Problemas de Agendamento: Contribuições Pedagógicas para o Aprendizado no Escopo da Teoria da Computação

Scheduling Problems: Pedagogical Contributions to Learning within the Scope of Computation Theory

Heloisa Rolins Ribeiro [1], Neci Oneides da Silva Fialho Neta [1], Daniel Martins da Silva [2] e Tanilson Dias dos Santos [1]

[1] *Universidade Federal do Tocantins, Curso de Ciência da Computação*
[2] *Universidade Federal do Norte do Tocantins,Curso de Logística*

Data de recebimento do manuscrito: 03/12/2025
Data de aceitação do manuscrito: 05/01/2026
Data de publicação: 10/02/2026

Resumo—Os problemas de agendamento constituem uma classe essencial de desafios em otimização e computação, especialmente em sistemas operacionais, processamento paralelo e aplicações em tempo real. Apesar de sua ampla utilização prática, diversas variantes permanecem computacionalmente intratáveis, mesmo sob fortes restrições estruturais. Este artigo investiga a NP-completude de duas versões específicas do problema de escalonamento: o agendamento com tempo de execução unitário e o agendamento com dois processadores em tarefas de duração igual a uma ou duas unidades. A fundamentação teórica baseia-se em reduções polinomiais clássicas, em particular a partir do problema 3-SAT, que permite codificar atribuições lógicas diretamente nas restrições de precedência e capacidade dos processadores. Além disso, transformações adicionais entre versões restritas do problema são utilizadas para preservar a equivalência estrutural das soluções. As contribuições incluem uma reconstrução didática das provas originais, a análise dos mecanismos que geram dureza computacional e uma discussão sobre as implicações práticas desses resultados em sistemas reais de escalonamento. Os resultados apresentados na literatura reforçam que mesmo cenários aparentemente simples apresentam comportamento NP-completo.

Palavras-chave—NP-completude; Agendamento; Redução polinomial; 3-SAT; Complexidade computacional.

Abstract—*Scheduling problems constitute a fundamental class of challenges in optimization and computing, particularly in operating systems, parallel processing, and real-time applications. Despite their wide practical use, many variants remain computationally intractable, even under strong structural restrictions. This article investigates the NP-completeness of two specific versions of the scheduling problem: scheduling with unit processing time and scheduling on two processors with tasks of duration one or two time units. The theoretical foundation relies on classical polynomial-time reductions, especially from the 3-SAT problem, which allows logical assignments to be encoded directly into precedence constraints and processor-capacity limitations. Furthermore, additional transformations between restricted versions of the problem are employed to preserve the structural equivalence of solutions. The contributions include a didactic reconstruction of the original proofs, an analysis of the mechanisms that give rise to computational hardness, and a discussion of the practical implications of these results in real scheduling systems. The results presented in the literature reinforce that even seemingly simple scenarios exhibit NP-complete behavior.*

Keywords—*NP-completeness; Scheduling; Polynomial reduction; 3-SAT; Computational complexity.*

I. Introdução

A Teoria da Computação estabelece os fundamentos formais para compreender os limites do que pode ser calculado de maneira eficiente. Nesse contexto, a teoria da complexidade computacional desempenha papel central ao classificar problemas quanto ao custo de suas soluções, destacando as classes $\mathcal{P}$, $\mathcal{NP}$ e NP-completo. Problemas NP-completos são aqueles para os quais não se conhece algoritmo polinomial e, ao mesmo tempo, qualquer problema em $\mathcal{NP}$ pode ser reduzido a eles em tempo polinomial. Assim, demonstrar que um problema pertence a essa classe significa evidenciar sua provável intratabilidade.

Nesse contexto, os problemas de agendamento (scheduling problems) ocupa posição de destaque. Eles modelam situações onde tarefas devem ser distribuídas ao longo do

Dados de contato: Neci Oneides da Silva Fialho Neta, neci.silva@uft.edu.br

tempo ou entre múltiplos processadores, respeitando restrições de precedência, limites de duração e capacidade. Tais problemas surgem em sistemas operacionais, manufatura, computação paralela, arquiteturas multinúcleo e otimização industrial. Entretanto, mesmo versões altamente restritas do escalonamento podem exibir comportamento computacional complexo.

O presente artigo aborda duas variantes específicas: (i) o agendamento em que todas as tarefas possuem tempo de execução unitário e (ii) o agendamento em dois processadores com tarefas de duração igual a 1 ou 2 unidades. Apesar da simplicidade aparente dessas restrições, ambas as versões são NP-completas.

O propósito deste trabalho é apresentar uma análise das provas de NP-completude desses dois problemas, contextualizando-as dentro da Teoria da Computação e explicando, passo a passo, como reduções polinomiais, especialmente a partir do problema 3-SAT, permitem codificar instâncias lógicas dentro de modelos de escalonamento. Além disso, discute-se como restrições de precedência, janelas de execução e limitações de processadores funcionam como dispositivos para simular atribuições booleanas.

As principais contribuições deste artigo são a reconstrução didática das demonstrações clássicas, tornando-as mais acessíveis a estudantes e pesquisadores; a análise conceitual dos mecanismos responsáveis pela complexidade computacional dos problemas estudados; a integração entre teoria e prática, discutindo implicações para sistemas reais de escalonamento e algoritmos modernos; e a organização clara e sistemática das relações entre as variantes do problema, destacando cadeias de reduções e interdependências.

Com isso, o artigo busca não apenas demonstrar formalmente a NP-completude das variantes analisadas, mas também oferecer uma compreensão mais profunda sobre por que tais problemas permanecem intratáveis mesmo em cenários simples.

Para organizar a discussão, o artigo está estruturado da seguinte forma: na Seção II (Preliminares), apresentam-se os conceitos preliminares necessários para compreender a complexidade dos problemas estudados, incluindo definições formais, modelos de agendamento e a cadeia de reduções utilizada. A Seção III (Trabalhos Relacionados) revisa trabalhos clássicos e contemporâneos relacionados ao tema, situando P2 e P3 no contexto mais amplo da teoria de escalonamento. A Seção IV (Descrição do Problema) descreve formalmente as variantes analisadas e suas aplicações, ilustrando seus aspectos combinatórios. Na Seção V (Demonstração e Contribuições) são desenvolvidas as provas de NP-completude de P2 e P3, com ênfase nas reduções polinomiais que conectam esses problemas ao 3-SAT. A Seção VI (Resultados e Reflexões) apresenta reflexões e interpretações sobre os resultados obtidos, destacando implicações teóricas e pedagógicas. Por fim, a Seção VII (Considerações Finais) reúne as considerações finais e aponta possíveis direções para investigações futuras.

II. PRELIMINARES

As preliminares apresentadas nesta seção têm o objetivo de estabelecer todas as definições, notações e convenções formais utilizadas ao longo deste trabalho. Como as demonstrações de NP-completude reconstruídas aqui envolvem cadeias de reduções, relações de precedência, funções de escalonamento e estruturas lógicas, é importante que os símbolos e conceitos empregados sejam apresentados de modo claro e unificado antes de aparecerem nas seções posteriores.

TABELA 1: DESCRIÇÃO FORMAL DO PROBLEMA 3-SAT.

3-SAT

Entrada: Uma fórmula booleana φ em forma normal conjuntiva,

$$\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_k,$$

onde cada cláusula possui exatamente três literais:

$$C_i = (\ell_1 \vee \ell_2 \vee \ell_3), \qquad \ell_j \in \{x, \neg x\}.$$

Objetivo: Decidir se existe uma atribuição de valores verdade às variáveis que satisfaça todas as cláusulas de φ.
Saída: SIM, se φ é satisfatível; NÃO, caso contrário.

Para iniciar, adotamos as classes de complexidade usuais da Teoria da Computação. A classe $\mathcal{P}$ contém todos os problemas de decisão solucionáveis por algoritmos determinísticos cujo tempo de execução é polinomial no tamanho da entrada. A classe $\mathcal{NP}$ reúne problemas cujas soluções podem ser verificadas em tempo polinomial por um verificador determinístico, dado um certificado apropriado. Um problema π é dito *NP-completo* se satisfaz duas condições: (i) $\pi \in \mathcal{NP}$; e (ii) para todo problema π' já conhecido por ser NP-completo, existe uma redução polinomial de π' para π. Denotamos tal redução pela notação:

$$\pi' \leq_p \pi,$$

que indica que qualquer instância de π' pode ser transformada, em tempo polinomial, em uma instância equivalente de π. Essa notação será empregada repetidas vezes ao longo deste artigo.

Como ponto de partida das reduções, utilizamos o problema 3-SAT, cuja importância histórica foi estabelecida por Cook em 1971 [1]. Empregamos as notações padrão: *variáveis booleanas* $x_1, \ldots, x_m$, que são entidades que podem assumir os valores verdadeiro ou falso; *literais* $\ell \in \{x_i, \neg x_i\}$, onde cada literal representa uma variável booleana ou sua negação; *cláusulas* $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$, que são disjunções de exatamente três literais; e *fórmulas booleanas* em forma normal conjuntiva (CNF) da forma

$$\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_n,$$

que consistem em conjunções de múltiplas cláusulas. Uma fórmula é *satisfatível* quando existe uma atribuição de valores às *variáveis booleanas* que torna todas as *cláusulas* verdadeiras. Como o problema 3-SAT é o ponto de origem da cadeia de reduções analisada, apresentamos a seguir sua definição formal apresentada na tabela 1.

Passamos agora ao modelo formal de agendamento adotado em todas as variantes do problema. Uma instância é composta por um conjunto de tarefas $S = \{J_1, \ldots, J_n\}$, uma relação parcial $<$ indicando precedências (por exemplo, $J < J'$ significa que J deve terminar antes do início de

ISSN: 2675-3588

J'), uma função duração $W : S \to \mathbb{Z}^+$ que associa a cada tarefa seu tempo de execução em unidades de tempo, um número de processadores k, e um horizonte de tempo t. Um escalonamento é descrito por uma função

$$f : S \to \{0, 1, \ldots, t-1\},$$

que define para cada tarefa seu instante de início dentro do intervalo de tempo disponível, onde $f(J)$ determina o instante de início de J. Esse escalonamento é válido quando satisfaz: (i) $f(J) + W(J) \leq f(J')$ sempre que $J < J'$; (ii) em cada unidade de tempo, no máximo k tarefas executam simultaneamente; e (iii) todas as tarefas terminam antes do tempo limite t. Essa notação será usada constantemente nas definições e nas construções das reduções.

As variantes específicas de escalonamento reconstruídas neste trabalho são as mesmas introduzidas por Ullman (1975) [2]. O problema P2 consiste em escalonar tarefas com duração unitária sobre k processadores, sob um conjunto arbitrário de precedências. O problema P3 envolve dois processadores e tarefas cujas durações pertencem ao conjunto $\{1,2\}$. O problema P4 é semelhante a P2, exceto pelo fato de que, em vez de um número fixo de processadores, cada unidade de tempo possui uma capacidade própria $c_0, c_1, \ldots, c_{t-1}$. Já o problema P5 corresponde a uma versão onde todos os processadores devem permanecer ocupados durante toda a execução, isto é, exatamente k tarefas devem estar em execução em cada instante.

Como as reduções de Ullman empregam estruturas visuais e padrões temporais específicos, adotamos também notações auxiliares internas às construções. Cadeias verticais de tarefas representam literais ou suas negações; barras sobre variáveis, como $\bar{x}_i$, indicam negação; blocos D_{ij} agrupam tarefas associadas a cláusulas ou a estruturas auxiliares; e, na redução para P3, os termos *banda* e *quebra* designam segmentos longos e curtos de execução no segundo processador, respectivamente. O par de tarefas J' e J, ambas de duração 2, será utilizado para preservar precedências ao converter instâncias de P5 para P3. Finalmente, o termo *certificado* será entendido sempre como um escalonamento candidato cuja verificação é realizada em tempo polinomial.

Com todas essas convenções estabelecidas, apresentamos a cadeia de reduções que estrutura a demonstração da NP-completude dos problemas estudados:

$$\text{3-SAT} \leq_p P4 \leq_p P5 \leq_p P2, \qquad P5 \leq_p P3.$$

Cada ocorrência do símbolo $\leq_p$ será detalhada nas seções subsequentes, com construções explícitas e demonstrações de validade. Assim, esta seção reúne todo o aparato matemático necessário para sustentar as provas desenvolvidas ao longo do artigo.

III. Trabalhos Relacionados

Os estudos sobre a complexidade de problemas de escalonamento possuem uma trajetória consolidada na literatura, e o presente trabalho se insere nesse contexto ao analisar variantes restritas que permanecem NP-completas. O trabalho de referência fundamental é o de Ullman [2], cujo objetivo foi demonstrar formalmente que versões simplificadas do problema de scheduling continuam a exibir dureza combinatória. Por meio de reduções formais iniciadas em 3-SAT, o autor constrói progressivamente instâncias dos problemas P4, P5, P2 e P3, utilizando cadeias de tarefas, precedências rígidas e janelas de execução que representam diretamente a lógica das fórmulas booleanas. Seu principal resultado é estabelecer que tanto o agendamento com tempos unitários quanto o agendamento em dois processadores com tarefas de duração 1 ou 2 são NP-completos, servindo como base teórica direta para as análises reconstruídas neste artigo.

O survey clássico de Graham, Lawler, Lenstra e Rinnooy Kan [3] também está intimamente relacionado a este trabalho. Seu objetivo foi organizar e classificar modelos determinísticos de escalonamento, descrevendo algoritmos, limites de complexidade, estruturas de precedência e resultados de aproximação. A metodologia consiste em sistematizar o campo usando a notação de três campos $(\alpha|\beta|\gamma)$, além de situar diversos problemas dentro de categorias de tratabilidade ou NP-dificuldade. O survey demonstra que a interação entre precedências e múltiplas máquinas é uma das principais fontes de intratabilidade, o que contextualiza de maneira abrangente os problemas P2 e P3 analisados aqui.

Outro trabalho relevante é o de Brucker e Kravchenko [4], cujo foco é o escalonamento em máquinas paralelas quando todos os tempos de processamento são iguais. Seu objetivo foi investigar como a presença de precedências e janelas temporais afeta a complexidade do problema. Por meio de reduções polinomiais baseadas em problemas clássicos de particionamento, os autores demonstram que mesmo instâncias homogêneas tornam-se NP-difíceis quando combinadas com dependências. Essa conclusão reforça diretamente o caso de P2 estudado neste artigo.

Também se destaca a obra de Pinedo [5], cujo objetivo é oferecer uma visão abrangente dos modelos de escalonamento utilizados em sistemas industriais, computacionais e de produção. É uma referência técnica essencial para compreender como modelos com múltiplas máquinas, precedências e janelas temporais se comportam na prática, contextualizando os cenários teóricos tratados neste trabalho.

Por fim, o trabalho de Baptiste, Leung e Smith [6] aprofunda limites de complexidade em modelos de escalonamento com restrições de precedência, janelas de disponibilidade e múltiplas máquinas. Seu objetivo é mapear rigorosamente a fronteira entre casos polinomiais e NP-difíceis, empregando técnicas de construção temporal similares às utilizadas por Ullman. Seus resultados mostram que até variantes aparentemente simples tornam-se NP-completas quando precedências e tempos variados interagem, conectando-se diretamente às reduções que caracterizam P3.

Coletivamente, esses estudos situam claramente P2 e P3 dentro do panorama teórico do escalonamento, reforçando que tais variantes representam casos emblemáticos na fronteira entre tratabilidade e intratabilidade na Teoria da Computação.

Em complemento a esses estudos específicos de escalonamento, a monografia clássica de Garey e Johnson [7] fornece o pano de fundo teórico geral sobre NP-completude e técnicas de redução polinomial. Embora trate de uma ampla variedade de problemas e não se concentre exclusivamente

TABELA 2: DESCRIÇÃO FORMAL DO PROBLEMA P2.

P2 – Escalonamento com tempo de execução unitário
Entrada: Um conjunto de n tarefas, cada uma levando exatamente 1 unidade de tempo para ser concluída. Há relações de precedência entre algumas tarefas, existem m processadores idênticos disponíveis e um tempo máximo total D para executar todas elas. **Objetivo:** Decidir se existe uma forma de agendar todas as tarefas nos m processadores de modo que todas sejam concluídas até o tempo limite D. **Saída:** SIM, se existe um escalonamento que termina todas as tarefas dentro de D; NÃO, caso contrário.

TABELA 3: DESCRIÇÃO FORMAL DO PROBLEMA P3.

P3 – Escalonamento com tempos de execução variados
Entrada: Um conjunto de n tarefas, cada uma com um tempo de execução definido. Há relações de precedência entre certas tarefas. Existem m processadores idênticos disponíveis e um tempo limite total D para concluir todas as tarefas. **Objetivo:** Determinar se existe um escalonamento válido que aloque todas as tarefas aos m processadores de forma a respeitar as precedências e terminar tudo até o tempo D. **Saída:** SIM, se existe tal escalonamento dentro de D; NÃO, caso contrário.

em modelos de escalonamento como P2 e P3, essa obra é uma referência útil para o enquadramento conceitual deste trabalho, especialmente no que diz respeito à definição formal das classes $\mathcal{P}$, $\mathcal{NP}$ e dos problemas NP-completos.

Além da literatura técnica sobre escalonamento, este trabalho também se apoia em produções pedagógicas da área de Teoria da Computação. O artigo de Lassance et al. [8] discute práticas de ensino envolvendo decidibilidade, NP-completude e transformações polinomiais, oferecendo uma base didática que auxiliou na organização conceitual dos fundamentos teóricos utilizados, ainda que não trate diretamente dos problemas de scheduling analisados aqui.

IV. DESCRIÇÃO DO PROBLEMA

Os problemas de agendamento tratam da organização de um conjunto de tarefas ao longo do tempo ou entre múltiplos recursos, respeitando restrições estruturais como precedência, duração e capacidade de processamento. Em sua formulação clássica, busca-se determinar em que momento cada tarefa deve ser executada, de modo a cumprir dependências e limitações de recursos, garantindo que todas sejam concluídas antes de um tempo máximo permitido. As variantes analisadas neste trabalho — o agendamento com tempo de execução unitário (P2) e o agendamento em dois processadores com tarefas de duração igual a uma ou duas unidades (P3) — representam versões restritas desse modelo geral, mas preservam a complexidade combinatória presente em cenários mais amplos.

A definição formal do problema de escalonamento com tempo unitário é apresentada na tabela 2.

Embora o problema P2 trate exclusivamente de tarefas com duração unitária, o que permite certas simplificações em sua análise estrutural, muitas aplicações práticas exigem considerar tarefas com tempos distintos de execução. Essa generalização leva naturalmente à formulação do problema P3, apresentada a seguir na tabela 3.

Enquanto P2 e P3 representam variantes fundamentais do modelo de escalonamento com restrições de precedência e limite global de tempo, a análise de sua complexidade costuma recorrer a versões intermediárias mais expressivas. Entre elas destacam-se os problemas P4 e P5, que introduzem novos elementos — como tempos de liberação e prazos individuais — permitindo construir reduções mais detalhadas e modularizadas ao longo da prova de NP-completude.

TABELA 4: DESCRIÇÃO FORMAL DO PROBLEMA P4.

P4 – Escalonamento com tempos de liberação
Entrada: Um conjunto de n tarefas, cada uma com um tempo de duração e um instante mínimo no qual está autorizada a começar; algumas tarefas devem ocorrer antes de outras; há m processadores idênticos disponíveis; e existe um limite total D para finalizar todas as tarefas. **Objetivo:** Determinar se há uma forma de escalonar todas as tarefas nos m processadores, respeitando os tempos de liberação, as precedências e o tempo máximo permitido. **Saída:** SIM, se existe um escalonamento válido dentro de D; NÃO, caso contrário.

TABELA 5: DESCRIÇÃO FORMAL DO PROBLEMA P5.

P5 – Escalonamento com precedências arbitrárias
Entrada: Um conjunto de n tarefas, cada uma com tempo de duração e um prazo individual; um conjunto de dependências indicando quais tarefas devem anteceder outras; e m processadores idênticos disponíveis. **Objetivo:** Determinar se existe um escalonamento que respeite tanto os prazos individuais como todas as dependências entre as tarefas. **Saída:** SIM, se existe um escalonamento válido que satisfaça todos os prazos e dependências; NÃO, caso contrário.

Para estabelecer a complexidade computacional dos problemas P2 e P3, utilizamos dois problemas intermediários nas reduções, conforme proposto por Ullman [2]: P4 (escalonamento com tempos de liberação),definido formalmente na tabela 4, e P5 (escalonamento com precedências arbitrárias),apresentado na tabela 5. Estes servem como etapas intermediárias na cadeia de reduções que parte do problema 3-SAT e culmina na demonstração de NP-completude de P2 e P3.

Embora P4 e P5 compartilhem a estrutura básica de problemas de escalonamento, diferenciam-se pelas restrições específicas que impõem. Enquanto P4 introduz *tempos de liberação* como restrições adicionais ao início das tarefas, P5 generaliza as relações de precedência e incorpora *prazos individuais* para cada tarefa. Essa progressão na complexidade das restrições é fundamental para a cadeia de reduções, permitindo que se estabeleça a NP-dificuldade de P2 através de transformações sucessivas partindo do 3-SAT.

Uma forma intuitiva de visualizar esses problemas é por

ISSN: 2675-3588

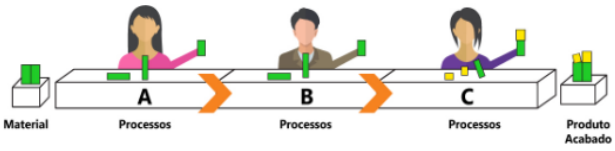

Figura 1: Linha de produção ilustrando dependências, capacidade limitada e fluxo sequencial — elementos que caracterizam os problemas de escalonamento P2 e P3.

meio de uma linha de produção simplificada, como ilustrado na Figura 1. Nessa representação, o material bruto entra pela esquerda e percorre três etapas de processamento (A, B e C) até tornar-se produto acabado. Cada estação representa um conjunto de tarefas que deve ser executado em uma ordem específica, pois A precisa concluir sua parte antes que B possa começar, e o mesmo vale para a transição de B para C. Essa metáfora captura precisamente a ideia de restrições de precedência presentes nos problemas de escalonamento. Além disso, cada operador da linha só consegue manipular uma peça por vez, analogamente ao limite de capacidade dos processadores ou máquinas em um modelo computacional. Ao observarmos a linha em funcionamento, percebemos que, mesmo que as peças tenham tamanhos semelhantes, pequenas dependências ou variações de duração podem provocar bloqueios, esperas desnecessárias ou gargalos — efeitos que modelam diretamente a complexidade de P2 e P3.

No contexto dessa metáfora, o problema P2 corresponde a uma situação em que todas as tarefas duram exatamente uma unidade de tempo. Isso seria equivalente a imaginar que cada operador leva sempre o mesmo tempo para processar qualquer peça que receba. Ainda assim, dependências rígidas entre etapas podem impedir um fluxo contínuo, e o desafio consiste em verificar se existe uma forma de organizar essas execuções dentro de um limite global de tempo. Já o problema P3 se aproxima de uma linha de produção com dois operadores trabalhando simultaneamente, mas com tarefas que podem durar uma ou duas unidades de tempo. Nesse cenário, algumas peças exigem mais trabalho em uma etapa específica, o que gera desequilíbrios e requer um planejamento cuidadoso para evitar que o segundo operador fique sobrecarregado ou ocioso em momentos críticos.

Esses modelos, embora simples, surgem naturalmente em sistemas operacionais, computação paralela, engenharia industrial e processamento em tempo real. A analogia da linha de produção evidencia de forma clara como neles coexistem dois fatores cruciais: a necessidade de obedecer a dependências estritas e a limitação de recursos. Mesmo exemplos cotidianos, como essa sequência organizada de operações A→B→C, são suficientes para ilustrar como a ordem de execução e a duração das tarefas influenciam diretamente a viabilidade de um cronograma. Basta imaginar um operador ficando sem peças para trabalhar devido ao atraso na etapa anterior — um fenômeno equivalente à espera imposta pela precedência entre tarefas — ou dois operadores disputando o processamento simultâneo de peças, representando o conflito pela capacidade dos processadores.

Assim, a metáfora da linha de produção ajuda a visualizar por que os problemas P2 e P3, apesar de parecerem simples, capturam estruturas lógicas suficientemente ricas para simular decisões combinatórias complexas. Em particular, a interação entre tempos de execução, dependências e recursos limitados cria padrões que não apenas se aproximam do fluxo real de sistemas industriais, mas também constituem o núcleo das dificuldades teóricas que tornam esses problemas NP-completos.

V. Demonstração e Contribuições

Nesta seção apresentamos, as provas de NP-completude das duas variantes de escalonamento estudadas: o problema de tempo de execução unitário (P2) e o problema de dois processadores com tarefas de duração 1 ou 2 (P3). A demonstração é organizada em duas etapas principais para cada problema: (i) prova de NP-pertinência (isto é, mostrar que o problema pertence à classe $\mathcal{NP}$) e (ii) prova de NP-dificuldade (isto é, mostrar que o problema é ao menos tão difícil quanto um problema base conhecido como NP-completo).

a. P2 $\in$ NP e P3 $\in$ NP

Para demonstrar que os problemas P2 e P3 pertencem à classe $\mathcal{NP}$, utilizamos o conceito de verificador polinomial. Em ambos os casos, o certificado natural é um possível escalonamento das tarefas: para cada tarefa J, o certificado descreve o instante de início $f(J)$ e, no caso de P3, também o processador onde ela é executada.

Dado esse certificado, o verificador deve apenas conferir se o escalonamento obedece a todas as restrições impostas pelo problema. O procedimento consiste em:

1. verificar se cada tarefa termina antes do limite de tempo t;
2. verificar todas as relações de precedência, confirmando que, para cada $J < J'$, o término de J ocorre antes ou no momento do início de J';
3. para cada unidade de tempo, contar quantas tarefas estão sendo executadas simultaneamente, garantindo que esse valor não ultrapassa o número de processadores disponíveis (em P3, exatamente dois).

Cada uma dessas verificações pode ser feita em tempo polinomial no número de tarefas e no número de relações de precedência. Especificamente, a validação das precedências requer verificar cada relação individualmente, com complexidade $O(|<|)$, onde $|<|$ é o número de relações de precedência. A verificação de sobrecarga por unidade de tempo pode ser feita em $O(n \cdot t)$, mantendo-se contadores para cada instante. Não é necessário explorar todas as possíveis execuções, mas apenas validar a execução proposta no certificado. Portanto, existe um verificador determinístico polinomial tanto para P2 quanto para P3, o que implica:

$$P2 \in NP \qquad \text{e} \qquad P3 \in NP.$$

b. P2 $\in$ NP-Difícil

A prova de NP-dificuldade de P2 segue a estratégia de reduzir um problema clássico NP-completo, o 3-SAT, a uma instância de escalonamento com tempo de execução unitário.

A ideia é construir um conjunto de tarefas, precedências e limites de tempo tais que:

3-SAT é satisfatível $\iff$ P2 admite escalonamento válido.

Como abordado na Seção IV (Descrição do Problema), a demonstração completa é feita em etapas, por meio de P4 e P5.

Etapa 1: 3-SAT $\leq_p$ P4

Define-se inicialmente um problema intermediário, P4, que é uma versão do escalonamento com tempo unitário, mas com número de processadores variando ao longo do tempo. Em vez de um k fixo, P4 recebe uma sequência de capacidades $c_0, c_1, \ldots, c_{t-1}$, indicando quantas tarefas podem ser executadas em cada unidade de tempo.

A redução 3-SAT $\leq_p$ P4 constrói um conjunto de tarefas cuja viabilidade de escalonamento reflete diretamente a satisfatibilidade da fórmula. Para cada variável x_i são criadas duas cadeias disjuntas, uma associada a x_i e outra a $\neg x_i$, ligadas por restrições de precedência internas. A escolha de qual cadeia iniciar primeiro, e de qual literal será "verdadeiro", é forçada por tarefas auxiliares y_i e $\neg y_i$ que, via precedências e pela ocupação precisa dos slots de capacidade impostos pela sequência c_t, garantem que exatamente uma das duas cadeias progrida.

Em seguida, para cada cláusula C_j com três literais, introduz-se um bloco de sete tarefas $D_{j1}, \ldots, D_{j7}$ e arestas de precedência que as conectam às cadeias de variáveis. O instante crítico de execução dessas tarefas só pode ser alcançado se pelo menos uma das cadeias correspondentes a literais verdadeiros já estiver sido escalonada; caso contrário, a capacidade disponível naquele momento torna-se insuficiente e o escalonamento quebra. A sequência de capacidades $c_0, \ldots, c_{t-1}$ é escolhida de forma a apertar o espaço de processamento: em cada unidade de tempo o número de posições é exatamente o necessário para comportar as tarefas "verdadeiras" e as auxiliares, de modo que qualquer desvio da codificação correta — isto é, qualquer tentativa de satisfazer simultaneamente x_i e $\neg x_i$ ou de falsificar todas as cláusulas — impede a conclusão de todas as tarefas dentro do horizonte dado.

A complexidade desta construção é polinomial: para uma instância de 3-SAT com m variáveis e n cláusulas, o número total de tarefas geradas é da ordem de $O(m^2+n)$, assim como o número de relações de precedência. A construção pode ser implementada por algoritmos que percorrem variáveis e cláusulas com laços aninhados de profundidade constante, resultando em tempo polinomial no tamanho da fórmula original.

Comentário: a construção faz com que "rodar" certas tarefas em tempos específicos corresponda exatamente a atribuir verdadeiro ou falso às variáveis. Se a fórmula é satisfatível, existe um modo de encaixar todas as tarefas dentro do limite de tempo; se não é, faltará espaço em algum instante, e o escalonamento será impossível.

A Figura 2 ilustra a estrutura típica utilizada na redução 3-SAT $\rightarrow$ P4. Cada literal é convertido em uma cadeia vertical de tarefas — cadeias sem barra representam literais positivos, enquanto cadeias com barra representam negativas. Os blocos D_{ij} funcionam como pontos de verificação para cada

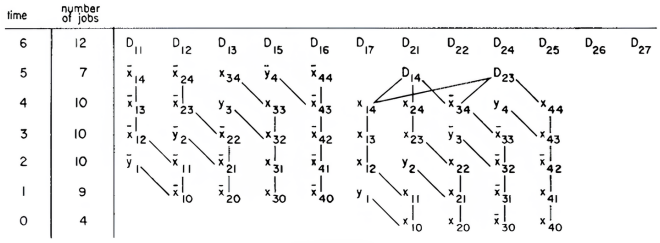

Figura 2: Estrutura geral da redução de uma instância de 3-SAT para o problema P4. Figura retirada de Ullman [2].

cláusula, garantindo que apenas escalonamentos compatíveis com uma atribuição satisfatória permitam preencher os instantes críticos impostos pelas capacidades temporais. Embora vários blocos apareçam na figura, apenas um bloco D_{ij} por cláusula desempenha o papel de validar a cláusula; os demais são estruturas auxiliares introduzidas para forçar o alinhamento temporal da construção.

Para tornar a construção mais intuitiva, a Figura 2 apresenta um exemplo concreto de como uma fórmula 3-SAT é convertida em cadeias de tarefas no problema P4. A fórmula booleana correspondente ao diagrama é:

$$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_3 \vee x_4).$$

Nessa representação, cada literal aparece como uma cadeia vertical de tarefas. Cadeias sem barra correspondem ao literal positivo (como x_{14}, x_{24}, x_{34}), enquanto cadeias com barra representam o literal negado (como $\bar{x}_{33}, \bar{x}_{32}, \bar{x}_{31}$). As relações de precedência ligam os elementos de cada cadeia, garantindo que a posição temporal em que uma tarefa pode ser executada codifica a escolha "verdadeiro" ou "falso" para cada variável.

Além das cadeias de literais, o diagrama inclui vários blocos D_{ij}. Esses blocos têm funções distintas dentro da redução. Apenas alguns deles correspondem diretamente às cláusulas da fórmula; os demais fazem parte da estrutura geral da construção e servem para controlar capacidade temporal, sincronizar cadeias ou criar janelas de execução obrigatórias. Assim, embora muitos blocos apareçam no diagrama, somente dois deles representam efetivamente as cláusulas da fórmula de exemplo.

Para cada cláusula C_j, Ullman [2] insere exatamente um bloco D_{ij} que atua como ponto de verificação: esse bloco só pode ser executado caso pelo menos um dos três literais da cláusula tenha sido marcado como verdadeiro pela estrutura de precedência construída. A indexação segue o padrão usado no artigo: o índice i refere-se à variável principal associada ao bloco, enquanto j indica a cláusula da qual aquele bloco participa.

No exemplo da figura, as duas cláusulas da fórmula aparecem como: D_{14} (cláusula 1), D_{23} (cláusula 2).

Os demais blocos, como $D_{11}, D_{12}, \ldots, D_{27}$, não representam cláusulas. Eles compõem apenas a estrutura auxiliar da redução e não têm correspondência com fórmulas booleanas; funcionam como "slots de tempo" usados para forçar a organização correta das cadeias de variáveis.

Dessa forma, a Figura 2 ilustra como cada literal, cada cláusula e cada restrição temporal são traduzidos para tarefas do problema P4, permitindo que a satisfatibilidade de φ

ISSN: 2675-3588

seja refletida diretamente na existência de um escalonamento viável.

Etapa 2: P4 $\leq_p$ P5

O problema P5 é definido como uma versão de P2 com a restrição adicional de que todos os processadores devem estar ocupados em todas as unidades de tempo; isto é, o número de tarefas é exatamente $n = kt$ e o escalonamento deve preencher completamente a capacidade disponível.

Para reduzir P4 a P5, transforma-se o cronograma de capacidades variáveis $c_0, \ldots, c_{t-1}$ em um quadro de k processadores constantes simplesmente completando, em cada instante i, as $k - c_i$ posições ociosas com tarefas "de preenchimento". Essas tarefas são introduzidas em número exato para que, em cada unidade de tempo, o total de tarefas ativas seja exatamente k; além disso, elas são conectadas por uma única cadeia de precedências lineares que as obriga a executar sequencialmente, impedindo que sobreponham ou conflitem com as tarefas originais. Como suas durações são unitárias e suas janelas de execução são rigidamente controladas, qualquer escalonamento válido de P5 descarta automaticamente as tarefas de preenchimento e recupera, nos instantes restantes, um escalonamento válido para P4; reciprocamente, todo escalonamento de P4 pode ser estendido a um de P5 incluindo as tarefas de preenchimento nos slots vazios.

Esta transformação é polinomial: o número de tarefas de preenchimento adicionadas é proporcional à diferença entre a capacidade máxima e o número de tarefas já previstas em P4. Como o horizonte de tempo t e as capacidades c_i são limitados por funções polinomiais no tamanho da instância de P4, o tempo de construção é polinomial.

Comentário: essa etapa padroniza a capacidade ao longo do tempo, transformando capacidades variáveis em um número fixo de processadores que precisam estar sempre ocupados.

Etapa 3: P5 $\leq_p$ P2

Por fim, observa-se que P5 é apenas um caso particular de P2: trata-se do mesmo problema de escalonamento com tempo de execução unitário, mas com a condição adicional de $n = kt$. Portanto, qualquer instância de P5 é uma instância de P2 com uma restrição extra, e a redução é imediata, com complexidade linear no tamanho da instância de P5.

Juntando as etapas, temos:

$$\text{3-SAT} \leq_p P4 \leq_p P5 \leq_p P2,$$

onde cada redução é computável em tempo polinomial, o que implica que P2 é NP-difícil. Como já foi mostrado que P2 pertence a NP, conclui-se que P2 é NP-completo.

c. P3 $\in$ NP-Difícil

Para demonstrar que P3 é NP-difícil, utiliza-se P5 como problema intermediário. A ideia é reduzir uma instância de P5 para uma instância de P3, de forma que:

$$\text{P5 é solucionável} \iff \text{P3 construída é solucionável.}$$

A construção explora a presença de dois processadores e tarefas com pesos 1 ou 2 para criar um padrão de *bandas* e *quebras* no segundo processador.

Etapa: P5 $\leq_p$ P3

Dada uma instância de P5 com tempo limite t, número de processadores k e conjunto de tarefas S de tamanho kt parcialmente ordenado por $<$, constrói-se uma instância de P3 sobre dois processadores de velocidade $s = 2$ da seguinte forma.

Primeiro, cria-se uma sequência contínua de tarefas X_i de peso 1 que ocupam exclusivamente o primeiro processador durante todo o horizonte t, impedindo qualquer outra tarefa de executar nele. Em seguida, no segundo processador, dispõem-se tarefas Y_{ij} também de peso 1 de modo a gerar um padrão regular de "quebras" e "bandas": após cada intervalo de $2k$ unidades de tempo livres (banda), insere-se uma única unidade de tempo ocupada (quebra), repetindo esse ciclo até cobrir o horizonte total.

Cada tarefa original $J \in S$ é substituída por um par (J', J), ambas de peso 2. A precedência $J' \prec J$ garante que J só possa iniciar após J' terminar, e as relações $J \prec K$ do conjunto original tornam-se $J \prec K'$ na nova instância, preservando a ordem parcial. O peso 2 impede que qualquer dessas tarefas seja executada durante uma quebra (apenas uma unidade de tempo livre); portanto, J' e J são forçadas a se alinharem inteiramente dentro de uma banda de $2k$ unidades consecutivas. Como cada banda oferece exatamente $2k$ unidades de capacidade e o número total de tarefas de peso 2 é $2kt$, o preenchimento completo de todas as bandas corresponde biunivocamente a um escalonamento válido de P5: cada par (J', J) posicionado numa banda representa a execução da tarefa original J num dos k processadores de P5, enquanto as quebras funcionam como divisores naturais entre os k instantes de tempo.

Esta construção é polinomial: o número total de tarefas em P3 é limitado por uma função polinomial no número de tarefas e no horizonte de tempo da instância de P5. Especificamente, são criadas $O(kt)$ tarefas, e a construção pode ser implementada por laços que percorrem o conjunto de tarefas originais e o intervalo de tempo sem recursão excessiva.

O efeito desta construção é o seguinte: as tarefas X_i garantem que o primeiro processador esteja sempre ocupado, enquanto as tarefas Y_{ij} consomem parte do tempo do segundo processador, de modo que restam segmentos de tempo contínuos (bandas) suficientemente longos apenas para acomodar as tarefas de peso 2 (J) e pequenos espaços (quebras) onde se encaixam as tarefas J'. Adicionalmente, devido às dependências, cada tarefa J deve ser executada na banda correspondente ao instante em que seria processada na solução de P5, enquanto J' ocupa a quebra associada.

Com isso, qualquer solução para a instância de P3 construída induz um escalonamento válido para a instância original de P5 (interpretando cada banda como uma unidade de tempo de P5). Reciprocamente, qualquer solução de P5 pode ser "expandida" para uma solução de P3, alocando J' nas quebras e J nas bandas apropriadas.

Comentário: as bandas funcionam como "janelas compactadas" que representam cada unidade de tempo da

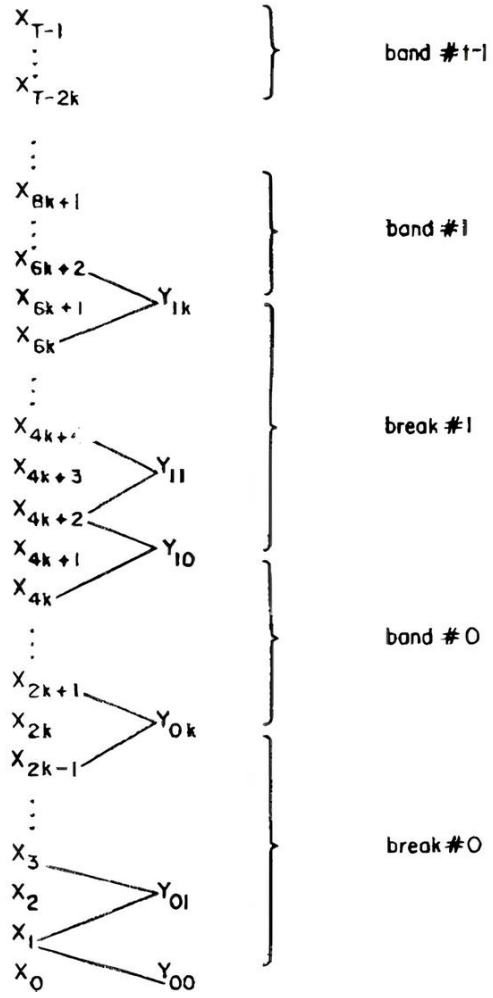

Figura 3: Organização das tarefas na redução do problema P5 para P3. Figura retirada de Ullman [2].

instância original de P5. Preencher corretamente essas bandas com tarefas de peso 2 equivale a decidir, para cada unidade de tempo, quais tarefas estão sendo executadas no modelo com k processadores.

A Figura 3 mostra como a redução de P5 para P3 utiliza dois processadores para reproduzir o comportamento temporal de uma instância original. O primeiro processador permanece ocupado continuamente pelas tarefas X_i, enquanto o segundo alterna entre segmentos curtos (*quebras*), que acomodam as tarefas J', e segmentos longos (*bandas*), nos quais são escalonadas as tarefas J de duração 2. Cada banda corresponde exatamente a uma unidade de tempo da instância de P5, preservando precedências e garantindo que o escalonamento resultante em P3 reflita corretamente a execução original em P5.

Como P5 é NP-completo e P5 $\leq_p$ P3 com complexidade polinomial, segue que P3 é NP-difícil. Já demonstramos anteriormente que P3 pertence a NP, portanto P3 é NP-completo.

d. Comentários Finais sobre a Demonstração

As provas apresentadas mostram que tanto P2 quanto P3 não são apenas variantes artificiais, mas modelos ricos o suficiente para simular a lógica de um problema canônico como o 3-SAT. As construções utilizadas exploram intensivamente a codificação de variáveis e atribuições como escolhas de tarefas executadas em tempos específicos, o uso de precedência para impor dependências lógicas, e o controle do número de processadores (ou da capacidade por unidade de tempo) para forçar o preenchimento exato de janelas de execução.

Esses mecanismos fazem com que qualquer tentativa de encontrar um algoritmo geral e eficiente para P2 ou P3 esbarre na mesma dificuldade encontrada para 3-SAT e outros problemas NP-completos. Assim, as demonstrações de NP-completude justificam, do ponto de vista teórico, o uso de heurísticas e algoritmos aproximados em problemas práticos de escalonamento.

VI. Resultados e Reflexões

A análise das variantes P2 e P3 permitiu confirmar formalmente sua classificação como problemas NP-completos, por meio da reconstrução detalhada das reduções apresentadas por Ullman (1975). A replicação dessas provas evidenciou como estruturas aparentemente simples (tarefas de duração unitária, dois processadores e precedências básicas) são suficientes para simular o comportamento lógico de fórmulas booleanas. Esse resultado reforça um dos princípios fundamentais da Teoria da Computação: a dificuldade computacional não depende apenas da complexidade aparente do modelo, mas da capacidade de representar decisões combinatórias por meio das restrições do problema.

Do ponto de vista metodológico, o processo revelou desafios relevantes. A cadeia de reduções 3-SAT $\rightarrow$ P4 $\rightarrow$ P5 $\rightarrow$ P2 exigiu um entendimento cuidadoso das construções intermediárias, especialmente na definição das capacidades variáveis de P4 e no uso das tarefas auxiliares que forçam a codificação de variáveis e cláusulas. Já a redução P5 $\rightarrow$ P3 mostrou-se particularmente difícil devido à alternância entre "bandas" e "quebras", que exige atenção à sincronização temporal e ao mapeamento entre segmentos de tempo e precedências. Esses aspectos demonstram que provas de NP-completude vão além de manipulações algébricas: tratam-se de construções conceituais sofisticadas que demandam rigor, visualização estrutural e compreensão profunda dos modelos envolvidos.

As contribuições deste trabalho situam-se tanto no campo da compreensão teórica quanto no campo pedagógico. Ao reorganizar e explicar as reduções de forma sistemática, com figuras, comentários e interpretações intuitivas, o estudo oferece um material mais acessível a estudantes e pesquisadores que desejam compreender NP-completude aplicada a problemas de escalonamento. Além disso, ao contextualizar os problemas P2 e P3 dentro da Teoria da Computação, o trabalho evidencia como reduções podem servir como ferramenta para analisar casos reais de sistemas operacionais, arquiteturas de processadores, computação paralela e engenharia de produção.

No âmbito acadêmico, a aplicabilidade desta investigação é ampla. A reconstrução didática das provas pode servir como apoio em disciplinas de Estruturas de Dados, Análise de Algoritmos, Teoria da Computação, Sistemas Operacionais e Escalonamento. O estudo também pode auxiliar estudantes na compreensão de reduções polinomiais, frequentemente uma das maiores dificuldades no apren-

 ISSN: 2675-3588

dizado de NP-completude, oferecendo exemplos concretos, visuais e contextualizados. Por fim, a exposição das limitações teóricas desses modelos reforça a importância de heurísticas, algoritmos aproximados e métodos experimentais em cenários reais onde soluções exatas são inviáveis.

Assim, os resultados alcançados não apenas reafirmam a NP-completude das variantes analisadas, mas também destacam o valor formativo do tema, demonstrando como problemas clássicos podem ser reinterpretados, visualizados e aplicados em contextos educacionais e práticos. O trabalho, portanto, contribui tanto para o rigor científico quanto para o fortalecimento do ensino da complexidade computacional.

VII. Considerações Finais

Este trabalho teve como objetivo analisar, formalizar e demonstrar a NP-completude das variantes de escalonamento P2 e P3, com base nas construções apresentadas por Ullman (1975). A partir da reconstrução detalhada das reduções a partir do 3-SAT — passando pelos problemas intermediários P4 e P5, foi possível compreender, de maneira estruturada e visual, como modelos de escalonamento aparentemente simples podem capturar a complexidade combinatória de problemas booleanos clássicos. Em síntese, demonstrou-se que tanto P2 quanto P3 pertencem à classe $\mathcal{NP}$ e são NP-difíceis, concluindo-se formalmente que ambos são NP-completos.

Durante o desenvolvimento da pesquisa, algumas dificuldades se mostraram centrais. A primeira refere-se à própria interpretação das construções utilizadas nas reduções, que exigem uma leitura atenta dos padrões de precedência, capacidade e sincronização temporal criados para simular variáveis e cláusulas. A segunda diz respeito ao esforço de transformar essas construções abstratas em explicações claras, diagramas compreensíveis e justificativas coerentes, mas essencial para consolidar o entendimento. Além disso, adaptar as provas originais para uma perspectiva didática, mantendo rigor matemático, demandou uma revisão cuidadosa da literatura e um tratamento sistemático das etapas envolvidas.

Apesar dessas dificuldades, os resultados obtidos ampliam a compreensão acadêmica sobre redução polinomial e sobre o caráter intratável de problemas de escalonamento. A abordagem adotada reforça o valor pedagógico das provas de NP-completude, especialmente quando apoiadas por esquemas visuais e interpretações intuitivas. O estudo também evidencia que a complexidade computacional permanece relevante não apenas em termos teóricos, mas também como fundamento para decisões práticas em sistemas operacionais, arquiteturas de processadores e modelos de produção.

Como perspectivas futuras, sugere-se aprofundar a análise de variações modernas dos problemas de escalonamento, incluindo modelos com preempção, janelas de tempo flexíveis, pesos múltiplos e ambientes heterogêneos. Outra linha de investigação envolve a exploração de algoritmos aproximados e heurísticas, fundamentais para aplicações reais nas quais soluções exatas são inviáveis devido à NP-completude. Por fim, estudos comparativos entre provas clássicas e abordagens contemporâneas de complexidade podem contribuir para o ensino, permitindo compreender como a teoria evoluiu e como esses problemas permanecem centrais na ciência da computação.

Assim, este trabalho não apenas consolida formalmente a NP-completude de P2 e P3, mas também contribui para o fortalecimento do entendimento teórico e pedagógico sobre reduções polinomiais, oferecendo bases sólidas para investigações futuras e para a aplicação prática desses conceitos em diferentes áreas da computação.

Referências

[1] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC '71)*. New York, NY, USA: Association for Computing Machinery, 1971, pp. 151–158.

[2] J. D. Ullman, "Np-complete scheduling problems," *Journal of Computer and System Sciences*, vol. 10, no. 3, pp. 384–393, 1975.

[3] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan, "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[4] P. Brucker and S. A. Kravchenko, "Scheduling jobs with equal processing times on parallel machines," in *Discrete Optimization and Operations Research*, P. M. Pardalos and V. Zhukovskii, Eds. Berlin: Springer, 2008, pp. 38–49.

[5] M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, 4th ed. New York: Springer, 2016.

[6] P. Baptiste, J. Y.-T. Leung, and C. Smith, *Scheduling: Algorithms and Complexity*. Berlin: Springer, 2001.

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

[8] Y. Lassance *et al.*, "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, Oct. 2025.

ISSN: 2675-3588