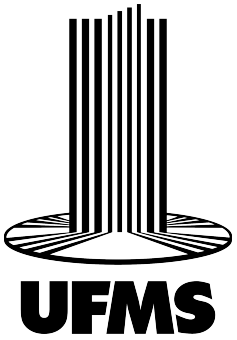
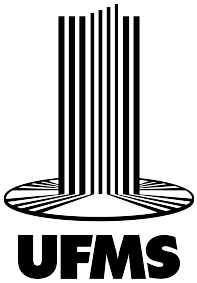


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE COMPUTAÇÃO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Uma Abordagem Matheurística baseada em Large Neighborhood Search para o Problema da Distribuição de Disciplinas

Deivid Reinke Schutz

Campo Grande - MS
23 de janeiro de 2026



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE COMPUTAÇÃO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Uma Abordagem Matheurística baseada em Large Neighborhood Search para o Problema da Distribuição de Disciplinas

Deivid Reinke Schutz

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal de Mato Grosso do Sul – UFMS.

Orientador: Profa. Dra. Edna Ayako Hoshino

Campo Grande - MS
23 de janeiro de 2026

Uma Abordagem Matheurística baseada em Large Neighborhood Search para o Problema da Distribuição de Disciplinas

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal de Mato Grosso do Sul – UFMS.

Banca Examinadora:

Profa. Dra. Edna Ayako Hoshino

Prof. Dr. Francisco Eloi Soares de Araújo

Prof. Dr. Fábio Henrique Viduani Martinez

Campo Grande - MS

23 de janeiro de 2026

Agradecimentos

À minha família.

Resumo

A atribuição de disciplinas a professores em instituições de ensino superior é um problema combinatório recorrente, classificado como NP-difícil por sua relação com o *Generalized Assignment Problem*, cuja complexidade cresce com o número de docentes, disciplinas e restrições operacionais envolvidas, tornando métodos exatos baseados em Programação Linear Inteira (PLI) potencialmente inviáveis em tempo computacional para instâncias de maior porte, ao passo que abordagens puramente heurísticas não oferecem garantias de qualidade. Este trabalho propõe, implementa e avalia uma matheurística baseada em Large Neighborhood Search (LNS) com reotimização por sub-MIP, em um esquema do tipo *fix-and-optimize*, integrada como heurística primal ao solver SCIP 7.0, para o Problema da Distribuição de Disciplinas (PDD), modelado como um programa linear inteiro binário cuja função objetivo maximiza a qualidade da atribuição com base em preferências docentes e compatibilidade de área, sujeito a restrições de cobertura obrigatória de todas as disciplinas, carga horária mínima anual e máxima semestral por professor. A heurística LNS opera como mecanismo de intensificação integrado ao Branch-and-Bound (B&B): a partir de uma solução incumbente, identifica as atribuições ativas, ordena os candidatos segundo uma métrica de versatilidade docente e libera uma fração controlada das variáveis para reotimização em um subproblema auxiliar (sub-MIP) com limite de tempo, incorporando a solução resultante ao solver principal caso represente melhoria. Três parâmetros internos — taxa de destruição, sentido de ordenação dos candidatos e tempo máximo do sub-MIP — foram calibrados por meio de uma abordagem incremental *one-factor-at-a-time*, utilizando subconjuntos de 10 instâncias extraídos de um conjunto de 42 instâncias sintéticas, com limite de tempo global de 400 segundos por instância. A combinação selecionada foi de 75% de destruição, ordenação crescente e 200 segundos de sub-MIP. No comparativo final entre B&B puro, LNS, heurística construtiva *Greedy Randomized Adaptive Search Procedure* (GRASP) e variante híbrida GRASP|LNS, foram avaliadas todas as 42 instâncias sintéticas e uma instância real. Nenhuma abordagem dominou todas as métricas: o B&B puro obteve o menor tempo total e o menor gap médio; GRASP|LNS apresentou o menor total de nós restantes; e B&B puro e LNS empataram com 32 das 43 instâncias resolvidas até a otimalidade. Na instância real, LNS e B&B puro provaram a otimalidade, com pequena vantagem temporal para o LNS. Os resultados mostram que a contribuição das heurísticas depende da métrica e da instância, destacando a redução da árvore proporcionada pela variante híbrida e o desempenho competitivo do LNS no caso real, sem evidência de superioridade global sobre o método exato.

Palavras-chave: Distribuição de Disciplinas. Programação Linear Inteira. Branch-and-Bound. Matheurística. Large Neighborhood Search. Fix-and-Optimize. SCIP.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

A gestão acadêmica semestral em instituições de ensino superior envolve um conjunto amplo de decisões interdependentes, entre as quais a atribuição de disciplinas a professores ocupa papel central. Trata-se de um problema recorrente, sensível ao contexto e de natureza combinatória: a cada período, mudanças no quadro docente, na oferta de disciplinas ou nas regras de carga horária podem alterar significativamente o espaço de soluções viáveis. Na literatura de pesquisa operacional, esse tipo de problema é estudado há décadas sob denominações como *class–faculty assignment* e *course timetabling* (CARTER; LAPORTE, 1998; GUNAWAN; NG; ONG, 2008), o que reflete tanto a sua ubiquidade institucional quanto a sua complexidade computacional.

A abordagem padrão para modelar problemas de alocação docente é a Programação Linear Inteira (PLI), em que decisões binárias representam atribuições entre professores e disciplinas, e a função objetivo quantifica a qualidade da alocação por meio de pesos que combinam preferência e compatibilidade de área (WOLSEY, 2020). Restrições lineares codificam requisitos operacionais, como carga horária mínima anual e máxima semestral por professor e cobertura obrigatória de todas as disciplinas. No contexto de atribuição de docentes com preferências, trabalhos como Al-Yakoob e Sherali (2006) e Domenech e Lusa (2016) demonstram a eficácia dessa formulação para capturar exigências institucionais de forma sistemática. Modelos de PLI são resolvidos por métodos exatos como o Branch-and-Bound (B&B), implementado em solvers modernos como o *Solving Constraint Integer Programs* (SCIP) (GAMRATH et al., 2020), que utiliza relaxações lineares como limites para guiar a enumeração e podar regiões improdutivas da árvore de busca (VANDERBEI, 2014).

Apesar da expressividade dos modelos exatos, instâncias de maior porte ou com restrições mais detalhadas podem dificultar a obtenção da prova de otimalidade dentro de limites de tempo práticos: o B&B pode explorar um número exponencial de nós sem convergir, devolvendo soluções com gap significativo. No contexto específico do Problema da Distribuição de Disciplinas (PDD), Silva (2024a) investigou uma abordagem exata via PLI, enquanto Pedrotti e Barreto (2025) desenvolveu uma heurística construtiva baseada em *Greedy Randomized Adaptive Search Procedure* (GRASP). O presente trabalho avança sobre essas contribuições ao propor uma matheurística de melhoria baseada em Large Neighborhood Search (LNS), combinando a qualidade de soluções iniciais obtidas por GRASP com uma etapa de refinamento via subproblema de Programação Inteira Mista (sub-MIP). A integração de matheurísticas baseadas em programação matemática — em particular, a LNS em um esquema do tipo *fix-and-optimize* como heurística primal do SCIP — permanecia inexplorada para o PDD, configurando a lacuna metodológica que

motiva o presente trabalho.

Esta monografia propõe, implementa e avalia uma matheurística baseada em LNS integrada ao solver SCIP para o PDD. O problema é modelado como um PLI binário e resolvido por B&B, enquanto a LNS atua como mecanismo de intensificação: a partir de uma solução incumbente, uma fração controlada das atribuições é liberada para reotimização em um subproblema auxiliar (sub-MIP) com limite de tempo, e, caso uma solução de melhor qualidade seja encontrada, ela é incorporada ao solver principal como nova incumbente (PISINGER; ROPKE, 2010). Três parâmetros internos são analisados sistematicamente — taxa de destruição, sentido de ordenação dos candidatos e tempo máximo do sub-MIP — por meio de uma abordagem *one-factor-at-a-time*. O desempenho da LNS é então comparado com três estratégias alternativas: o B&B puro, a heurística GRASP (PEDROTTI; BARRETO, 2025) e uma variante híbrida GRASP|LNS, na qual a LNS é aplicada como etapa de melhoria a partir de soluções construídas pelo GRASP (BOSCHETTI; LETCHFORD; MANIEZZO, 2023; MANIEZZO; STÜTZLE; VOSS, 2021).

A distribuição de disciplinas é realizada a cada semestre e está sujeita a prazos institucionais que demandam respostas rápidas, mesmo quando a prova de otimalidade não é exigida. Nesse cenário, a capacidade de gerar incumbentes de alta qualidade em estágios iniciais do B&B tem impacto direto na eficiência da busca: soluções melhores permitem podas mais agressivas da árvore, acelerando a convergência (NETO; HOSHINO; PEDROTTI, 2023). Matheurísticas do tipo *fix-and-optimize* exploram precisamente essa propriedade ao resolver, repetidamente, versões restritas do problema original com estrutura preservada da incumbente corrente.

A partir desta introdução, no Capítulo 2 é apresentada a fundamentação teórica, cobrindo conceitos de otimização inteira, heurísticas e matheurísticas relevantes ao estudo. O Capítulo 3 define o PDD formalmente e descreve tanto o modelo matemático exato quanto a matheurística LNS proposta. O Capítulo 4 apresenta os resultados experimentais e a análise computacional, incluindo a calibração dos parâmetros internos da LNS e as comparações de desempenho entre as abordagens. Por fim, o Capítulo 5 conclui o trabalho e discute limitações e possibilidades de trabalhos futuros.

2 Fundamentação Teórica

2.1 Otimização Combinatória e Programação Linear Inteira (PLI)

Problemas de alocação em ambientes acadêmicos podem ser formulados como problemas de otimização combinatória, nos quais se busca selecionar uma atribuição que maximize uma medida de qualidade (por exemplo, preferências) e, ao mesmo tempo, respeite restrições operacionais (carga horária, conflitos e requisitos mínimos). Na literatura, problemas próximos incluem o *class–faculty assignment* e variantes de alocação docente com preferências (AL-YAKOOB; SHERALI, 2006; AL-YAKOOB; SHERALI, 2013; DOMENECH; LUSA, 2016), bem como estudos em *course timetabling* (CARTER; LAPORTE, 1998; AVELLA; VASILYEV, 2005).

Uma forma padrão de modelar tais problemas é por Programação Linear Inteira (PLI), em que variáveis de decisão (frequentemente binárias) representam escolhas discretas, e a função objetivo quantifica a qualidade da solução. De modo geral, um modelo pode ser escrito conforme a Equação (2.1):

$$\max \sum_{p \in P} \sum_{c \in C} w_{p,c}\, x_{p,c} \quad \text{sujeito a restrições lineares e } x_{p,c} \in \{0,1\}, \tag{2.1}$$

em que $x_{p,c} = 1$ indica que o professor p é designado à disciplina c, e $w_{p,c}$ representa o peso/preferência. A PLI permite representar restrições de compatibilidade e limites de capacidade de forma explícita, mantendo uma interpretação direta do modelo (VANDERBEI, 2014; WOLSEY, 2020).

2.2 O Método Branch-and-Bound e Relaxação Linear

Modelos de PLI são, em geral, NP-difíceis, de modo que solucionadores utilizam métodos exatos baseados em limites e enumeração inteligente. Um componente central é a relaxação linear, obtida ao substituir a integralidade por $0 \leq x \leq 1$, produzindo um problema de Programação Linear (PL) que fornece um limite para a solução inteira ótima (VANDERBEI, 2014; WOLSEY, 2020).

O Branch-and-Bound (B&B) explora uma árvore de subproblemas: a cada nó, resolve-se uma relaxação para obter um limite e, quando necessário, ramifica-se em subproblemas impondo decisões de integralidade (por exemplo, $x_{p,c} = 0$ ou $x_{p,c} = 1$). Nós cujos limites não superam a melhor solução inteira conhecida podem ser podados, reduzindo a busca (WOLSEY, 2020). Na prática, solucionadores modernos integram B&B com técnicas adicionais; neste trabalho, a resolução exata é realizada com o SCIP (GAMRATH et al., 2020).

2.3 Heurísticas e Metaheurísticas

Embora métodos exatos forneçam garantias de otimalidade, instâncias reais podem apresentar grande dimensão e múltiplas restrições, tornando necessário recorrer a estratégias aproximadas. Heurísticas constroem boas soluções em tempo reduzido, enquanto metaheurísticas organizam mecanismos de exploração e intensificação para escapar de ótimos locais e melhorar soluções iterativamente.

Para problemas de alocação docente, há abordagens heurísticas e metaheurísticas que exploram diferentes representações e movimentos de vizinhança, como algoritmos genéticos (GUNAWAN; NG; ONG, 2008) e heurísticas voltadas a contextos específicos de atribuição docente — como a atribuição de assistentes de ensino a sessões de laboratório (HOSNY, 2013) e a incorporação de competências curriculares na alocação de professores (SZWARC; BACH-DÁBROWSKA; BOCEWICZ, 2018). Esse corpo de técnicas é especialmente útil quando se deseja produzir soluções de boa qualidade sob limites de tempo, ou quando o modelo exato é utilizado como referência/validação.

Uma distinção conceitual importante separa heurísticas, metaheurísticas e matheurísticas (BLUM; ROLI, 2003; MANIEZZO; STÜTZLE; VOSS, 2021). Heurísticas são procedimentos específicos ao problema que exploram sua estrutura para construir ou melhorar soluções de forma eficiente, sem garantia de otimalidade; exemplos clássicos incluem heurísticas gulosas construtivas e buscas locais simples por troca de pares. Metaheurísticas são estratégias de alto nível, independentes de domínio, que orquestram mecanismos de intensificação (exploração em torno de soluções promissoras) e diversificação (exploração de regiões ainda não visitadas), visando escapar de ótimos locais (BLUM; ROLI, 2003). Em suas formas clássicas, heurísticas e metaheurísticas operam diretamente sobre representações de soluções, sem recorrer necessariamente a um modelo matemático formal. Matheurísticas, categoria distinta, embutem um solver de otimização (como um solver de MIP) como operador de melhoria ou construção, aproveitando simultaneamente a estrutura matemática do modelo — em particular, limites de qualidade e garantias de viabilidade — e a flexibilidade das estratégias de busca (MANIEZZO; STÜTZLE; VOSS, 2021; BALL, 2011). Essa combinação é especialmente vantajosa quando o problema admite formulação exata, mas as instâncias práticas excedem a capacidade dos solvers em tempo razoável.

2.3.1 Greedy Randomized Adaptive Search Procedure (GRASP)

O GRASP é uma metaheurística em duas fases: (i) uma construção gulosa aleatorizada, que produz uma solução inicial por escolhas guiadas por um critério de qualidade com componente probabilística; e (ii) uma busca local, que tenta melhorar a solução por movimentos em uma vizinhança definida. A aleatoriedade permite amostrar diferentes

regiões do espaço de soluções, enquanto a busca local promove intensificação.

No contexto do Problema da Distribuição de Disciplinas (PDD), uma estratégia GRASP é um caminho natural por combinar regras gulosas (por exemplo, priorizar pares professor–disciplina com maior peso) com fases de melhoria que reatribuem disciplinas para reduzir violações e aumentar a qualidade (PEDROTTI; BARRETO, 2025).

2.4 Matheurísticas

Matheurísticas combinam programação matemática com componentes heurísticos/metaheurísticos, buscando aproveitar simultaneamente a força de modelagem e os limites fornecidos por modelos exatos, e a flexibilidade de estratégias de busca em grandes espaços combinatórios (BALL, 2011; BOSCHETTI; LETCHFORD; MANIEZZO, 2023; MANIEZZO; STÜTZLE; VOSS, 2021). Em geral, a ideia é usar submodelos (ou modelos restritos) como operadores de melhoria, resolver subproblemas por Programação Inteira Mista (MIP)/CPLEX/SCIP, ou ainda usar informações do solucionador (limites, incumbentes, cortes) para orientar a busca.

Em problemas do tipo mochila e variantes, como o Problema da Mochila com Conjuntos de Penalidades (D'AMBROSIO et al., 2023), há linhas de trabalho que empregam resoluções exatas em subestruturas, gerando melhorias sistemáticas em torno de incumbentes (JOVANOVIC; VOSS, 2024; SILVA, 2024b). Essa mesma filosofia se estende ao PDD: parte-se de um modelo de PLI e explora-se uma vizinhança definida sobre variáveis binárias, reotimizando subproblemas com tempo controlado.

Aplicações reais de matheurísticas LNS foram reportadas em problemas estruturalmente próximos ao PDD. Em roteamento de veículos, Ropke e Pisinger (2006) propuseram uma Large Neighborhood Search Adaptativa que destrói e repara soluções por meio de múltiplos operadores selecionados por pesos adaptativos, obtendo resultados de estado da arte no *Pickup and Delivery Problem with Time Windows* — um problema de alocação e sequenciamento com restrições de capacidade análogas às de carga horária. Em escala de pessoal (*workforce scheduling*), estratégias de reotimização por MIP com parte das decisões da incumbente fixadas são amplamente empregadas para resolver instâncias industriais de alocação de enfermeiros e funcionários a turnos (MANIEZZO; STÜTZLE; VOSS, 2021), domínio cuja estrutura — atribuição de recursos humanos respeitando restrições de disponibilidade e demanda mínima — guarda simetria direta com a distribuição de disciplinas docentes. Esses resultados consolidam a LNS com reotimização por sub-MIP como uma estratégia robusta e bem motivada para o PDD.

2.4.1 Large Neighborhood Search (LNS) e Reotimização por Sub-MIP

O Large Neighborhood Search (LNS) alterna etapas de *destroy* e *repair*: uma porção da solução corrente é liberada (ou removida) e, em seguida, o problema restrito é resolvido para reparar e potencialmente melhorar a solução. A principal vantagem é permitir movimentos de grande alcance, mantendo a viabilidade por meio de um procedimento de reparo eficaz (AHUJA et al., 2002; PISINGER; ROPKE, 2010). Uma implementação prática e amplamente adotada é usar um solucionador MIP na etapa de reparo, o que transforma o LNS em uma matheurística.

Na literatura, diferentes esquemas de reotimização podem ser combinados ao LNS. Em procedimentos construtivos do tipo *relax-and-fix* ou *fix-and-relax*, parte das variáveis é fixada, parte permanece inteira e outra parte tem a integralidade relaxada, produzindo uma sequência de subproblemas usados para construir soluções. Já em abordagens de melhoria do tipo *fix-and-optimize*, parte das variáveis da incumbente é fixada e um subconjunto permanece livre e inteiro para reotimização, mantendo-se a estrutura do modelo original por meio de limites ou restrições adicionais (PISINGER; ROPKE, 2010). Para o PDD, o mecanismo adotado neste trabalho se alinha a essa segunda classe: a vizinhança é definida a partir da incumbente, a maior parte das variáveis binárias é mantida fixa e um subconjunto de atribuições é liberado para um sub-MIP temporariamente restrito.

Esse esquema de fixação parcial de variáveis pode ser visto como uma forma estruturada de definir vizinhanças: fixa-se a maior parte das decisões da incumbente e libera-se um subconjunto selecionado para reotimização, normalmente sob um limite de tempo. Como as variáveis liberadas permanecem binárias, o reparo é efetuado por um sub-MIP que preserva as restrições do modelo original e busca recombinar decisões de modo consistente. Assim, o método proposto neste trabalho utiliza o SCIP (GAMRATH et al., 2020) como motor de reotimização em vizinhanças geradas a partir de uma solução incumbente.

Por fim, este trabalho se insere na linha de abordagens para o PDD que exploram tanto modelos exatos quanto estratégias de melhoria, dialogando com trabalhos recentes no problema (SILVA, 2024a) e com a motivação geral de matheurísticas de melhoria para problemas combinatórios de grande porte (SILVA, 2024b).

3 O Problema da Distribuição de Disciplinas (PDD)

3.1 Definição do Problema

O Problema da Distribuição de Disciplinas (PDD) consiste em atribuir professores a disciplinas de modo a respeitar restrições operacionais (carga horária mínima anual e máximas por semestre para cada professor, cobertura obrigatória de todas as disciplinas) e, simultaneamente, maximizar a qualidade da alocação. A qualidade de cada atribuição é mensurada por um peso que combina a preferência do professor pela disciplina e a compatibilidade entre a área de atuação do professor e a área da disciplina. Quando um professor é designado a uma disciplina fora de suas áreas de especialização, o modelo aplica uma penalidade (`area_penalty`), resultando em um coeficiente negativo na função objetivo.

Antes da resolução de cada instância, define-se *a priori* um conjunto finito $\mathcal{A}$ de áreas de conhecimento. Cada professor $i \in P$ é associado a um subconjunto $\mathcal{A}_i \subseteq \mathcal{A}$, que pode conter uma ou mais áreas nas quais ele possui competência. De modo análogo, cada disciplina $j \in C$ é associada a um subconjunto $\mathcal{A}_j \subseteq \mathcal{A}$ que descreve as áreas de conhecimento por ela abrangidas. Uma atribuição entre o professor i e a disciplina j é considerada compatível quando $\mathcal{A}_i \cap \mathcal{A}_j \neq \emptyset$, isto é, quando ambos compartilham pelo menos uma área. Essa classificação é um dado fixo da instância e não é alterada durante a otimização.

Formalmente, temos um conjunto de P professores e um conjunto de C disciplinas. Cada disciplina $j \in C$ possui uma carga horária associada ch_j (em horas/aula ou créditos) e deve ser atribuída a exatamente um professor. Cada professor $i \in P$ possui uma carga horária mínima anual $\min_i$ e cargas horárias máximas $\max_{i,1}$ e $\max_{i,2}$ para o primeiro e segundo semestres, respectivamente. O problema é NP-difícil. O PDD pode ser visto como um caso especial do *Generalized Assignment Problem* (GAP), no qual recursos (professores) devem ser atribuídos a tarefas (disciplinas) respeitando restrições de capacidade individuais (GAREY; JOHNSON, 1979). A NP-dificuldade do GAP é bem estabelecida: o problema contém o Problema da Mochila como caso particular, cujo caráter NP-difícil no sentido fraco é demonstrado em Garey e Johnson (1979). Adicionalmente, problemas de escalonamento de aulas — variantes estreitamente relacionadas ao PDD — foram demonstrados NP-completos por Even, Itai e Shamir (1976) por meio de redução a partir do problema de coloração de grafos. No contexto do PDD especificamente, Silva (2024a) observa que instâncias de médio porte já demandam tempo computacional expressivo

para o Branch-and-Bound puro, evidenciando na prática o comportamento esperado de um problema NP-difícil. Esses resultados justificam a adoção de abordagens heurísticas e matheurísticas, onde a modelagem como Programação Linear Inteira possibilita o uso de solvers exatos de forma integrada à estratégia de busca.

3.2 Modelo Matemático Exato

O PDD é formulado como um programa linear inteiro binário, instanciando a estrutura geral de PLI apresentada na Equação (2.1). Os elementos do modelo são organizados em três categorias — conjuntos e índices, parâmetros e variável de decisão — todos reunidos na Tabela 1.

Os **conjuntos** $P = \{1, \ldots, |P|\}$ e $C = \{1, \ldots, |C|\}$ representam, respectivamente, o conjunto de professores e o conjunto de disciplinas. Nas formulações a seguir, i e j são utilizados para representar, respectivamente, um professor e uma disciplina. O conjunto de disciplinas é ainda particionado em $C_1 \subseteq C$ (disciplinas ofertadas no primeiro semestre) e $C_2 \subseteq C$ (disciplinas ofertadas no segundo semestre), permitindo tratar separadamente as restrições de carga horária de cada período letivo.

Os **parâmetros** do modelo são: $w_{i,j}$, o peso associado à atribuição do professor i à disciplina j, que assume o valor da preferência declarada quando o professor possui competência na área da disciplina, ou $w_{i,j} = -\text{area\_penalty} < 0$ quando a disciplina está fora de suas áreas de especialização, penalizando alocações incompatíveis; ch_j, a carga horária em créditos da disciplina j; $\min_i$, a carga horária mínima anual obrigatória do professor i; e $\max_{i,1}$ e $\max_{i,2}$, as cargas horárias máximas permitidas para o professor i no primeiro e no segundo semestres, respectivamente.

A **variável de decisão** $x_{i,j} \in \{0, 1\}$ é binária: assume valor 1 se o professor i é designado à disciplina j, e 0 caso contrário.

Tabela 1 – Notação do modelo matemático do PDD.

Símbolo	**Descrição**
Conjuntos e Índices	
$P = \{1, \ldots, \lvert P\rvert\}$	Conjunto de professores; $i \in P$ denota um professor genérico
$C = \{1, \ldots, \lvert C\rvert\}$	Conjunto de disciplinas; $j \in C$ denota uma disciplina genérica
$\mathcal{A}$	Conjunto de áreas de conhecimento definido *a priori* para a instância
$\mathcal{A}_i$	Subconjunto de áreas de competência do professor i
$\mathcal{A}_j$	Subconjunto de áreas de conhecimento abrangidas pela disciplina j
$C_1 \subseteq C$	Subconjunto de disciplinas do primeiro semestre
$C_2 \subseteq C$	Subconjunto de disciplinas do segundo semestre
Parâmetros	
$w_{i,j}$	Peso da atribuição do professor i à disciplina j: assume o valor da preferência declarada se o professor tem competência na área da disciplina; caso contrário, $w_{i,j} = -\text{area\_penalty} < 0$
ch_j	Carga horária (créditos) da disciplina j
$\min_i$	Carga horária mínima anual do professor i
$\max_{i,1}$	Carga horária máxima do professor i no primeiro semestre
$\max_{i,2}$	Carga horária máxima do professor i no segundo semestre
Variável de Decisão	
$x_{i,j} \in \{0, 1\}$	1 se o professor i é designado à disciplina j; 0 caso contrário

O modelo completo do PDD é apresentado no Programa (3.1):

$$\max \; Z \quad = \sum_{i \in P} \sum_{j \in C} w_{i,j}\, x_{i,j} \tag{3.1a}$$

sujeito a:

$$\sum_{i \in P} x_{i,j} = 1, \qquad \forall\, j \in C, \tag{3.1b}$$

$$\sum_{j \in C} ch_j\, x_{i,j} \geq \min_i, \qquad \forall\, i \in P, \tag{3.1c}$$

$$\sum_{j \in C_1} ch_j\, x_{i,j} \leq \max_{i,1}, \qquad \forall\, i \in P, \tag{3.1d}$$

$$\sum_{j \in C_2} ch_j\, x_{i,j} \leq \max_{i,2}, \qquad \forall\, i \in P, \tag{3.1e}$$

$$x_{i,j} \in \{0, 1\}, \qquad \forall\, i \in P,\ j \in C. \tag{3.1f}$$

A função objetivo (3.1a) maximiza a soma ponderada de todas as atribuições rea-

lizadas. Quando o professor possui competência na área da disciplina, o coeficiente $w_{i,j}$ é positivo e reflete a preferência declarada, contribuindo para aumentar a qualidade da solução. Quando a atribuição ocorre fora da área de especialização, $w_{i,j} = -\text{area\_penalty}$ é negativo, penalizando explicitamente alocações incompatíveis e desincentivando sua ocorrência na solução final.

A restrição (3.1b) é a restrição de cobertura: garante que cada disciplina j seja atribuída a exatamente um professor, assegurando que nenhuma disciplina fique sem docente responsável. A restrição (3.1c) impõe que a soma das cargas horárias das disciplinas atribuídas ao professor i ao longo do ano seja pelo menos igual à sua carga mínima anual $\min_i$, respeitando obrigações contratuais. As restrições (3.1d) e (3.1e) estabelecem limites superiores para a carga de cada professor em cada semestre, evitando sobrecarga em um único período letivo. Por fim, o domínio (3.1f) restringe as variáveis ao conjunto binário $\{0, 1\}$, garantindo que cada par professor–disciplina seja ou inteiramente atribuído ou não atribuído.

3.3 Matheurística LNS Proposta

A matheurística desenvolvida neste trabalho baseia-se no Large Neighborhood Search (LNS) com reotimização por sub-MIP, em um esquema de fixação parcial de variáveis do tipo *fix-and-optimize*. A cada iteração, a heurística parte de uma solução incumbente $\bar{x}$ e define uma vizinhança ao fixar a maior parte das variáveis binárias e liberar um subconjunto controlado de atribuições para reotimização. A estratégia integra-se ao framework de heurísticas primais do SCIP.

3.3.1 Definição de Candidatos e Ordenação

A partir da solução incumbente $\bar{x}$, o algoritmo identifica todas as atribuições ativas (i.e., $\bar{x}_{i,j} = 1$), formando um conjunto de candidatos. Cada candidato corresponde a um par (professor, disciplina) e possui associado o valor `avgPreferenceWeight` do professor, uma métrica calculada conforme a Equação (3.2):

$$\text{avgPreferenceWeight}_i = \frac{\sum_{j \in J_i} w_{i,j}}{|J_i|} + \frac{|C|}{|J_i|}, \tag{3.2}$$

em que C é o conjunto de todas as disciplinas da instância; portanto, $|C|$ representa a quantidade total de disciplinas. Já $J_i \subseteq C$ é o conjunto de disciplinas para as quais o professor i declarou preferência, de modo que $|J_i|$ representa a quantidade de preferências desse professor. O primeiro termo representa a média aritmética dos pesos de preferência. O segundo termo ($|C|/|J_i|$) introduz um bônus inversamente proporcional à quantidade de preferências declaradas: para um mesmo conjunto de disciplinas, quanto menor for $|J_i|$, maior será o valor dessa fração. Assim, durante a construção do conjunto de candidatos,

professores com menos preferências recebem maior prioridade para reotimização, pois dispõem de menos alternativas de atribuição.

O conjunto de candidatos é ordenado segundo o parâmetro `lns_order`:

- `"decrescente"` (padrão): candidatos com maior `avgPreferenceWeight` aparecem primeiro, priorizando professores mais especializados (menor número de preferências). Estes professores possuem menos opções alternativas de atribuição, sendo candidatos naturais para reotimização;
- `"crescente"`: ordena em ordem ascendente, priorizando professores mais versáteis (maior número de preferências). Esta estratégia busca explorar a flexibilidade de realocação dos professores generalistas.

Seja K o número de candidatos identificados na solução incumbente. A construção dessa lista requer uma varredura dos pares professor–disciplina e possui custo $O(|P| \cdot |C|)$ no pior caso. Em seguida, a ordenação dos K candidatos pelo valor de `avgPreferenceWeight` possui custo $O(K \log K)$. Esses custos referem-se apenas à construção e à ordenação da vizinhança; a resolução do sub-MIP é tratada separadamente e não possui limite polinomial geral.

3.3.2 Destruição (Destroy)

Após a ordenação, o algoritmo seleciona os primeiros $\lfloor \text{nCands} \times \text{lns\_perc} \rfloor$ candidatos, onde `lns_perc` $\in (0, 1)$ é a taxa de destruição configurada pelo usuário. Para cada candidato selecionado, a variável $x_{i,j}$ correspondente é liberada (i.e., seu valor é desfixado), removendo a atribuição da solução incumbente. As demais variáveis permanecem fixadas nos valores da incumbente.

3.3.3 Reparo (Repair) via Sub-MIP

Com o subconjunto de variáveis liberado, o algoritmo resolve um sub-MIP no SCIP auxiliar (mantido em memória pela estrutura `heurdata->subscip`). O sub-MIP possui:

- Todas as restrições do problema original;
- Todas as variáveis binárias do modelo original, com limites ajustados conforme o vetor `fixed[]`;
- Limite de tempo configurado pelo parâmetro `lns_time`;
- Limite de função objetivo (`objlimit`) ajustado para parar assim que uma solução melhor que a incumbente for encontrada.

A resolução do sub-MIP busca reatribuir as disciplinas liberadas de modo a melhorar a função objetivo, respeitando todas as restrições de carga horária e cobertura. Se uma solução melhor é encontrada, ela é transferida para o SCIP principal e passa a ser a nova incumbente.

3.3.4 Integração com o SCIP

A heurística LNS é registrada como uma heurística primal do SCIP com prioridade e frequência configuráveis. A cada chamada, a heurística:

1. Verifica se existe uma solução incumbente viável;
2. Extrai os candidatos ativos e ordena conforme `lns_order`;
3. Libera uma fração `lns_perc` das atribuições;
4. Resolve o sub-MIP com limite de tempo `lns_time`;
5. Caso o sub-MIP retorne uma solução de melhor qualidade, adiciona a nova solução ao pool do SCIP principal.

A abordagem permite explorar vizinhanças de tamanho controlado, equilibrando a intensificação (ao focar em atribuições específicas) e a diversificação (ao permitir reatribuições significativas). Os parâmetros `lns_perc`, `lns_order` e `lns_time` oferecem controle sobre o trade-off entre qualidade da solução e tempo computacional, sendo objeto de análise experimental detalhada no próximo capítulo.

4 Resultados e Análise Computacional

Este capítulo apresenta os experimentos computacionais realizados para calibrar e avaliar a matheurística LNS proposta. Inicialmente, a Seção 4.1 descreve as instâncias de teste, o ambiente computacional e a metodologia de calibração adotada. Em seguida, a Seção 4.2 detalha a análise incremental dos três parâmetros da LNS — taxa de destruição, sentido de ordenação dos candidatos e tempo máximo do sub-MIP —, conduzida conforme a abordagem *one-factor-at-a-time*. Por fim, a Seção 4.3 apresenta o comparativo de desempenho entre a combinação de parâmetros selecionada para a LNS, o B&B puro, a heurística GRASP e a variante híbrida GRASP|LNS nas 42 instâncias sintéticas e em uma instância real, avaliando tempo de execução, gap de otimalidade, número de nós restantes na árvore de busca e execuções efetivas da LNS.

4.1 Instâncias e Ambiente de Testes

Os experimentos computacionais foram conduzidos utilizando um conjunto de 42 instâncias de teste geradas sinteticamente, abrangendo diferentes cenários do Problema da Distribuição de Disciplinas (PDD). As instâncias variam em tamanho (62 a 214 disciplinas), número de professores (22 a 69), complexidade das restrições de carga horária e distribuição de preferências entre áreas de conhecimento. Para o comparativo final, acrescentou-se uma instância real, identificada nas tabelas como `entrada_real`, composta por 224 disciplinas e 61 professores. Dessa forma, a avaliação geral considera 43 instâncias, mantendo a instância real distinguível das sintéticas e, simultaneamente, incorporada às métricas agregadas.

As instâncias foram produzidas pelo procedimento sintetizado no Algoritmo 4.1. A quantidade de tipos de disciplina foi fornecida como parâmetro e assumiu os valores 12, 16, 20, 25 e 32 nas instâncias utilizadas. Um tipo de disciplina representa uma categoria-base, como Cálculo ou Programação, e define uma área de conhecimento. Para cada tipo, o gerador cria entre 1 e 10 ofertas; cada oferta corresponde a uma disciplina individual do PDD, com carga horária própria, que deverá ser atribuída a um professor.

O fator de limite de professores foi mantido igual a 2 e utilizado somente para calcular a quantidade máxima de docentes que poderia ser gerada em cada área, dada por $\max\{1, \lfloor \text{número de ofertas}/2 \rfloor\}$. A quantidade efetiva de professores é sorteada entre 1 e esse limite; portanto, o fator não estabelece uma proporção exata entre disciplinas e professores. Em seguida, são gerados os limites de carga horária e as preferências docentes. O ajuste final da carga máxima assegura que a capacidade docente da área seja suficiente para cobrir a carga horária de suas ofertas.

Algoritmo 4.1 – Pseudocódigo para geração das instâncias sintéticas.

```
ENTRADA:
    numero_tipos <- valor de entrada em {12, 16, 20, 25, 32}
    fator_limite_professores <- 2

PARA cada tipo de disciplina FACA
    criar uma area de conhecimento exclusiva para o tipo
    numero_ofertas <- inteiro aleatorio entre 1 e 10

    PARA cada oferta do tipo FACA
        carga_horaria <- inteiro aleatorio entre 0 e 100
        registrar a oferta no primeiro semestre e na area do tipo
    FIM PARA

    carga_total_area <- soma das cargas horarias das ofertas
    maximo_professores <- max(1, piso(numero_ofertas /
                                      fator_limite_professores))
    numero_professores <- inteiro aleatorio entre 1 e
                          maximo_professores
    carga_media <- piso(carga_total_area / numero_professores)

    PARA cada professor da area FACA
        sortear carga minima entre 0 e carga_media
        sortear carga maxima entre a carga minima e carga_total_area
        numero_preferencias <- inteiro aleatorio entre 1 e
                               numero_ofertas
        selecionar, sem repeticao, esse numero de ofertas da area

        PARA cada oferta selecionada FACA
            peso_preferencia <- inteiro aleatorio entre 0 e 10
        FIM PARA
    FIM PARA

    SE a soma das cargas maximas nao superar carga_total_area ENTAO
        aumentar a carga maxima do ultimo professor ate que a soma
        supere carga_total_area
    FIM SE
FIM PARA

gravar totais, ofertas, professores e preferencias em arquivo CSV
```

Na calibração interna apresentada na Seção 4.2, foram selecionadas 10 instâncias dentre as 42 sintéticas disponíveis em cada etapa, mantendo o tempo total de experimentação viável. As mesmas instâncias foram utilizadas entre os cenários comparados dentro de cada etapa, assegurando comparabilidade direta dos parâmetros. Após a calibração, o comparativo final da Seção 4.3 foi reexecutado com todas as 42 instâncias sintéticas e com a instância real.

Todas as execuções foram realizadas com limite de tempo global de 400 segundos por instância, em uma máquina equipada com processador AMD Ryzen 5 5600G, 64 GB de memória RAM e sistema operacional Arch Linux. O solver utilizado foi o SCIP 7.0 (GAMRATH et al., 2020), integrado com as heurísticas LNS e GRASP desenvolvidas neste trabalho. A calibração dos parâmetros da heurística LNS foi conduzida de forma incremental, conforme a abordagem *one-factor-at-a-time* (OFAT) (MONTGOMERY, 2017): primeiro determinou-se a melhor taxa de destruição, em seguida o melhor sentido de ordenação e, finalmente, o melhor limite de tempo para o sub-MIP. Essa estratégia ajusta um parâmetro por vez, mantendo os demais fixados, o que permite isolar o efeito individual de cada fator e identificar a configuração mais adequada de forma sistemática. A escolha pelo OFAT foi motivada principalmente pelo custo computacional de uma calibração exaustiva: um projeto fatorial completo sobre os três parâmetros (4 níveis de taxa de destruição $\times$ 2 sentidos de ordenação $\times$ 4 níveis de tempo do sub-MIP $=$ 32 combinações), com 10 instâncias e limite de 400 segundos cada, demandaria até $32 \times 10 \times 400 = 128.000$ segundos ($\approx$ 35,6 horas) de execução, contra $10 \times 10 \times 400 = 40.000$ segundos ($\approx$ 11,1 horas) da abordagem OFAT — uma redução de aproximadamente 69% no esforço experimental. Reconhece-se que o OFAT não captura possíveis interações entre parâmetros, limitação discutida na Seção de conclusões. Contudo, os três parâmetros da LNS governam aspectos distintos e relativamente independentes do algoritmo — o tamanho da vizinhança (`lns_perc`), a estratégia de seleção de candidatos (`lns_order`) e o orçamento computacional por chamada (`lns_time`) —, o que reduz a probabilidade de interações fortes e torna a abordagem adequada para este contexto.

Para a seleção do melhor valor de cada parâmetro, adotou-se a seguinte hierarquia de critérios de desempate: (i) tempo de execução, (ii) gap de otimalidade e (iii) número de nós restantes na árvore de Branch-and-Bound (*nodes left*). Quando os critérios de maior prioridade apresentaram empate entre os cenários avaliados, o critério subsequente foi utilizado como desempate. Durante a calibração, a identificação do melhor ajuste considera exclusivamente os níveis do parâmetro em avaliação; o B&B puro é apresentado apenas como referência e não participa dessa classificação. A comparação direta entre o B&B puro e as abordagens heurísticas é realizada somente no comparativo final, após a calibração da LNS.

4.2 Análise Interna dos Parâmetros da LNS

4.2.1 Impacto da Taxa de Destruição (10%, 25%, 50% e 75%)

A primeira análise experimental investigou o impacto da taxa de destruição (`lns_perc`) na qualidade das soluções e no desempenho computacional. Foram avaliados quatro cenários: 10%, 25%, 50% e 75% de destruição, mantendo fixos o tempo máximo do LNS (30s por chamada), a ordenação (crescente) e o tempo total de execução (400s).

Tempo de Execução. A Tabela 2 apresenta os tempos de execução para as 10 instâncias selecionadas. A variante B&B puro (sem LNS) obteve o menor tempo total, com **623 segundos**, contra 753 segundos do melhor cenário LNS (75%). Esse resultado é esperado, pois o *overhead* introduzido pelas chamadas ao sub-MIP do LNS aumenta o tempo total, sobretudo em instâncias que já são resolvidas de forma ótima pelo Branch-and-Bound puro. Nove das dez instâncias foram resolvidas até a otimalidade por todos os cenários, e apenas a instância input7 atingiu o limite de tempo. Embora o B&B puro tenha sido claramente superior, as diferenças entre os quatro cenários LNS foram pequenas (753s a 756s de tempo total), não sendo suficientes para justificar a escolha de um cenário em detrimento dos demais.

Tabela 2 – Tempo de execução (segundos) para diferentes taxas de destruição do LNS. Cenário avaliado: ordenação crescente, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**10%**	**25%**	**50%**	**75%**	**Ajuste**
input2	167	52	4,04	17,07	17,66	16,98	17,04	50%
input5	214	69	113,03	145,27	145,04	145,71	143,63	75%
input6	79	24	1,07	2,47	2,40	2,33	2,22	75%
input7	84	29	400,02	400,02	400,02	400,02	400,02	Empate
input9	104	35	17,06	72,52	72,33	72,57	73,48	25%
input13	100	34	29,13	60,21	59,56	59,04	59,43	50%
input19	95	29	2,11	1,39	1,44	1,42	1,47	10%
input20	104	37	0,95	4,69	4,56	4,59	5,02	25%
input26	114	38	33,73	25,50	24,67	24,81	24,38	75%
input34	161	54	21,92	26,68	26,08	26,38	26,41	25%
Total	1.222	401	623	756	754	754	**753**	**75%**

Gap de Otimalidade. Em termos de gap, todos os cenários — incluindo o B&B puro — apresentaram desempenho equivalente (0,60% de gap total), com apenas a instância input7 permanecendo sem solução ótima comprovada (ver Tabela 3). Esse empate impediu o uso do gap como critério de seleção.

Nós Restantes (*Nodes Left*). Uma vez que o tempo de execução não apresentou diferenças significativas entre os cenários LNS e o gap resultou em empate absoluto, o critério de desempate utilizado foi a quantidade de nós restantes na árvore de Branch-and-Bound ao final da execução, métrica que reflete o progresso em direção à prova de

Tabela 3 – Gap de otimalidade (%) para diferentes taxas de destruição. Combinação de parâmetros mantida fixa: ordenação crescente, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**10%**	**25%**	**50%**	**75%**	**Ajuste**
input2	167	52	0,00	0,00	0,00	0,00	0,00	Empate
input5	214	69	0,00	0,00	0,00	0,00	0,00	Empate
input6	79	24	0,00	0,00	0,00	0,00	0,00	Empate
input7	84	29	0,60	0,60	0,60	0,60	0,60	Empate
input9	104	35	0,00	0,00	0,00	0,00	0,00	Empate
input13	100	34	0,00	0,00	0,00	0,00	0,00	Empate
input19	95	29	0,00	0,00	0,00	0,00	0,00	Empate
input20	104	37	0,00	0,00	0,00	0,00	0,00	Empate
input26	114	38	0,00	0,00	0,00	0,00	0,00	Empate
input34	161	54	0,00	0,00	0,00	0,00	0,00	Empate
Total	1.222	401	0,60	0,60	0,60	0,60	0,60	**Empate**

otimalidade. A Tabela 4 revela que o cenário com 75% de destruição alcançou o menor total de nós restantes: **58.921 nós**, comparado a 66.230 (B&B puro), 62.628 (10%), 60.540 (25%) e 61.056 (50%). Todos os cenários obtiveram resultado idêntico em 9 das 10 instâncias (resolvidas até a otimalidade), mas na instância crítica input7, a taxa de 75% demonstrou maior capacidade de exploração e poda da árvore. A Figura 1 apresenta visualmente a comparação dos totais, evidenciando a superioridade do cenário de 75%.

Tabela 4 – Número de nós restantes na árvore de Branch-and-Bound para diferentes taxas de destruição. Ajuste dos demais parâmetros: ordenação crescente, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**10%**	**25%**	**50%**	**75%**	**Ajuste**
input2	167	52	0	0	0	0	0	–
input5	214	69	0	0	0	0	0	–
input6	79	24	0	0	0	0	0	–
input7	84	29	66.230	62.628	60.540	61.056	**58.921**	**75%**
input9	104	35	0	0	0	0	0	–
input13	100	34	0	0	0	0	0	–
input19	95	29	0	0	0	0	0	–
input20	104	37	0	0	0	0	0	–
input26	114	38	0	0	0	0	0	–
input34	161	54	0	0	0	0	0	–
Total	1.222	401	66.230	62.628	60.540	61.056	**58.921**	**75%**

Conclusão Parcial. A taxa de destruição de **75%** foi selecionada para as análises subsequentes, pois apresentou a maior redução da árvore de busca na instância difícil avaliada. Para as instâncias testadas, esse resultado sugere que vizinhanças maiores podem permitir explorações mais amplas do espaço de soluções, reforçando a estratégia de intensificação adotada pela matheurística.

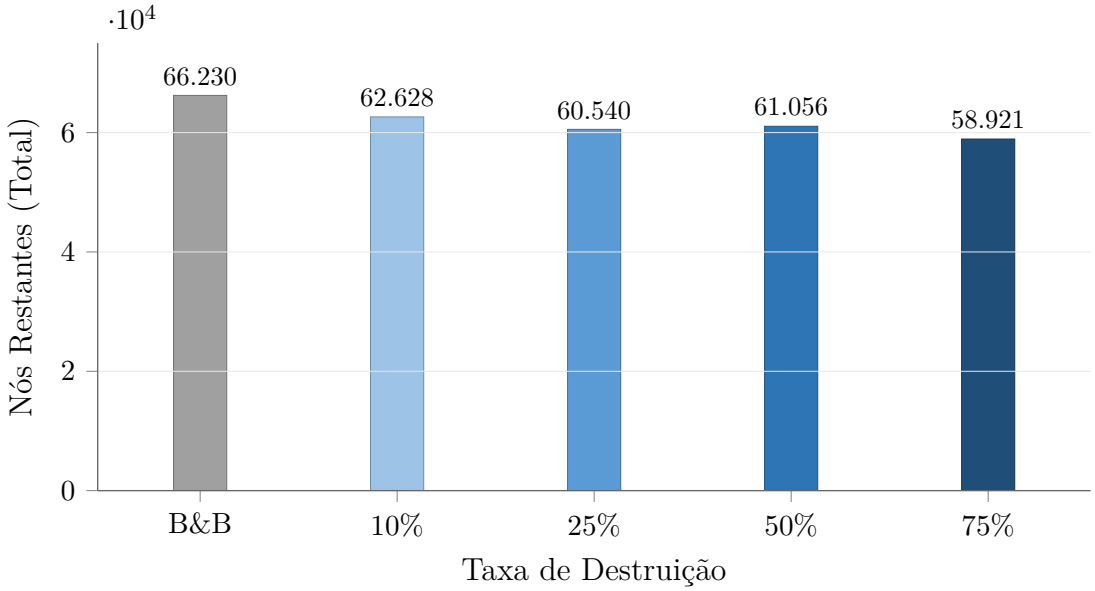


Figura 1 – Número total de nós restantes na árvore de Branch-and-Bound para diferentes taxas de destruição do LNS. O cenário com 75% apresentou o melhor desempenho, com 58.921 nós restantes.

4.2.2 Impacto do Sentido de Ordenação (Crescente vs. Decrescente)

Com a taxa de destruição fixada em 75%, investigou-se o impacto do sentido de ordenação dos candidatos pelo valor de `avgPreferenceWeight` ((3.2)). Conforme descrito no Capítulo 3, a ordenação crescente prioriza a liberação de professores mais versáteis (maior número de preferências declaradas), enquanto a decrescente prioriza professores mais especializados (menor número de preferências).

Tempo de Execução. A Tabela 5 revela desempenho temporal praticamente idêntico entre as duas estratégias: 1.803,48 segundos (crescente) versus 1.804,48 segundos (decrescente) no tempo total, com diferença de apenas 1 segundo. O B&B puro obteve o menor tempo total com **1.792,24 segundos**. Das 10 instâncias testadas, quatro atingiram o limite de tempo em ambas as variantes (input4, input15, input17 e input33), e as demais apresentaram variações inferiores a 1 segundo. O tempo, portanto, não permitiu diferenciar as estratégias de ordenação.

Gap de Otimalidade. Ambas as variantes mantiveram gap total idêntico de 1,62%, distribuído entre quatro instâncias não resolvidas de forma ótima (input4, input15, input17 e input33), com gaps individuais variando de 0,20% a 0,73% (ver Tabela 6). O empate no gap impediu, novamente, seu uso como critério de seleção.

Nós Restantes (*Nodes Left*). A métrica de nós restantes revelou vantagem para a ordenação crescente. A Tabela 7 mostra que a variante crescente resultou em **131.997 nós restantes** no total, enquanto a decrescente deixou 133.674 nós — uma redução de 1.677 nós (1,25%). A diferença se concentra em duas instâncias críticas: input15 (61.546 vs. 62.452 nós) e input33 (46.243 vs. 47.063 nós). A Figura 2 apresenta a comparação visual dos totais.

Tabela 5 – Tempo de execução (segundos) comparando ordenação crescente e decrescente. Cenário avaliado: 75% de destruição, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**Crescente**	**Decrescente**	**Ajuste**
input4	123	46	400,05	400,04	400,05	Crescente
input5	214	69	114,18	140,64	141,47	Crescente
input12	115	37	5,79	25,70	25,84	Crescente
input15	94	30	400,03	400,02	400,02	Empate
input16	89	28	3,07	0,72	0,72	Empate
input17	108	38	400,04	400,04	400,03	Decrescente
input20	104	37	0,90	4,40	4,32	Decrescente
input33	104	36	400,03	400,03	400,03	Empate
input37	152	53	68,00	31,61	31,62	Crescente
input38	62	22	0,16	0,28	0,28	Empate
Total	1.165	396	1.792,24	**1.803,48**	1.804,48	**Crescente**

Tabela 6 – Gap de otimalidade (%) comparando ordenação crescente e decrescente. Combinação de parâmetros mantida fixa: 75% de destruição, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**Crescente**	**Decrescente**	**Ajuste**
input4	123	46	0,20	0,20	0,20	Empate
input5	214	69	0,00	0,00	0,00	Empate
input12	115	37	0,00	0,00	0,00	Empate
input15	94	30	0,73	0,73	0,73	Empate
input16	89	28	0,00	0,00	0,00	Empate
input17	108	38	0,20	0,20	0,20	Empate
input20	104	37	0,00	0,00	0,00	Empate
input33	104	36	0,49	0,49	0,49	Empate
input37	152	53	0,00	0,00	0,00	Empate
input38	62	22	0,00	0,00	0,00	Empate
Total	1.165	396	1,62	1,62	1,62	**Empate**

Tabela 7 – Número de nós restantes comparando ordenação crescente e decrescente. Ajuste dos demais parâmetros: 75% de destruição, tempo LNS = 30s, limite global = 400s.

Instância	**Disciplinas**	**Profs**	**B&B**	**Crescente**	**Decrescente**	**Ajuste**
input4	123	46	19.797	19.541	19.507	Decrescente
input5	214	69	0	0	0	–
input12	115	37	0	0	0	–
input15	94	30	64.711	61.546	62.452	Crescente
input16	89	28	0	0	0	–
input17	108	38	4.673	4.667	4.652	Decrescente
input20	104	37	0	0	0	–
input33	104	36	50.279	46.243	47.063	Crescente
input37	152	53	0	0	0	–
input38	62	22	0	0	0	–
Total	1.165	396	139.460	**131.997**	133.674	**Crescente**

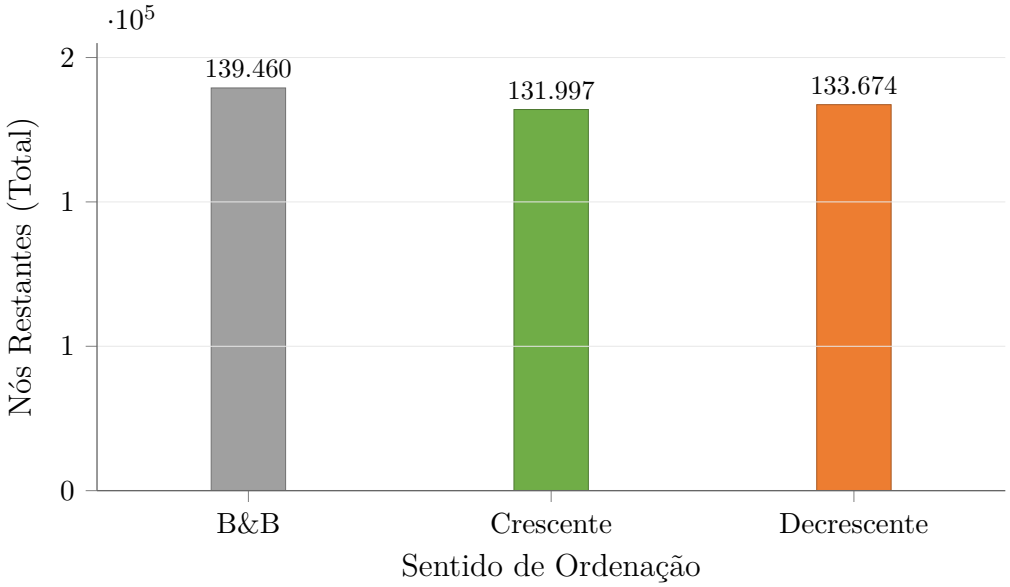


Figura 2 – Comparação do número total de nós restantes entre ordenação crescente e decrescente. A ordenação crescente (priorizando professores versáteis) obteve melhor desempenho com 131.997 nós restantes.

Interpretação. A superioridade da ordenação **crescente** sugere que, para o PDD, priorizar a realocação de professores mais versáteis (generalistas) oferece maior flexibilidade para o sub-MIP encontrar rearranjos que melhorem a solução global. Professores generalistas possuem preferências por um maior número de disciplinas, o que amplia o espaço de busca do sub-MIP e facilita a satisfação simultânea de múltiplas restrições de carga horária. A ordenação crescente foi, portanto, adotada para as análises subsequentes.

4.2.3 Impacto do Tempo Máximo do LNS (10s, 50s, 100s e 200s)

Com os parâmetros de taxa de destruição (75%) e ordenação (crescente) definidos, investigou-se o impacto do limite de tempo do sub-MIP (`lns_time`). Foram testados quatro valores: 10s, 50s, 100s e 200s, mantendo o limite global de 400 segundos por instância.

Tempo de Execução. A Tabela 8 apresenta resultados contraintuitivos: embora se esperasse que limites maiores de LNS aumentassem o tempo total, observou-se comportamento não monotônico. O tempo total foi de 1.629,17s (B&B puro), 1.647,01s (10s), 1.313,60s (50s), **1.039,68s (100s)** e 1.090,00s (200s). Destaca-se a instância input15, em que os cenários de 100s e 200s resolveram o problema em ∼77s, contra 400s nos demais. O ajuste de 100s obteve o menor tempo total, porém a diferença para 200s (50,32s) foi pequena, não permitindo uma conclusão unívoca por este critério.

Essa redução de tempo com limites maiores de LNS pode ser explicada pela capacidade de encontrar soluções de melhor qualidade mais cedo: uma incumbente mais forte permite podas mais agressivas na árvore de Branch-and-Bound, acelerando a convergência. Cenários com tempo insuficiente (10s) produzem melhorias marginais que não

Tabela 8 – Tempo de execução (segundos) para diferentes limites de tempo do sub-MIP. Cenário avaliado: 75% de destruição, ordenação crescente, limite global = 400s.

Inst.	**Disc.**	**Prof.**	**B&B**	**10s**	**50s**	**100s**	**200s**	**Ajuste**
input6	79	24	1,19	2,40	2,14	2,15	2,14	Empate
input14	99	28	14,21	22,31	68,47	118,44	168,12	10s
input15	94	30	400,03	400,02	400,02	76,31	77,09	100s
input21	104	31	6,98	3,02	3,04	3,05	3,10	10s
input22	133	41	400,05	400,04	400,04	400,04	400,04	Empate
input23	118	35	1,55	11,41	12,80	12,66	12,50	10s
input27	125	39	4,93	7,45	7,34	7,33	7,31	200s
input29	103	39	400,03	400,03	19,42	19,40	19,39	200s
input33	104	36	400,03	400,03	400,03	400,03	400,03	Empate
input38	62	22	0,16	0,28	0,28	0,28	0,28	Empate
Total	1,021	325	1.629,17	1.647,01	1.313,60	**1.039,68**	1.090,00	**100s**

compensam o *overhead* da heurística.

Gap de Otimalidade. A Tabela 9 mostra que os cenários de 100s e 200s empataram com gap total de **0,72%**, substancialmente inferior ao obtido pelo B&B puro e pelo LNS com 10s (ambos com 1,73%). O destaque vai para a instância input15, resolvida até a otimalidade (gap 0,00%) apenas pelos cenários de 100s e 200s, e para a instância input33, cujo gap caiu de 0,49% (B&B puro) para 0,24% com 100s e 200s. Entretanto, o empate entre 100s e 200s impediu o uso do gap como critério final.

Tabela 9 – Gap de otimalidade (%) para diferentes limites de tempo do sub-MIP. Combinação de parâmetros mantida fixa: 75% de destruição, ordenação crescente, limite global = 400s.

Inst.	**Disc.**	**Prof.**	**B&B**	**10s**	**50s**	**100s**	**200s**	**Ajuste**
input6	79	24	0,00	0,00	0,00	0,00	0,00	Empate
input14	99	28	0,00	0,00	0,00	0,00	0,00	Empate
input15	94	30	0,73	0,73	0,73	0,00	0,00	Empate
input21	104	31	0,00	0,00	0,00	0,00	0,00	Empate
input22	133	41	0,32	0,32	0,32	0,48	0,48	Empate
input23	118	35	0,00	0,00	0,00	0,00	0,00	Empate
input27	125	39	0,00	0,00	0,00	0,00	0,00	Empate
input29	103	39	0,19	0,19	0,00	0,00	0,00	Empate
input33	104	36	0,49	0,49	0,49	0,24	0,24	Empate
input38	62	22	0,00	0,00	0,00	0,00	0,00	Empate
Total	1.021	325	1,73	1,73	1,54	**0,72**	**0,72**	**Empate**

Nós Restantes (*Nodes Left*). A Tabela 10 foi o critério de desempate definitivo. O total de nós restantes foi: 135.986 (B&B puro), 144.674 (10s), 130.424 (50s), 51.935 (100s) e **33.539 (200s)**. O cenário de 200s reduziu os nós em 75,3% em relação ao B&B puro e em 76,8% em relação ao LNS com 10s. As instâncias input15 e input29 foram resolvidas completamente (0 nós restantes) pelos cenários de 100s e 200s, enquanto permaneceram com dezenas de milhares de nós nos cenários inferiores. Na instância input33,

o cenário de 200s deixou apenas 539 nós, contra 1.524 com 100s. A Figura 3 apresenta a comparação visual dos totais, evidenciando a clara superioridade do limite de 200s.

Tabela 10 – Número de nós restantes para diferentes limites de tempo do sub-MIP. Ajuste dos demais parâmetros: 75% de destruição, ordenação crescente, limite global = 400s.

Inst.	**Disc.**	**Prof.**	**B&B**	**10s**	**50s**	**100s**	**200s**	**Ajuste**
input6	79	24	0	0	0	0	0	–
input14	99	28	0	0	0	0	0	–
input15	94	30	54.355	63.056	59.819	0	0	Empate
input21	104	31	0	0	0	0	0	–
input22	133	41	21.528	29.161	25.100	50.411	33.000	50s
input23	118	35	0	0	0	0	0	–
input27	125	39	0	0	0	0	0	–
input29	103	39	12.643	4.347	0	0	0	Empate
input33	104	36	47.460	48.110	45.505	1.524	539	200s
input38	62	22	0	0	0	0	0	–
Total	1.021	325	135.986	144.674	130.424	51.935	**33.539**	**200s**

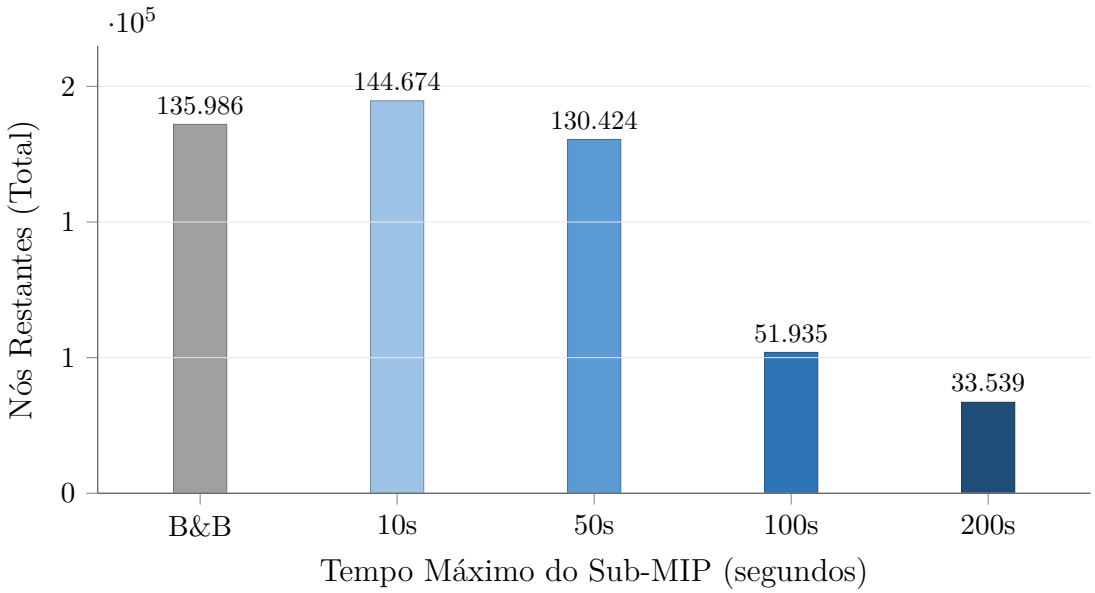


Figura 3 – Impacto do limite de tempo do sub-MIP no número de nós restantes. O cenário com 200s demonstrou o melhor desempenho, com 33.539 nós, representando uma redução de 75,3% em relação ao B&B puro.

Conclusão Parcial. Entre os cenários avaliados, o limite de **200 segundos** apresentou o melhor desempenho global, reduzindo a árvore de busca e resolvendo instâncias que os demais cenários não resolveram. Para as instâncias testadas, os resultados sugerem que investir mais tempo em cada chamada do LNS pode produzir soluções de maior qualidade e acelerar a convergência do Branch-and-Bound. Nesse contexto experimental, o *trade-off* entre intensidade (tempo por chamada) e frequência (número de chamadas) favoreceu a intensidade.

4.2.4 Resumo da Combinação de Parâmetros Selecionada

A Tabela 11 resume os parâmetros selecionados ao longo da análise incremental.

Tabela 11 – Combinação final de parâmetros da heurística LNS selecionada pela análise incremental.

Parâmetro	**Valores Testados**	**Selecionado**	**Critério Decisivo**
Taxa de destruição (`lns_perc`)	10%, 25%, 50%, 75%	75%	Nós restantes
Ordenação (`lns_order`)	Crescente, Decrescente	Crescente	Nós restantes
Tempo do sub-MIP (`lns_time`)	10s, 50s, 100s, 200s	200s	Nós restantes

Nota-se que, em todas as etapas, o tempo de execução não apresentou diferenças significativas entre os cenários LNS e o gap de otimalidade resultou em empate, sendo a métrica de nós restantes o critério de desempate definitivo. Esse padrão sugere que, para as instâncias avaliadas, o LNS atua predominantemente como acelerador da prova de otimalidade, reduzindo o esforço do Branch-and-Bound por meio de incumbentes de melhor qualidade.

4.3 Comparativo de Desempenho: LNS, GRASP e Modelo Exato

Com os parâmetros da LNS calibrados (75% de destruição, ordenação crescente e tempo de sub-MIP de 200s), realizou-se um comparativo final entre quatro variantes: (i) **B&B puro** (sem heurísticas primais customizadas), (ii) **LNS** (apenas a heurística LNS proposta), (iii) **GRASP** (apenas a heurística GRASP, com `max_iter=1`, $\alpha = 0{,}8$ e sem busca local (PEDROTTI; BARRETO, 2025)) e (iv) **GRASP|LNS** (ambas as heurísticas ativadas simultaneamente). As quatro variantes foram executadas sobre as mesmas 43 instâncias: 42 sintéticas e uma real. Nas tabelas detalhadas, a instância real é identificada explicitamente e destacada em cinza; seus resultados também integram os totais e as médias gerais.

4.3.1 Análise do Tempo de Execução

A Tabela 12 apresenta os tempos de execução por instância. Considerando o conjunto completo, o **B&B puro** obteve o menor tempo total, com **5.232,81 segundos**, seguido por LNS (6.102,82s), GRASP|LNS (6.176,44s) e GRASP (6.989,55s). Em relação ao B&B puro, os tempos totais foram 16,6% maiores para o LNS, 18,0% maiores para o GRASP|LNS e 33,6% maiores para o GRASP. O resultado evidencia que o custo das heurísticas não foi compensado por reduções sistemáticas no tempo de resolução ao longo das 43 instâncias.

Tabela 12 – Tempo de execução (segundos) nas 42 instâncias sintéticas e na instância real, destacada em cinza.

Instância	**Disc.**	**Profs**	**B&B**	**LNS**	**GRASP**	**GRASP|LNS**	**Melhor**
input1	194	51	13,87	14,04	55,33	32,23	B&B
input2	167	52	5,07	5,97	19,07	9,25	B&B
input3	180	50	3,15	3,12	11,20	11,26	LNS
input4	123	46	11,45	208,94	38,47	228,73	B&B
input5	214	69	33,99	212,68	122,70	253,37	B&B
input6	79	24	0,73	2,00	1,79	0,49	GRASP|LNS
input7	84	29	400,02	400,02	400,02	400,02	Empate
input8	166	46	400,06	209,41	400,07	231,67	LNS
input9	104	35	15,30	55,09	50,07	54,58	B&B
input10	113	31	1,11	1,22	3,39	0,73	GRASP|LNS
input11	100	31	3,41	1,14	0,95	0,57	GRASP|LNS
input12	115	37	4,76	6,17	10,93	1,76	GRASP|LNS
input13	100	34	98,87	242,55	252,49	45,39	GRASP|LNS
input14	99	28	54,35	66,69	131,38	139,80	B&B
input15	94	30	45,62	113,87	400,03	58,63	B&B
input16	89	28	0,24	0,34	0,46	0,37	B&B
input17	108	38	400,04	400,04	400,04	400,04	Empate
input18	121	38	20,21	3,00	7,96	4,32	LNS
input19	95	29	1,52	1,25	2,40	0,72	GRASP|LNS
input20	104	37	9,56	19,69	20,31	19,71	B&B
input21	104	31	8,24	4,84	18,14	7,58	LNS
input22	133	41	400,05	400,05	400,05	400,05	Empate
input23	118	35	1,32	1,78	2,87	1,07	GRASP|LNS
input24	117	33	40,76	400,03	400,04	400,03	B&B
input25	107	34	5,43	39,20	9,78	38,95	B&B
input26	114	38	400,04	400,04	400,04	400,04	Empate
input27	125	39	6,44	8,23	19,44	11,18	B&B
input28	114	39	78,83	400,04	400,04	400,04	B&B
input29	103	39	4,19	9,01	8,60	50,69	B&B
input30	109	36	10,68	2,77	19,29	3,20	LNS
input31	113	40	1,55	1,48	2,86	1,87	LNS
input32	97	36	400,03	400,03	400,03	400,03	Empate
input33	104	36	400,03	400,03	400,03	400,04	Empate
input34	161	54	4,33	9,28	12,56	16,99	B&B
input35	169	53	400,08	400,08	400,08	400,09	Empate
input36	185	60	5,10	6,64	15,71	14,78	B&B
input37	152	53	290,53	40,14	400,07	113,44	LNS
input38	62	22	0,18	0,45	0,65	0,41	B&B
input39	130	43	53,71	12,11	146,95	15,22	LNS
input40	126	42	400,05	400,05	400,05	400,05	Empate
input41	137	42	400,05	400,05	400,05	400,05	Empate
input42	142	35	1,55	5,89	2,98	6,91	B&B
entrada_real (Real)	224	61	396,31	393,38	400,14	400,12	LNS
Total sintético	–	–	4.836,50	5.709,44	6.589,40	5.776,32	B&B
Total geral	–	–	**5.232,81**	**6.102,82**	**6.989,55**	**6.176,44**	**B&B**

Na instância real, o LNS apresentou o menor tempo, com 393,38s, seguido pelo B&B puro, com 396,31s, uma diferença de 2,93s (0,74%). GRASP e GRASP|LNS atingiram aproximadamente 400s. Portanto, embora o B&B puro tenha sido mais eficiente no

agregado, o LNS obteve pequena vantagem temporal no caso real avaliado. A Figura 4 sintetiza os tempos totais.

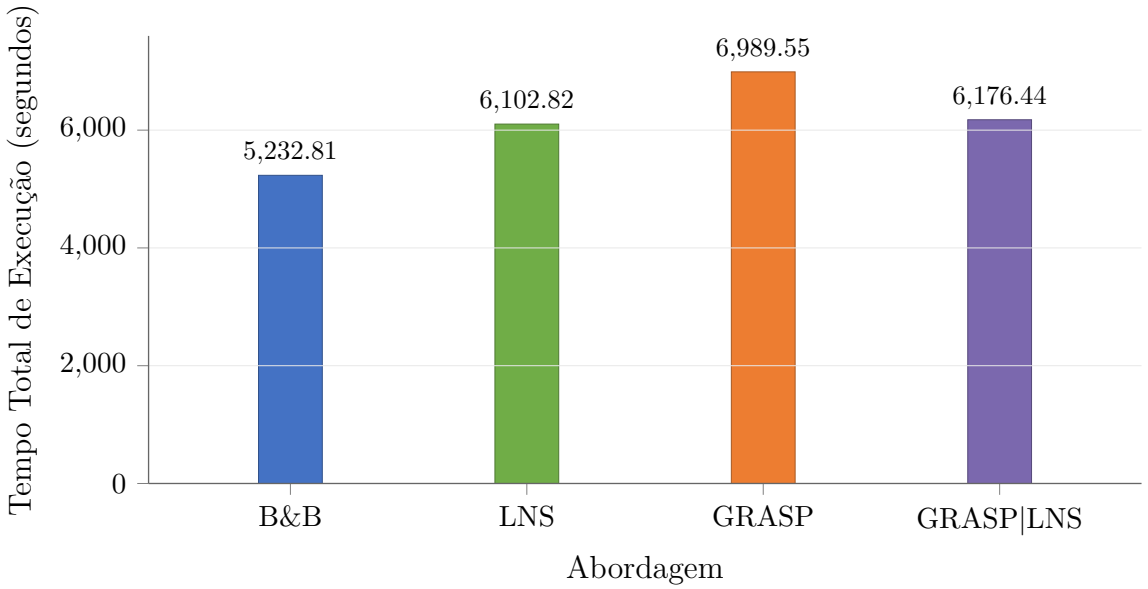


Figura 4 – Tempo total de execução nas 42 instâncias sintéticas e na instância real. O B&B puro apresentou o menor total.

4.3.2 Análise do Gap de Otimalidade

A Tabela 13 apresenta o gap de otimalidade em cada instância. Como gaps de instâncias distintas não constituem uma única taxa acumulável, a comparação agregada utiliza o **gap médio percentual**, acompanhado da quantidade de instâncias resolvidas até a otimalidade. O B&B puro obteve o menor gap médio, 0,3473%, seguido por LNS (0,4682%), GRASP|LNS (0,5420%) e GRASP (0,7099%). B&B puro e LNS resolveram 32 das 43 instâncias até a otimalidade, enquanto GRASP|LNS resolveu 31 e GRASP, 28.

Tabela 13 – Gap de otimalidade (%) nas 42 instâncias sintéticas e na instância real, destacada em cinza.

Instância	Disc.	Profs	B&B	LNS	GRASP	GRASP|LNS	Melhor
input1	194	51	0,0000	0,0000	0,0000	0,0000	Empate
input2	167	52	0,0000	0,0000	0,0000	0,0000	Empate
input3	180	50	0,0000	0,0000	0,0000	0,0000	Empate
input4	123	46	0,0000	0,0000	0,0000	0,0000	Empate
input5	214	69	0,0000	0,0000	0,0000	0,0000	Empate
input6	79	24	0,0000	0,0000	0,0000	0,0000	Empate
input7	84	29	0,2994	1,8237	0,2994	1,8237	Empate
input8	166	46	0,1592	0,0000	0,1592	0,0000	Empate
input9	104	35	0,0000	0,0000	0,0000	0,0000	Empate
input10	113	31	0,0000	0,0000	0,0000	0,0000	Empate
input11	100	31	0,0000	0,0000	0,0000	0,0000	Empate
input12	115	37	0,0000	0,0000	0,0000	0,0000	Empate
input13	100	34	0,0000	0,0000	0,0000	0,0000	Empate
input14	99	28	0,0000	0,0000	0,0000	0,0000	Empate

Continua na próxima página.

Tabela 13 – continuação

Instância	Disc.	Profs	B&B	LNS	GRASP	GRASP|LNS	Melhor
input15	94	30	0,0000	0,0000	0,2410	0,0000	Empate
input16	89	28	0,0000	0,0000	0,0000	0,0000	Empate
input17	108	38	0,3893	1,9923	2,1963	1,9923	B&B
input18	121	38	0,0000	0,0000	0,0000	0,0000	Empate
input19	95	29	0,0000	0,0000	0,0000	0,0000	Empate
input20	104	37	0,0000	0,0000	0,0000	0,0000	Empate
input21	104	31	0,0000	0,0000	0,0000	0,0000	Empate
input22	133	41	4,8414	4,8414	4,8414	4,8414	Empate
input23	118	35	0,0000	0,0000	0,0000	0,0000	Empate
input24	117	33	3,6163	0,2008	3,3769	0,2008	Empate
input25	107	34	0,0000	0,0000	0,0000	0,0000	Empate
input26	114	38	0,2079	5,2402	5,2402	5,2402	B&B
input27	125	39	0,0000	0,0000	0,0000	0,0000	Empate
input28	114	39	0,0000	0,1805	0,1805	0,1805	B&B
input29	103	39	0,0000	0,0000	0,0000	0,0000	Empate
input30	109	36	0,0000	0,0000	0,0000	0,0000	Empate
input31	113	40	0,0000	0,0000	0,0000	0,0000	Empate
input32	97	36	0,2597	0,2597	1,8470	0,2597	Empate
input33	104	36	0,2427	0,4866	0,7317	0,2427	Empate
input34	161	54	0,0000	0,0000	0,0000	0,0000	Empate
input35	169	53	0,7849	0,1560	1,1024	0,1560	Empate
input36	185	60	0,0000	0,0000	0,0000	0,0000	Empate
input37	152	53	0,0000	0,0000	2,0043	0,0000	Empate
input38	62	22	0,0000	0,0000	0,0000	0,0000	Empate
input39	130	43	0,0000	0,0000	0,0000	0,0000	Empate
input40	126	42	0,2193	0,2193	0,2193	0,2193	Empate
input41	137	42	3,9139	4,7337	4,7337	4,7337	B&B
input42	142	35	0,0000	0,0000	0,0000	0,0000	Empate
entrada_real (Real)	224	61	0,0000	0,0000	3,3528	3,4147	Empate
Média sintética	–	–	0,3556	0,4794	0,6470	0,4736	B&B
Média geral	–	–	**0,3473**	**0,4682**	**0,7099**	**0,5420**	**B&B**

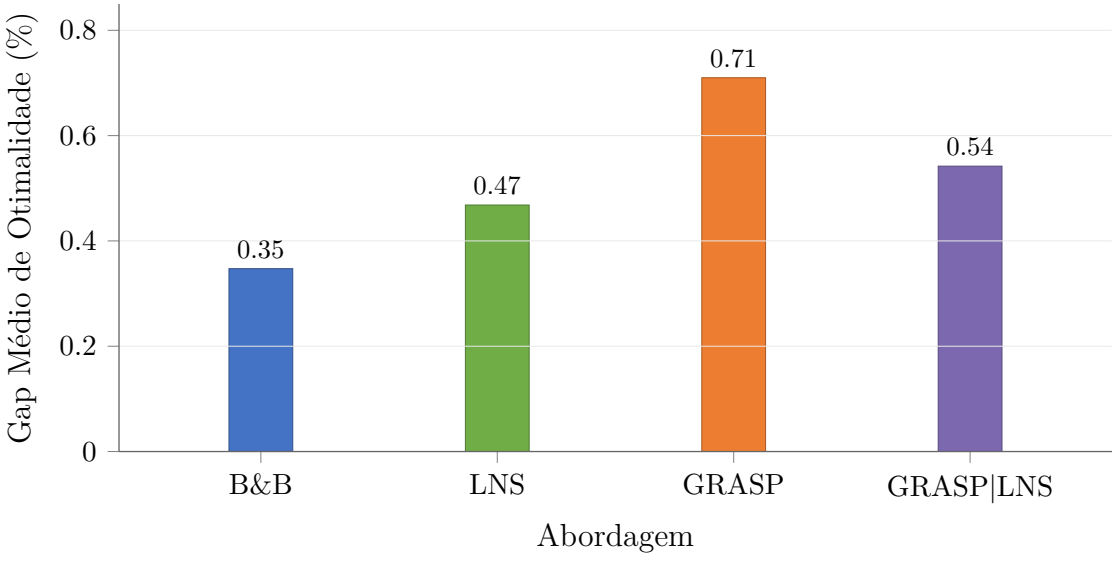


Figura 5 – Gap médio de otimalidade nas 42 instâncias sintéticas e na instância real. O B&B puro obteve o menor valor médio.

Na instância real, B&B puro e LNS alcançaram gap de 0,0000%, ao passo que GRASP encerrou com 3,3528% e GRASP|LNS com 3,4147%. Esse resultado reforça que a presença das duas heurísticas em conjunto não assegura melhoria da qualidade da incumbente ou da prova de otimalidade. A Figura 5 apresenta os gaps médios das 43 instâncias.

4.3.3 Análise dos Nós Restantes

A Tabela 14 mostra que a variante **GRASP|LNS** obteve o menor total de nós restantes, com **187.673 nós**, contra 416.125 do B&B puro, 462.967 do LNS e 467.336 do GRASP. A redução da variante híbrida foi de 54,9% em relação ao B&B puro, 59,5% em relação ao LNS e 59,8% em relação ao GRASP. Embora não tenha apresentado o menor gap médio nem o menor tempo total, a combinação GRASP|LNS avançou mais na exploração e poda das árvores que permaneceram abertas.

Tabela 14 – Número de nós restantes nas 42 instâncias sintéticas e na instância real, destacada em cinza.

Instância	**Disc.**	**Profs**	**B&B**	**LNS**	**GRASP**	**GRASP|LNS**	**Melhor**
input1	194	51	0	0	0	0	Empate
input2	167	52	0	0	0	0	Empate
input3	180	50	0	0	0	0	Empate
input4	123	46	0	0	0	0	Empate
input5	214	69	0	0	0	0	Empate
input6	79	24	0	0	0	0	Empate
input7	84	29	1.405	69.086	2.027	37.759	B&B
input8	166	46	7.283	0	4.552	0	Empate
input9	104	35	0	0	0	0	Empate
input10	113	31	0	0	0	0	Empate
input11	100	31	0	0	0	0	Empate
input12	115	37	0	0	0	0	Empate
input13	100	34	0	0	0	0	Empate
input14	99	28	0	0	0	0	Empate
input15	94	30	0	0	2.843	0	Empate
input16	89	28	0	0	0	0	Empate
input17	108	38	34.042	64.524	52.649	31.854	GRASP|LNS
input18	121	38	0	0	0	0	Empate
input19	95	29	0	0	0	0	Empate
input20	104	37	0	0	0	0	Empate
input21	104	31	0	0	0	0	Empate
input22	133	41	152.910	91.398	56.289	28.601	GRASP|LNS
input23	118	35	0	0	0	0	Empate
input24	117	33	14.077	3.083	66.822	2.147	GRASP|LNS
input25	107	34	0	0	0	0	Empate
input26	114	38	3.853	79.390	64.343	33.110	B&B
input27	125	39	0	0	0	0	Empate
input28	114	39	0	6.138	3.895	8.344	B&B
input29	103	39	0	0	0	0	Empate
input30	109	36	0	0	0	0	Empate
input31	113	40	0	0	0	0	Empate
input32	97	36	2.327	1.797	74.630	3.380	LNS
input33	104	36	2.514	36.423	25.783	3.287	B&B

Continua na próxima página.

Tabela 14 – continuação

Instância	Disc.	Profs	B&B	LNS	GRASP	GRASP|LNS	Melhor
input34	161	54	0	0	0	0	Empate
input35	169	53	21.909	5.593	12.876	4.965	GRASP|LNS
input36	185	60	0	0	0	0	Empate
input37	152	53	0	0	29.896	0	Empate
input38	62	22	0	0	0	0	Empate
input39	130	43	0	0	0	0	Empate
input40	126	42	1.453	1.267	1.118	775	GRASP|LNS
input41	137	42	174.352	104.268	57.079	27.989	GRASP|LNS
input42	142	35	0	0	0	0	Empate
entrada_real (Real)	224	61	0	0	12.534	5.462	Empate
Total sintético	–	–	416.125	462.967	454.802	182.211	GRASP|LNS
Total geral	–	–	**416.125**	**462.967**	**467.336**	**187.673**	**GRASP|LNS**

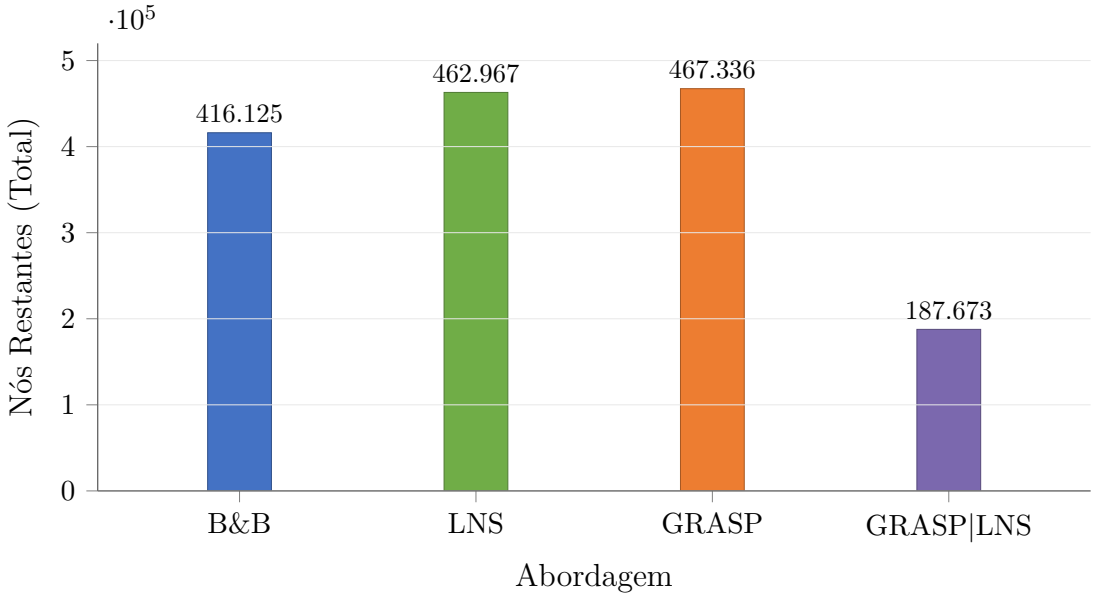


Figura 6 – Total de nós restantes nas 42 instâncias sintéticas e na instância real. A variante GRASP|LNS apresentou o menor total.

Na instância real, B&B puro e LNS encerraram a árvore de busca sem nós restantes. GRASP deixou 12.534 nós e GRASP|LNS, 5.462, mostrando que a combinação com LNS reduziu a árvore remanescente do GRASP, mas não foi suficiente para provar a otimalidade dentro do limite adotado.

4.3.4 Análise das Execuções da LNS

A Tabela 15 apresenta o número de execuções efetivas da LNS. Essa métrica é aplicável apenas às variantes LNS e GRASP|LNS; por esse motivo, B&B puro e GRASP são indicados por "–". Nas 42 instâncias sintéticas, foram registradas 88 execuções no LNS e 80 no GRASP|LNS. Na instância real, ocorreram 4 execuções no LNS e 10 no GRASP|LNS, resultando em totais gerais de 92 e 90, respectivamente.

Tabela 15 – Número de execuções efetivas da LNS nas 42 instâncias sintéticas e na instância real. O símbolo – indica que a abordagem não utiliza LNS; a instância real aparece destacada.

Instância	**Disc.**	**Profs**	**B&B**	**LNS**	**GRASP**	**GRASP|LNS**
input1	194	51	–	2	–	2
input2	167	52	–	1	–	1
input3	180	50	–	0	–	0
input4	123	46	–	2	–	2
input5	214	69	–	2	–	2
input6	79	24	–	1	–	1
input7	84	29	–	7	–	7
input8	166	46	–	1	–	1
input9	104	35	–	1	–	2
input10	113	31	–	1	–	1
input11	100	31	–	1	–	1
input12	115	37	–	2	–	1
input13	100	34	–	6	–	2
input14	99	28	–	1	–	1
input15	94	30	–	1	–	2
input16	89	28	–	2	–	1
input17	108	38	–	2	–	2
input18	121	38	–	1	–	1
input19	95	29	–	2	–	1
input20	104	37	–	1	–	1
input21	104	31	–	2	–	2
input22	133	41	–	1	–	1
input23	118	35	–	1	–	1
input24	117	33	–	3	–	3
input25	107	34	–	1	–	1
input26	114	38	–	1	–	1
input27	125	39	–	1	–	1
input28	114	39	–	3	–	2
input29	103	39	–	1	–	1
input30	109	36	–	1	–	1
input31	113	40	–	1	–	1
input32	97	36	–	5	–	3
input33	104	36	–	2	–	2
input34	161	54	–	2	–	2
input35	169	53	–	2	–	2
input36	185	60	–	2	–	2
input37	152	53	–	2	–	2
input38	62	22	–	1	–	1
input39	130	43	–	2	–	2
input40	126	42	–	15	–	15
input41	137	42	–	1	–	1
input42	142	35	–	1	–	1
entrada_real (Real)	224	61	–	4	–	10
Subtotal sintético	–	–	–	88	–	80
Total geral	–	–	–	**92**	–	**90**

O número de execuções representa a frequência com que o mecanismo de reotimização foi efetivamente acionado, e não uma medida direta de qualidade. Na instância real, por exemplo, o GRASP|LNS executou a LNS mais vezes que o LNS isolado, mas

terminou com gap positivo; portanto, mais execuções não implicaram necessariamente melhor convergência.

4.3.5 Discussão dos Resultados

O comparativo completo não identifica uma abordagem dominante em todas as métricas. A Tabela 16 sintetiza os indicadores das 43 instâncias, incluindo a instância real.

Tabela 16 – Resumo comparativo das quatro variantes nas 42 instâncias sintéticas e na instância real.

Métrica	**B&B puro**	**LNS**	**GRASP**	**GRASP|LNS**
Tempo total (s)	**5.232,81**	6.102,82	6.989,55	6.176,44
Tempo médio (s)	**121,69**	141,93	162,55	143,64
Gap médio (%)	**0,3473**	0,4682	0,7099	0,5420
Instâncias ótimas	**32/43**	**32/43**	28/43	31/43
Nós restantes (total)	416.125	462.967	467.336	**187.673**
Execuções LNS	–	92	–	90

A análise consolidada permite destacar quatro observações:

1. **O B&B puro foi mais eficiente no agregado.** A abordagem exata apresentou o menor tempo total e o menor gap médio, além de empatar com o LNS no maior número de instâncias ótimas. Para o conjunto completo, o custo adicional das heurísticas não produziu ganho temporal global.

2. **O GRASP|LNS promoveu maior redução das árvores abertas.** A variante híbrida deixou menos da metade dos nós restantes do B&B puro. Esse avanço, contudo, não se converteu no menor gap médio nem no menor tempo total, demonstrando que o número de nós restantes deve ser interpretado em conjunto com as demais métricas.

3. **O LNS apresentou resultado competitivo na instância real.** LNS e B&B puro provaram a otimalidade e encerraram a árvore, com vantagem de 2,93s para o LNS. Trata-se de evidência favorável à matheurística no caso real estudado, mas insuficiente para generalização a outros contextos institucionais.

4. **O GRASP isolado apresentou o desempenho agregado menos favorável, mas contribuiu quando combinado com a LNS.** Em relação ao GRASP isolado, a variante GRASP|LNS reduziu o tempo total em 11,6%, diminuiu o gap médio de 0,7099% para 0,5420%, aumentou de 28 para 31 o número de instâncias ótimas e reduziu em 59,8% os nós restantes. Esses ganhos indicam complementaridade entre a diversificação do GRASP e a intensificação da LNS, embora não demonstrem superioridade global da combinação.

5 Conclusão

Este trabalho propôs, implementou e avaliou uma matheurística baseada em Large Neighborhood Search (LNS) com reotimização por sub-MIP, em um esquema do tipo *fix-and-optimize*, integrada como heurística primal ao solver SCIP 7.0, para o Problema da Distribuição de Disciplinas (PDD). O problema foi modelado como um programa linear inteiro binário, cuja função objetivo maximiza a qualidade da atribuição de disciplinas a professores, respeitando restrições de carga horária mínima anual e máxima semestral.

A calibração dos parâmetros internos da LNS foi conduzida via abordagem *one-factor-at-a-time*, avaliando três parâmetros: taxa de destruição, sentido de ordenação dos candidatos e tempo máximo do sub-MIP. Em todas as etapas, a métrica de nós restantes na árvore de Branch-and-Bound foi o critério decisivo, uma vez que tempo de execução e gap de otimalidade não apresentaram diferenças significativas entre os cenários avaliados. Para as instâncias testadas, a combinação de parâmetros selecionada — 75% de destruição, ordenação crescente por `avgPreferenceWeight` e 200 segundos de sub-MIP — esteve associada a incumbentes de melhor qualidade e a uma convergência mais rápida do Branch-and-Bound.

No comparativo final, foram avaliadas quatro abordagens: B&B puro, LNS, GRASP e a variante híbrida GRASP|LNS. O conjunto experimental reuniu todas as 42 instâncias sintéticas e uma instância real. Os resultados não indicaram uma abordagem dominante em todas as métricas. O B&B puro obteve o menor tempo total (5.232,81s) e o menor gap médio (0,3473%). A variante GRASP|LNS apresentou o menor total de nós restantes (187.673), uma redução de 54,9% em relação ao B&B puro. B&B puro e LNS empataram no número de instâncias resolvidas até a otimalidade, com 32 de 43. O GRASP isolado apresentou o maior tempo total, o maior gap médio e o menor número de instâncias ótimas.

Na instância real, composta por 224 disciplinas e 61 professores, LNS e B&B puro provaram a otimalidade e encerraram a árvore de busca. O LNS foi 2,93s mais rápido, enquanto GRASP e a variante híbrida GRASP|LNS atingiram o limite temporal com gaps positivos. Esse resultado demonstra desempenho competitivo do LNS no caso real avaliado, mas uma única instância não permite generalizar essa vantagem para outras instituições ou configurações do PDD.

Os resultados obtidos mostram que a integração de heurísticas ao Branch-and-Bound produz efeitos distintos conforme a métrica observada. A LNS não superou globalmente o método exato, mas permaneceu competitiva na quantidade de soluções ótimas e na instância real. O GRASP mostrou-se especialmente relevante como mecanismo comple-

mentar à LNS: enquanto sua construção aleatorizada favorece a diversificação e a geração de incumbentes em diferentes regiões do espaço de busca, a LNS realiza a intensificação por meio da reotimização de grandes vizinhanças. Essa complementaridade apareceu nos resultados. Em relação ao GRASP isolado, a variante GRASP|LNS reduziu o tempo total em 11,6%, diminuiu o gap médio de 0,7099% para 0,5420%, aumentou de 28 para 31 o número de instâncias ótimas e reduziu em 59,8% os nós restantes. Na instância real, a combinação também reduziu os nós remanescentes do GRASP de 12.534 para 5.462. Por outro lado, a variante híbrida não superou a LNS isolada em tempo, gap médio ou quantidade de ótimos; portanto, os resultados evidenciam um potencial de cooperação ainda não plenamente explorado, e não uma superioridade global da combinação. O trabalho cumpriu os objetivos propostos: modelagem do problema como PLI, implementação de uma heurística LNS do tipo *fix-and-optimize* no SCIP, análise do impacto dos parâmetros internos e comparação abrangente com abordagens alternativas.

5.1 Limitações

As principais limitações deste trabalho são:

1. A calibração *one-factor-at-a-time* ajusta os parâmetros de forma sequencial, sem explorar possíveis interações entre eles. Uma análise fatorial completa ou métodos de *design of experiments* poderiam revelar combinações mais eficientes.

2. A avaliação incluiu apenas uma instância real. Embora ela complemente as 42 instâncias sintéticas e permita observar o comportamento das abordagens em um caso aplicado, não representa a diversidade de regras, preferências e dimensões encontrada entre instituições de ensino.

3. A variante do GRASP utilizada na comparação (`max_iter=1`, sem busca local) representa uma versão simplificada da heurística. Embora a combinação GRASP|LNS tenha melhorado todas as métricas agregadas em relação ao GRASP isolado, essa configuração limitada pode ter impedido que a variante híbrida explorasse plenamente a complementaridade entre diversificação e intensificação.

4. O limite de tempo global fixo de 400 segundos por instância não permite avaliar o comportamento das abordagens sob restrições temporais mais rígidas ou mais flexíveis.

5.2 Trabalhos Futuros

Como direções para trabalhos futuros, destacam-se:

1. Investigar estratégias adaptativas para os parâmetros do LNS, ajustando dinamicamente a taxa de destruição e o tempo do sub-MIP com base no progresso da busca.

2. Avaliar a matheurística em múltiplas instâncias reais, provenientes de diferentes instituições e períodos acadêmicos, incluindo problemas de maior porte e restrições adicionais como preferências de horário e indisponibilidades.

3. Explorar critérios alternativos de seleção de candidatos à destruição, como métricas baseadas na contribuição marginal de cada atribuição à função objetivo.

4. Investigar mecanismos de equidade e de resistência à declaração estratégica de preferências. Como a maximização da qualidade total pode concentrar ganhos em parte dos professores, um docente poderia declarar preferência por poucas disciplinas para tentar aumentar a prioridade de suas escolhas. Formulações multiobjetivo ou do tipo max-min, combinadas com informações como qualificação, histórico de atribuições ou limites mínimos de opções aceitáveis, podem reduzir essa vantagem e equilibrar a satisfação entre os professores.

5. Aprimorar e calibrar o GRASP antes de uma nova avaliação da variante híbrida, considerando um número maior de iterações, diferentes valores de α, busca local, múltiplas construções com sementes controladas e estratégias reativas. A redução de tempo, gap e nós restantes observada em relação ao GRASP isolado indica que essa integração é uma direção promissora.

6. Comparar o desempenho com solvers comerciais (Gurobi, CPLEX) e com outras metaheurísticas, como Busca Tabu e Algoritmos Genéticos.

Referências Bibliográficas

AHUJA, R. K. et al. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, v. 123, n. 1-3, p. 75–102, 2002. Disponível em: https://www.sciencedirect.com/science/article/pii/S0166218X01003389. Citado na página 15.

AL-YAKOOB, S.; SHERALI, H. A column generation mathematical programming approach for a class-faculty assignment problem with preferences. *Computational Management Science*, v. 12, 2013. Disponível em: https://link.springer.com/article/10.1007/s10287-013-0163-9. Citado na página 12.

AL-YAKOOB, S. M.; SHERALI, H. D. Mathematical programming models and algorithms for a class–faculty assignment problem. *European Journal of Operational Research*, v. 173, n. 2, p. 488–507, 2006. ISSN 0377-2217. Disponível em: https://www.sciencedirect.com/science/article/pii/S0377221705002067. Citado 2 vezes nas páginas 10 e 12.

AVELLA, P.; VASILYEV, I. A computational study of a cutting plane algorithm for university course timetabling. *Journal of Scheduling*, v. 8, p. 497–514, 2005. Citado na página 12.

BALL, M. O. Heuristics based on mathematical programming. *Surveys in Operations Research and Management Science*, v. 16, n. 1, p. 21–38, 2011. ISSN 1876-7354. Disponível em: https://www.sciencedirect.com/science/article/pii/S1876735410000036. Citado 2 vezes nas páginas 13 e 14.

BLUM, C.; ROLI, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, v. 35, n. 3, p. 268–308, 2003. Citado na página 13.

BOSCHETTI, M. A.; LETCHFORD, A. N.; MANIEZZO, V. Matheuristics: survey and synthesis. *International Transactions in Operational Research*, v. 30, n. 6, p. 2840–2866, 2023. Disponível em: https://onlinelibrary.wiley.com/doi/10.1111/itor.13301. Citado 2 vezes nas páginas 11 e 14.

CARTER, M. W.; LAPORTE, G. Recent developments in practical course timetabling. In: BURKE, E.; CARTER, M. (Ed.). *Practice and Theory of Automated Timetabling II*. Springer Berlin Heidelberg, 1998. p. 3–19. ISBN 978-3-540-49803-2. Disponível em: https://link.springer.com/chapter/10.1007/BFb0055878. Citado 2 vezes nas páginas 10 e 12.

D'AMBROSIO, C. et al. The knapsack problem with forfeit sets. *Computers & Operations Research*, v. 151, p. 106093, 2023. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0305054822003239. Citado na página 14.

DOMENECH, B.; LUSA, A. A milp model for the teacher assignment problem considering teachers' preferences. *European Journal of Operational Research*, v. 249, n. 3, p. 1153–1160, 2016. ISSN 0377-2217. Disponível em:

https://www.sciencedirect.com/science/article/pii/S0377221715008139. Citado 2 vezes nas páginas 10 e 12.

EVEN, S.; ITAI, A.; SHAMIR, A. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, v. 5, n. 4, p. 691–703, 1976. Citado na página 16.

GAMRATH, G. et al. *The SCIP Optimization Suite 7.0*. 2020. Technical report, Optimization Online. Disponível em: http://www.optimization-online.org/DB_-HTML/2020/03/7705.html. Citado 4 vezes nas páginas 10, 12, 15 e 24.

GAREY, M. R.; JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman, 1979. Citado na página 16.

GUNAWAN, A.; NG, K.; ONG, H. A genetic algorithm for the teacher assignment problem for a university in indonesia. *International Journal of Information and Management Sciences*, v. 19, 2008. Disponível em: https://www.researchgate.net/publication/237502795_-A_Genetic_Algorithm_for_the_Teacher_Assignment_Problem_for_a_University_-in_Indonesia. Citado 2 vezes nas páginas 10 e 13.

HOSNY, M. A heuristic algorithm for solving the faculty assignment problem. *Journal of Communications and Computers*, v. 10, p. 287–294, 2013. Citado na página 13.

JOVANOVIC, R.; VOSS, S. Matheuristic fixed set search applied to the multidimensional knapsack problem and the knapsack problem with forfeit sets. *OR Spectrum*, p. 1–37, 2024. Citado na página 14.

MANIEZZO, V.; STÜTZLE, T.; VOSS, S. *Matheuristics: Algorithms and Implementations*. [S.l.]: Springer, 2021. Citado 3 vezes nas páginas 11, 13 e 14.

MONTGOMERY, D. C. *Design and Analysis of Experiments*. 9. ed. New York: John Wiley & Sons, 2017. Citado na página 24.

NETO, F. F. L.; HOSHINO, E. A.; PEDROTTI, V. Aplicação de matheurísticas em modelos estendidos. *LV Simpósio Brasileiro de Pesquisa Operacional*, p. 1–12, 2023. Disponível em: https://proceedings.science/p/175208?lang=pt-br. Citado na página 11.

PEDROTTI, V.; BARRETO, Jhonatan Duarte. Meta-heurísticas para o problema da alocação de disciplinas. In: INTEGRA UFMS, 2025, Campo Grande. *Anais eletrônicos [...]*. Campo Grande: UFMS, 2025. Disponível em: https://integra.ufms.br/files/2025/12/Anais-do-Integra-2025.pdf. Acesso em: 19 jul. 2026. Citado 4 vezes nas páginas 10, 11, 14 e 32.

PISINGER, D.; ROPKE, S. Large neighborhood search. In: *Handbook of Metaheuristics*. [S.l.]: Springer US, 2010. p. 399–419. Citado 2 vezes nas páginas 11 e 15.

ROPKE, S.; PISINGER, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, v. 40, n. 4, p. 455–472, 2006. Citado na página 14.

SILVA, L. S. *Uma abordagem exata para o problema da distribuição de disciplinas*. 2024. Trabalho de Conclusão de Curso, Universidade Federal de Mato Grosso do Sul (UFMS), Campo Grande, MS. Citado 3 vezes nas páginas 10, 15 e 16.

SILVA, P. P. A. de Paula e. *Uma Matheurística de Melhoria para o Problema da Mochila com Conjuntos de Penalidades*. 2024. Trabalho de Conclusão de Curso, Universidade Federal de Mato Grosso do Sul (UFMS), Campo Grande, MS. Citado 2 vezes nas páginas 14 e 15.

SZWARC, E.; BACH-DÁBROWSKA, I.; BOCEWICZ, G. Competence management in teacher assignment planning. In: *24th International Conference, ICIST 2018, Vilnius, Lithuania, October 4–6, 2018, Proceedings*. [S.l.]: Springer, 2018. p. 449–460. ISBN 978-3-319-99971-5. Citado na página 13.

VANDERBEI, R. J. *Linear Programming: Foundations and Extensions*. [S.l.]: Springer, 2014. Citado 2 vezes nas páginas 10 e 12.

WOLSEY, L. A. *Integer Programming*. [S.l.]: John Wiley & Sons, Ltd., 2020. Citado 2 vezes nas páginas 10 e 12.