positivo representando o tamanho mínimo da clique procurada, a função de redução f produz uma instância (F', K') do problema Max-2SAT, transformando o grafo G em uma fórmula booleana F' em forma normal conjuntiva (onde cada cláusula contém no máximo dois literais) e o parâmetro k em um novo parâmetro K' que representa o número mínimo de cláusulas a serem satisfeitas. A seguir, descreve-se passo a passo como F' e K' são construídos a partir dos elementos de G e do valor k .

A construção de f pode ser realizada em tempo polinomial. A criação das variáveis booleanas requer

$O(|V(G)|)$ operações para os n vértices, além de $O(1)$ para a variável auxiliar z . As cláusulas de seleção totalizam $2 \cdot |V(G)|$ cláusulas, cada uma construída em tempo constante, resultando em $O(|V(G)|)$. As cláusulas de incompatibilidade correspondem a uma cláusula para cada par de vértices não-adjacentes, o que no pior caso representa $|\bar{E}| = O(|V(G)|^2)$ cláusulas. Por fim, o cálculo de K' envolve apenas $O(1)$ operações aritméticas. Portanto, a complexidade total da redução é $O(|V(G)|^2)$, que é polinomial no tamanho da entrada.

A construção da fórmula F' envolve a criação de variáveis booleanas e três tipos de cláusulas que trabalham em conjunto para codificar a estrutura do grafo. Inicialmente, são definidas as variáveis que representarão os vértices do grafo. Para cada vértice $v_i \in V$, cria-se uma variável booleana x_i que indica se o vértice v_i faz parte da clique candidata. Além dessas variáveis, é introduzida uma variável auxiliar adicional z , cujo papel será explicado no contexto das cláusulas de seleção.

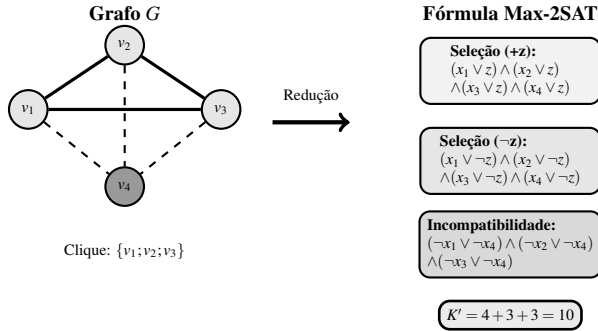
O primeiro tipo de cláusula são as cláusulas de seleção, que incentivam a escolha de vértices e permitem controlar o tamanho da clique. Para cada vértice v_i do grafo original, são inseridas duas cláusulas na fórmula: $(x_i \vee z)$ e $(x_i \vee \neg z)$. Essas cláusulas funcionam juntas para distinguir vértices selecionados de vértices não selecionados. Quando $x_i = 1$, indicando que o vértice v_i foi escolhido para compor a clique, ambas as cláusulas são satisfeitas, independentemente do valor atribuído à variável auxiliar z . Já quando $x_i = 0$, apenas uma das duas cláusulas pode ser satisfeita, dependendo do valor de z : se $z = 1$, a cláusula $(x_i \vee z)$ é satisfeita e $(x_i \vee \neg z)$ é violada; se $z = 0$, ocorre o inverso. Essa diferença de uma cláusula satisfeita entre vértices escolhidos e não escolhidos permite controlar o tamanho da clique por meio do parâmetro K' , garantindo que apenas seleções com exatamente k vértices produzam o número exigido de cláusulas satisfeitas.

O segundo tipo de cláusula são as cláusulas de incompatibilidade, que garantem que apenas vértices mutuamente adjacentes sejam selecionados simultaneamente. Para cada par de vértices (v_i, v_j) que não são adjacentes no grafo original, isto é, para cada par onde $(v_i, v_j) \notin E$, adiciona-se à fórmula a cláusula $(\neg x_i \vee \neg x_j)$. Essa cláusula impõe uma restrição essencial: vértices não adjacentes não podem ser selecionados ao mesmo tempo para compor a clique. Se ambos x_i e x_j recebem o valor 1, a cláusula $(\neg x_i \vee \neg x_j)$ se torna falsa, penalizando essa escolha inválida. Por outro lado, se ao menos um dos vértices não for selecionado (isto é, se uma das variáveis for 0), a cláusula é satisfeita. Dessa forma, qualquer valoração que satisfaça um número elevado de cláusulas deve corresponder a um conjunto de vértices que forma uma clique no grafo original.

Finalmente, define-se o parâmetro K' que estabelece o número mínimo de cláusulas a serem satisfeitas. O parâmetro K' , que estabelece o número mínimo de cláusulas a serem satisfeitas na instância de Max-2SAT, é definido de modo a manter equivalência exata com o problema Clique. O parâmetro é definido como $K' = |V| + k + |\bar{E}|$, onde $|V(G)|$ é o número total de vértices do grafo (e, portanto, o número de cláusulas do tipo $(x_i \vee z)$), k é o tamanho da clique procurada no problema original (refletindo o ganho adicional obtido nas cláusulas $(x_i \vee \neg z)$ quando k vértices são

TABELA 4: RESUMO DA REDUÇÃO CLIQUE \leq_p MAX-2SAT.

Componente	Cláusulas	Função (Gadget)	Contrib. p/ K'
Var. de vértice	$x_i, v_i \in V$	Indica se v_i pertence à clique	—
Var. auxiliar	z	Controla contagem de cláusulas	—
Seleção positiva	$(x_i \vee z)$	Satisfeita se $z=1$; base fixa	$ V $
Seleção negativa	$(x_i \vee \neg z)$	Satisfeita se $x_i=1$; mede seleção	k
Incompatib.	$(\neg x_i \vee \neg x_j), (v_i, v_j) \notin E$	Impede seleção de não adjacentes	$ \bar{E} $
Total:			$ V +k+ \bar{E} $

**Figura 5:** Transformação completa: grafo com clique $\{v_1, v_2, v_3\}$ e fórmula Max-2SAT resultante.

escolhidos), e $|\bar{E}|$ é o número de não-arestas, isto é, de pares de vértices não adjacentes (que correspondem às cláusulas $(\neg x_i \vee \neg x_j)$). Essa definição garante que a solução do Max-2SAT reproduza a condição do problema da clique.

Para facilitar a compreensão da construção, a Tabela 4 consolida os componentes da redução, explicitando o papel de cada tipo de cláusula, o significado dos gadgets empregados e a contribuição de cada bloco para o parâmetro K' .

A Figura 5 ilustra o processo de redução em três etapas, mostrando como cada elemento do grafo original é traduzido para componentes da fórmula Max-2SAT. Na primeira etapa, apresentamos o grafo de entrada com seus vértices e arestas. Na segunda etapa, mostramos os gadgets de seleção, que consistem nas cláusulas $(x_i \vee z)$ e $(x_i \vee \neg z)$ para cada vértice, representando o mecanismo que diferencia vértices escolhidos de não escolhidos. Na terceira etapa, exibimos os gadgets de incompatibilidade, que são as cláusulas $(\neg x_i \vee \neg x_j)$ geradas para cada par de vértices não adjacentes, impedindo a seleção simultânea de vértices que não formam aresta.

Para entender como esse valor opera, considere uma valoração que corresponde a uma clique válida de tamanho k . Todas as $|V(G)|$ cláusulas do tipo $(x_i \vee z)$ são satisfeitas, contribuindo $|V(G)|$ para a contagem. As k cláusulas $(x_i \vee \neg z)$ associadas aos vértices escolhidos também são satisfeitas, acrescentando k ao total. Além disso, todas as $|\bar{E}|$ cláusulas de incompatibilidade são satisfeitas, pois nenhum par de vértices não adjacentes foi selecionado ao mesmo tempo. Somando essas contribuições, obtém-se $K' = |V| + k + |\bar{E}|$. Já qualquer valoração que não represente uma clique de tamanho pelo menos k ou que viole alguma cláusula de incompatibilidade satisfará um número menor de cláusulas, o que garante a equivalência entre os dois problemas.

A Figura 5 apresenta uma visualização completa da transformação, mostrando lado a lado o grafo de entrada e a fórmula Max-2SAT resultante, com destaque para a correspondência entre cada elemento do grafo e as cláusulas geradas. Na parte esquerda da figura, observa-se o grafo G com a clique $\{v_1, v_2, v_3\}$ destacada, onde as linhas contínuas representam arestas e as linhas tracejadas representam não-arestas. Na parte direita, é exibida a fórmula completa organizada em três blocos: as cláusulas de seleção positivas $(x_i \vee z)$, as cláusulas de seleção negativas $(x_i \vee \neg z)$, e as cláusulas de incompatibilidade $(\neg x_i \vee \neg x_j)$. A organização visual evidencia como cada vértice do grafo origina suas cláusulas de seleção e como cada par de vértices não adjacentes (linhas tracejadas) gera sua respectiva cláusula de incompatibilidade.

Para ilustrar como o parâmetro K' é obtido, considere o grafo mostrado na Figura 5, com $V(G) = \{v_1; v_2; v_3; v_4\}$ e $E(G) = \{(v_1, v_2); (v_2, v_3); (v_1, v_3)\}$. Suponha que desejamos verificar se existe uma clique de tamanho $k = 3$ nesse grafo.

Primeiro, são identificados os componentes necessários para o cálculo:

O número de vértices é $|V(G)| = 4$, o que gera quatro cláusulas do tipo $(x_i \vee z)$, uma para cada vértice do grafo. Essas cláusulas sempre serão satisfeitas quando $z = 1$, independentemente de quais vértices forem escolhidos.

O tamanho da clique desejada é $k = 3$, que corresponde ao número de cláusulas adicionais do tipo $(x_i \vee \neg z)$ que esperamos satisfazer quando o número de vértices selecionados é três.

Para determinar o número de não-arestas $|\bar{E}|$, conta-se quantos pares de vértices não são adjacentes. Em um grafo com quatro vértices, existem $\binom{4}{2} = 6$ pares possíveis. Como o grafo possui $|E(G)| = 3$ arestas, o número de não-arestas é $|\bar{E}| = 6 - 3 = 3$. Os pares não adjacentes são: (v_1, v_4) , (v_2, v_4) e (v_3, v_4) . Para cada um desses pares, inclui-se uma cláusula de incompatibilidade.

Aplicando a fórmula $K' = |V(G)| + k + |\bar{E}|$, obtém-se:

$$K' = 4 + 3 + 3 = 10$$

Assim, a instância de Max-2SAT correspondente pergunta: "É possível satisfazer pelo menos 10 cláusulas da fórmula construída?" Uma resposta positiva equivale a afirmar que o grafo original possui uma clique de tamanho pelo menos 3.

Para verificar a construção, observe que o grafo contém uma clique de tamanho 3 formada pelos vértices $\{v_1, v_2, v_3\}$. Atribuindo $x_1 = x_2 = x_3 = 1, x_4 = 0$ e $z = 1$, temos:

As 4 cláusulas $(x_1 \vee z)$, $(x_2 \vee z)$, $(x_3 \vee z)$ e $(x_4 \vee z)$ são todas satisfeitas porque $z = 1$, contribuindo com 4 cláusulas.

Das 4 cláusulas do tipo $(x_i \vee \neg z)$, apenas as três correspondentes aos vértices selecionados são satisfeitas: $(x_1 \vee \neg z)$, $(x_2 \vee \neg z)$ e $(x_3 \vee \neg z)$, pois $x_1 = x_2 = x_3 = 1$. Isso contribui com 3 cláusulas adicionais.

As 3 cláusulas de incompatibilidade $(\neg x_1 \vee \neg x_4)$, $(\neg x_2 \vee \neg x_4)$ e $(\neg x_3 \vee \neg x_4)$ são todas satisfeitas porque $x_4 = 0$, contribuindo com 3 cláusulas.

O total é $4 + 3 + 3 = 10 = K'$, o que confirma que a valoração satisfaz exatamente o número exigido de cláusulas. Esse exemplo mostra que o parâmetro K' reflete de forma precisa a estrutura da clique por meio da contagem de cláusulas satisfeitas.

Estabelecida a construção, é necessário agora provar sua corretude demonstrando a equivalência entre as instâncias. Para estabelecer que a redução está correta, é preciso demonstrar que $(G, k) \in \text{Clique}$ se e somente se $(F', K') \in \text{Max-2SAT}$. Cada direção é provada separadamente.

Ida (\Rightarrow): Se G possui uma clique de tamanho k , então (F', K') é satisfatível. Suponha que exista um conjunto $S \subseteq V(G)$ de vértices formando uma clique de tamanho pelo menos k , isto é, $|S| \geq k$ e para todo par de vértices distintos $v_i, v_j \in S$, existe uma aresta $(v_i, v_j) \in E(G)$. Constrói-se uma valoração que satisfaz pelo menos K' cláusulas da fórmula F' .

Defina a valoração da seguinte forma: para cada variável x_i , atribua $x_i = 1$ se o vértice v_i pertence ao conjunto S e $x_i = 0$ caso contrário. Adicionalmente, atribua $z = 1$ à variável auxiliar. Analisa-se quantas cláusulas são satisfeitas por essa valoração.

Primeiro, são analisadas as cláusulas de seleção. Para cada vértice $v_i \in V(G)$, há duas cláusulas: $(x_i \vee z)$ e $(x_i \vee \neg z)$. Como foi definido $z = 1$, todas as cláusulas do tipo $(x_i \vee z)$ são satisfeitas, independentemente do valor de x_i , o que contribui com $|V(G)|$ cláusulas satisfeitas. Já para as cláusulas $(x_i \vee \neg z)$, temos $\neg z = 0$ nessa valoração, de modo que elas são satisfeitas apenas quando $x_i = 1$. Como foi atribuído $x_i = 1$ aos k vértices do conjunto S , k cláusulas desse tipo são satisfeitas.

Agora são consideradas as cláusulas de incompatibilidade. Para cada par não-adjacente $(v_i, v_j) \notin E(G)$, temos a cláusula $(\neg x_i \vee \neg x_j)$. Essa cláusula é falsa apenas quando ambos $x_i = 1$ e $x_j = 1$, o que aconteceria se ambos os vértices v_i e v_j pertencessem ao conjunto S . No entanto, por hipótese, S é uma clique, portanto todos os pares de vértices em S são adjacentes. Isso significa que não existe nenhum par $(v_i, v_j) \notin E(G)$ com ambos $v_i, v_j \in S$. Consequentemente, para todo par não-adjacente $(v_i, v_j) \notin E(G)$, pelo menos um dos vértices não pertence a S , garantindo que pelo menos uma das variáveis x_i ou x_j vale 0, o que torna a cláusula $(\neg x_i \vee \neg x_j)$ verdadeira. Portanto, todas as $|\bar{E}|$ cláusulas de incompatibilidade são satisfeitas.

Somando as contribuições, obtém-se $|V(G)| + k + |\bar{E}| = K'$ cláusulas satisfeitas, provando que $(F', K') \in \text{Max-2SAT}$.

Volta (\Leftarrow): Se (F', K') é satisfatível, então G possui uma clique de tamanho k . Suponha agora que exista uma valoração que satisfaz pelo menos K' cláusulas da fórmula F' . É demonstrado que pode-se extrair dessa valoração um conjunto de vértices que forma uma clique de tamanho pelo menos k no grafo G .

Sem perda de generalidade, pode-se assumir que a variável auxiliar z recebe o valor 1 nesta valoração. Caso $z = 0$ na valoração original, considera-se uma valoração alternativa onde invertemos o valor de z para 1 e mantemos os valores de todas as variáveis x_i inalterados. Como cada vértice contribui com duas cláusulas $(x_i \vee z)$ e $(x_i \vee \neg z)$, inverter z apenas troca qual dessas cláusulas é satisfeita para vértices com $x_i = 0$, sem alterar o total de cláusulas satisfeitas. Portanto, é possível trabalhar com uma valoração onde $z = 1$.

Com $z = 1$, todas as $|V(G)|$ cláusulas da forma $(x_i \vee z)$ são automaticamente satisfeitas. Isso deixa espaço para $K' - |V(G)| = k + |\bar{E}|$ cláusulas adicionais serem satisfeitas. Essas cláusulas adicionais vêm de duas fontes: as cláusulas $(x_i \vee \neg z)$ para vértices com $x_i = 1$ e as cláusulas de

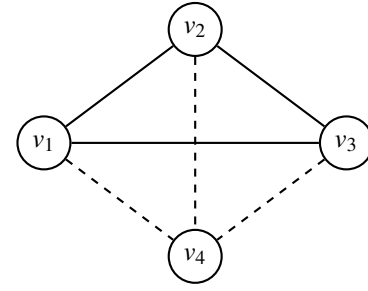


Figura 6: Grafo de entrada para a redução: clique $\{v_1, v_2, v_3\}$ e vértice isolado v_4 .

incompatibilidade $(\neg x_i \vee \neg x_j)$.

Defina $S = \{v_i \mid x_i = 1\}$ como o conjunto de vértices cujas variáveis foram atribuídas como verdadeiras. Para satisfazer pelo menos $k + |\bar{E}|$ cláusulas adicionais, é necessário ter pelo menos $|S|$ cláusulas do tipo $(x_i \vee \neg z)$ satisfeitas (uma para cada vértice em S) e todas as $|\bar{E}|$ cláusulas de incompatibilidade satisfeitas.

Se alguma cláusula de incompatibilidade $(\neg x_i \vee \neg x_j)$ não for satisfeita, isso significa que ambos $x_i = 1$ e $x_j = 1$, mas $(v_i, v_j) \notin E(G)$. Cada cláusula de incompatibilidade violada reduz o número total de cláusulas satisfeitas em uma unidade. Para manter o total em pelo menos K' , seria necessário que mais cláusulas $(x_i \vee \neg z)$ fossem satisfeitas, o que requereria mais vértices em S . No entanto, adicionar mais vértices aumenta o risco de violar mais cláusulas de incompatibilidade. De fato, pode-se verificar algebricamente que violar qualquer cláusula de incompatibilidade torna impossível atingir exatamente K' cláusulas satisfeitas com a construção apresentada.

Portanto, todas as $|\bar{E}|$ cláusulas de incompatibilidade devem ser satisfeitas, o que garante que não existe nenhum par $(v_i, v_j) \notin E(G)$ com ambos $v_i, v_j \in S$. Logo, S forma uma clique de tamanho pelo menos k no grafo G .

Para consolidar a compreensão da redução, é apresentado um exemplo completo que percorre todas as etapas da transformação, desde o grafo de entrada até a verificação da valoração resultante. Considere um grafo com quatro vértices $V(G) = \{v_1, v_2, v_3, v_4\}$ e arestas $E(G) = \{(v_1, v_2), (v_2, v_3), (v_1, v_3)\}$. A Figura 6 ilustra esse grafo, onde os vértices v_1, v_2 e v_3 formam uma clique de tamanho 3, representada pelas linhas contínuas que conectam cada par desses três vértices. O vértice v_4 não possui arestas com nenhum dos outros vértices, o que é indicado pelas linhas tracejadas que representam os pares não adjacentes (v_1, v_4) ; (v_2, v_4) ; (v_3, v_4) . Essa distinção visual entre arestas presentes e ausentes é fundamental para compreender como a redução constrói as cláusulas de incompatibilidade.

A partir da clique de tamanho 3 identificada no grafo, a construção gera as cláusulas:

$$(x_i \vee z), \quad (x_i \vee \neg z) \quad \text{para } i = 1, 2, 3, 4,$$

e, para as não-arestas:

$$(\neg x_1 \vee \neg x_4), \quad (\neg x_2 \vee \neg x_4), \quad (\neg x_3 \vee \neg x_4).$$

A valoração $x_1 = x_2 = x_3 = 1$, $x_4 = 0$ e $z = 1$ satisfaz exatamente K' cláusulas, como requerido. A Figura 7 ilustra detalhadamente essa verificação, mostrando quais cláusulas

Verificação: $x_1 = x_2 = x_3 = 1, x_4 = 0, z = 1$

Seleção ($x_i \vee z$): **4 cláusulas**

Seleção ($x_i \vee \neg z$): **3 cláusulas**

Incompatibilidade: **3 cláusulas**

Total: $4 + 3 + 3 = 10 = K' \checkmark$

Figura 7: Verificação detalhada da valoração que satisfaz $K' = 10$ cláusulas.

são satisfeitas e quais não são, organizadas por tipo. Na coluna da esquerda, observa-se que todas as cláusulas de seleção positiva ($x_i \vee z$) são satisfeitas porque $z = 1$. Na coluna central, as cláusulas de seleção negativa ($x_i \vee \neg z$) são satisfeitas apenas para os vértices selecionados (x_1, x_2, x_3), totalizando 3 cláusulas, enquanto a cláusula ($x_4 \vee \neg z$) não é satisfeita pois $x_4 = 0$ e $\neg z = 0$. Na coluna da direita, todas as cláusulas de incompatibilidade são satisfeitas porque $x_4 = 0$ torna verdadeira qualquer cláusula da forma ($\neg x_i \vee \neg x_4$).

Esse exemplo demonstra concretamente como a estrutura da clique no grafo original é preservada na fórmula Max-2SAT: os três vértices da clique correspondem aos três vértices com $x_i = 1$, que são exatamente os responsáveis por satisfazer as cláusulas extras de seleção negativa. Simultaneamente, o fato de esses três vértices serem mutuamente adjacentes garante que nenhuma cláusula de incompatibilidade é gerada entre eles, permitindo que todas as cláusulas de incompatibilidade (que envolvem apenas v_4) sejam satisfeitas.

A redução com a variável auxiliar z evidencia diversos aspectos pedagógicos relevantes. Ela mostra como cláusulas de dois literais podem impor restrições estruturais fortes sobre as possíveis valorações e como a seleção de vértices válidos depende da satisfação simultânea de múltiplas cláusulas independentes. Além disso, as cláusulas negativas traduzem de maneira direta as relações de incompatibilidade no grafo, reforçando a conexão entre propriedades combinatórias e expressões booleanas. A definição precisa do parâmetro K' demonstra como controlar o tamanho da clique desejada por meio da contagem de cláusulas satisfeitas. Por fim, essa construção evidencia por que reduções entre problemas de grafos e fórmulas booleanas constituem ferramenta fundamental no estudo de \mathcal{NP} -completude.

VI. RESULTADOS E REFLEXÕES

Apresentada a demonstração de \mathcal{NP} -completude do Max-2SAT, esta seção discute os principais resultados obtidos, bem como reflexões conceituais e pedagógicas sobre o processo de construção da redução e sobre os elementos que tornaram essa abordagem útil para o aprendizado em Teoria da Computação.

A demonstração apresentada confirmou que o problema Max-2SAT pertence à classe \mathcal{NP} , uma vez que o número de cláusulas satisfeitas por uma valoração pode ser verificado em tempo linear no tamanho da instância. Basta percorrer cada cláusula uma única vez, avaliar se ela é verdadeira sob a valoração fornecida como certificado, e contar quantas são

satisfeitas. Como cada cláusula contém no máximo dois literais, essa avaliação é realizada em tempo constante por cláusula, resultando em complexidade total $O(m)$, onde m é o número de cláusulas.

Além disso, foi demonstrado que Max-2SAT é \mathcal{NP} -difícil, pois o problema Clique, que é \mathcal{NP} -completo conforme estabelecido por Karp [6], foi reduzido a ele por meio de uma função de transformação computável em tempo polinomial. A redução constrói uma fórmula booleana cujo tamanho é polinomial no tamanho do grafo de entrada: o número de variáveis é $|V| + 1$ e o número de cláusulas é $2|V| + |\bar{E}|$, onde $|\bar{E}| \leq \binom{|V|}{2}$. A construção de cada cláusula requer tempo constante, portanto a transformação completa opera em tempo $O(|V|^2)$.

A formulação com a variável auxiliar z mostrou-se útil, pois permite controlar o número total de cláusulas satisfeitas sem recorrer a construções mais extensas. Essa técnica evita a criação de cláusulas com mais de dois literais e preserva a estrutura típica do Max-2SAT. A variável z funciona como um mecanismo de balanceamento que garante que vértices escolhidos contribuam com exatamente uma cláusula a mais do que vértices não escolhidos, traduzindo o tamanho da clique diretamente na quantidade de cláusulas satisfeitas.

O valor limite K' foi definido para refletir com precisão a estrutura combinatória do grafo original. A decomposição $K' = |V| + k + |\bar{E}|$ incorpora três componentes distintos: a base fixa de cláusulas sempre satisfeitas, o ganho proporcional ao tamanho da clique e a penalização associada à violação das cláusulas de incompatibilidade. Essa construção assegura que uma valoração que satisfaça exatamente K' cláusulas corresponda a uma clique de tamanho k , estabelecendo equivalência completa entre os dois problemas.

Esses resultados mostram que pequenas alterações estruturais em problemas que parecem simples, como a transição de 2-SAT para Max-2SAT, podem alterar de forma profunda sua complexidade. Enquanto 2-SAT admite solução em tempo linear por meio do grafo de implicações, sua versão de maximização torna-se tão difícil quanto qualquer problema em NP, evidenciando a fronteira entre tratabilidade e intratabilidade.

O desenvolvimento da redução Clique \leq_p Max-2SAT revelou diversos aspectos importantes para o ensino e compreensão de problemas \mathcal{NP} -completos. A transformação de relações de adjacência em cláusulas de dois literais torna explícito como propriedades estruturais de grafos podem ser modeladas por fórmulas booleanas. Cada aresta ou não-aresta no grafo corresponde de forma direta a uma restrição lógica, estabelecendo um dicionário claro entre os dois domínios. Essa correspondência evidencia que problemas aparentemente distintos compartilham estrutura matemática profunda, sendo manifestações diferentes de uma mesma dificuldade computacional subjacente.

A expressão ($\neg x_i \vee \neg x_j$) traduz a proibição de escolher dois vértices não adjacentes, mostrando como restrições combinatórias são mapeadas para restrições lógicas. Essa cláusula funciona como uma "barreira lógica" que impede configurações inválidas, e sua violação resulta na redução do número total de cláusulas satisfeitas. Compreender esse mecanismo de penalização é fundamental para desenvolver

intuição sobre como problemas de otimização combinatória podem ser codificados em fórmulas booleanas.

A introdução da variável auxiliar z simplifica a contagem de cláusulas, garantindo equilíbrio entre as expressões do tipo $(x_i \vee z)$ e $(x_i \vee \neg z)$ e permitindo a definição precisa do parâmetro K' . Sem essa variável, seria necessário construir gadgets mais complexos ou usar cláusulas maiores, evidenciando a fronteira. A técnica de usar variáveis auxiliares para controlar comportamentos globais da fórmula é aplicável em construções de reduções, e sua apresentação neste contexto fornece modelo útil para outros problemas.

O ajuste de K' evidencia uma técnica comum em reduções, na qual o número de cláusulas satisfeitas reflete diretamente o tamanho da estrutura procurada no problema original. Essa correspondência numérica precisa entre parâmetros do problema fonte e problema alvo é característica essencial de reduções bem construídas. Estudantes muitas vezes têm dificuldade em determinar parâmetros corretos para problemas de otimização; o exemplo apresentado demonstra metodologia sistemática baseada na análise de contribuições independentes de cada componente da construção.

Provar ambas as direções da equivalência, isto é, que uma clique gera uma valoração válida e que uma valoração válida gera uma clique, reforça o raciocínio formal necessário para reduções corretas. Muitos estudantes cometem o erro de provar apenas uma direção ou de assumir equivalência sem justificativa rigorosa. A demonstração cuidadosa apresentada serve como modelo de argumentação matemática, enfatizando a importância de considerar todas as possibilidades e eliminar casos degenerados.

Analisar como cada valoração influencia o conjunto das cláusulas satisfeitas ajuda a desenvolver intuição sobre como problemas booleanos capturam propriedades de grafos. Ao considerar sistematicamente os efeitos de atribuir valores verdadeiro ou falso a cada variável, estudantes desenvolvem compreensão profunda de como restrições locais (satisfação de cláusulas individuais) emergem como propriedades globais (existência de cliques). Essa conexão entre nível local e global é fundamental não apenas em complexidade computacional, mas em toda ciência da computação.

Ao final, a redução analisada oferece uma visão sólida sobre a intratabilidade do Max-2SAT e sobre a versatilidade das reduções polinomiais. A construção estudada demonstra, de forma acessível e estruturada, como problemas de natureza combinatória e lógica podem ser relacionados de maneira precisa, contribuindo significativamente para o aprendizado prático de \mathcal{NP} -completude.

+30 material desenvolvido pode ser empregado em diversos contextos de ensino. Em disciplinas de *graduação* em Teoria da Computação ou Análise de Algoritmos, a redução pode ser apresentada como estudo de caso após a introdução dos conceitos de \mathcal{NP} -completude, permitindo que estudantes acompanhem passo a passo uma demonstração completa antes de desenvolverem suas próprias provas. Em cursos de *pós-graduação*, o material pode servir como ponto de partida para discussões sobre técnicas avançadas de redução, limites de aproximabilidade e conexões com outros problemas de satisfatibilidade. Como *atividade prática*, sugere-se propor aos estudantes a verificação manual da redução para grafos pequenos, a implementação computacional do algoritmo de transformação, ou a adaptação da técnica para variantes

como Max-3SAT ou Weighted Max-2SAT. A estrutura modular da apresentação — com figuras, tabelas-resumo e exemplos comentados — facilita a segmentação do conteúdo em múltiplas aulas ou a utilização em metodologias ativas baseadas em seminários, conforme discutido por Lassance, Bianchini e Santos [11].

O material apresentado serve tanto como recurso didático para compreensão de técnicas específicas quanto como exemplar metodológico para desenvolvimento de novas reduções, cumprindo assim o objetivo pedagógico central deste trabalho.

VII. CONSIDERAÇÕES FINAIS

Este artigo apresentou uma demonstração formal e didática da \mathcal{NP} -completude do problema Max-2SAT por meio de redução polinomial a partir do problema Clique. A prova foi estruturada demonstrando-se primeiro a pertinência à classe \mathcal{NP} mediante certificado verificável em tempo polinomial, e em seguida a \mathcal{NP} -dificuldade através de transformação polinomial que preserva equivalência entre instâncias.

A construção proposta utiliza variável auxiliar z para controlar o número de cláusulas satisfeitas, ajustando o parâmetro K' para refletir de forma precisa o tamanho da clique desejada. As cláusulas de seleção incentivam a escolha de vértices, enquanto as cláusulas de incompatibilidade impedem seleção simultânea de vértices não adjacentes, evidenciando como relações combinatórias em grafos são codificadas por fórmulas booleanas.

Entre as dificuldades encontradas, destacam-se a escolha adequada do parâmetro K' e a formalização rigorosa da prova de corretude em ambas as direções. A compreensão do papel da variável z exigiu análise cuidadosa de como cada tipo de cláusula contribui para a contagem total.

Como limitações, o trabalho concentrou-se na versão de decisão do Max-2SAT e na redução a partir de Clique. Outras reduções, como baseadas em 3-SAT ou Vertex Cover, podem oferecer perspectivas complementares. Para trabalhos futuros, sugere-se explorar variantes parametrizadas, analisar complexidade em classes especiais de grafos, desenvolver algoritmos aproximativos e investigar aplicações com SAT solvers modernos.

O material produzido serve como recurso didático para disciplinas de Teoria da Computação, contribuindo para a formação de estudantes em Ciência da Computação. Espera-se que inspire novas produções que articulem rigor técnico e finalidade pedagógica, fortalecendo a comunidade de ensino e pesquisa em Teoria da Computação.

REFERÊNCIAS

- [1] S. A. Cook, "The complexity of theorem-proving procedures," *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pp. 151–158, 1971.
- [2] B. Aspvall, M. F. Plass, and R. E. Tarjan, "A linear-time algorithm for testing the truth of certain quantified boolean formulas," *Information Processing Letters*, vol. 8, no. 3, pp. 121–123, 1979.
- [3] M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified np-complete graph problems," *Theoretical Computer Science*, vol. 1, no. 3, pp. 237–267, 1976.
- [4] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

- [5] M. Sipser, *Introduction to the Theory of Computation*, 2nd ed. Boston: Thomson Course Technology, 2006.
- [6] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York, NY; London: Plenum Press, 1972, pp. 85–103.
- [7] C. H. Papadimitriou, *Computational Complexity*. Addison-Wesley, 1994.
- [8] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.
- [9] L. Trevisan, G. B. Sorkin, M. Sudan, and D. P. Williamson, "Gadgets, approximation, and linear programming," in *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, 1996, pp. 617–626. [Online]. Available: <https://ieeexplore.ieee.org/document/548521>
- [10] S. Khanna, M. Sudan, L. Trevisan, and D. P. Williamson, "The approximability of constraint satisfaction problems," *SIAM Journal on Computing*, vol. 30, no. 6, pp. 1863–1920, 2001. [Online]. Available: <https://doi.org/10.1137/S0097539799349948>
- [11] Y. M. Lassance Di Vilhena Y Cantãñede, G. d. B. Bianchini, and T. D. d. Santos, "Reflexões e práticas pedagógicas no escopo da disciplina de teoria da computação," *Academic Journal on Computing, Engineering and Applied Mathematics*, vol. 6, no. 2, pp. 10–17, October 2025.