COPPE
Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia
UFRJ

DOIS PROBLEMAS DE OTIMIZAÇÃO UTILIZANDO TÉCNICAS DE GERAÇÃO DE COLUNAS: PROBLEMA INTEIRO E PROGRAMAÇÃO GENERALIZADA

Amanda Matos Ferreira

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Nelson Maculan Filho

Rio de Janeiro
Setembro de 2025

DOIS PROBLEMAS DE OTIMIZAÇÃO UTILIZANDO TÉCNICAS DE GERAÇÃO DE COLUNAS: PROBLEMA INTEIRO E PROGRAMAÇÃO GENERALIZADA

Amanda Matos Ferreira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Nelson Maculan Filho

Aprovada por: Prof. Nelson Maculan Filho
Prof. Renan Vicente Pinto
Prof. Lennin Malllma Ramirez

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2025

Matos Ferreira, Amanda

Dois Problemas de Otimização Utilizando Técnicas de Geração de Colunas: Problema Inteiro e Programação Generalizada/Amanda Matos Ferreira. – Rio de Janeiro: UFRJ/COPPE, 2025.

XI, 71 p.: il.; 29, 7cm.

Orientador: Nelson Maculan Filho

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2025.

Referências Bibliográficas: p. 67 – 71.

1. Geração de Colunas. 2. Otimização Combinatória. 3. Técnicas de Decomposição. I. Maculan Filho, Nelson. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

"Se a educação sozinha não
transforma a sociedade, sem ela
tão pouco a sociedade muda."
Paulo Freire

Agradecimentos

Gostaria de agradecer primeiramente aos meu queridos pais, Andréa de Souza e Valceir Ferreira, por me permitirem acesso aos estudos mesmo diante das dificuldades enfrentadas. Estou concluindo este mestrado graças aos seus esforços e à dedicação diária, que persistem há anos. Obrigada por me permitirem ter um futuro acadêmico promissor e leve, sem me preocupar com nada além dos estudos.

Gostaria de agradecer ao Lucas Fernando, uma pessoa mais que importante durante essa trajetória, uma pessoa essencial. Obrigada por todo apoio e incentivo, sempre acreditante e apostando em mim.

Agradeço aos meus orientadores da graduação, Marcela Lima e Ronaldo Gregório, que sempre me incentivaram a ingressar no mestrado. Com certeza, sem o apoio de vocês, eu não teria iniciado essa etapa desafiadora da minha vida. Obrigada por sempre acreditarem em mim, e obrigada pelo prazer que me proporcionaram de ter trabalhado ao lado de vocês. Até hoje levo comigo todos os ensinamentos de nossas alegres reuniões (e que saudades delas!).

Agradeço ao meu atual orientador, Nelson Maculan, que desde o meu primeiro dia na UFRJ me senti acolhida e cuidada. Sou muito grata de ter um orientador que não só admiro profissionalmente, mas também admiro o ser humano gigante que és. A cada reunião (sempre com um bom cafezinho rs) cresço profissionalmente e aprendo um pouco mais sobre empatia, humildade, Política e História. Obrigada por toda paciência, por toda conversa e por todas explicações.

Agradeço a minha querida amiga Beatris Xavier que pude ter a felicidade de reencontrar no PESC. Nossa amizade foi/é um alicerce para eu chegar até aqui hoje, finalizando o mestrado. Obrigada por acreditar em mim, por ser minha ouvinte e por cada palavra positiva proferida em momentos cirúrgicos. Obrigada pelo espaço de empatia e acolhimento que desfrutei durante os períodos árduos e conturbados, e ainda, obrigada pelo entusiasmo de vibrar (e partilhar) comigo os momentos de alegria e de sucesso.

Também gostaria de agradecer aos meus amigos queridos pessoais de longa data (e da vida todo) pela escuta acolhedora, apoio e palavras positivas durante o mestrado. Beatriz França, Bruno Pimentel, Gabriel Pereira, Isabella Veloso, Juliana Fernandes, Letícia Nunes e Mariana Amorim. Sou muito grata pela compreensão, e

por me estimularem sempre seguir em frente.

Minha gratidão se estende também às instituições que viabilizaram financeiramente esta jornada. Primeiramente ao CNPq por meio da bolsa de pesquisa que me ofereceu tempo e a tranquilidade necessários em boa parte do meu mestrado. Além disso, e não menos importante, agradeço a COPPETEC e CAPGOV por também contribuírem para minha manutenção neste período. Sou muito grata pela oportunidade de está atuando no Projeto Verde (Defensoria Pública do Rio de Janeiro), onde estou tendo a chance de contribuir com a equipe de testes - a equipe mais feliz, segundo minha "chefa". Além do âmbito profissional e respeitoso, existe uma relação de amizade que ultrapassa totalmente o horário de expediente. Sou muito feliz de poder trabalhar/aprender ao lado de profissionais incríveis, em especial: Aline, Beatris, Caroline, Diego, Jéssica, Juliana, Marcella, Matheus, Victória e Wanderson.

Estendo também minha gratidão à Acolhe COPPE, onde pude encontrar tranquilidade e empatia em momentos conturbados. O apoio e o acolhimento oferecidos foram essenciais para enfrentar os desafios pessoais e do mestrado, proporcionando um ambiente de estudo mais saudável e equilibrado. Obrigada Josi, Márcia e Soraya.

Agradeço à UFRJ por manter um espaço de conhecimento e inovação, que inspira e forma profissionais. Reconheço a excelência de uma instituição que, impulsionada por políticas públicas sociais, tem se tornado um espaço cada vez mais amplo e plural, proporcionando um ensino de alta qualidade. Agradeço também toda a sua comunidade, incluindo os corpos docente, técnico e discente, em especial: Dona Rosa, Guty, Lurdes, o irmão, Ricardo.

Agradeço a Renan, uma pessoa querida, animada e empática, que carinhosamente se tornou meu "coorientador do coração". Mais do que os momentos felizes que compartilhamos, a minha gratidão é principalmente pelo suporte e pela ajuda constante. Sua paciência em me oferecer inúmeras explicações e sua valiosa disponibilidade para reuniões no fim do dia foram cruciais para a conclusão desta dissertação. Obrigada por tudo!

Gostaria de agradecer também ao Lennin, por fazer parte da minha banca. Uma pessoa querida e prestativa desde o início do mestrado. Obrigada por toda conversa que, com certeza, contribuiu para eu finalizar esta dissertação.

Por fim, meu muito obrigado a cada um que me acolheu e me deu a mão nesse momento desafiador que foi o mestrado. O incentivo e apoio que recebi durante este período foram inestimáveis.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

DOIS PROBLEMAS DE OTIMIZAÇÃO UTILIZANDO TÉCNICAS DE GERAÇÃO DE COLUNAS: PROBLEMA INTEIRO E PROGRAMAÇÃO GENERALIZADA

Amanda Matos Ferreira

Setembro/2025

Orientador: Nelson Maculan Filho

Programa: Engenharia de Sistemas e Computação

Apresenta-se, nesta dissertação, um estudo sobre a aplicação da técnica de decomposição Geração de Colunas na resolução de dois Problemas de Otimização. É abordado o destaque desse algoritmo por sua eficiência do método na redução da Complexidade Computacional aplicado na solução de um Problema Inteiro e de um Problema Não-Linear e Não-Convexo nesta pesquisa. O Cutting Stock Problem, um clássico de Otimização Combinatória, será o primeiro problema abordado e utilizado como fator motivador e explicativo para a aplicação da técnica. O segundo problema corresponde a um problema não-linear e não-convexo, resolvido mediante a Programação Linear Generalizada, cuja modelagem é reformulada para empregar a conceituação de Geração de Colunas para sua solução.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

THESIS TITLE

Amanda Matos Ferreira

September/2025

Advisor: Nelson Maculan Filho

Department: Systems Engineering and Computer Science

In this dissertation, a study on the application of the Column Generation decomposition technique to solve two Optimization Problems is presented. This algorithm is highlighted for the efficiency of its method in reducing the Computational Complexity applied to the solution of an Integer Problem and a Non-Linear and Non-Convex Problem in this research. The Cutting Stock Problem, a classic in Combinatorial Optimization, will be the first problem addressed and used as a motivating and explanatory factor for the application of the technique. The second problem corresponds to a non-linear and non-convex problem, solved using Generalized Linear Programming, whose model is reformulated to employ the Column Generation concept for its solution.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Introdução

Nesta dissertação, estudamos a técnica de *Geração de Colunas* aplicada em dois problemas de otimização distintos: o *Cutting Stock Problem* (CSP), amplamente investigado pela sua relevância em processos de corte de materiais, e um problema não linear e não convexo, resolvido por meio da Programação Linear Generalizada. Esta última é uma técnica de reformulação baseada em mudanças de variáveis, que permite recuperar a solução ótima do problema original a partir da solução do problema reformulado.

A escolha da Geração de Colunas se justifica pela sua eficiência em lidar com problemas de larga escala, especialmente aqueles com estrutura combinatória complexa. Em vez de considerar simultaneamente todas as variáveis do modelo, a técnica trabalha com um subconjunto reduzido delas, o que contribui para a diminuição do esforço computacional e torna viável a resolução de problemas que, de outra forma, seriam intratáveis.

O *Cutting Stock Problem* foi adotado inicialmente como exemplo introdutório, servindo para demonstrar o funcionamento do algoritmo de Geração de Colunas em um contexto já consolidado na literatura. Para isso, utilizamos instâncias de pequena escala, de modo a apresentar passo a passo o processo de resolução. Além da formulação matemática, são oferecidas explicações detalhadas a cada iteração, o que facilita a compreensão da lógica do método.

Na sequência, discutimos a Programação Linear Generalizada, conforme apresentada por LASDON (1970), e sua aplicação a problemas não lineares e não convexos. Essa abordagem é válida quando o vetor de coluna, formado pelo coeficiente da função objetivo c_j e pela coluna associada à variável x_j, pertence a um conjunto convexo S_j, cujos vértices são previamente conhecidos para cada $j \in J$. Tal método serviu como motivação para o desenvolvimento do algoritmo proposto nesta dissertação.

O novo algoritmo difere do apresentado em LASDON (1970), embora mantenha analogia com o método Simplex. Ele é adaptado para lidar com problemas não lineares, permitindo que os coeficientes a_{ij} sejam tratados como variáveis que podem assumir valores dentro de intervalos $[\alpha_{ij}, \beta_{ij}]$, enquanto os parâmetros c_j permanecem fixos e os valores de x_j variam. Seu funcionamento pode ser descrito, de forma simplificada, como um processo em que, a cada iteração, é resolvido um subproblema de otimização. A solução deste subproblema determina se uma nova coluna a_j deve ou não ser incorporada à base, com o intuito de melhorar a solução corrente.

Assim, o objetivo central deste trabalho é investigar o potencial da técnica de Geração de Colunas, tanto em sua aplicação clássica ao *Cutting Stock Problem* quanto em sua adaptação para problemas não lineares e não convexos, avaliando como essa abordagem pode contribuir para a redução significativa do esforço computacional.

A estrutura desta dissertação está organizada da seguinte forma: no Capítulo 2, apresenta-se uma revisão bibliográfica sobre Otimização Não Linear e Não Convexa, além de tópicos de Otimização Combinatória, com destaque para Complexidade Computacional e Relaxações Linear e Contínua. Também são discutidos o Algoritmo Simplex, a técnica de Geração de Colunas e o *Cutting Stock Problem*.

No Capítulo 3, é explorada a aplicação do algoritmo de Geração de Colunas tanto ao *Cutting Stock Problem* quanto à Programação Linear Generalizada. Por fim, é introduzido o algoritmo proposto para a resolução de problemas não lineares e não convexos, acompanhado de um exemplo ilustrativo, e assim como, o mesmo resolvido pelo método de LASDON (1970).

Por fim, no capítulo 4 foi apresentado uma comparação de resultados com instâncias distintas em tamanho. Os resultados mostram que o algoritmo proposto supera o IPOPT em termos de tempo, mesmo em instâncias maiores, sem perder qualidade da solução.

Capítulo 2

Revisão Bibliográfica

2.1 Otimização Não-Linear e Não-Convexa

Problemas de Otimização Não Linear e Não Convexa (PONLNC) são definidos por estruturas nas quais a função objetivo e/ou as restrições não apresentam comportamento linear e, tampouco, satisfazem as propriedades de convexidade. Esses problemas assumem formas geométricas complexas e, muitas vezes, desafiadoras do ponto de vista computacional. Uma formulação geral de um PONLNC pode ser representada em (2.1), onde foi adaptada das teorias dos modelos de problemas convexos e lineares encontrados em BOYD e VANDENBERGHE (2004):

$$\begin{aligned} \min \quad & f(x) \\ \text{sujeito a:} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h(x) = 0 \\ & x \in X \end{aligned} \tag{2.1}$$

Onde:

- $f(x)$: É a função objetivo não linear e/ou não convexa;
- $g_i(x)$: São as restrição não lineares e/ou não convexas;
- $h(x)$: São as restrição de igualdade não-lineares e/ou não convexas;
- $x \in X$: Define o conjunto viável.

Para entender o impacto da não convexidade na otimização, é essencial compreender as definições formais de conjunto convexo e função convexa. As definições aqui utilizadas também seguem a formulação apresentada por BOYD e VANDENBERGHE (2004), amplamente reconhecida na literatura.

Definição 2.1. *Um conjunto C é convexo se o segmento de reta entre quaisquer dois pontos em C estiver contido em C, isto é, se para quaisquer $x_1, x_2 \in C$ e qualquer θ com $0 \leq \theta \leq 1$, nós temos:*

$$\theta x_1 + (1 - \theta)x_2 \in C.$$

Definição 2.2. *Uma função $f : \mathbb{R}^n \to \mathbb{R}$ é convexa se $domf$ for um conjunto convexo e se, para todos $x, y \in domf$, e θ com $0 \leq \theta \leq 1$, tivermos:*

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y).$$

A definição de conjunto convexo, apresentada na Definição 2.1, estabelece que, se tomarmos quaisquer dois pontos pertencentes a esse conjunto, todo ponto situado no segmento de reta que os une também deverá pertencer ao conjunto. Em outras palavras, o conjunto compreende o caminho entre dois pontos, sem lacunas ou reentrâncias.

Essa propriedade garante que a região viável de um problema de otimização convexo seja bem definida do ponto de vista geométrico, isto é, livre de buracos ou formas irregulares que possam dificultar a busca por uma solução ótima. Na imagem (2.1), é possível visualizar o hexágono como um gráfico de um conjunto convexo, enquanto as duas demais figuras representam conjuntos não-convexos, com buracos e formas irregulares, respectivamente.

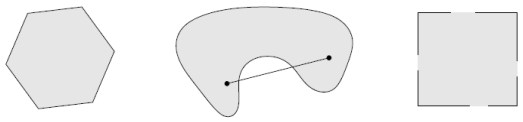

Figura 2.1: Definição geométrica de conjuntos convexos. Retirada de BOYD e VANDENBERGHE (2004).

De forma análoga, a função convexa, definida formalmente na Definição 2.2, é aquela em que, para quaisquer dois pontos x e y em seu domínio e qualquer $\theta \in [0, 1]$, temos $f(\theta x+(1-\theta)y) \leq \theta f(x)+(1-\theta)f(y)$. Isso significa que o gráfico da função se mantém sempre abaixo (ou no máximo sobreposto a) qualquer corda que ligue dois pontos da curva, evitando picos ou vales inesperados, como ilustrado na imagem (2.2):

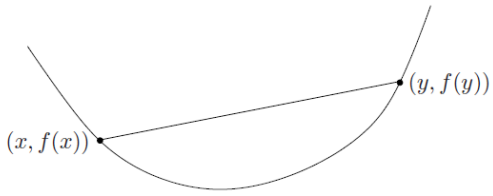

Figura 2.2: Gráfico de uma função convexa. Retirada de BOYD e VANDENBERGHE (2004).

Como consequência direta, em funções convexas, qualquer mínimo local é também o mínimo global, eliminando ambiguidade na solução ótima. Além disso, a epigraph da função — o conjunto de todos os pontos que estão no plano acima de seu gráfico — também constitui um conjunto convexo, o que reforça a regularidade da função do ponto de vista geométrico.

A linearidade é uma propriedade mais restrita. Toda função linear é convexa, mas o contrário não é verdadeiro. Isso ocorre porque a linearidade exige que a função preserve combinações lineares exatas dos pontos - ou seja, $f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$ para quaisquer $\alpha, \beta \in \mathbb{R}$, o que é mais rigoroso do que a desigualdade exigida pela convexidade.

A função quadrática $f(x) = ax^2 + bx + c$, com $a > 0$ e $x \in \mathbb{R}$, é um exemplo clássico de função convexa que não é linear. Sua convexidade pode ser verificada por meio da segunda derivada, uma vez que $f''(x) = 2a > 0$ para todo x, satisfazendo a condição de convexidade segundo a definição padrão. Apesar o gráfico dessa função apresente a curvatura voltada para cima, característica típica das funções convexas, ela não cumpre os critérios de linearidade.

Conforme discutido por BOYD e VANDENBERGHE (2004), funções lineares constituem um subconjunto das funções convexas, o que torna a convexidade uma generalização natural da linearidade no contexto de problemas de otimização.

É importante destacar a distinção entre não linearidade e não convexidade, uma vez que esses conceitos, embora relacionados, possuem implicações distintas no contexto da otimização. A não linearidade refere-se ao fato de que a função objetivo ou as restrições do problema não são lineares em sua forma matemática. Já a não convexidade está associada à ausência de propriedades de convexidade tanto no conjunto viável quanto na função objetivo, o que impacta diretamente a complexidade e a estrutura das soluções possíveis. Essa diferenciação conceitual é fundamental para a compreensão da natureza dos PONLNC.

De forma complementar, é importante destacar também que um problema de otimização não convexo pode conter restrições convexas, ainda que a estrutura global do problema não preserve a convexidade. Essa ocorrência é comum em formulações

mistas ou problemas com múltiplos níveis de complexidade estrutural, exigindo um tratamento específico para cada componente do modelo.

Consideremos o problema de minimizar $f(x) = \cos(x)$, sujeito à restrição $x \in [0, 2]$, exemplo do qual ilustra um caso em que a região viável é convexa, mas o problema como um todo não é, devido à forma da função objetivo. Embora a função objetivo não seja convexa no intervalo considerado, a restrição define um conjunto convexo.

Quando um problema de otimização é convexo — isto é, quando a função objetivo é convexa e o conjunto viável também apresenta convexidade — é possível garantir a unicidade da solução ótima global, além de assegurar a convergência eficiente de diversos algoritmos clássicos de otimização. Isso ocorre porque essas estruturas apresentam propriedades que favorecem a tratabilidade computacional. A convexidade do conjunto viável garante que qualquer combinação convexa de pontos factíveis continue sendo factível, o que simplifica a exploração do espaço de busca.

De modo análogo, e já discutido neste trabalho, a convexidade da função objetivo assegura que qualquer mínimo local seja também o mínimo global, eliminando ambiguidades na seleção da solução ótima.

Em contrapartida, problemas de otimização não convexos perdem essas garantias fundamentais. A presença de múltiplos mínimos locais compromete a robustez da solução obtida, a análise de dualidade torna-se restrita, e a escolha do ponto inicial passa a influenciar fortemente o resultado final, podendo até comprometer a convergência do algoritmo. Além disso, esses problemas são, em geral, associados a maior complexidade computacional. Frequentemente classificados como NP-difíceis, exigem tempo de resolução que pode crescer exponencialmente conforme o aumento da dimensão do problema.

Para suavizar essas dificuldades, empregam-se estratégias como Programação Global, métodos estocásticos, heurísticas e técnicas de relaxação. A escolha da abordagem depende da estrutura específica do problema, do nível de precisão desejado e das restrições práticas impostas pelo contexto computacional.

Neste trabalho, será apresentado um PONLNC resolvido por meio da técnica de Geração de Colunas, fornecendo uma fundamentação teórica sólida acerca da natureza desses problemas, situando o leitor quanto aos conceitos fundamentais que justificam as dificuldades inerentes à sua resolução.

2.2 Otimização Combinatória

Problemas clássicos da Otimização Combinatória desempenharam um papel central no desenvolvimento de algoritmos de otimização. Um exemplo clássico é o Problema das Sete Pontes de Königsberg, apresentado por EULER (1741), que investigou a

possibilidade de encontrar um caminho que atravessasse todas as pontes da cidade exatamente uma vez, retornando ao ponto de partida. Euler demonstrou que tal percurso não era possível, introduzindo assim o conceito de passeio euleriano e introduzindo as bases da Teoria dos Grafos.

Outro problema de grande relevância na literatura é o Traveling Salesman Problem (TSP), cuja formulação moderna é datada século XX. O TSP busca determinar o percurso mínimo que permita visitar um conjunto de cidades uma única vez, retornando à cidade de origem. Este é um problema pertencente à classe dos NP-difíceis, para os quais não se conhecem algoritmos que garantam solução ótima em tempo polinomial para todas as instâncias.

A Otimização Combinatória visa encontrar a melhor solução no espaço de busca diante de um conjunto finito de possibilidades, geralmente representadas por variáveis discretas. Contudo, a obtenção de suas soluções ótimas raramente é trivial. Embora a formulação de problemas de Otimização Combinatória seja muitas vezes simples, encontrar a solução ideal pode representar um grande desafio computacional, que se agrava com a complexidade das instâncias e restrições. Para problemas de grande porte, o número de possibilidades viáveis pode ser tão vasto que listar cada alternativa se torna impraticável, levando à explosão combinatória.

Para atenuar os efeitos da explosão combinatória, diversas estratégias algorítmicas são utilizadas. Os métodos exatos, como a Programação Inteira e o algoritmo Branch-and-Bound, garantem a obtenção da solução ótima, porém sua aplicabilidade costuma se restringir a instâncias de pequeno ou médio porte, devido ao elevado custo computacional associado. Já para problemas de maior escala, é comum o uso de heurísticas e meta-heurísticas, que, embora não assegurem a otimalidade da solução, buscam aproximações de alta qualidade com menor esforço computacional. Algoritmos Genéticos, Simulated Annealing e Busca Tabu são exemplos clássicos da literatura de abordagens utilizadas nesse contexto, sendo especialmente eficazes na resolução de problemas complexos com múltiplos mínimos locais e espaços de busca extensos.

WOLSEY (2020) descreve que Problemas de Otimização Inteira e Combinatória possuem modelos robustos, e muitos desses modelos podem ser ricamente representados com variáveis discretas. CAMPELLO (1994) apontam que Otimização Combinatória não visa encontrar a quantidade total de arranjos existentes em um problema, mas sim obter somente um arranjo ótimo diante dos inúmeros existentes. PAPADIMITRIOU e STEIGLITZ (1998a) exploram a relação entre o tempo de execução, a eficiência e a estabilidade dos algoritmos em função do tamanho de suas instâncias, além de abordar o estudo da Complexidade Computacional que avalia o desempenho dos algoritmos em termos de tempo e espaço de execução.

Assim, a convergência da Otimização Combinatória e Complexidade Computa-

cional reside na busca por algoritmos e sistemas capazes de lidar de forma eficiente com os desafios intrínsecos aos problemas do mundo real, garantindo a viabilidade de encontrar soluções ótimas em tempo razoável.

2.2.1 Complexidade Computacional

A Complexidade Computacional estuda os recursos computacionais - tempo e memória - necessários para resolução de problemas. O desempenho de um algoritmo é avaliado principalmente sob duas métricas: a complexidade de tempo e a complexidade de espaço.

A Complexidade de Tempo de um algoritmo mensura seu desempenho em termos de tempo de execução, em função do tamanho da entrada. Essa análise é significativa porque permite tanto a comparação da eficiência dos algoritmos, facilitando a seleção dos mais adequados para resolver problemas específicos, quanto a estimativa do período necessário para que um algoritmo processe diferentes instâncias, considerando as restrições computacionais de cada aplicação.

A Complexidade de Espaço, por sua vez, refere-se à quantidade de memória necessária para a execução de um algoritmo, também em função do tamanho da entrada, contudo priorizando a avaliação da memória consumida em relação ao porte das instâncias processadas.

Historicamente, na década de 60, a avaliação do tempo de execução era realizada de forma empírica, mensurando o tempo real de execução de um algoritmo em um computador específico. No entanto, essa abordagem apresenta limitações devido à variação dos resultados em função do hardware, software e outros fatores do ambiente de execução. Atualmente, adota-se uma abordagem analítica e não dependente da máquina utilizada. A eficiência dos algoritmos é descrita por meio de notações assintóticas, como $\mathcal{O}(n)$, $\Omega(n)$ e $\Theta(n)$ CORMEN *et al.* (2022). A notação $\mathcal{O}$ representa o limite superior do tempo de execução (pior caso), Ω indica o limite inferior (melhor caso) e Θ descreve o crescimento exato da função (caso médio ou típico). Essas notações permitem abstrair o desempenho dos algoritmos de forma teórica, facilitando a comparação entre diferentes estratégias de solução, especialmente para grandes volumes de dados.

Para informações mais detalhadas, acerca dos métodos e técnicas utilizados para calcular a Complexidade de Tempo de algoritmos, pode ser consultado o livro de SZWARCFITER e MARKENZON (1994).

O estudo da Complexidade Computacional é, particularmente, relevante em Problemas de Otimização Combinatória (POCs), dado o vasto tamanho do espaço de busca que esses problemas frequentemente assumem. A utilização dos recursos de Modelagem Matemática e Computacional viabiliza representações mais ajustadas e

eficientes.

Algoritmos de força bruta, por exemplo, enumeram todas as possibilidades de soluções viáveis e identificam a solução ótima por meio da enumeração exaustiva de todas as possibilidades. Contudo, o tempo necessário para encontrar a solução ótima pode resultar em um custo computacional indesejado, inviabilizando sua aplicação prática, mesmo para máquinas mais avançadas. Com o objetivo de solucionar essa limitação, a literatura emprega alternativas de encontrar soluções mais eficientes, como modelar um mesmo problema por maneiras distintas, por exemplo. Modelagens eficientes também contribuem para uma boa formulação matemática, onde, ao simplificar o problema, é possível reduzir sua complexidade.

Problemas de Otimização Combinatória são problemas de decisão fundamentalmente baseados na Matemática Discreta. Sua modelagem é frequentemente realizada através da Programação Inteira, cujas formulações matemáticas são capazes de descrever a essência do problema prático e capturar sua combinatória na busca pela solução ótima.

Consideramos um conjunto finito $N = \{1, 2, \ldots, n\}$, do qual extraímos subconjuntos $S \subseteq N$ que formam o conjunto de soluções viáveis F. Esses subconjuntos S representam as combinações possíveis de elementos que satisfazem todas as restrições do problema. Ou seja, $F = S : S \subseteq N$ de elementos viáveis, a saber, esses elementos de F que serão relevantes na resolução dos problemas, uma vez que só estamos interessados nas soluções viáveis.

Sua solução ótima é aquela que maximiza ou minimiza uma determinada função objetivo, sujeita às restrições do problema. WOLSEY (2020) define em (2.2) a formulação de um Problema de Otimização Combinatória, com critério de minimização, em encontrar um único elemento de F que garanta o menor custo C_j total, onde $C_j : j \in N$:

$$\text{minimizar} \quad \left\{ \sum_{j \in S} C_j : S \in F \right\} \tag{2.2}$$

Métodos exatos como Branch-and-Bound, Relaxação Linear e Geração de Colunas são amplamente utilizados com o objetivo de reduzir o crescimento exponencial do espaço de busca, acelerando o processo de convergência para o ótimo.

No entanto, é importante destacar que, em muitos casos práticos, a busca por uma solução ótima global não é o objetivo primordial para o modelador e, dessa maneira, encontrar soluções viáveis e praticáveis pode ser uma boa alternativa. Em problemas de grande escala, ou dados incertos, por exemplo, obter a solução ótima global pode ser computacionalmente cara, e até desnecessária em certos aspectos. Soluções heurísticas ou aproximadas, são mais inteligentes por ser contraporem à

busca exaustiva de uma solução ótima final, que uma vez encontrada, podem se portar inviáveis de implementação devido a restrições do mundo real não estarem plenamente descritas no modelo.

A seguir, apresentaremos, de forma breve, a técnica de Relaxação Linear e o Algoritmo Simplex, que servirão de base para a técnica de decomposição Geração de Colunas. O uso da Relaxação Linear será fundamental para execução do Cutting Stock Problem, e o algoritmo Simplex para a manipulação da técnica de decomposição, e do algoritmo proposto.

2.2.2 Relaxação Linear e Contínua

A ideia de relaxação, em termos gerais, consiste em tornar um problema original mais tratável do ponto de vista computacional, removendo ou suavizando algumas de suas restrições, principalmente as relacionadas à integralidade ou à não linearidade.

Em otimização, duas relaxações são amplamente utilizadas: a relaxação linear, voltada a problemas lineares inteiros ou mistos, e a relaxação contínua, empregada em modelos não lineares inteiros. A relaxação contínua é amplamente utilizada, onde as restrições de integralidade das variáveis são relaxadas, permitindo que estas assumam valores reais contínuos. Quando o problema resultante dessa relaxação é linear, a técnica é especificamente denominada relaxação linear

Relaxação Linear

A técnica de Relaxação Linear desempenha um papel fundamental na resolução de problemas de Programação Linear Inteira (PLI) e Otimização Combinatória, pois permite a obtenção de aproximações eficientes, especialmente em problemas de grande porte. Essa abordagem ganhou relevância com os trabalhos de GOMORY (2009), ao introduzira os cortes fracionários — restrições adicionais que eliminam soluções fracionárias preservando a região viável inteira do modelo.

A Relaxação Linear está, por definição, associada a problemas de Programação Linear Inteira ou Mista, especialmente para modelagens matemáticas mais complexas e robustas. Trata-se de relaxar as restrições de integralidade das variáveis, para assim, permitir que estas possam assumir valores reais contínuos.

Problemas de Programação Inteira pertencem à classe NP-difícil, frequentemente tornam sua resolução impraticável através de métodos exatos. Nesses casos, a Relaxação Linear é uma estratégia comum para obter limites inferiores, ou superiores para o valor ótimo da solução inteira.

Considere o problema em (2.3) para critério de exemplificação para a relaxação linear:

$$\begin{aligned} \min \quad & c^T x \\ \text{sujeito a:} \quad & Ax \leq b \\ & x \in \mathbb{Z}_+^n \end{aligned} \tag{2.3}$$

Ao relaxar a condição de integralidade das variáveis $x \in \mathbb{Z}_+^n$, obtém-se o problema relaxado (2.4) a seguir, agora com as variáveis assumindo valores contínuos:

$$\begin{aligned} \min \quad & c^T x \\ \text{sujeito a:} \quad & Ax \leq b \\ & x \in \mathbb{R}_+^n \end{aligned} \tag{2.4}$$

A Relaxação Linear viabiliza o uso de algoritmos eficientes de Programação Linear, como o Simplex em problemas inteiros. Essas informações são essenciais para algoritmos como Branch-and-Bound e Branch-and-Cut, pois fornecem uma aproximação inicial da solução WOLSEY (2020); CORMEN *et al.* (2022).

Pode haver uma diferença entre o resultado da solução relaxada com a solução inteira, de forma que os valores das funções objetivos possam não coincidirem. A diferença entre esses valores é denominada Gap de Otimalidade e, este serve como uma métrica da qualidade da aproximação. Um valor pequeno indica uma boa aproximação e sugere um modelo mais ajustado, enquanto valores elevados podem indicar a necessidade de reforços no modelo, como a adição de planos de corte para reduzir o espaço de busca e melhorar a qualidade do limite, por exemplo.

Relaxação Contínua

Para problemas de Otimização Não Linear Inteira, a técnica adequada é denominada Relaxação Contínua. Nesse caso, relaxa-se a integralidade das variáveis, permitindo que estas assumam valores reais contínuos, porém sem exigências de linearidade na função objetivo ou nas restrições.

Considere o seguinte problema de Programação Inteira Não Linear em (2.5):

$$\begin{aligned} \min \quad & f(x) \\ \text{sujeito a:} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & x \in \mathbb{Z}^n \end{aligned} \tag{2.5}$$

Uma Relaxação Contínua dessa modelagem pode ser visualizada em (2.6):

$$\begin{aligned} \min \quad & f(x) \\ \text{sujeito a:} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m \\ & x \in \mathbb{R}^n \end{aligned} \tag{2.6}$$

Essa abordagem é essencial em problemas não convexos e não lineares, como o

discutido neste trabalho, possibilitando o uso de algoritmos de otimização contínua. A análise do gap de dualidade entre a solução relaxada e a melhor solução inteira disponível permite avaliar a qualidade da aproximação e orientar estratégias de refinamento.

Em resumo, ambas as técnicas são ferramentas cruciais para reduzir significativamente a complexidade computacional, e viabilizar a resolução de problemas de larga escala. A Relaxação Linear é indicada para modelos com estrutura linear inteira ou mista, enquanto a Relaxação Contínua estende sua aplicabilidade a problemas com funções e restrições não lineares.

2.3 Simplex

O método Simplex, introduzido por DANTZIG (1951), é uma técnica fundamental para a resolução de Problemas de Programação Linear (PPL). Sua principal caracterização é explorar a estrutura poliédrica da região factível de um problema para encontrar o valor ótimo da função objetivo.

Esse percurso é realizado ao longo das arestas, partindo de um vértice inicial e movendo-se para os seus vértices adjacentes, almejando sempre uma melhoria no valor da função objetivo, conforme cada iteração. MACULAN e FAMPA (2006), de forma mais sucinta, destacaram a comutação dos vértices relacionados com a viabilidade a cada iteração como:

Citação 1. *A ideia do método é partir de uma solução básica de $Ax = b$ satisfazendo $x \geq 0$, isto é, uma solução básica primal viável, passar para outra solução básica primal viável sem que o valor da função objetivo diminua (no caso de maximização). (Maculan e Fampa, 2004: 19)*

Consideremos um problema de Programação Linear (PL) com critério de minimização, na forma padrão:

$$\begin{aligned} (P): \quad & \min \sum_{j=1}^{p} x_j \\ \text{sujeito a:} \quad & \sum_{j=1}^{p} a_{ij} x_j = d_i, \quad i = 1, \ldots, m, \\ & x_j \geq 0, \quad j = 1, 2, \ldots, p, \end{aligned} \tag{2.7}$$

A seguir, são descritos os passos fundamentais do algoritmo Simplex, para um problema de minimização, ilustrados também no fluxograma da Figura (2.3).

1. **Modelo no formato padrão e Solução Básica Viável:** Inicialmente, o problema é formulado no modelo formato padrão. Em seguida, determina-se

uma Solução Básica Viável (SBV), identificada por x_B, correspondente a um subconjunto de m variáveis básicas que satisfazem o sistema $Ax = b$, com $x \geq 0$.

2. **Teste de Otimalidade:** Realiza-se o Teste de Otimalidade, com o cálculo dos custos reduzidos $c_j - z_j$, onde z_j representa o valor associado à variável não básica x_j, calculado a partir das variáveis duais.

3. **Condição de Otimalidade:** Se todos os custos reduzidos satisfazem $c_j - z_j \geq 0$, a solução atual é ótima e o algoritmo é encerrado.

4. **Seleção da Variável a Entrar na Base:** Caso a condição de otimalidade não seja satisfeita (ou seja, existe algum $c_j - z_j < 0$), seleciona-se uma variável para entrar na base.

5. **Critério de Entrada na Base:** A variável a entrar na base é aquela, dentre as não básicas, que possui o maior custo reduzido negativo. Isso pode ser formalizado como:

$$\arg\max \{c_j - z_j \mid x_j \in x_N,\ c_j - z_j < 0\},$$

onde x_N é o conjunto de variáveis não básicas.

6. **Condição de Otimalidade:** Após a seleção da variável de entrada, deve-se verificar se todos os $c_j - z_j$ são não-negativos. Se esta condição for satisfeita, a solução é ótima e o algoritmo é encerrado .

7. **Seleção da Variável a Sair da Base:** Se a solução ainda não for ótima após as verificações anteriores, uma variável básica é selecionada a sair da base.

8. **Teste da Razão Mínima:** Para determinar qual variável básica deixará a base, realiza-se o Teste da Razão Mínima:

$$\theta = \min \left\{ \frac{d_i}{a_{ij}} \mid a_{ij} > 0 \right\},$$

onde d_i são os valores atuais das variáveis básicas e a_{ij} são os coeficientes da coluna da variável de entrada na base transformados.

9. **Atualização da Solução Básica Viável:** A solução básica viável é então atualizada com a nova composição de variáveis, resultando em um novo ponto extremo da região factível.

10. **Retorno e Repetição do Processo:** O processo retorna ao passo 2 (Teste de Otimalidade) e se repete até que a condição de otimalidade seja satisfeita em qualquer uma das verificações, passos 3 ou 6.

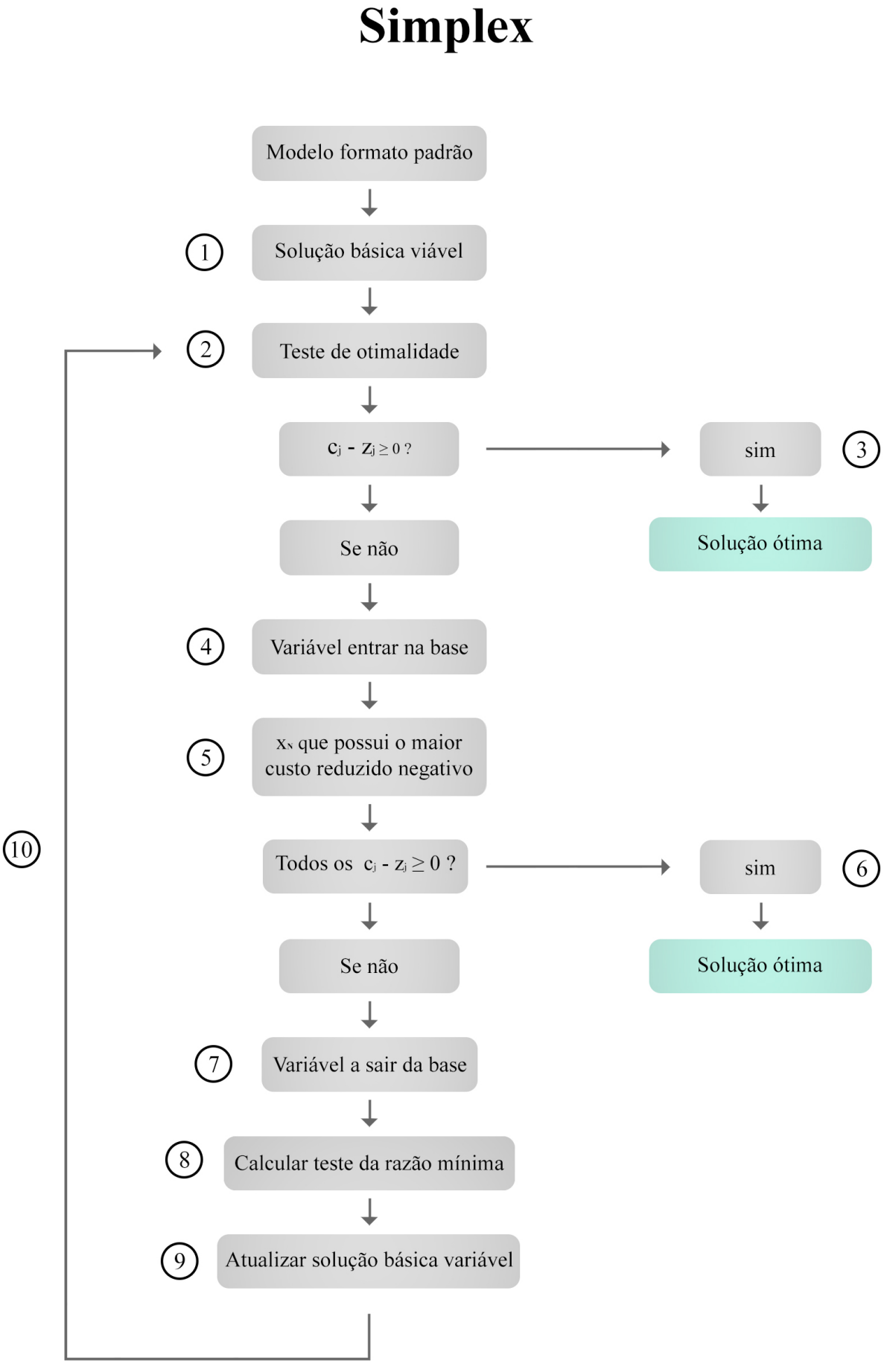

Figura 2.3: Fluxograma correlacionado com os passos descritos

No contexto desta dissertação, o algoritmo Simplex é utilizado como resolvedor do Problema Mestre Restrito dentro da técnica de Geração de Colunas. A cada iteração, resolve-se o problema atual utilizando o conjunto de colunas disponíveis, atualizando a solução primal, bem como as variáveis duais, que posteriormente são utilizados na formulação e resolução do Subproblema gerador de colunas.

Método Simplex Duas Fases

Nem todo PPL pode ser inicialmente aplicado ao método Simplex devido má formulação para o mesmo e, também sua solução básica viável não pertencer ao conjunto factível padrão. Isso ocorre pelo motivo do problema não apresentar uma solução básica viável inicial, ou seja, a forma algébrica que compõe a matriz de restrições não possui uma submatriz identidade, como também, não possui variáveis que permite compor uma base inicial, pois violam a condição de não negatividade.

As razões das estruturas desses problemas que aplicam o método Simplex de duas fases é bem retratada nos trabalhos de BAZARAA *et al.* (2011) e VANDERBEI (2014). Os autores analisam as restrições de modelos que impedem o uso imediado do algoritmo simplex. Segue abaixo um resumo de cada restrição complicador

- Restrições de igualdade onde não é possível adicionar variáveis de folga, já que essas variáveis servem só para restrições caracterizadas por inequações;
- Restrições compostas por inequações "maior ou igual a"para serem transformadas em uma igualdade. Neste caso, é preciso subtrair uma variável de excesso, e com isso, essa variável entra com coeficiente negativo na equação, o que não ajuda a formar uma coluna da matriz identidade;
- Coeficientes negativos no lado direito da restrição ($b < 0$) são contornado ao multiplicarem a inequação por -1, o que também decai no caso anterior.

Nesses casos, para contornar esses empecilhos, é aplicado o método Simplex Duas Fases, um clássico documentado na literatura pelos trabalhos de KANIYAR (2025), MACULAN e FAMPA (2006) e BAZARA *et al.* (2011), por exemplo, cuja finalidade é garantir a inicialização do Simplex a partir de um ponto viável, independentemente desse ponto pertencer diretamente ao espaço do problema original.

Suponha que temos um Problema de Programação Linear (PL) na forma:

$$\begin{aligned} \min \quad & c^T x \\ \text{sujeito a} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{2.8}$$

Para a Primeira Fase do método é introduzido de variáveis artificiais x^a em (2.8) para "forçar"a obtenção de uma base viável. Em (2.9), temos um novo modelo

designado de Problema Auxiliar, já com o acréscimo dessas variáveis:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} x_j^a \\ \text{sujeito a} \quad & Ax + Ix^a = b \\ & x, x^a \geq 0 \end{aligned} \tag{2.9}$$

Onde:

- x^a são variáveis artificiais adicionadas, uma para cada equação de igualdade ou restrição do tipo $\geq$.
- I é a matriz identidade que permite formar uma base inicial viável.

O objetivo desse Problema Auxiliar é minimizar a soma das variáveis artificiais, pois desejamos eliminá-las do problema abordado.

Ao encontrar uma solução ótima em (2.9), logo encontraremos uma solução viável para o Problema Auxiliar (2.8). É importante destacar que, quando o problema original admite uma solução básica viável, o problema auxiliar da Primeira Fase produzirá, necessariamente, uma solução ótima onde todas as variáveis artificiais assumem valor nulo.

Isso ocorre porque, nesse caso, existe uma combinação das variáveis originais que satisfaz integralmente as restrições do modelo, tornando desnecessária a contribuição das variáveis artificiais. Logo, o valor ótimo da função objetivo de (2.9) igual a zero, e todas as variáveis artificiais introduzidas poderão ser removidas do modelo sem prejuízo à viabilidade, possibilitando o início da Fase 2, a partir de agora, de uma base real e factível.

Contudo, caso a solução ótima da Primeira Fase resulte em pelo menos uma variável artificial com valor estritamente positivo, isto é, se $x_j^a > 0$ para algum j, isso indica que não foi possível satisfazer o conjunto de restrições do problema original sem o auxílio das variáveis artificiais introduzidas. Ou seja, a presença de variáveis artificiais ativas na solução ótima do Problema Auxiliar implica que o conjunto viável do problema original é vazio.

Portanto, nesse cenário, conclui-se que o problema original não admite solução básica viável, o que inviabiliza a aplicação da Fase 2 do método Simplex. Assim, o processo é encerrado ainda na Fase 1, sendo declarado o problema como inviável.

Após a Primeira Fase assegurar a viabilidade da solução, então, pode-se iniciar a segunda, e última fase. Uma vez obtida uma solução ótima para o Problema Auxiliar, considera-se que foi identificada uma Solução Básica Viável para o problema original (2.8), isso porque todas as variáveis artificiais assumem valor zero. Essa solução passa a ser utilizada como ponto de partida para a Fase 2.

O procedimento agora consiste, em sequência, da remoção das colunas associadas às variáveis artificiais da matriz de restrições. Redefinir a função objetivo do problema original, denotada por $c^T x$, anteriormente substituída na Fase 1.

Por fim, o algoritmo Simplex é retomado a partir da base viável já encontrada, operando agora com foco exclusivo na otimalidade.

Método da Penalidade – Big M

Assim como o método Simplex Duas Fases, o método da penalidade, conhecido como método Big M, é empregado em PPL nos quais as restrições não se apresentam diretamente na forma padrão exigida pelo algoritmo Simplex. Isso ocorre, como no Simplex Duas Fases, quando há restrições de igualdade ou restrições do tipo maior ou igual ($\geq$), o que impede a obtenção direta de uma Solução Básica Viável inicial.

Nesse método, também são introduzidas variáveis artificiais ao modelo. Contudo, diferentemente da abordagem do Simplex Duas Fases, não se constrói um problema auxiliar. Em vez disso, essas variáveis artificiais são incorporadas diretamente na função objetivo original, com coeficientes penalizadores de magnitude extremamente elevada — comumente representados por uma constante M, suficientemente grande. O objetivo é tornar a presença dessas variáveis indesejáveis na solução ótima.

O ponto central do método Big M está na introdução controlada dessas variáveis artificiais, que são estrategicamente posicionadas nas restrições para completar a matriz de coeficientes com uma submatriz identidade. Isso garante a existência de uma base viável inicial, requisito necessário para a aplicação do algoritmo Simplex. Embora essas variáveis não pertençam ao modelo original, a penalização imposta via $M \gg 0$ assegura que, durante as iterações do algoritmo, o Simplex tenderá a eliminá-las da base sempre que houver uma solução viável composta apenas pelas variáveis originais.

No caso de um problema de minimização, os coeficientes das variáveis artificiais são positivos e muito grandes, enquanto em problemas de maximização, são muito negativos. A formulação típica da função objetivo penalizada assume a forma:

$$\min \quad Z = c^T x + M \sum x_j^a \tag{2.10}$$

Onde:

- x_j^a representando as variáveis artificiais adicionadas ao modelo;
- $M \gg 0$ indicando uma penalidade suficientemente elevada, de modo a tornar sua permanência na base economicamente indesejável.

Com a estrutura da função objetivo definida em (2.10), é possível aplicar o algoritmo Simplex diretamente ao problema modificado, em uma única etapa, sem necessidade de fase preliminar.

Entretanto, essa conveniência apresenta limitações importantes. A escolha do valor de M é arbitrária, e embora deva ser suficientemente grande para garantir a eliminação das variáveis artificiais, não existe um critério teórico absoluto para sua definição. Como a notação $M \gg 0$ indica uma penalidade altamente elevada, espera-se que sua contribuição domine a função objetivo e force sua exclusão da solução ótima. No entanto, essa magnitude deve ser calibrada com cautela, pois valores excessivos podem introduzir instabilidades numéricas, como perda de precisão e erros de arredondamento — especialmente em problemas com escalas heterogêneas ou mal dimensionados.

Conforme discutido por KANIYAR (2025), essas fragilidades numéricas podem comprometer a confiabilidade do método, especialmente em implementações computacionais de larga escala. De modo semelhante, **?** apontam que o método Big M pode ser inadequado para ambientes sensíveis a precisão, sendo mais apropriado para aplicações didáticas ou modelos de pequena dimensão.

Dessa forma, o método Big M é geralmente recomendado em situações em que o controle sobre os dados e a escala do problema permitem ajustes adequados de penalização. Nesta dissertação, utilizamos esse método para garantir uma base viável inicial na resolução do segundo problema abordado.

2.4 Geração de Colunas

O objetivo desta seção é apresentar conceitos teóricos e fundamentais do uso das técnicas de decomposição, assim como introduzir o algoritmo de Geração de Colunas utilizado nesta pesquisa. Sua funcionalidade é aplicada para lidar com problemas lineares que possuem muitas variáveis manipuladas no modelo, em comparação com suas restrições. Ou seja, essa abordagem é destinada a resolver Problemas de Otimização Linear em larga escala, caracterizados por possuir um número elevado de variáveis, que em muitos casos, são impraticáveis de enumeração.

2.4.1 Evolução Histórica das Técnicas de Decomposição

Historicamente, as primeiras abordagens para resolução de Problemas de Otimização Linear podem ser atribuídas ao trabalho de KANTOROVICH (1939), que introduziu, além da ideia inicial de Programação Linear, a ideia de particionar problemas maiores em problemas reduzidos. Essa ideia é associada aos Subproblemas, que, para estes, serão discorridos com minuciosidade mais à frente. Apesar do trabalho original de Kantorovich não se referir a técnicas de decomposição, seu estudo conduziu pesquisas futuras e promissoras sobre o tema.

DANTZIG e WOLFE (1960), abordaram a resolução de Problemas de Programação Linear dividindo-os em Problema Mestre e Subproblema. Contudo, FORD JR e FULKERSON (1958) já haviam explorado técnicas de decomposição em problemas de fluxo em redes, separando o fluxo total em fluxos individuais para cada commodity. Embora Ford e Fulkerson não tenham trabalhado diretamente com Geração de Colunas, seus estudos também introduziram técnicas de decomposição que, mais tarde, influenciariam abordagens em Otimização como as de Dantzig-Wolfe.

Diante disso, o método de decomposição Dantzig-Wolfe estabeleceu formalmente o conceito de decomposição para problemas lineares gerais. Seu funcionamento baseia-se na resolução de problemas lineares de grande porte que possuem características de blocos independentes na matriz de restrição, possibilitando a repartição do problema original em subproblemas menores e mais tratáveis. Essa decomposição ocorre por meio da representação das soluções de cada subproblema como combinações convexas das soluções extremas da região factível.

Em um processo iterativo, o Problema Mestre Restrito fornece uma solução viável ao resolver um modelo reduzido do problema original, enquanto o Subproblema identifica novas colunas a serem incorporadas ao Problema Mestre Restrito, condensando-o com o acúmulo de colunas. É importante destacar que, apesar desse método ser destinado a problemas lineares originalmente, ele também serviu como base para técnicas posteriores que abordaram problemas inteiros, como a Branch-and-Price. Uma aplicação clássica que utilizou esse método é a resolução do Cutting Stock Problem, cujo objetivo é a minimização dos desperdícios gerados após cada corte, proposta por GILMORE e GOMORY (1961).

Gilmore e Gomory introduziram a técnica de Geração de Colunas para obter cortes eficientes de materiais como papel, madeira e metal. Essa abordagem foi ajustada diante da modelagem inteira do Cutting Stock Problem em uma formulação linear, utilizando a técnica de Dantzig e Wolfe, permitindo resolver problemas de grande escala.

A formulação proposta estabelece que cada padrão de corte viável corresponde a uma coluna da matriz de restrição, permitindo uma abordagem iterativa. O subproblema tem como objetivo identificar novos padrões de corte promissores ao problema original, incorporando-os iterativamente ao Problema Mestre Restrito, ao invés de acrescentar todos os possíveis padrões desde o início.

O processo se repete até que não existam mais padrões de corte capazes de reduzir o custo da função objetivo, garantindo assim a otimalidade da solução. Gilmore e Gomory inovaram na resolução de Problemas Combinatórios e possibilitaram o desenvolvimento de diversas variantes para o Cutting Stock Problem.

Após a consolidação da Geração de Colunas no Cutting Stock Problem, a técnica começou a ser aplicada em problemas ainda mais complexos. DESROSIERS

et al. (1984) aplicaram as técnicas da decomposição de Dantzig e Wolfe junto com Branch-and-Bound e Geração de Colunas em problemas de roteamento e transporte. Demonstraram a eficácia da Geração de Colunas aplicada de forma inovadora em problemas de roteamento com janelas de tempo.

Nos anos 1990, BARNHART *et al.* (1998) estabeleceram a técnica de Branch-and-Price na resolução de problemas inteiros de grande escala. Esse algoritmo firmou a aplicação de Geração de Colunas com Branch-and-Bound.

Com o desenvolvimento de algoritmos híbridos e de heurísticas, o trabalho de LÜBBECKE e DESROSIERS (2005) permitiu um avanço maior na técnica de Geração de Colunas para resolução de problemas de otimização mais complexos. Lübbecke e Desrosiers relataram em seu artigo, tanto a teoria detalhada, como técnicas práticas de resolução de problemas com degenerescência e convergência lenta, além de descreverem a evolução do uso da Geração de Colunas na Programação Inteira ao longo dos anos.

Atualmente, os trabalhos concentram-se na utilização do algoritmo focando na resolução de problemas inteiros de grande porte. É possível observar nos trabalhos PESSOA *et al.* (2020), UCHOA *et al.* (2024), DESROSIERS *et al.* (2024), um estudo detalhado acerca de Problemas de Otimização Combinatória e Geração de Colunas.

2.4.2 Algoritmo de Geração de Colunas

Nesta seção, apresenta-se a técnica de Geração de Colunas, que será aplicada aos dois problemas abordados nesta dissertação. A seguir, descreve-se o funcionamento do algoritmo em associação com o método Simplex, com ênfase no processo de decomposição em Problema Mestre (PM) e Subproblema.

O objetivo da Geração de Colunas é gerar, de forma iterativa e sistemática, novas variáveis (colunas) que aperfeiçoem progressivamente a solução de um modelo de grande porte. Em muitos problemas de grande escala, como já discutido, o número de variáveis é tão elevado que torna inviável sua enumeração completa, devido à explosão combinatória. A Geração de Colunas permite contornar essa dificuldade ao processar, em cada iteração, apenas um subconjunto reduzido de colunas, promovendo uma redução significativa da complexidade computacional.

O problema original é denominado Problema Mestre (PM), e é reformulado em um Problema Mestre Restrito (PMR), que contém apenas um conjunto inicial reduzido em colunas viáveis. A solução do PMR fornece uma solução viável para o PM, porém não necessariamente ótima, já que representa apenas um limitante inferior para o ótimo do problema completo. A Geração de Colunas pode ser aplicada tanto em Problemas de Programação Linear quanto em Problemas Inteiros relaxados linearmente.

O processo de Geração de Colunas baseia-se na resolução iterativa de um Subproblema, cujo objetivo é identificar colunas que, ao serem incorporadas ao PMR, promovam a melhoria da solução corrente. A eficiência do algoritmo depende da possibilidade de calcular os custos das colunas do PMR com base em sua própria estrutura, o que permite que o Subproblema identifique colunas promissoras, com base no critério de custo reduzido.

A resolução inicial do PMR processa-se através de uma solução básica viável, obtida comumente por heurísticas, garantindo que todas as restrições sejam satisfeitas. Embora que não seja ótima, a solução inicial garante viabilidade para a execução do algoritmo Simplex.

Com a viabilidade inicial garantida, inicia-se o processo iterativo, da qual, a cada iteração, o PMR é resolvido utilizando o algoritmo Simplex, fornecendo uma solução primal λ_j^* e um vetor dual u^*. Esses vetores duais, associados às restrições do PMR, são fundamentais para a fase de precificação de colunas.

O Subproblema, formulado com base na estrutura do Problema Mestre e parametrizado por u, tem como objetivo identificar uma nova coluna (variável) j que, se adicionada ao PMR, irá aprimorar o valor da função objetivo.

Mais formalmente, o Subproblema busca uma coluna a_j, tal que seu custo reduzido $\bar{c}_j = c_j - (u^T a_j)$ seja o mais negativo possível, em problemas com critério de minimização. A natureza e a formulação específica do Subproblema são fundamentalmente determinadas pelas propriedades estruturais do problema original, bem como detalhado em revisões abrangentes para as formulações de corte e empacotamento, como as apresentadas em WÄSCHER *et al.* (2007a) com suas respectivas topologias.

A seguir, apresenta-se a formulação geral do PMR, adaptada de LÜBBECKE e DESROSIERS (2005), cujo objetivo é encontrar a solução ótima mínima z^* com base em um subconjunto de colunas representadas por λ_j:

$$
\begin{aligned}
z^* := \min \quad & \sum_{j \in J} c_j \lambda_j \\
\text{sujeito a} \quad & \sum_{j \in J} a_j \lambda_j \geq b, \\
& \lambda_j \geq 0, \quad j \in J.
\end{aligned}
$$

Nesta formulação:

- J representa o conjunto de índices das colunas (variáveis) atualmente presentes no Problema Mestre Restrito (PMR).
- c_j é o custo associado à variável λ_j.
- λ_j é a variável de decisão que indica a quantidade da j-ésima coluna.

- a_j é o vetor coluna da j-ésima variável de decisão nas restrições do PMR.
- b é o vetor dos recursos disponíveis.

O objetivo é minimizar o custo total z^*, sujeita às restrições de demanda/capacidade e não-negatividade das variáveis.

Como discutido, o Subproblema é resolvido com o objetivo de identificar colunas que contribuam para a melhoria para o PMR. Isso é feito calculando o custo reduzido das variáveis não básicas e, assim, identificando aquelas com potencial para aprimorar a solução atual.

Segundo DESROSIERS *et al.* (2024), cada nova variável (coluna), incorporada ao PMR, corresponde a uma solução obtida do Subproblema, sendo essa escolha guiada pela análise dos custos reduzidos.

Esse comportamento pode ser compreendido com maior profundidade a partir do trabalho de LÜBBECKE e DESROSIERS (2005), o qual descreve formalmente o critério de precificação de colunas. Especificamente, o Subproblema é responsável por identificar, dentre o conjunto implicitamente dado A, uma coluna $a \in A$ que minimiza o custo reduzido. Se o valor mínimo $\bar{c}^*$ for não negativo, então a solução primal λ e a solução dual $\bar{u}$ do PMR são ótimas para o Problema Mestre original. Caso contrário, a coluna que gerou o custo reduzido mais negativo é adicionada ao PMR e o processo é repetido, conforme explicitado na citação a seguir:

Citação 2. *Assuming that we have a feasible solution, let λ and $\bar{u}$ be primal and dual optimal solutions of the RMP, respectively. When columns a_j, $j \in J$, are implicitly given as elements of a set $A \neq \emptyset$, and the cost coefficient c_j can be computed from a_j, then the subproblem or oracle $\bar{c}^* := \min\{c(a) - \bar{u}^T a \mid a \in A\}$ returns an answer to the pricing problem. If $\bar{c}^* \geq 0$, no reduced cost coefficient $\bar{c}_j$ is negative and λ (embedded in $\mathbb{R}^{|J|}$) optimally solves the MP as well. Otherwise, we add to the RMP a column derived from the oracle's answer, and repeat with re-optimizing the RMP. (Lübbecke e Desrosiers, 2005: 1008)*

A utilização de técnicas de decomposição que particionam Problemas de Otimização em reformulações baseadas em Programação Linear constitui uma alternativa eficiente frente ao desempenho insatisfatório de métodos exaustivos, como o Branch-and-Bound, em instâncias de grande porte.

Nessas situações, a limitação computacional inviabiliza a exploração completa da árvore de decisão. Ainda que técnicas de decomposição não garantam sempre limites inferiores de boa qualidade, especialmente em problemas altamente simétricos, o uso de Geração de Colunas em Programação Inteira se mostra promissor, principalmente em modelos com grandes matrizes de restrição. O algoritmo busca eliminar colunas

irrelevantes e inserir, a cada iteração, apenas aquelas que contribuem efetivamente para a melhora da solução.

Degenerescência e Ciclagem em Geração de Colunas

A degenerescência e a ciclagem em Geração de Colunas, especialmente em problemas de grande escala, podem comprometer a convergência e eficiência do algoritmo. Um problema é degenerado, no contexto da Programação Linear, quando uma solução básica viável apresenta mais variáveis básicas com valor zero que o necessário para qualificar um ponto extremo da região factível.

No algoritmo Geração de Colunas, mais precisamente no Problema Mestre Restrito, as soluções ótimas podem não retratar avanços promissores. Isso porque, mesmo com a entrada de novas colunas através do subproblema, o valor da função objetivo pode não ter alterações, desestabilizando os multiplicadores duais.

Esse comportamento é delicado devido esses multiplicadores serem utilizados como coeficiente do Subproblema, e pequenas variações em seus valores propiciam a inserção de colunas semelhantes, ou ainda, idênticas às anteriores, não melhorando a solução global. DESROSIERS e LÜBBECKE (2005) destacam que colunas degeneradas geralmente entram com valor nulo e podem, ao longo das iterações, se acumulando gerando soluções excessivamente estáticas.

É exatamente nesse cenário que a ciclagem pode vir a se manifestar. Esse processo ocorre quando, devido as repetições degeneradas, o algoritmo passa a alternar entre as mesmas soluções básicas, ou entre conjuntos muito próximo de colunas. Ou seja, o algoritmo acaba "preso"em um ciclo de soluções estagnando o valor da função objetivo, resultando em um elevado número de iterações.

A literatura propõe diferentes estratégias para mitigar esses efeitos. Técnicas de estabilização da função dual são bem utilizadas na prática, em decorrência de controlar as oscilações dos multiplicadores duais. Isso é feito, por exemplo, através de penalizações a variação entre iterações consecutivas, de forma que o subproblema não gere colunas repetidas ou instáveis. Um exemplo amplamente adotado é o *Stabilized Column Generation*, detalhado em LÜBBECKE e DESROSIERS (2010).

Outras estratégias envolvem a inclusão de penalizações artificiais na função objetivo do Subproblema, ou mesmo sua reformulação. Tais soluções garantem uma maior flexibilidade ao algoritmo, podendo explorar soluções potenciais, e não se prender em um conjunto limitado de variáveis do espaço de busca.

No capítulo 3, apresentaremos a aplicação do método de Geração de Colunas no problema clássico do Cutting Stock, por meio de uma instância de pequeno porte, a fim de detalhar cada uma das etapas do algoritmo. Para isso, a seção seguinte introduz os principais conceitos fundamentais do problema abordado, ilustrando a aplicação detalhada do algoritmo.

2.5 Cutting Stock Problem

Cutting Stock Problem (CSP) é um problema clássico da Otimização Combinatória, com formulação matemática fundamentada na Programação Inteira. A essência desse problema reside na determinação do melhor processo de cortar materiais de dimensões grandes e padronizadas, visando gerar em peças/partes menores. Esse processo acontece de modo sempre a atender uma demanda previamente estabelecida de produção, para cada tipo de peça de cada modelo gerado e, ao mesmo tempo, minimizar o desperdício de material ou o número total de cortes realizados.

A sua relevância prática é notória em setores industriais como o madeireiro, têxtil, metalúrgico, e de embalagens, por exemplo, onde a definição de padrões eficientes de corte impactam diretamente a redução de custos e, do aproveitamento dos insumos. Nesses contextos, adicionalmente, o uso ineficiente da matéria-prima implica diretamente em elevados custos operacionais e perdas financeiras, o que evidencia a necessidade de soluções otimizadas para aprimorar a eficiência produtiva, e, assim como, a gestão empresarial.

Contexto Histórico e Aplicações

Suas aplicações geralmente são em indústrias que possuem um alto volume de produção diante de um material básico (matéria-prima), a ser dividido (cortado) inteligentemente em unidades menores. É indiscutivel que existam variações de modelagem diante de cada cenário industrial a ser aplicado o Cutting Stock, possibilitando restrições adicionais ao modelo referentes a produção desejada.

Um exemplo tratado por BELOV *et al.* (2007) relata uma formulação para o Cutting Stock unidimensional, do qual as posições dos cortes gerados são relevantes no modelo. É aplicado os cortes de Gomory para melhorar seu desempenho computacional.

GOULIMIS (2020) já alertava sobre as limitações dos algoritmos clássicos para solucionar Cutting Stock, enunciando contraexemplos práticos dos quais validavam sua ineficiência, como os métodos tradicionais do modelo de Gilmore-Gomory.

DE CARVALHO (2002) revisou diferentes formulações lineares para Cutiing Stock e Bin Packing, comparando os modelos clássicos dos quais combinam padrões de corte predefinidos, com modelos fundamentados em Fluxos em Rede. Esses possibilitam restrições adicionais como, por exemplo, sequência e ordem em que os cortes são realizados, ou também, restrições de perda mínima após os cortes.

Nos trabalhos de DYCKHOFF (1990) e de WÄSCHER *et al.* (2007b) é estudado a estrutura do Cutting Stock Problem como dois conjuntos, dos quais o primeiro é definido por elementos de objetos maiores, como por exemplo, a matéria-prima disponibilizada. Já o segundo, como um conjunto de objetos menores, as demandas a serem asseguradas.

Os objetos menores são posicionados de modo a preencher o espaço total disponível nos objetos maiores, sem sobreposição e sem ultrapassar seus limites. Os autores fazem essa interpretação associando o Packing Problem ao Cutting Stock Problem, pois ambos problemas compartilham características estruturais e matemáticas em comum, mesmo possuindo objetivos distintos. Nas indústrias, ambas aplicações podem ser utilizados através de modelos híbridos, como por exemplo, na ocasião em que uma empresa dispõe da necessitada de gerar/cortar peças de uma matéria-prima e empacotá-las em um contêiner, por exemplo.

Um classificação fundamental para o Cutting Stock Problem é a dimensionalidade do tipo de corte a ser efetuado, ou seja, as dimensões geométricas que envolvem os cortes do problema são consideráveis e impactam diretamente sua modelagem.

Problemas unidimensionais são caracterizados por cortes lineares (corte de barras e rolos de papel), problemas bidimensionais e tridimensionais são caracterizados por, cortes em duas dimensões (cortes em chapas metálicas e cortes de tecidos) e três dimensões (corte de blocos e vidros), respectivamente.

Usualmente, existem abordagem conceituadas para CSP, dois exemplos seriam a maximização da área total da matéria-prima disponível, enquanto a outra visa a minimização dos desperdícios gerados após cada corte. No entanto, para essa pesquisa, abordaremos a minimização dos cortes realizados ao longo do processo produtivo.

Formulação Matemática

A modelagem tradicional assume um estoque (ilimitado ou não) de barras de comprimento fixo, a partir das quais devem ser produzidas peças menores com demandas específicas. Cada padrão de corte representa uma forma viável de dividir uma barra do estoque em várias peças, e o objetivo final é minimizar o número total dessas barras utilizadas (ou, em casos mais gerais, o custo associado).

O objetivo nesta dissertação é encontrar a melhor combinação de cortes que minimize a quantidade de cortes realizados ao longo do processo, otimizando o uso da matéria-prima e reduzindo o desperdício com cortes desnecessários. Seu modelo clássico foi apresentado por GILMORE e GOMORY (1961).

Contudo, é possível generalizá-lo em um modelo que incorpora custos relacionados na função objetivo, diferenciando do modelo proposto por Gilmore e Gomory, o que chamaremos de P_{Geral} nesta dissertação.

$$
\begin{aligned}
(P_{\text{Geral}}): \quad & \min \sum_{j=1}^{p} c_j x_j \\
\text{sujeito a:} \quad & \sum_{j=1}^{p} a_{ij} x_j \geq d_i, \\
& x_j \in \mathbb{Z}^+, \quad j = 1, \ldots, p
\end{aligned}
\tag{2.11}
$$

O problema abordado envolve a otimização do corte de barras de comprimento padrão L_i como limitante para a execução dos cortes de cada modelo desejado, e, neste caso, adotaremos todas as barras com o mesmo comprimento L_i padrão e pré-definido.

Além disso, existirá uma quantidade finita de barras disponíveis $i = 1, 2, ..., m$ para a execução de cada corte x_j, no qual j representa a quantidade de vezes que cada corte é concretizado, e varia entre $j = 1, 2, ..., p$, onde p representa o conjunto de todos os padrões de cortes possíveis a serem cometidos.

Capítulo 3

Método Proposto

3.1 Geração de Colunas Aplicada ao CSP

A modelagem do CSP é desafiadora para instâncias de grande porte devido à vasta quantidade de padrões de corte possíveis. A aplicação da Geração de Colunas emerge como uma técnica eficiente para contornar essa complexidade, e nesse contexto, o Problema Mestre (PM) original, que representa o CSP completo, é linearmente relaxado e reformulado em um Problema Mestre Restrito (PMR), contendo apenas um subconjunto inicial de padrões de corte (variáveis).

A resolução iterativa do PMR, via método Simplex, fornece a solução primal λ^* e, crucialmente, os vetores duais ótimos, u^*, associado às suas restrições de demanda.

Com base nesse vetor u^*, o Subproblema é formulado e resolvido. Para o CSP, a formulação clássica para seu subproblema é o Knapsack Problem, onde se busca identificar um novo padrão de corte que maximize o valor dual sem exceder a capacidade da barra principal a ser cortada. Mais precisamente, o subproblema visa maximizar a redução de custo na função objetivo do PMR, respeitando a restrição de comprimento da barra da matéria-prima, e utilizando os vetores duais como "valores"dos itens (pedaços de demanda a serem gerados).

O critério para seleção desse novo padrão de corte, a ser incorporado ao PMR, é de possuir o menor custo reduzido não positivo ($\bar{c} \geq 0$), do qual indicará o potencial de melhora na função objetivo. Se nenhum padrão de corte com custo reduzido negativo for encontrado, a solução atual do PMR é considerada ótima para o CSP.

Para uma nova coluna - padrão de corte - identificada pelo subproblema ser inserida na base do PMR, é necessário determinar qual coluna básica deixará a base. Para isso, aplica-se o Teste Mínimo da Razão, um critério fundamental do algoritmo Simplex.

Este teste é essencial para manter a viabilidade da solução e garantir que o algoritmo continue progredindo em direção a uma solução ótima, decidindo eficien-

temente qual variável básica deve se tornar não básica.

A coluna que sai da base simplifica a solução, removendo variáveis que não contribuem mais para a melhoria do problema, permitindo que o modelo evolua progressivamente até que a otimalidade da função objetivo seja atingida. Seguidamente, a base é atualizada, substituindo a coluna que saiu pela nova coluna identificada no Subproblema.

A Geração de Colunas é uma técnica eficiente para resolver CSP devido à quantidade de padrões possíveis de corte surgirem à medida que o número de barras, e a diversidade dos cortes, aumentam. A grande dificuldade desse problema está correlacionada na decorrência de cada padrão de corte ser uma combinação única de peças cortadas, assim como o número de padrões possíveis aumenta espontaneamente devido a complexidade do problema, tornando sua descrição, e resolução, inviáveis em tempo hábil para instâncias de grande escala, como já discutido nesta dissertação.

Ao ser solucionado com Geração de Colunas no contexto do método Simplex, além da motivação sobre o tamanho excessivo das instâncias, uma outra vantagem é a diminuição de padrões de cortes equivalentes para o CSP. Frequentemente padrões de cortes repetitivos são gerados por ser um padrão mais eficiente, aproveitando melhor a matéria-prima utilizada, causando um menor desperdício, e por isso são fornecidos como ótimos.

É possível evitar a exploração de todo o espaço de busca ao utilizar Geração de Colunas, controlando a redução excessiva de padrões idênticos, dado que, a cada iteração do algoritmo, só se é adicionado uma nova coluna (padrão de corte) se for capacitado de melhoria para a função objetivo.

A solução inicial trivial para o PMR, quando aplicada ao CSP, consiste na formulação de padrões de cortes unitários, nos quais cada padrão é responsável por produzir apenas um único tipo de peça a partir de uma barra padrão. Embora tal abordagem resulte, em geral, um elevado índice de desperdício de matéria-prima - uma vez que não é otimizado o aproveitamento total das barras disponíveis - ela apresenta como principal vantagem a garantia de viabilidade da solução inicial.

Essa característica é especialmente relevante no contexto de algoritmos baseados em Geração de Colunas, que necessitam de uma solução inicial factível para dar início ao processo iterativo de refinamento dos padrões. A aplicabilidade dessa estratégia inicial pode ser compreendida por meio das ilustrações apresentadas a seguir.

No exemplo considerado, são definidas três demandas distintas, cada uma correspondendo a um tipo específico de peça a ser obtida a partir do corte. Conforme ilustrado na imagem (3.1), para cada demanda é gerado um padrão de corte exclusivo, resultando em cortes homogêneos.Já na imagem (3.2), visualiza-se os desperdícios por essa abordagem, representados em vermelho, indicando as áreas

que o material não foi aproveitado por completo.

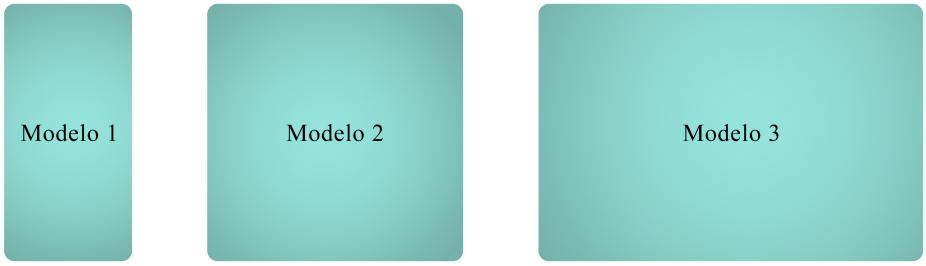

Figura 3.1: Demanda de modelos

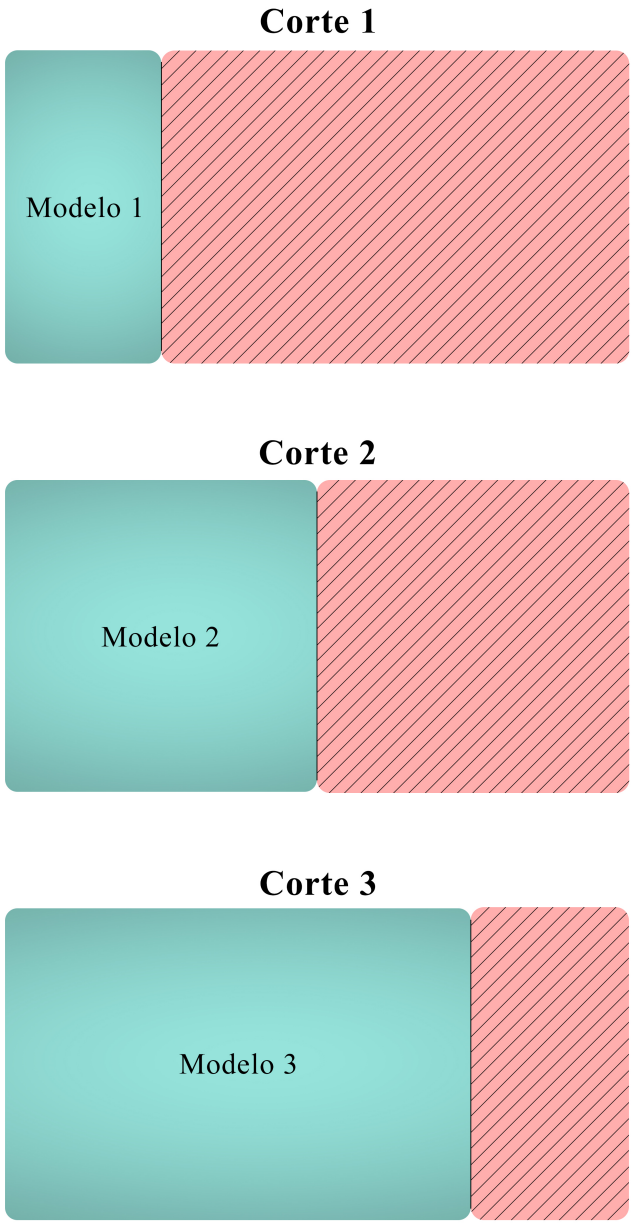

Figura 3.2: Desperdícios gerados após execução da solução trivial.

Esses desperdícios processados a partir da aplicação de padrões de cortes triviais poderão impactar significativamente o lucro final de uma produção. No entanto, tais soluções são comumente selecionadas heuristicamente para dar inicialização no algoritmo Geração de Colunas, sendo cruciais para acelerar a convergência do valor ótimo em um tempo plausível, explorando o espaço de busca de forma mais sagaz. A escolha da heurística inicial, nesse contexto, pode variar conforme as especificações do problema abordado.

A literatura apresenta diversas conjecturas e suposições que englobam a interpre-

tação final ótima final no Cutting Stock. No artigo *Counterexamples in the Cutting Stock Problem* GOULIMIS (2020) são discutidas conjecturas e suposições - comprovadas e não comprovadas - formuladas por autores relevantes na área, exibindo exemplos e contraexemplos. Goulimis discorre sobre arredondamentos de inteiros, relação mínima de itens mestres (matéria-prima) com os resíduos gerados ao final do problema (desperdícios), apresenta também questionamentos sobre divisão e contagens de padrões. Apesar de alguns indícios sugerirem a validade de determinadas conjecturas em classes específicas de instâncias, não há, até o presente momento, uma formulação teórica que comprove tais hipóteses de maneira geral.

O Cutting Stock Problem unidimensional é classificado como um problema $\mathcal{NP}$-difícil. Sua complexidade reside em selecionar, dentre um número exponencial de padrões de corte possíveis, um subconjunto que minimize o número de barras utilizadas ou o desperdício de material, enquanto atende às demandas de cada tipo de peça.

Ao resolver o CSP por meio da Geração de Colunas, o subproblema responsável por gerar novos padrões de corte ao modelo é naturalmente formulado como um *Integer Knapsack Problem*, que também é classificado como $\mathcal{NP}$-difícil MARTELLO e TOTH (1990). O objetivo do Knapsack Problem é selecionar um subconjunto de itens, com pesos e valores associados, de modo a maximizar o valor total sem exceder capacidade máxima de uma mochila.

A redução do CSP ao Knapsack no contexto da Geração de Colunas ocorre porque o subproblema busca um novo padrão de corte que maximize o "lucro"(ou minimize o "custo reduzido") a partir de uma barra de matéria-prima de comprimento limitado.

A relação entre ambos é formalmente estabelecida por meio da formulação do subproblema de precificação, cuja função é identificar um padrão de corte com custo reduzido negativo - ou seja, uma coluna que, ao ser adicionada ao Problema Mestre Restrito (PMR), potencialmente melhora a solução atual.

Enquanto o Knapsack Problem foca a maximizar o aproveitamento total de recursos e cumprir restrições de capacidade da mochila, no CSP essa lógica é migrada para a seleção de quantidades de peças a serem extraídas de uma única barra padrão, respeitando seu comprimento total e maximizando o valor dual associado às peças.

Dessa forma, a geração de padrões consiste na identificação de novas combinações de cortes da matéria-prima que reduzam o custo da solução corrente. Esse processo é validado através do Knapsack Problem, pois ele permite identificar a combinação de peças da demanda que melhor aproveita o espaço disponível na barra. E como resultado, obtem-se padrões de corte mais eficientes e, consequentemente, com menor desperdício ou o número de barras utilizadas.

O subproblema assume, a seguinte forma, conforme apresentado por MARTELLO e TOTH (1990):

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} u_i a_i \\ \text{sujeito a:} \quad & \sum_{i=1}^{n} w_i a_i \leq L, \\ & a_i \in \mathbb{Z}^+, \quad i = 1, \ldots, n \end{aligned} \tag{3.1}$$

onde:

- u_i representa o valor dual associado à restrição de demanda da peça do tipo i. É o "valor"que cada unidade da peça i contribui para a função objetivo do problema mestre.
- w_i é o comprimento da peça i.
- a_i é a quantidade de peças i no padrão de corte.
- L é o comprimento da barra padrão de matéria-prima.

3.1.1 Formulação do Problema

Cada barra a ser cortada possui um comprimento fixo, e os padrões de corte determinam como essa barra será dividida em pedaços, de acordo com as demandas exigidas. Esses pedaços podem variar em categoria e comprimento, conforme as necessidades do problema.

A formulação matemática será realizada com base na Relaxação Linear do problema apresentado em (2.11), considerando agora variáveis de decisão contínuas, denotamos por $x_j \geq 0$, que garantem a não negatividade e representam a quantidade de vezes que o padrão de corte j é utilizado.

A função objetivo visa minimizar o somatório dessas variáveis x_j, buscando assim a combinação mais eficiente de padrões de corte que atenda às exigências de demanda. O coeficiente a_{ij} indica, em unidades, os pedaços gerados a cada vez que um padrão j de corte é empregado na barra i do estoque. Embora cada padrão possa ter um custo associado c_j, neste modelo assumimos, por simplicidade, que todos os custos são unitários: $c_j = 1$.

Como existirá um limite de demanda d_l a ser garantida para as larguras de cada peça exigida, definiremos, para o PMR, a restrição $a_{ij}x_j = d_l$, que garante a produção da quantidade necessária de cada tipo de peça gerado.

O índice l representa os diferentes tipos de peças a serem produzidos e varia entre $l = 1, 2, ..., z$. Essas demandas individuais impõem um limitante mínimo de

produção para cada tipo de peças geradas, e o termo a_{ij} corresponde exatamente aos coeficientes da matriz de restrição.

Com isso, a formulação do problema PMR é dada por:

$$\begin{aligned}
(\bar{P}) : \ & \min \sum_{j=1}^{p} x_j \\
& \text{sujeito a:} \\
& \sum_{j=1}^{p} a_{ij} x_j = d_l, \quad l = 1, 2, ..., z \\
& x_j \geq 0, \qquad\qquad j = 1, 2, ..., p,
\end{aligned} \tag{3.2}$$

Cabe destacar que, nesta formulação $(\bar{P})$, não são consideradas explicitamente as restrições relacionadas às larguras das peças geradas. Essa escolha visa simplificar o modelo, focando apenas na quantidade total de peças a serem produzidas, sem incorporar as dimensões físicas de cada item.

Por fim, vale observar que cada padrão de corte j corresponde a uma configuração específica de cortes, e cada barra de matéria-prima será composta por diferentes combinações de peças - geradas pelos cortes j - conforme sua capacidade.

3.1.2 Resolução do Cutting Stock Problem utilizando Geração de Colunas

Nesta seção, ilustramos o funcionamento do algoritmo de Geração de Colunas aplicado a uma instância específica do Cutting Stock Problem. Esta etapa é fundamental para uma compreensão detalhada do processo e da lógica envolvida no algoritmo.

Consideramos um caso particular com instâncias reduzidas, no qual estão disponíveis placas de matéria-prima com comprimento de 7 metros. A demanda consiste em: 25 unidades de placas com 5 metros, 37 unidades de placas com 3 metros e 41 unidades de placas com 2 metros. A Figura (3.3) apresenta um resumo visual do problema.

Aplicamos a técnica de Geração de Colunas com o objetivo de minimizar a quantidade total de placas de matéria-prima utilizadas. Em instâncias maiores, esse algoritmo apresenta desempenho computacional consideravelmente superior aos métodos tradicionais de Programação Inteira.

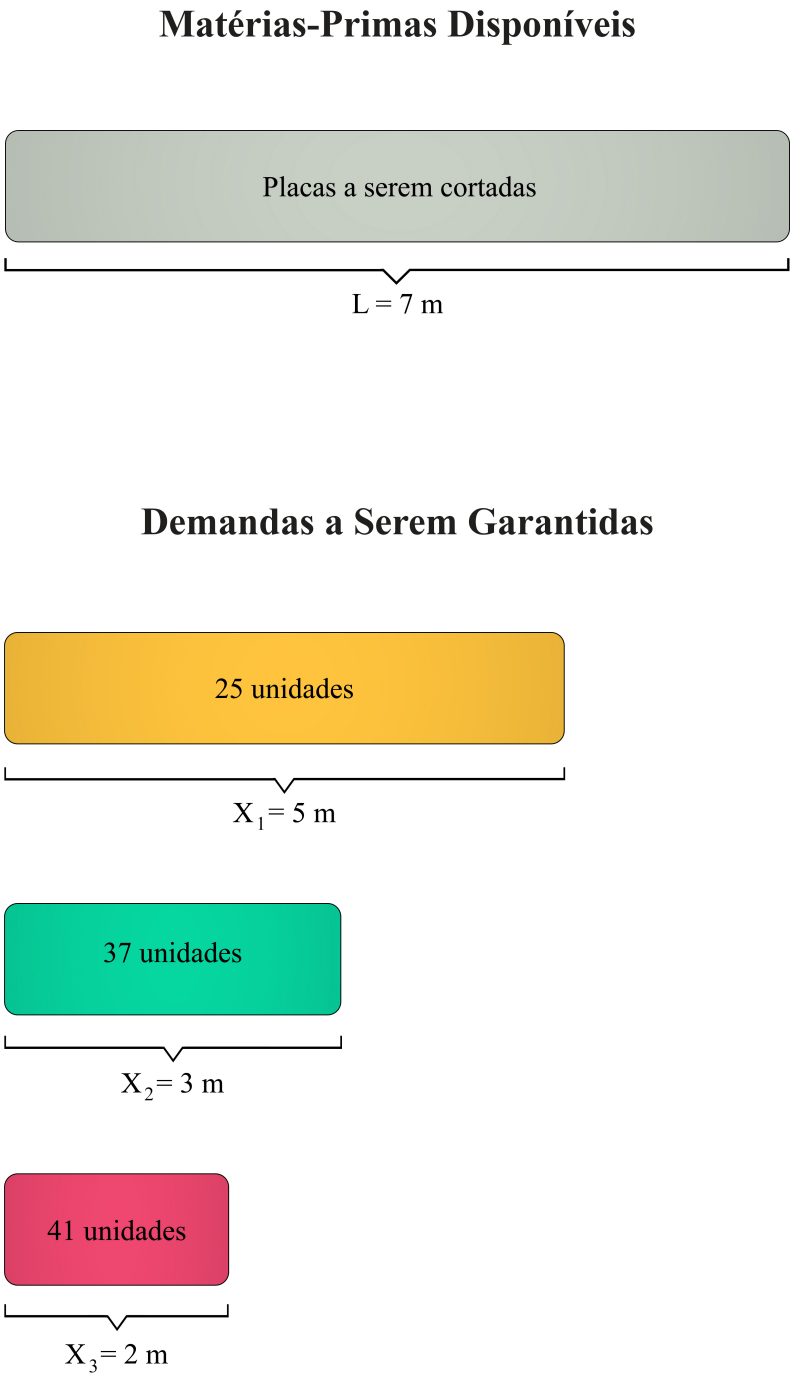

Figura 3.3: Modelo Resumido

Utilizaremos a formulação matemática apresentada em (3.2), assumindo que todos os cortes possuem custo unitário, ou seja, $c_j = 1$ para todos os padrões de corte $j = 1, 2, 3, 4, ..., p$.

Sabemos também que as placas disponíveis no estoque possuem largura fixa $L_i = 7$ metros, a serem cortadas em três tipos de placas l, com larguras x_l distintas

e previamente definidas. O índice $l = 1, 2, 3$ representa os diferentes tipos de placas demandadas após os cortes. As respectivas larguras são: $x_1 = 5$, $x_2 = 3$, $x_3 = 2$, e suas quantidades exigidas são $d_1 = 25$, $d_2 = 37$ e $d_3 = 41$.

Portanto, temos uma demanda a ser satisfeita por três tipos de placas com larguras conhecidas. O índice l refere-se a cada tipo de peça gerada. Assim, podemos reescrever o modelo como:

$$
\begin{array}{lrcrcrl}
(\bar{P}_1): \min & x_1 & + & x_2 & + & x_3 & \\
\text{sujeito a} & & & & & & \\
 & x_1 & & & & & = 25 \\
 & & & x_2 & & & = 37 \\
 & & & & & x_3 & = 41 \\
 & x_1 & , & x_2 & , & x_3 & \geq 0
\end{array}
\tag{3.3}
$$

Sabemos que existe uma relação fundamental entre os valores ótimos do Problema Inteiro (PM) e do Problema Relaxado (PMR). Ao remover a restrição de integralidade das variáveis no PMR, estas passam a assumir valores contínuos, o que amplia o conjunto de soluções viáveis em relação ao problema original.

Assim, podemos afirmar:

$$val(\bar{P}) \leq val(P_{Geral})$$

Essa desigualdade indica que o valor ótimo de $(\bar{P})$ serve como limite inferior para o valor ótimo de (P_{Geral}), já que o PMR representa uma relaxação do problema original. Esse limite inferior é útil para guiar a busca por soluções inteiras mais eficientes, em razão que o problema relaxado representa uma cota inferior para o problema inteiro, diminuindo assim o valor mínimo da função objetivo. A cota inferior encontrada será usada para alcançar uma melhor solução inteira para o problema original.

Utilizamos como solução inicial a configuração trivial, que consiste em realizar apenas um tipo de corte por placa de matéria-prima, conforme previamente mencionado e ilustrado na Figura (3.1). Como temos $x_j = d_l$, sendo $j = 1, 2, 3$ os padrões de corte e, $l = 1, 2, 3$ os respectivos tipos de peças. Essa solução estabelece que cada variável associada a um padrão de corte é igual à sua demanda correspondente.

Dessa forma, o vetor de demandas é definido por $d = (25\ 37\ 41)^T$, enquanto a matriz $\mathcal{A}$ é composta por m colunas que formam uma matriz identidade de ordem m. Isso equivale a dizer que $a_{kj} = e_{kj}$, o que caracteriza uma solução básica viável. Especificamente, neste problema com $m = 3$, a matriz $\mathcal{A}$ é composta pelas colunas a_{11}, a_{12} e a_{13}, definidas por: $a_{11}^T = (1\ 0\ 0)$, $a_{12}^T = (0\ 1\ 0)$ e $a_{13}^T = (0\ 0\ 1)$.

Esses vetores representam os padrões de corte utilizados na solução inicial trivial, na qual cada tipo de corte atende exclusivamente à sua respectiva demanda.

De forma geral, os elementos da matriz $\mathcal{A}$, denotados por a_{ij}, representam os padrões de corte gerados a cada iteração, em conformidade com as demandas para os diferentes tipos de placas a serem confeccionadas. A cada iteração do algoritmo de Geração de Colunas, uma nova coluna (isto é, um novo padrão de corte) é inserida na matriz $\mathcal{A}$ com o intuito de melhorar a solução atual, até que não seja mais possível obter melhorias.

Paralelamente, um padrão de corte menos eficiente pode ser removido, conforme o critério de razão mínima aplicado no PMR. Esse mecanismo iterativo assegura a convergência do algoritmo, promovendo uma redução significativa da complexidade computacional ao substituir colunas com baixo impacto por outras com maior potencial de otimização. Ou seja, uma coluna promissora a melhorar o problema é trocada por outra que não o influencia mais, em questões de melhoria da solução.

Para o problema abordado, com $m = 3$ e possuindo colunas iniciais a_{11}, a_{12} e a_{13}, a matriz $\mathcal{A}$ é representada, para a primeira iteração, como:

$$\mathcal{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.4}$$

A partir dessa matriz, podemos representar a relação $\mathcal{A}x = d$ da seguinte forma:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} \tag{3.5}$$

Como se trata de um sistema linear trivial, as soluções podem ser imediatamente identificadas: $x_1 = 25$, $x_2 = 37$ e $x_3 = 41$. Substituindo esses valores na função objetivo, obtém-se $(\bar{P}_1) = 103$.

Apesar da simplicidade da solução, é necessário verificar a otimalidade da base, e, para isso, calculamos o vetor dual u. Seja $\mathcal{B}$ uma matriz quadrada e inversível associada a essa solução básica viável, assim dizendo, $\mathcal{A} = \mathcal{B}$. Sabe-se também que a inversa de uma matriz identidade é a própria matriz, logo $\mathcal{B} = \mathcal{B}^{-1}$. E, com isso, teremos:

$$\mathcal{A} = \mathcal{B} = \mathcal{B}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.6}$$

Dessa forma, definiremos o cálculo da solução básica, a qual é estabelecida por $x_B = B^{-1}d$:

$$x_B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} \tag{3.7}$$

Com isso, podemos concluir e afirmar, de fato, a solução para essa iteração, como sendo:

$$x_B = \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} \tag{3.8}$$

Sabemos que os custo básicos são $c_B = (1\ 1\ \ldots\ 1)^T \in R$. Definiremos o vetor dual u e z_j como sendo, respectivamente, $u = c_B^T B^{-1}$ e $z_j = ua_j$. Ou ainda, ao substituirmos o vetor dual em z_j, podemos reescrevê-lo como, $z_j = c_B^T B^{-1} aj$.

Fazendo as devidas substituições na forma matricial, teremos:

$$u = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \tag{3.9}$$

Iniciamos, nesta etapa, o processo de cálculo dos custos reduzidos. No algoritmo Simplex, os custos reduzidos representam um critério essencial para a escolha da variável que deverá entrar na base em cada iteração, com o objetivo de melhorar o valor da função objetivo. Sua formulação é dada pela expressão em (3.10).

No método Geração de Colunas, o PMR desempenha o papel como o conjunto das variáveis básicas do Simplex, enquanto o Subproblema assume a função de identificar as variáveis não básicas. Dessa forma, em um processo iterativo, resolve-se o PMR utilizando o Simplex, e os vetores duais obtidos são utilizados na formulação do

Subproblema.

A função objetivo do Subproblema é formulada de modo que sua otimização identifique uma coluna que apresenta o menor (mais negativo) custo reduzido, mas sempre com base na solução dual corrente. Caso seja encontrada uma coluna com custo reduzido negativo, ela é inserida no PMR e o processo iterativo é reiniciado. Contudo, caso contrário, se o Subproblema não consegue encontrar uma coluna com custo reduzido negativo, isso indica que a solução atual do PMR é ótima.

A escolha da nova variável não básica a ser introduzida na base é guiada pelo critério de parada denominado Teste de Otimalidade. Em problemas de minimização, ele finaliza o algoritmo quando todos os valores dos custos reduzidos das variáveis não básicas são negativos (em problemas de maximização, esse critério é de positividade).

É importante destacar que existem outros critérios de parada ao longo do algoritmo, a saber, como problema ilimitado relacionado ao teste da razão e a viabilidade do problema.

O custo reduzido é associado a cada variável x_j de cada solução pertencente a base atual, e sua entrada na base depende do valor de $\bar{c}_j$. Reafirmando:

$$\bar{c}_j = c_j - z_j, j = 1, 2, ..., p \tag{3.10}$$

A etapa de análise dos custos reduzidos ocorre no procedimento conhecido como *Pricing*, no contexto do algoritmo Simplex. Nessa etapa, busca-se identificar a variável não básica que deverá entrar na base, com o intuito de melhorar a solução atual da função objetivo.

Como o problema em questão é de minimização, o objetivo é encontrar colunas cujo custo reduzido seja negativo, que ao serem adicionadas ao modelo, contribuam para a redução dos cortes realizados. Formalmente, deseja-se $\bar{c}_j \geq 0$ para todos os $j = 1, 2, ..., p$.

A formulação da função objetivo desse Subproblema, é realizada de modo a resumir o custo reduzido. Para isso, é utilizado substituições dos custos reais unitários, assim como, substituições de z_j pelos multiplicadores duais associados às restrições da solução corrente.

Assim, $z_j = u_k a_{kj}$, com $k = 1, 2, 3$ representando as linhas/restrições a serem inseridas no PMR a partir da coluna gerada no Subproblema (considerando as três demandas presentes na modelagem), e $j = 1, 2, ..., p$.

A função objetivo para a minimização dos tipos de corte é, portanto, expressa por:

$$\min\ 1 - u_k a_{kj},\ \forall j \tag{3.11}$$

Substituindo os valores de $k = 1, 2, 3$, obtém-se:

$$\min\left[1 - u_1 a_{1j} + u_2 a_{2j} + u_3 a_{3j}\right],\ \forall j \tag{3.12}$$

Considerando que o vetor dual nessa iteração é $u = (1\ 1\ 1)$, e que estamos analisando $j = 1, 2, ..., p$ colunas (tipos de cortes) a serem minimizadas, a função objetivo pode ser reescrita como:

$$\text{minimizar}\ \ 1 - (a_{1j}\ +\ a_{2j}\ + a_{3j}),\ \forall j \tag{3.13}$$

Com base na formulaçao acima, o Subproblema é definido como um *Knapsack Problem*, como já discutido nesta dissertação. Sua finalidade é identificar um novo padrão de corte viável que respeite o comprimento máximo permitido da barra de matéria-prima. Isso corresponde a resolver um Problema da Mochila Inteiro, cujo objetivo é maximizar a vantagem proporcionada por um novo padrão em relação aos existentes.

Nos trabalhos de GILMORE e GOMORY (1961, 1963) foi difundido o uso Knapsack como Subproblema para o CSP, dentro do contexto da Geração de Colunas. Neste cenário, cada "item"a ser disposto na mochila corresponde aos tipos de pedaços - com diferentes larguras - a serem cortados. O "peso"de cada item refere-se aos tamanhos dos pedaços cortado - demandas geradas.

Além disso, a "capacidade"da mochila é correlacionada com a dimensão total da matéria-prima a ser dividida. Por fim, o "benefício"de cada item a caber ma mochila está relacionado com o vetor dual u da restrição associada àquele tipo de pedaço. Isso ocorre porque os vetores duais refletem o ganho potencial na função objetivo do PMR associado a uma unidade adicional de demandas de cada tipo de pedaço da demanda, precificando o valor de um novo padrão de corte.

Com base no teorema da *Dualidade Forte*, é possível reformular um problema de minimização em circunstância de sua dualidade, dado que o valor ótimo primal será sempre idêntico ao valor ótimo dual. Em outras palavras, os valores ótimos de ambos problemas serão iguais em magnitude.

Assim, a função objetivo do Subproblema é expressa como (3.14), e ajustando-se a modelagem do Knapsack Problem, será expressa agora como:

$$- \max\ (a_{1j}\ +\ a_{2j}\ + a_{3j}),\ \forall j \tag{3.14}$$

As restrições do Subproblema envolve uma restrição de capacidade (demanda) e outra de integralidade. Assumindo que todas as barras a serem cortadas possuem o mesmo comprimento $L_i > 0$, e os j pedaços resultantes tenham comprimentos ordenados $0 < \Lambda_1 < \Lambda_2 < ... < \Lambda_j < L_i$, definidos por uma sequência crescente de valores entre 0 e L_i.

Dessa forma, a nova coluna a ser gerada é denotada por:

$$\alpha = (\alpha(1)\ \alpha(2)\ .\ .\ \alpha(p))^T$$

onde α representa a nova coluna a ser introduzida na base na iteração corrente, e $\alpha(k)$ representa a quantidade de peças do tipo k no padrão de corte gerado.

A soma ponderada $\Lambda_1\alpha(1) + \Lambda_2\alpha(2) + ... + \Lambda_p\alpha(p) \leq L_i$ é suficiente para definir a viabilidade dos cortes a serem executados, não ultrapassando o comprimento total da barra de matéria-prima. Assim, a restrição de demanda é formulada como:

$$l_k a_{kj} \leq L_i,\ k = 1, 2, 3\ ,\ j = 1, 2, ..., p \text{ e } i = 1, 2, ..., m \tag{3.15}$$

No caso específico do problema em questão, o comprimento fixo da barra é $L_i = 7\ metros$, e pedaços possuem larguras previamente estabelecidas de $x_l = 5, 3, 2$ para cada $l = 1,\ 2,\ 3$, respectivamente.

Portanto, o Subproblema G (3.16), com a função objetivo e as restrições de integralidade e demanda, é formalmente representado como:

$$\begin{aligned} (G):\ & - \max \sum_{k=1}^{3} u_{kj} a_{kj} \\ & \text{sujeito a: } \sum_{k=1}^{3} l_k a_{kj} \leq L_i, \\ & a_{kj} \in \mathbb{Z}^+ \quad \forall j \end{aligned} \tag{3.16}$$

Substituindo os dados numéricos de (3.14) e (3.15) em (3.16), teremos G_1 como nosso primeiro Subproblema:

$$
\begin{array}{rlllllll}
(G_1) : \text{- max} & a_{1j} & + & a_{2j} & + & a_{3j} & & \\
\text{sujeito a} & & & & & & & \\
& 5a_{1j} & + & 3a_{2j} & + & 2a_{3j} & \leq 7 & \\
& a_{1j} & , & a_{2j} & , & a_{3j} & \in \mathbb{Z}^+ &
\end{array}
\tag{3.17}
$$

Se o custo reduzido associado à solução de G_1 for negativo, o novo padrão de corte, representado pela coluna α, será incorporado à base do PMR. Caso contrário, a matriz B é considerada ótima para o problema relaxado, indicando a inexistência de padrões relevantes capazes de melhorar a solução atual.

O algoritmo de Geração de Colunas converge quando todos os custos reduzidos nas iterações são maiores ou iguais a zero - para minimização. É importante esclarecer que, a cada iteração, a nova coluna a ser inserida na base será definida por $\alpha(p)$, onde p denota o estilo de corte aplicado até o momento.

Para o Subproblema G_1, a solução ótima obtida é dada por $a_{14} = 0$, $a_{24} = 1$ e $a_{34} = 2$, resultando na coluna $\alpha(4) = (0\ 1\ 2)^T$. O custo reduzido correspondente $\bar{c}_4 = c_4 - z_4 = -2$, indica que o padrão gerado melhora efetivamente a solução. Consequentemente, essa nova coluna será adicionada à base do PMR, já definido em (3.3).

Logo, o problema $(\bar{P})$ poderá ser reformulado como $\bar{P'}_1$:

$$
\begin{array}{rlllllllll}
(\bar{P}'_1) : \text{minimizar} & x_1 & + & x_2 & + & x_3 & + & x_4 & & \\
\text{sujeito a} & & & & & & & & & \\
& x_1 & & & & & & & = & 25 \\
& & & x_2 & & & + & x_4 & = & 37 \\
& & & & & x_3 & + & 2x_4 & = & 41 \\
& x_1 & , & x_2 & , & x_3 & , & x_4 & \geq & 0
\end{array}
\tag{3.18}
$$

Com a adição dessa nova coluna, torna-se possível avaliar o impacto do novo padrão de corte $\alpha(4) = (0\ 1\ 2)^T$ na função objetivo e prosseguir com as próximas iterações do algoritmo. Para isso, o modelo $(\bar{P}_1)$ é reescrito conforme a forma padrão (3.19), utilizando a variável x_4 para representar o novo padrão de corte. Nessa equação, z_B corresponde ao valor da função objetivo corrente, e $\bar{c}$ representa o custo reduzido da nova coluna associado a variável que a representa:

$$
\begin{aligned}
&\min z = z_B + \bar{c}_j x_j \\
&x_B = B^{-1}b - B^{-1}a_j x_j \geq 0
\end{aligned}
\tag{3.19}
$$

Substituindo os dados do modelo (3.18) em (3.19), obtemos:

$$
\begin{aligned}
&(\bar{P}_1'') \min z = 103 - 2x_4 \\
&x_1 = 25 \\
&x_2 = 37 - x_4 \\
&x_3 = 41 - 2x_4
\end{aligned}
\tag{3.20}
$$

A seguir, aplica-se o Teste da Razão, fundamental nos algoritmos Simplex e Geração de Colunas para avaliar a viabilidade da solução e promover melhora a cada iteração. Na Geração de Colunas, em particular, o teste impacta diretamente que a adição da nova coluna - novo padrão de corte - não inviabilize o PMR.

Examinando ($\bar{P}''1$) em (3.20), observa-se-se que o novo padrão de corte x_4 não afeta diretamente x_1, dado que o coeficiente de x_4 na linha correspondente à variável x_1 é nulo. Portanto, a análise de factibilidade será concentrada em x_2 e x_3, cujos valores dependem diretamente de x_4.

Após a entrada de x_4, devemos garantir que x_2 e x_3 permaneceçam não negativos, o que solicita analisar o seguinte critério:

$$
\begin{aligned}
&x_2 : \ 37 - x_4 \geq 0 \\
&x_3 : \ 41 - 2x_4 \geq 0
\end{aligned}
\tag{3.21}
$$

Assim, o máximo valor que x_4 pode assumir será determinado pelo menor dos limites encontrados nas desigualdades anteriores, garantindo a não negatividade de x_2 e x_3 . O valor de x_4 será:

$$
\min(37, 20.5) = 20.5 \tag{3.22}
$$

Como a coluna associada a x_3 determina o mínimo no Teste da Razão, será x_3 a sair da base. Dessa forma, é possível fazer a substituições de variáveis. A coluna $\alpha(4)$, importada do Subproblema e correspondente agora a variável x_4 no PMR, entrará na base, reconfigurando o Problema Mestre Restrito para a iteração, conforme (3.24).

Logo, x_4 deve pertencer ao intervalo $0 \leq \ x_4 \leq 20,5$, garantindo que x_2 e x_3 - variáveis não básicas - permanecem viáveis e respeitando a solução factível. Dentro desse limite estabelecido, é possível ver que o valor de z em (3.20) diminuirá de 103 para 62, sinalizando a existência de mais uma iteração:

$$
z = 103 - 2(20.5) = 62 \tag{3.23}
$$

Portanto, o PMR atualizado para a segunda iteração é definido por $\bar{P}_2$:

$$
\begin{aligned}
(\bar{P}_2): \min \quad & x_1 + x_2 + x_4 \\
\text{sujeito a} \quad & \\
& x_1 = 25 \\
& x_2 + x_4 = 37 \\
& 2x_4 = 41 \\
& x_1 \;,\; x_2 \;,\; x_4 \geq 0
\end{aligned}
\tag{3.24}
$$

A nova base $\mathcal{B}$ é composta pelas colunas presentes no modelo $\bar{P}_2$ (3.24):

$$
\mathcal{A} = \mathcal{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \tag{3.25}
$$

Reformulando $\bar{P}_2$ (3.24) para forma matricial, $Ax = d$:

$$
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} \tag{3.26}
$$

Deve-se calcular a inversa $\mathcal{B}^{-1}$ para assim, calcular a solução básica $x_B = \mathcal{B}^{-1}d$. Então, teremos:

$$
\mathcal{B}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{pmatrix} \tag{3.27}
$$

$$
x_B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{pmatrix} \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} = \begin{pmatrix} 25 \\ 16.5 \\ 20.5 \end{pmatrix} \tag{3.28}
$$

Com isso, podemos calcular os vetores duais $u = c_B^T \mathcal{B}^{-1}$:

$$
u = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \tag{3.29}
$$

Dado isso, iremos modelar o Subproblema para a segunda iteração como G_2 (3.30). E sua solução ótima para o padrão de corte $j = 5$ é dada por $a_{15} = 0$, $a_{25} = 2$ e $a_{35} = 0$, ou seja, $\alpha(5) = (0\ 2\ 0)^T$. Assim, seu custo reduzido não positivo é dado por $\bar{c}_5 = 1 - 2 = -1$.

$$
\begin{array}{lrcrcrl}
(G_2) : \text{- max} & a_{1j} & + & a_{2j} & & & \\
\text{sujeito a} & & & & & & \\
 & 5a_{1j} & + & 3a_{2j} & + & 2a_{3j} & \leq 7 \\
 & a_{1j} & , & a_{2j} & , & a_{3j} & \in \mathbb{Z}^+
\end{array}
\tag{3.30}
$$

Dado isso, ao efetuar a mudança de variáveis, será adicionado o novo padrão de corte x_5 no PMR apresentado em $\bar{P}_2$ (3.24), e será reformulado para $\bar{P'}_2$:

$$
\begin{array}{lrcrcrcl}
(\bar{P}_2') : \min & x_1 & + & x_2 & + & x_4 & + & x_5 \\
\text{sujeito a} & & & & & & & \\
 & x_1 & & & & & & = 25 \\
 & & & x_2 & + & x_4 & + & 2x_5 = 37 \\
 & & & & & 2x_4 & & = 41 \\
 & x_1 & , & x_2 & , & x_4 & , & x_5 \geq 0
\end{array}
\tag{3.31}
$$

Verificaremos o impacto desse novo padrão de corte no valor da função objetivo desse problema. Para isso, definiremos $\bar{P''}_2$ como:

$$
\begin{aligned}
&(\bar{P}_2'') \min z = 62 - x_5 \\
&x_1 = 25 \\
&x_2 = 37 - 2x_5 \\
&x_4 = 20.5
\end{aligned}
\tag{3.32}
$$

É possível ver que os coeficientes x_1 e x_4 não possuem contribuição nesse padrão de corte x_5, então o Teste da Razão confirma que o novo padrão de corte não assuma valores negativos, já que $x_2 = 8.25$. Diante disso, a coluna a sair da base será a_2.

Logo podemos fazer a troca de a_2 por a_5 e calcular o valor de z como:

$$
z = 62 - 8.25 = 53.75 \tag{3.33}
$$

Sendo assim, podemos iniciar a terceira iteração e gerar um novo PMR definido por $\bar{P}_3$:

$$
\begin{array}{lllllllll}
(\bar{P}_3): \min & x_1 & + & x_5 & + & x_4 & & \\
\text{sujeito a} & & & & & & & \\
& x_1 & & & & & = & 25 \\
& & & 2x_5 & + & x_4 & = & 37 \\
& & & & & 2x_4 & = & 41 \\
& x_1 & , & x_5 & , & x_4 & \geq & 0
\end{array}
\tag{3.34}
$$

A nova $\mathcal{B}$ será dada pelas colunas de 3.34:

$$
\mathcal{A} = \mathcal{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix} \tag{3.35}
$$

Reformulando 3.24 para forma matricial $Ax = d$:

$$
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_5 \\ x_4 \end{pmatrix} = \begin{pmatrix} 25 \\ 37 \\ 41 \end{pmatrix} \tag{3.36}
$$

Calculando a inversa $\mathcal{B}^{-1}$ para gerar a solução básica $x_B = \mathcal{B}^{-1}d$, teremos:

$$
\mathcal{B}^{-1} = \begin{pmatrix} 1 & -0.5 & 0 \\ 0 & 0.5 & -0.5 \\ 0 & 0 & 0.5 \end{pmatrix} \tag{3.37}
$$

Com isso, podemos calcular os vetores duais $u = c_B^T \mathcal{B}^{-1}$:

$$
u = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & -0.5 & 0 \\ 0 & 0.5 & -0.5 \\ 0 & 0 & 0.5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \tag{3.38}
$$

Dado isso, iremos modelar o subproblema para a terceira iteração como G_3 (3.39) do qual gerará o novo padrão de corte $j = 6$.

$$
\begin{array}{lllllll}
(G_3): \text{- max} & a_{16} & & & & & \\
\text{sujeito a} & & & & & & \\
& 5a_{16} & + & 3a_{26} & + & 2a_{36} & \leq 7 \\
& a_{16} & , & a_{26} & , & a_{36} & \in \mathbb{Z}^+
\end{array}
\tag{3.39}
$$

A solução ótima para o padrão de corte $j = 6$ é dada por $a_{16} = 0$, $a_{26} = 1$ e $a_{36} = 0$, assim como seu custo reduzido é exatamente $\bar{c}_6 = 1 - 1 = 0$. Com isso, chega-se em

um critério de parada, e logo podemos descrever a solução ótima como $z^* = 53.75$, do qual gerou 25 peças com x_1, 20.5 de x_4 e $x_5 = 8.25$ encontrado na iteração anterior.

A solução ótima contínua para $\bar{P}$, ou seja, a relaxação linear do problema, teve valor de 53.75. Contudo, o problema original P é formulado como um Problema de Programação Inteira e, portanto, é necessário encontrar uma solução inteira viável.

Como estamos lidando com um problema de minimização linear, sabemos que a solução ótima do problema relaxado sempre satisfará $val(\bar{P}) \leq val(P_{Geral})$, ou seja, a solução relaxada fornece um limite inferior para a solução inteira ótima.

É frequentemente comum recorrer a técnicas de arredondamento de soluções após obter uma solução relaxada aplicada ao Cutting Stock Problem. GILMORE e GOMORY (1963) sugeriram o arredondamento das soluções fracionárias para encontrar facilmente uma solução inteira, combinando métodos heurísticos com a solução obtida na Relaxação Linear.

Como estamos com um problema de minimização, utilizar o arredondamento para cima das soluções contínuas garante o atendimento de todas demandas exigidas. Contudo, para esquivar de excesso de itens a serem cortados, a utilização de heurísticas estabeleceu uma boa estratégia para reduzir desperdícios, como, por exemplo, substituições de um padrão de corte por combinações alternativas ou pequenas modificações no padrão selecionado.

Dessa forma, afirmamos, através de uma heurística de arredondamento, que uma solução ótima para o problema original será $z = 54$ com as configurações de padrões de cortes sendo $x_1 = 25$, $x_5 = 8$ e $x_4 = 21$ atendendo a demanda exigida.

É importante destacar que, as permutações em cada barra a ser cortada é desconsiderada para essa solução, uma vez que estamos lidando com cortes unidimensionais. Na imagem (3.4) é possível visualizar os padrões de cortes selecionados, assim como as perdas de matéria-prima ocasionadas representadas pela cor vermelha hachurada.

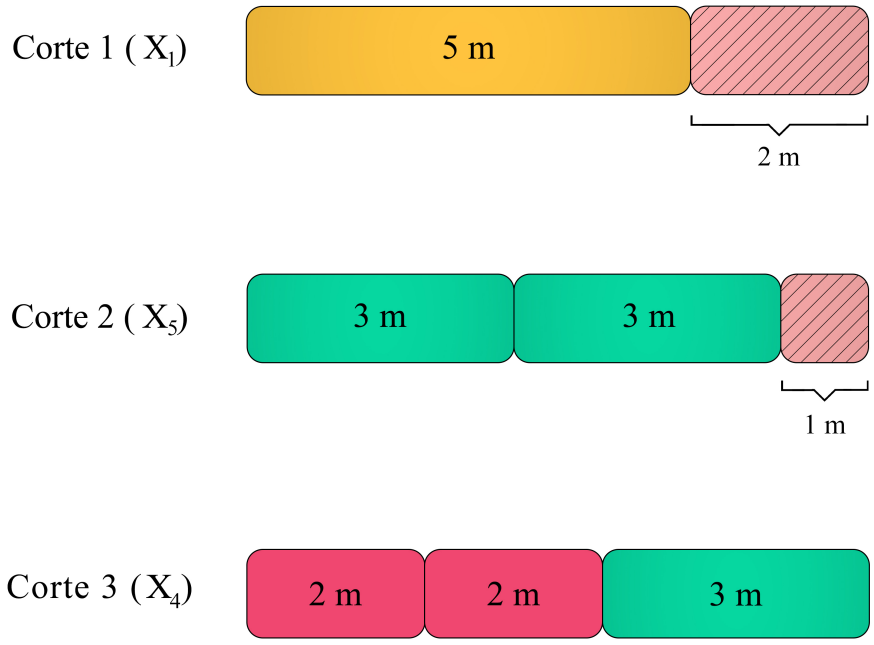

Figura 3.4: Solução final

3.2 Geração de Colunas em Programação Linear Generalizada

A Programação Linear Generalizada (PLG) abrange problemas que apresentam modificações em variáveis, colunas e/ou nos coeficientes de custos, permitindo a possibilidade da não linearidade. Essa abordagem amplia a flexibilidade na modelagem de problemas mais complexos, utilizando técnicas de aproximação e iteratividade para alcançar o ótimo.

O segundo problema analisado nesta dissertação tem como base o trabalho de LASDON (1970), apresentado em seu livro Optimization Theory for Large Systems. Neste é considerado manipulações aplicadas em problemas de otimização mais robustos, possuindo a característica de não linearidade das funções objetivo e/ou restrições envolvidas.

O autor apresentou a problemática de minimização, onde a variável x_i é não negativa, e o vetor $(c_i\ a_i)$, $\forall i$ é constante. Além disso, estabeleceu a hipótese de que $(c_i\ a_i) \in S_i$, sendo S_i é definido como um poliedro convexo limitado. Também afirmou que o presente problema, por envolver o produto entre variáveis, ou seja, quantidades desconhecida, é caracterizado como não linear, e ainda, possivelmente

não convexo. A seguir, apresenta-se sua modelagem:

$$\begin{aligned} \min \quad & z = \sum_i c_i x_i \\ \text{sujeito a} \quad & \sum_i a_i x_i = b, \quad x_i \geq 0 \\ & (a_i \ c_i) \in S_i \end{aligned} \tag{3.40}$$

Com o objetivo de transformar o problema acima (3.40) em um Problema Linear, o autor utilizou a hipótese de que $(c_i \ a_i) \in S_i$. Diante do exposto, reescreveu toda coluna $(c_i \ a_i)^T$ como uma combinação convexa, já que S_i é limitado:

$$\begin{pmatrix} c_i \\ a_i \end{pmatrix} = \sum_k \lambda_{ik} \begin{pmatrix} c_{ik} \\ a_{ik} \end{pmatrix}, \text{ com } \sum_k \lambda_{ik} = 1 \text{ e } \lambda_{ik} \geq 0. \tag{3.41}$$

Com isso, declarou $(c_{ik} \ a_{ik})^T$ como pontos extremos conhecidos do poliedro S_i. Á vista disso, reescreveu o modelo (3.40) realizando substituições em sua modelagem, com o propósito do problema tornar-se linear.

Substituindo a expressão (3.41) em (3.40), obteve:

$$\begin{aligned} \min \quad & \sum_i \sum_k \lambda_{ik} c_{ik} x_i \\ \text{s.a.} \quad & \sum_i \sum_k \lambda_{ik} a_{ik} x_i = b \end{aligned} \tag{3.42}$$

Definindo uma nova variável $u_{ik} = \lambda_{ik} x_i$ não negativa, o autor procedeu à mudança de variável (3.41), e obteve a seguinte modelagem:

$$\begin{aligned} \min \quad & \sum_i \sum_k c_{ik} u_{ik} \\ \text{s.a.} \quad & \sum_i \sum_k a_{ik} u_{ik} = b \\ & u_{ik} \geq 0 \end{aligned} \tag{3.43}$$

Como a formulação acima, ele atingiu um problema linear, uma vez que c_{ik}, a_{ik} e b são agora valores conhecidos, com isso, é possível solucioná-lo encontrando u^*_{ik} como solução ótima, através do algoritmo o Simplex, por exemplo. Entretanto, a formulação inicial em (3.40) precisa ser recuperada perante suas variáveis originais, a saber x_i, c_i e a_i. Fazendo operações que envolvem a variável u_{ik}, o autor recupera o valor da variável ótima x^*_i de (3.40) com a seguinte expressão:

$$x^*_i = \sum_k u^*_{ik} \tag{3.44}$$

E para a recuperação de c_i^* e a_i^*, o autor utilizou da formulação (3.41) para recuperá-los, uma vez que os vértices são todos conhecidos. No entanto, é preciso encontrar o valor de λ_{ik} para efetuar a devida substituição, que é expressada por:

$$\lambda_{ik} = \frac{\mu_{ik}}{x_i} \tag{3.45}$$

Dessa forma, é possível efetuar a recuperação momentânea dos valores de c_i^* e a_i^* a partir de (3.41), e encontra-se assim os valores de x_i^*, c_i^* e a_i^*.

3.2.1 O Problema Abordado

Este problema difere tanto do modelo clássico de Programação Linear, em que os parâmetros a_{ij}, b_i e c_j são todos conhecidos, quanto da formulação proposta por LASDON (1970). A diferença para esta nova proposta está na retratação de cada a_{ij} sendo considerada variáveis em um intervalo estabelecido entre $[\alpha_{ij},\ \beta_{ij}]$, ou seja, α_{ij} e β_{ij} são parâmetros conhecidos em um problema.

Diante disso, o modelo transfigura-se em um modelo de Otimização Não-Linear e Não-Convexo, como em (3.46):

$$\begin{aligned} min \quad & z = \sum_{j \in J} c_j x_j \\ \text{s. a} \quad & \sum_{j \in J} a_{ij} x_j = b_i, \quad i \in I \\ & x_j \geq 0, \quad j \in J \\ & \alpha_{ij} \leq a_{ij} \leq \beta_{ij}, \quad i \in I, j \in J \end{aligned} \tag{3.46}$$

Conjectura: O valor ótimo da variável a_{ij} em (3.46), será ou α_{ij} ou β_{ij}, para todo $i \in I$ e para todo $j \in J$.
O problema descrito abrange uma quantidade finita e não-enumerável de colunas $a_j = (a_{1j}, a_{2j}, \ldots, a_{mj})^T$, cada uma pertencente a um paralelepípedo em $\mathbb{R}^m$, definido pela chamada *restrição de caixa* $[\alpha_{ij}, \beta_{ij}]$. Cada paralelepípedo possui 2^m vértices, que serão considerados candidatos à entrada na base.

A partir dessa conjectura, o modelo disporá uma quantidade finita de restrições e variáveis, isso porque será considerado apenas as colunas correspondentes aos vértices desses paralelepípedos. Por essa razão, desenvolvemos um algoritmo inspirado no método Simplex, mas incorporando os princípios da técnica de Geração de Colunas, no qual uma nova coluna a_j é avaliada a cada iteração para possível inclusão na base.

3.2.2 O algoritmo desenvolvido

Seja $A = (a_{ij})$ a matriz $m \times n$ associadas às variáveis a. Definimos uma base B para ser uma matriz $m \times m$ composta por m colunas linearmente independentes de A, correspondentes às variáveis cujos valores foram previamente fixados. De maneira análoga ao método Simplex, o primeiro passo é verificar a otimalidade da base B considerada. No caso de um problema de minimização, a condição necessária será:

$$c_B B^{-1} N - c_N \leq 0. \tag{3.47}$$

Note que, neste contexto, não consideramos explicitamente a matriz N, uma vez que se trata de uma submatriz de A composta de variáveis. Mas, ainda sim, é preciso verificar a existência de valores possíveis para essas variáveis em uma dada coluna j que violem o teste de otimalidade. Em outras palavras, dado a_j, a j-ésima coluna de A, procuramos identificar valores admissíveis para a_j tais que:

$$c_B B^{-1} a_j - c_j > 0 \tag{3.48}$$

Com uma abordagem de Geração de Colunas, resolveremos um subproblema w (3.50), objetivando verificar a possibilidade de valores - em uma dada coluna a_j - que não satisfaça o teste de otimalidade. Nesse caso, a coluna a_j entraria na base. Para verificá-la, resolveremos o problema:

$$\begin{aligned} w \;=\; & \max_{j \in J} \quad (c_B B^{-1} a_j - c_j) && (3.49) \\ & \text{s. to} \quad \alpha_{ij} \;\leq\; a_{ij} \;\leq\; \beta_{ij}\,, \qquad i \in I && (3.50) \end{aligned}$$

Este é um problema simples que pode ser resolvido por inspeção diante das restrições de caixa. Seja $u = c_B B^{-1}$, o vetor de coeficientes na função objetivo de (3.50). Sendo assim, para todo $j \in J$,

- se $u_i > 0$, defina $a_{ij} = \beta_{ij}$;
- caso contrário, defina $a_{ij} = \alpha_{ij}$.

Se $w > 0$, uma nova coluna é gerada e poderá ser incorporada à base. É importante destacar que, diferentemente do clássico algoritmo Simplex, a condição de otimalidade deve ser verificada também para as colunas associadas à base. Assim, podem existir outros valores para essas variáveis que resultem em uma redução do valor da função objetivo em (3.50).

A escolha da coluna que deve deixará a base segue a mesma regra adotada no algoritmo Simplex, considerando a coluna recém-gerada. Utilizando $\bar{x}_B = B^{-1}b$, a coluna substituída na base é aquela de índice i, tal que:

$$i = \arg \min_{i \in B,\, \bar{x}_i/a_{ij} > 0} \left\{ \frac{\bar{x}_i}{a_{ij}} \right\}.$$

Em cada escolha de mudança de base, será enumerado todos os valores $c_j - z_j,\ j = 1, 2, ..., n$ para obter:

$$c_k - z_k = \min_{j=1,2,\ldots,n} (c_j - z_j). \tag{3.51}$$

Um pseudo-algoritmo é dado em Algoritmo 1.

Algorithm 1: Algoritmo proposto.

Entrada: c, b, α, β e base inicial $\mathcal{B}$
Saída : A, x

```
1
2  atualizado = verdadeiro
3  while atualizado do
4      atualizado = falso
5      for j = 1, ..., n do
6          w, a_j = Resolver subproblema (3.50)
7          if w > 0 then
8              Coluna a_j entra na base
9              Coluna i = arg min_{i∈B} {x̄_i/a_ij | a_ij > 0} sai da base
10             atualizado = verdadeiro
11             interromper
```

Note também que, durante o procedimento descrito no Algoritmo 1, pode ocorrer de uma mesma base conter múltiplas colunas associadas a uma mesma variável x_j, em contraste com o algoritmo Simplex.

Por exemplo, suponha $m = 3$, $n = 5$ e que as colunas em $\mathcal{B}$ estejam associadas às variáveis x_1, x_2 e x_5. Na linha 6 do Algoritmo 1, outra coluna associada a x_2 (com custo reduzido positivo) é selecionada para entrar na base. Em seguida, a regra da linha 9 do Algoritmo 1 determina que a variável x_1 deve deixá-la, de modo que a nova base passa a conter colunas associadas a x_2, x_2 e x_5.

Para distringuir essas duas ocorrência de cópias de x_2, iremos denotá-las por $x_2^{(1)}$ e $x_2^{(2)}$.

Quando o Algoritmo 1 se encerra - isto é - quando todas as colunas possuem custo reduzido menor ou igual a zero - obtém-se uma solução ótima para o problema não-linear original (3.46), de forma análoga ao procedimento descrito em (3.44) e (3.45).

Seja R_j, para $j \in J$, a quantidade de colunas associadas a x_j na base. Então:

$$x_j^* = \sum_{k=1}^{R_j} \overline{x}_{j(k)} \tag{3.52}$$

e

$$a_j^* = \frac{1}{x_j^*} \sum_{k=1}^{R_j} \overline{x}_{j(k)}\, a_{j(k)} \tag{3.53}$$

Em outras palavras, o valor ótimo da coluna a_j corresponde a uma combinação convexa dos valores das colunas em $\mathcal{B}$ associadas à variável x_j.

3.2.3 Exemplo Numérico

Considere a seguinte instância do problema (3.46), com $m = 2$ e $n = 6$:

$$\begin{aligned}
\min \quad & z = 2x_1 + 3x_2 + 4x_3 + 4x_4 + 2x_5 + x_6 \\
\text{s. to} \quad & a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{15}x_5 + a_{16}x_6 = 240 \\
& a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + a_{25}x_5 + a_{26}x_6 = 500 \\
& x_1, \ldots, x_6 \geq 0, \\
& 4 \leq a_{11} \leq 10, \quad 6 \leq a_{21} \leq 8, \\
& 3 \leq a_{12} \leq 5, \quad 8 \leq a_{22} \leq 10, \\
& 2 \leq a_{13} \leq 8, \quad 5 \leq a_{23} \leq 15, \\
& 7 \leq a_{14} \leq 12, \quad 10 \leq a_{24} \leq 16, \\
& 5 \leq a_{15} \leq 7, \quad 8 \leq a_{25} \leq 10, \\
& 15 \leq a_{16} \leq 22, \quad 20 \leq a_{26} \leq 30.
\end{aligned} \tag{3.54}$$

A base inicial pode ser escolhida, por exemplo, a partir das colunas a_1 e a_2, com valores $a_1 = (4\ 6)^T$ e $a_2 = (3\ 8)^T$. Posteriormente, será discutido de que forma a seleção da base é definida neste algoritmo. Neste momento, o foco recai sobre a descrição do seu funcionamento.

Com isso, teremos a base $\mathcal{B}$ e sua inversa $\mathcal{B}^{-1}$:

$$\mathcal{B} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 6 & 8 \end{pmatrix} \tag{3.55}$$

$$\mathcal{B}^{-1} = \frac{1}{14}\begin{pmatrix} 8 & -3 \\ -6 & 4 \end{pmatrix} \tag{3.56}$$

Nós temos $x_{\mathcal{B}} = (x_1\ x_2)^T$ e $b = (b_1\ b_2)^T = (240\ 500)^T$. Então:

$$\overline{x}_{\mathcal{B}} = \mathcal{B}^{-1}b = (30\ \ 40)^T \tag{3.57}$$

Considerando todas as outras variáveis x com valor de custo zero, temos:

$$z = 2x_1 + 3x_2 = 2 \cdot 30 + 3 \cdot 40 = 60 + 120 = 180 \tag{3.58}$$

Obtivemos uma solução básica viável que fornece um valor para a função objetivo de (3.46) no valor de $z = 180$. Para verificar se é uma solução ótima, será necessário verificar a existência de uma coluna a_k, com $a_k,\ 1 \leq k \leq 6$, que possa entrar na base com objetivo de reduzir o valor de z.

Temos $c_{\mathcal{B}} = (c_1\ c_2) = (2\ 3)$ e $u = c_{\mathcal{B}}\,\mathcal{B}^{-1} = \frac{1}{7}(-1\ 3)$. Para cada $j \in J$, o subproblema apresentado em (3.50), será escrito como:

$$\begin{aligned} \max\ & \frac{1}{7}(-1\ 3)\begin{pmatrix} a_{1j} \\ a_{2j} \end{pmatrix} - c_j \\ \text{s. a}\ & \\ & \alpha_{ij}\ \leq\ a_{ij}\ \leq\ \beta_{ij}\,, \qquad i \in I \end{aligned} \tag{3.59}$$

ou ainda,

$$\begin{aligned} \max\ & -\frac{1}{7}\,a_{1j} + \frac{3}{7}\,a_{2j} - c_j \\ \text{s. a}\ & \\ & \alpha_{ij}\ \leq\ a_{ij}\ \leq\ \beta_{ij}\,, \qquad i \in I \end{aligned} \tag{3.60}$$

Escolhendo arbitrariamente $j = 6$, temos o subproblema a seguir:

$$\begin{aligned} \max\ & -\frac{1}{7}\,a_{16} + \frac{3}{7}\,a_{26} - 1 \\ \text{s. a}\ & \\ & 15\ \leq\ a_{16}\ \leq\ 22 \\ & 20\ \leq\ a_{26}\ \leq\ 30\,. \end{aligned} \tag{3.61}$$

É fácil visualizar que o valor máximo de a_6 será quando $a_{16} = 15$ e $a_{26} = 30$ em (3.61), correspondendo a $z_6 = \frac{68}{7} > 0$. E com isso, a coluna $a_6 = (15\ 30)^T$ entrará na base.

Como $\mathcal{B}^{-1}a_6 = \frac{1}{7}(15\ 15)^T$, o problema será reescrito como:

$$\begin{array}{ll} \max & z = 180 - \frac{68}{7} x_6 \\ & x_1 = 30 - \frac{15}{7} x_6 \quad \geq 0 \\ & x_2 = 40 - \frac{15}{7} x_6 \quad \geq 0 \end{array} \tag{3.62}$$

Verifica-se ainda que o valor máximo assumido por x_6 pode atingir, enquanto possuir $x_1 \geq 0$ e $x_2 \geq 0$, será $x_6 = 14$, obtido quando $x_1 = 0$ e $x_2 = 10$, resultando em $z = 44$.

Com isso, a nova base $\mathcal{B} = (a_6\ a_2)$ será, em termos matriciais:

$$\mathcal{B} = \begin{pmatrix} 15 & 3 \\ 30 & 8. \end{pmatrix} \tag{3.63}$$

É necessário verificar se a solução obtida é ótima. Para isso, procede o cálculo da inversa da base selecionada:

$$\mathcal{B}^{-1} = \frac{1}{30} \begin{pmatrix} 8 & -3 \\ -30 & 15 \end{pmatrix} \tag{3.64}$$

Nesse ponto, tem-se $c_{\mathcal{B}} = (c_6\ c_2) = (1\ 3)$ e $u = c_{\mathcal{B}}\,\mathcal{B}^{-1} = (-41/15\ \ 7/5)$. Para cada $j \in J$, o subproblema (3.50) pode então ser expresso da seguinte forma:

$$\begin{array}{ll} \max & -\dfrac{41}{15}\, a_{1j} + \dfrac{7}{5}\, a_{2j} - c_j \\ \text{s. to} & \\ & \alpha_{ij} \ \leq \ a_{ij} \ \leq \ \beta_{ij}\,, \qquad i \in I \end{array} \tag{3.65}$$

Nós temos:

- $z_1 = \max\{-\frac{41}{15}\, a_{11} + \frac{7}{5}\, a_{21} - 2 \mid 4 \leq a_{11} \leq 10\,,\ 6 \leq a_{21} \leq 8\} \ = \ -\frac{26}{15}$, obtido quando $a_{11} = 4$ e $a_{21} = 8$.

- $z_3 = \max\{-\frac{41}{15}\, a_{13} + \frac{7}{5}\, a_{23} - 4 \mid 2 \leq a_{13} \leq 8\,,\ 5 \leq a_{23} \leq 15\} \ = \ -\frac{173}{15}$, obtido quando $a_{13} = 2$ e $a_{23} = 15$.

- $z_4 = \max\{-\frac{41}{15}\, a_{14} + \frac{7}{5}\, a_{24} - 4 \mid 7 \leq a_{14} \leq 12\,,\ 10 \leq a_{24} \leq 16\} \ = \ -\frac{11}{15}$, obtido quando $a_{14} = 7$ e $a_{24} = 16$.

- $z_5 = \max\{-\frac{41}{15}\, a_{15} + \frac{7}{5}\, a_{25} - 2 \mid 5 \leq a_{15} \leq 7\,,\ 8 \leq a_{25} \leq 10\} \ = \ -\frac{5}{3}$, obtido quando $a_{15} = 5$ e $a_{25} = 10$.

Podemos observar que todas as colunas não-básicas, associadas a $j \in \mathcal{N}$, apresentam custo reduzido negativo. No entanto, conforme abordado na seção (3.2.2), é essencial estender a verificação às colunas pertencentes da base. Essa etapa é necessária, visto poderão haver outros valores para as variáveis dessas colunas também

podem resultar em um custo reduzido negativo. Assim, a análise prossegue com a verificação de z_2 e z_6:

- $z_2 = \max\{-\frac{41}{15}\,a_{12} + \frac{7}{5}\,a_{22} - 3 \mid 3 \leq a_{12} \leq 5\,,\ 8 \leq a_{22} \leq 10\} = \frac{14}{5}$, obtido quando $a_{12} = 3$ e $a_{22} = 10$.
- $z_6 = \max\{-\frac{41}{15}\,a_{16} + \frac{7}{5}\,a_{26} - 1 \mid 15 \leq a_{16} \leq 22\,,\ 20 \leq a_{26} \leq 30\} = 0$, obtido quando $a_{16} = 15$ e $a_{26} = 30$.

Conforme observado, o valor de $z_2 > 0$ indica que a solução ótima não foi encontrada ainda. Com a substituição dos valores em a_2 por $a_2 = (3\ 10)^T$, a próxima base e sua matriz inversa serão obtidas:

$$\mathcal{B} = \begin{pmatrix} 15 & 3 \\ 30 & 10 \end{pmatrix} \tag{3.66}$$

e

$$\mathcal{B}^{-1} = \frac{1}{60}\begin{pmatrix} 10 & -3 \\ -30 & 15 \end{pmatrix} \tag{3.67}$$

Recalculando x_6 e x_2, obtemos $(x_6\ \ x_2)^T = \mathcal{B}^{-1}b = (15\ \ 5)^T$. A função objetivo assume o valor de $z = 15x_6 + 5x_2 = 15 * 1 + 5 * 3 = 30$.

Para confirmar a otimalidade dessa solução, ou seja, para confirmar que $x_6 = 15$ e $x_2 = 5$ é a solução ótima para $z = 30$, calcularemos o mínimo de cada $c_j - z_j,\ j = 1, 2, 3, 4, 5, 6$:

$$c_{\mathcal{B}}\mathcal{B}^{-1} = (1\ \ 3)\mathcal{B}^{-1} = \frac{1}{60}(-80\ \ 42) \tag{3.68}$$

$z_1 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{60}(-82a_{11} + 42a_{21})$, cujo máximo será em $a_{11} = 4$ e $a_{21} = 8$.

$\min(c_1 - z_1) = 2 - \frac{1}{60}(-80 * 4\ +\ 42 * 8) = 2 - \frac{16}{60} > 0$.

$z_2 = c_{\mathcal{B}}\mathcal{B}^{-1}a_2 = \frac{1}{60}(-82a_{12} + 42a_{22})$, cujo máximo será em $a_{12} = 3$ e $a_{22} = 10$.

$\min(c_2 - z_2) = 3 - \frac{1}{60}(-80 * 3\ +\ 42 * 10) = 3 - \frac{180}{60} = 0$, porque está associado a uma coluna de $\mathcal{B}$.

$z_3 = c_{\mathcal{B}}\mathcal{B}^{-1}a_3 = \frac{1}{60}(-80a_{13} + 42a_{23})$, cujo máximo será em $a_{13} = 2$ e $a_{23} = 15$.

$\min(c_3 - z_3) = 4 - \frac{1}{60}(-80 * 2 \ + \ 42 * 15) = 4 - \frac{160}{60} = 4 - 6.15 < 0.$

$z_4 = c_{\mathcal{B}}\mathcal{B}^{-1}a_4 = \frac{1}{360}(-80a_{14} + 42a_{24})$, cujo máximo será em $a_{14} = 7$ e $a_{24} = 16$.

$\min(c_4 - z_4) = 4 - \frac{1}{60}(-80 * 7 \ + \ 42 * 16) = 4 - \frac{92}{60} > 0.$

$z_5 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{60}(-80a_{15} + 42a_{25})$, cujo máximo será em $a_{15} = 5$ e $a_{25} = 10$.

$\min(c_5 - z_5) = 2 - \frac{1}{60}(-80 * 5 \ + \ 42 * 10) = 2 - \frac{20}{60} > 0.$

$z_6 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{60}(-80a_{16} + 42a_{26})$, cujo máximo será em $a_{16} = 15$ e $a_{26} = 30$.

$\min(c_5 - z_5) = 1 - \frac{1}{60}(-80 * 15 \ + \ 42 * 30) = 1 - \frac{60}{60} = 1 - 1 = 0$, porque está associado a uma coluna de $\mathcal{B}$.

Como $\min(c_3 - z_3) < 0$, o valor da função objetivo z poderá ser otimizado (decrescer). Por essa razão, a coluna $a_3 = (2 \ \ 15)^T$ entrará na base, substituindo a coluna a_2 na nova matriz $\mathcal{B}$:

$$\mathcal{B} = \begin{pmatrix} 15 & 2 \\ 30 & 15 \end{pmatrix} \tag{3.69}$$

Isto é, $\mathcal{B} = (a_6 \ \ a_3)$.

E obtendo sua inversa, teremos:

$$\mathcal{B}^{-1} = \frac{1}{165}\begin{pmatrix} 15 & -2 \\ -30 & 15 \end{pmatrix} \tag{3.70}$$

Com isso, obtemos:

$$(x_6 \ \ x_3)^T = \mathcal{B}^{-1}b = (\frac{520}{33} \ \ \frac{60}{33})^T, \ z = \frac{760}{33} \tag{3.71}$$

ou

$$x_6 = \frac{520}{33} = 15,7575..., \ x_3 = \frac{60}{33} = 1,8181..., \ z = \frac{760}{33} = 23,0303... \tag{3.72}$$

Calculando o mínimo de cada $c_j - z_j, \ j = 1, 2, 3, 4, 5, 6.$, teremos:

$$c_{\mathcal{B}}\mathcal{B}^{-1} = (1 \;\; 4)\mathcal{B}^{-1} = \frac{1}{165}(-105 \;\; 58) \tag{3.73}$$

$z_1 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{165}(-105a_{11} + 58a_{21})$, cujo máximo será para $a_{11} = 4$ e $a_{21} = 8$.

$\min(c_1 - z_1) = 2 - \frac{1}{165}(-105 * 4 + 58 * 8) = 2 - \frac{44}{165} > 0$.

$z_2 = c_{\mathcal{B}}\mathcal{B}^{-1}a_2 = \frac{1}{165}(-105a_{12} + 58a_{22})$, cujo máximo será para for $a_{12} = 3$ e $a_{22} = 10$.

$\min(c_2 - z_2) = 3 - \frac{1}{165}(-105 * 3 + 58 * 10) = 3 - \frac{265}{165} > 0$.

$z_3 = c_{\mathcal{B}}\mathcal{B}^{-1}a_3 = \frac{1}{165}(-105a_{13} + 58a_{23})$, cujo máximo será de $a_{13} = 2$ e $a_{23} = 15$.

$\min(c_3 - z_3) = 4 - \frac{1}{165}(-105 * 2 + 58 * 15) = 4 - \frac{660}{165} = 4 - 4 = 0$, porque está associado com a coluna de $\mathcal{B}$.

$z_4 = c_{\mathcal{B}}\mathcal{B}^{-1}a_4 = \frac{1}{165}(-105a_{14} + 58a_{24})$, cujo máximo será de $a_{14} = 7$ e $a_{24} = 16$.

$\min(c_4 - z_4) = 4 - \frac{1}{165}(-105 * 7 + 58 * 16) = 4 - \frac{193}{165} > 0$.

$z_5 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{165}(-105a_{15} + 58a_{25})$, cujo máximo será de $a_{15} = 5$ e $a_{25} = 10$.

$\min(c_5 - z_5) = 2 - \frac{1}{165}(-105 * 5 + 58 * 10) = 2 - \frac{55}{165} > 0$.

$z_6 = c_{\mathcal{B}}\mathcal{B}^{-1}a_1 = \frac{1}{165}(-105a_{16} + 58a_{26})$, cujo máximo será de $a_{16} = 15$ e $a_{26} = 30$.

$\min(c_5 - z_5) = 1 - \frac{1}{165}(-105 * 15 + 58 * 30) = 1 - \frac{165}{165} = 1 - 1 = 0$, por-

que está associado com a coluna de $\mathcal{B}$.

Com isso, podemos afirmar com exatidão a solução do exemplo como sendo:

$x_6 = \frac{520}{33},\ x_3 = \frac{60}{33},\ \ x_j = 0,\ j = 1, 2, 4, 5,$ implicando $z = \frac{760}{33} = 23.030303....$

Nota: se $x_j = 0$, a_{ij} pode assumir qualquer valor entre α_{ij} e $\beta_{ij},\ i = 1, 2.$

3.2.4 Exemplo numérico utilizando LASDON (1970)

Vamos considerar novamente o problema (3.46) com os dados do nosso exemplo (3.2.3.

Como todos os paralelepípedos estão em $R^2,$ existem apenas duas restrições associadas a $x_j,\ j = 1, 2, ..., 6$ e existem apenas 4 vértices em cada paralelepípedo.

Relembrando os dados do exemplo: $m = 2,\ n = 6,\ b_1 = 240,\ b_2 = 500,\ c_1 = 2, \| c_2 = 3,\ c_3 = 4,\ c_4 = 4,$
$c_5 = 2,\ c_6 = 1.$

- **Paralelepípedo associado à variável x_1:**

$$4 \leq a_{11} \leq 10, \quad 6 \leq a_{21} \leq 8$$

Vértices:

$$\{(4\ \ 6)^T,\ (10\ \ 6)^T,\ (4\ \ 8)^T,\ (10\ )^T\}$$

- **Paralelepípedo associado à variável x_2:**

$$3 \leq a_{12} \leq 5, \quad 8 \leq a_{22} \leq 10$$

Vértices:

$$\{(3\ \ 8)^T,\ (5\ \ 8)^T,\ (5\ \ 10)^T,\ (3\ \ 10)^T\}$$

- **Paralelepípedo associado à variável x_3:**

$$2 \leq a_{13} \leq 8, \quad 5 \leq a_{23} \leq 15$$

Vértices:

$$\{(2\ \ 5)^T,\ (8\ \ 5)^T,\ (8\ \ 15)^T,\ (2\ \ 15)^T\}$$

- **Paralelepípedo associado à variável x_4:**

$$7 \leq a_{14} \leq 12, \quad 10 \leq a_{24} \leq 16$$

Vértices:

$$\{(7 \;\; 10)^T, \; (12 \;\; 10)^T, \; (12 \;\; 16)^T, \; (7 \;\; 16)^T\}$$

- **Paralelepípedo associado à variável x_5:**

$$5 \leq a_{15} \leq 7, \quad 8 \leq a_{25} \leq 10$$

Vértices:

$$\{(5 \;\; 8)^T, \; (7 \;\; 8)^T, \; (7 \;\; 10)^T, \; (5 \;\; 10)^T\}$$

- **Paralelepípedo associado à variável x_6:**

$$15 \leq a_{16} \leq 22, \quad 20 \leq a_{26} \leq 30$$

Vértices:

$$\{(15 \;\; 20)^T, \; (22 \;\; 20)^T, \; (22 \;\; 30)^T, \; (15 \;\; 30)^T\}$$

Vamos considerar as variáveis $\lambda_{jk}, k = 1, 2, 3, 4 \; j = 1, 2, 3, 4, 5, 6$, tais que:

$$\sum_{k=1}^{4} \lambda_{jk} = 1, \quad j = 1, 2, 3, 4, 5, 6, \lambda_{jk} \geq 0, \quad j = 1, 2, 3, 4, 5, 6; \quad k = 1, 2, 3, 4. \quad (3.74)$$

Chamaremos v_{jk} um vértice do paralelepípedo associado à variável x_j, onde $j = 1, 2, 3, 4, 5, 6$ e $k = 1, 2, 3, 4$.

Consideremos $a_j = (a_{1j} \; a_{2j})^T, \; j = 1, 2, 3, 4, 5, 6$. Podemos escrever que:

$$\min \; z = \sum_{j=1}^{6} \left(\sum_{k=1}^{4} \lambda_{jk} c_j \right) x_j \quad (3.75)$$

$$\text{sujeito a:} \sum_{j=1}^{6} \left(\sum_{k=1}^{4} \lambda_{jk} v_{jk} \right) x_j = (240 \;\; 500)^T, \quad x_j \geq 0, \quad j = 1, 2, 3, 4, 5, 6. \quad (3.76)$$

A função objetivo z será escrita da seguinte forma:

$$z = z_1 + z_2, \quad (3.77)$$

onde :

$$z_1 = (2\lambda_{11}+2\lambda_{12}+2\lambda_{13}+2\lambda_{14})x_1+(3\lambda_{21}+3\lambda_{22}+3\lambda_{23}+3\lambda_{24})x_2+(4\lambda_{31}+4\lambda_{32}+4\lambda_{33}+4\lambda_{34})x_3$$

$$z_2 = (4\lambda_{41}+4\lambda_{42}+4\lambda_{43}+4\lambda_{44})x_4+(2\lambda_{51}+2\lambda_{52}+2\lambda_{53}+2\lambda_{54})x_5+(\lambda_{61}+\lambda_{62}+\lambda_{63}+\lambda_{64})x_6$$

Restrições $g = 240$ e $h = 500$, onde:

$$g_1 = (4\lambda_{11}+10\lambda_{12}+4\lambda_{13}+10\lambda_{14})x_1+(3\lambda_{21}+5\lambda_{22}+5\lambda_{23}+3\lambda_{24})x_2+(2\lambda_{31}+8\lambda_{32}+8\lambda_{33}+2\lambda_{34})x_3$$

$$g_2 = (7\lambda_{41}+12\lambda_{42}+12\lambda_{43}+7\lambda_{44})x_4+(5\lambda_{51}+7\lambda_{52}+7\lambda_{53}+5\lambda_{54})x_5+(15\lambda_{61}+22\lambda_{62}+22\lambda_{63}+15\lambda_{64})x_6$$

$$g = g_1 + g_2 = 240$$

$$h_1 = (6\lambda_{11}+6\lambda_{12}+15\lambda_{13}+15\lambda_{14})x_1+(8\lambda_{21}+8\lambda_{22}+10\lambda_{23}+10\lambda_{24})x_2+(5\lambda_{31}+5\lambda_{32}+15\lambda_{33}+15\lambda_{34})x_3$$

$$h_2 = (10\lambda_{41}+10\lambda_{42}+16\lambda_{43}+16\lambda_{44})x_4+(8\lambda_{51}+8\lambda_{52}+10\lambda_{53}+10\lambda_{54})x_5+(20\lambda_{61}+20\lambda_{62}+30\lambda_{63}+30\lambda_{64})x_6$$

$$h = h_1 + h_2 = 500$$

Para resolver o problema $\min z$, s. a: $g = 240$, $h = 500$ faremos a mudança de variável abaixo:

$$\lambda_{jk}x_j = u_{jk} \geq 0 \tag{3.78}$$

Consideramos implicitamente que:

$$\lambda_{jk} \geq 0,\ k = 1, 2, 3, 4 \quad j = 1, 2, 3, 4, 5, 6. \tag{3.79}$$

Tal que:

$$\sum_{k=1}^{4} \lambda_{jk} = 1,\ j = 1, 2, 3, 4, 5, 6 \tag{3.80}$$

O novo problema de otimização será:

$$(\bar{P}): \ \min \ \bar{z} = \bar{z_1} + \bar{z_2} \tag{3.81}$$

$$\bar{z_1} = 2u_{11} + 2u_{12} + 2u_{13} + 2u_{14} + 3u_{21} + 3u_{22} + 3u_{23} + 3u_{24} + 4u_{31} + 4u_{32} + 4u_{33} + 4u_{34}$$

$$\bar{z_2} = 4u_{41} + 4u_{42} + 4u_{43} + 4u_{44} + 2u_{51} + 2u_{52} + 2u_{53} + 2u_{54} + u_{61} + u_{62} + u_{63} + u_{64}$$

sujeito a:

$$\bar{g} = 240, \ e \ \bar{h} = 500 \ e \ u_{jk} \geq 0, j = 1, 2, 3, 4, 5, 6 \ e \ k = 1, 2, 3, 4.$$

Onde $\bar{g} = \bar{g_1} + \bar{g_2} \ e \ \bar{h} = \bar{h_1} + \bar{h_2}$

$$\bar{g_1} = 4u_{11} + 10u_{12} + 4u_{13} + 10u_{14} + 3u_{21} + 5u_{22} + 5u_{23} + 3u_{24} + 2u_{31} + 8u_{32} + 8u_{33} + 2u_{34}$$

$$\bar{g_2} = 7u41 + 12u_{42} + 12u_{43} + 7u_{44} + 5u_{51} + 7u_{52} + 7u_{53} + 5u_{54} + 15u_{61} + 22u_{62} + 22u_{63} + 15u_{64}$$

$$\bar{h_1} = 6u_{11} + 6u_{12} + 8u_{13} + 8u_{14} + 8u_{21} + 8u_{22} + 10u_{23} + 10u_{24} + 5u_{31} + 5u_{32} + 15u_{33} + 15u_{34}$$

$$\bar{h_2} = 10u_{41} + 10u_{42} + 16u_{43} + 16u_{44} + 8u_{51} + 8u_{52} + 10u_{53} + 10u_{54} + 20u_{61} + 20u_{62} + 30u_{63} + 30u_{64}$$

$(\bar{P})$ é um problema de otimização linear com duas restrições e 24 variáveis.

Consideramos $\bar{u}_{jk}, \ j = 1, 2, 3, 4, 5, 6; \ k = 1, 2, 3, 4$ uma solução ótima de $(\bar{P})$, sabemos que se $\bar{u}_{jk} = 0, \ $ para todo k, então $\bar{x}_j = 0$. Por outro lado, se $\bar{u}_{jk} > 0$, para algum k, podemos escrever:

$$\bar{\lambda}_{jk} = \frac{\bar{u}_{jk}}{\bar{x}_j} \tag{3.82}$$

Ou ainda que:

$$\sum_{k=1}^{4} \bar{\lambda}_{jk} = \frac{\sum_{k=1}^{4} \bar{u}_{jk}}{\bar{x}_j} = 1 \tag{3.83}$$

Obteremos a solução ótima de (3.46):

$$\bar{x}_j = \sum_{k=1}^{4} \bar{u}_{jk}, \ j = 1, 2, 3, , 4, 5, 6 \tag{3.84}$$

Com a solução ótima encontrada: $u34 = 1.8182 = x3$, $u64 = 15.7576 = x6$ e $z = 23.0303$.

Neste exemplo, cada coluna pertence a um retângulo, que tem apenas 4 vértices, algo que pode ser facilmente calculado. No entanto, quando há muitas restrições (m pode ser muito grande), o método também é usado para gerar colunas, conforme apresentado em LASDON (1970).

3.2.5 Solução Básica Inicial Viável para (3.46)

A solução inicial viável para Algoritmo 1 será encontrada introduzindo variáveis artificiais $s_i \geq 0, \quad i \in I$. O método aqui utilizado é denominado Big M, como já discutido nesta dissertação. Ele consiste na criação de um problema auxiliar de otimização (3.88), do qual é formulado a partir do problema original (3.46) com o acréscimo de termos de penalidade significativos.

$$\min \quad z = \sum_{j \in J} c_j \, x_j + M \sum_{i \in I} s_i \tag{3.85}$$

$$\text{s. to} \quad \sum_{j \in J} a_{ij} \, x_j + s_i = b_i \, , \qquad i \in I \tag{3.86}$$

$$x_j \geq 0 \quad s_i \geq 0, \quad j \in J, \ i \in I \tag{3.87}$$

$$\alpha_{ij} \leq a_{ij} \leq \beta_{ij} \, , \qquad i \in I, \, j \in J \tag{3.88}$$

Para a inicialização do algoritmo, uma solução básica viável inicial será construída para o problema auxiliar (3.88). Essa solução será estabelecida a partir das colunas associadas às variáveis artificiais s_i. Estas variáveis são penalizadas através de um custo M de uma magnitude extremamente grande.

Após a aplicação do algoritmo, a análise dos resultados permite duas conclusões sobre a viabilidade e otimalidade do problema original (3.46) estudado:

1. **Solução Ótima:** Se, na solução ótima do problema auxiliar (3.88), todas as variáveis artificiais s_i forem nulas ($s_i = 0, \forall i \in I$), a solução obtida para as variáveis x_j corresponde à solução ótima do problema original (3.46).

2. **Inviabilidade:** Caso contrário, implica que o problema original (3.46 não possui uma solução viável.

Capítulo 4

Resultados Computacionais

Nesta seção são apresentados os resultados computacionais obtidos a partir da aplicação do algoritmo proposto em diferentes instâncias do problema. O objetivo dos experimentos foi avaliar a qualidade das soluções encontradas e comparar o desempenho do método com outros algoritmos disponíveis na literatura e em solvers amplamente utilizados.

Inicialmente, foram geradas 10 instâncias de pequeno porte, com características resumidas a seguir:

- número de colunas (variáveis x) variando entre 5 e 10;
- número de restrições variando entre 3 e 5;
- limites inferiores ℓ para as variáveis a_{ij} no intervalo de 0 a 9;
- limites superiores para as variáveis a_{ij} variando de $\ell + 0$ até $\ell + 9$;
- valores do lado direito variando de $50\,n + 1$ até $50\,n + 50$;
- coeficientes da função objetivo no intervalo de 1 a 9.

Os valores ótimos da função objetivo para essas instâncias encontram-se na Tabela 4.1. Cada coluna corresponde a um método de otimização distinto: o algoritmo proposto (QuadSimplex), o método descrito por LASDON (1970) e o solver Ipopt.

Observa-se que, para as instâncias menores, todos os métodos retornaram essencialmente os mesmos valores ótimos, com exceção de uma das instâncias, na qual o Ipopt apresentou leve discrepância. Esse resultado sugere que o algoritmo proposto é consistente na obtenção da solução ótima, apresentando desempenho equivalente ou superior aos métodos de referência.

Na sequência, foram geradas 10 instâncias adicionais de maior porte, cada uma contendo 100 colunas e 50 restrições. Nessas instâncias, o método de LASDON

Lasdon	QuadSimplex	Ipopt
34.655	34.655	34.655
45.939	45.939	45.939
119.731	119.731	119.731
142.412	142.412	142.412
73.486	73.486	73.486
150.441	150.441	150.441
87.000	87.000	87.000
133.873	133.873	134.792
91.959	91.959	91.959
140.312	140.312	140.312

Tabela 4.1: Resultados para instâncias geradas aleatoriamente.

(1970) não pôde ser aplicado, devido ao elevado número de variáveis, que excedeu a capacidade de processamento do AMPL. Os resultados são apresentados na Tabela 4.2.

QuadSimplex	Ipopt
640.86	641.78
607.87	612.42
613.37	613.37
631.66	633.88
710.28	710.28
646.83	647.22
684.64	684.97
669.34	669.34
709.10	709.10
739.81	739.81

Tabela 4.2: Resultados para instâncias maiores.

Nos experimentos com instâncias maiores, o algoritmo proposto manteve sua robustez, encontrando a solução ótima em todas as situações. Além disso, o tempo de execução foi significativamente reduzido em comparação ao solver Ipopt, que demandou aproximadamente 10 segundos para finalizar, enquanto o QuadSimplex obteve os resultados em menos de 1 segundo.

Por fim, em todas as 20 instâncias testadas, tanto nas menores quanto nas maiores, o solver Gurobi foi capaz de encontrar a solução ótima, servindo como referência de validação para os resultados obtidos.

Referências Bibliográficas

LASDON, L. S. *Optimization theory for large systems*. Mineola, New York, Courier Corporation, 1970.

BOYD, S., VANDENBERGHE, L. *Convex optimization*. Cambridge university press, 2004.

EULER, L. "Solutio problematis ad geometriam situs pertinentis", *Commentarii academiae scientiarum Petropolitanae*, pp. 128–140, 1741.

WOLSEY, L. A. *Integer programming*. Hoboken, New Jersey, John Wiley & Sons, 2020.

CAMPELLO, R. E. *Algoritmos e heurísticas: desenvolvimento e avaliação de performance*. Rio de Janeiro, EDUFF, 1994.

PAPADIMITRIOU, C. H., STEIGLITZ, K. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998a.

CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., et al. *Introduction to algorithms*. MIT press, 2022.

SZWARCFITER, J. L., MARKENZON, L. *Estruturas de Dados e seus Algoritmos*, v. 2. Rio de Janeiro, Livros Técnicos e Científicos, 1994.

GOMORY, R. E. "Outline of an algorithm for integer solutions to linear programs and an algorithm for the mixed integer problem". In: *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, Springer, pp. 77–103, 2009.

DANTZIG, G. B. "Maximization of a linear function of variables subject to linear inequalities", *Activity analysis of production and allocation*, v. 13, pp. 339–347, 1951.

MACULAN, N., FAMPA, M. H. C. "Otimização linear", *Editora Universidade de Brasilia: Brasilia*, 2006.

BAZARAA, M. S., JARVIS, J. J., SHERALI, H. D. *Linear programming and network flows*. John Wiley & Sons, 2011.

VANDERBEI, R. J. *Linear Programming: Foundations and Extensions*. Springer, 2014.

KANIYAR, T. *Linear and Nonlinear Programming Essentials*. Educohack Press, 2025.

BAZARA, M. S., JARVIS, J. J., SHERALI, H. D. *Linear programming and network flows*. John Wiley & Sons, 2011.

KANTOROVICH, L. V. "Mathematical methods of organizing and planning production", *Management science*, v. 6, n. 4, pp. 366–422, 1939.

DANTZIG, G. B., WOLFE, P. "Decomposition principle for linear programs", *Operations research*, v. 8, n. 1, pp. 101–111, 1960.

FORD JR, L. R., FULKERSON, D. R. "A suggested computation for maximal multi-commodity network flows", *Management Science*, v. 5, n. 1, pp. 97–101, 1958.

GILMORE, P. C., GOMORY, R. E. "A linear programming approach to the cutting-stock problem", *Operations research*, v. 9, n. 6, pp. 849–859, 1961.

DESROSIERS, J., SOUMIS, F., DESROCHERS, M. "Routing with time windows by column generation", *Networks*, v. 14, n. 4, pp. 545–565, 1984.

BARNHART, C., JOHNSON, E. L., NEMHAUSER, G. L., et al. "Branch-and-price: Column generation for solving huge integer programs", *Operations research*, v. 46, n. 3, pp. 316–329, 1998.

LÜBBECKE, M. E., DESROSIERS, J. "Selected topics in column generation", *Operations research*, v. 53, n. 6, pp. 1007–1023, 2005.

PESSOA, A., UCHOA, E., POGGI, M., et al. "A Robust Branch-and-Price Algorithm for Vehicle Routing Problems", *Transportation Science*, v. 54, n. 2, pp. 409–432, 2020.

UCHOA, E., PESSOA, A., MORENO, L. "Optimizing with Column Generation: Advanced Branch-Cut-and-Price Algorithms (Part I)", 2024.

DESROSIERS, J., LÜBBECKE, M., DESAULNIERS, G., et al. *Branch-and-price*. GERAD, HÉC Montréal, 2024.

WÄSCHER, G., HAUSSNER, H., SCHUMANN, H. "An improved typology of cutting and packing problems", *European journal of operational research*, v. 183, n. 3, pp. 1109–1130, 2007a.

DESROSIERS, J., LÜBBECKE, M. E. "A primer in column generation". In: *Column generation*, Springer, pp. 1–32, 2005.

LÜBBECKE, M. E., DESROSIERS, J. "Column generation", *Operations Research*, v. 53, n. 6, pp. 1007–1023, 2010.

BELOV, G., SCHEITHAUER, G., ALVES, C., et al. "Gomory cuts from a position-indexed formulation of 1D stock cutting". In: *Intelligent Decision Support: Current Challenges and Approaches*, Springer, pp. 3–14, 2007.

GOULIMIS, C. "Counterexamples in the CSP", *arXiv preprint arXiv:2004.01937*, 2020.

DE CARVALHO, J. V. "LP models for bin packing and cutting stock problems", *European Journal of Operational Research*, v. 141, n. 2, pp. 253–273, 2002.

DYCKHOFF, H. "A typology of cutting and packing problems", *European journal of operational research*, v. 44, n. 2, pp. 145–159, 1990.

WÄSCHER, G., HAUSSNER, H., SCHUMANN, H. "An improved typology of cutting and packing problems", *European journal of operational research*, v. 183, n. 3, pp. 1109–1130, 2007b.

MARTELLO, S., TOTH, P. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

GILMORE, P. C., GOMORY, R. E. "A linear programming approach to the cutting stock problem—Part II", *Operations research*, v. 11, n. 6, pp. 863–888, 1963.

MACAMBIRA, A. F. U., SIMONETTI, L., RODRIGUES, R. D. F., et al. "Tópicos em otimização inteira", *UFRJ*, 2022.

PAPADIMITRIOU, C. H., STEIGLITZ, K. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998b.

KORTE, B., VYGEN, J. *Combinatorial optimization: theory and algorithms*. Springer, 2008.

WOLSEY, L. A., NEMHAUSER, G. L. *Integer and combinatorial optimization*. John Wiley & Sons, 1999.

MITCHELL, J. E., PARDALOS, P. M., RESENDE, M. G. "Interior point methods for combinatorial optimization". In: *Handbook of Combinatorial Optimization: Volume1–3*, Springer, pp. 189–297, 1998.

VAZIRANI, V. V. *Approximation algorithms*, v. 1. Springer, 2001a.

LAND, A. H., DOIG, A. G. "An automatic method for solving discrete programming problems". In: *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, Springer, pp. 105–132, 2009.

BUENO, E. F. *Geração de Colunas em Problemas de Otimização Combinatória*. Tese de Doutorado, Tese de Mestrado, COPPE/UFRJ, Engenharia de Sistemas e Computaçao, 2005a.

BUENO, E. F. *Geração de Colunas em Problemas de Otimização Combinatória*. Tese de Doutorado, Tese de Mestrado, COPPE/UFRJ, Engenharia de Sistemas e Computaçao, 2005b.

DE LIMA TOSTAS, R. G. *BUSCA DE AGRUPAMENTOS DE DADOS UTILIZANDO GERAC AO DE COLUNAS EM PROGRAMAC AO INTEIRA*. Tese de Doutorado, Universidade Federal do Rio de Janeiro, 2012.

LODI, A., MARTELLO, S., MONACI, M. "Two-dimensional packing problems: A survey", *European journal of operational research*, v. 141, n. 2, pp. 241–252, 2002.

BERTSIMAS, D., TSITSIKLIS, J. N. *Introduction to linear optimization*, v. 6. Athena scientific Belmont, MA, 1997.

BELOV, G., LETCHFORD, A. N., UCHOA, E. "A node-flow model for 1D stock cutting: Robust branch-cut-and-price", *University of Dresden, Lancaster University and Universidade Federal Fluminense*, 2005.

HILLIER, F. S. *Introduction to operations research*. McGrawHill, 2005.

TAHA, H. A. *Operations research: an introduction*. Pearson Education India, 2013.

WINSTON, W. L. *Operations research: applications and algorithm*. Thomson Learning, Inc., 2004.

MARCOTTE, O. "The cutting stock problem and integer rounding", *Mathematical Programming*, v. 33, n. 1, pp. 82–92, 1985.

ASQUITH, J. "Linear programming", *The Mathematical Gazette*, v. 69, n. 448, pp. 151–151, 1985. Book review of: Linear programming, by Vašek Chvátal (WH Freeman, 1983), pp 478. ISBN 0-7167-1195-8.

VANCE, P. H., BARNHART, C., JOHNSON, E. L., et al. "Solving binary cutting stock problems by column generation and branch-and-bound", *Computational optimization and applications*, v. 3, n. 2, pp. 111–130, 1994.

DANTZIG, G. B. *Linear programming and extensions*. Princeton university press, 2016.

MACULAN, N., PASSINI, M. D. M., BRITO, J. A. D. M., et al. "Column-generation in integer linear programming", *RAIRO-Operations Research-Recherche Opérationnelle*, v. 37, n. 2, pp. 67–83, 2003.

WRIGHT, S., NOCEDAL, J. "Numerical optimization", *Springer Science*, v. 35, n. 67-68, pp. 7, 1999.

SALHI, A. "Global optimization: Deterministic approaches", *Journal of the Operational Research Society*, v. 45, n. 5, pp. 595–597, 1994.

AMENTA, N. "Helly theorems and generalized linear programming". In: *Proceedings of the ninth annual symposium on Computational geometry*, pp. 63–72, 1993.

Desaulniers, G., Desrosiers, J., Solomon, M. M. (Eds.). *Column generation*, v. 5. Springer Science & Business Media, 2006.

GAREY, M. R., JOHNSON, D. S. "A Guide to the Theory of NP-Completeness". In: *Computers and Intractability*, WH Freeman & Co., pp. 37–79, 1990.

VAZIRANI, V. V. *Approximation algorithms*, v. 1. Berlin, Springer, 2001b.

WÄCHTER, A., BIEGLER, L. T. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming", *Mathematical Programming*, v. 106, pp. 25–57. ISSN: 0025-5610.