Ministério da Educação
Universidade Federal de Ouro Preto
Instituto de ciências exatas e aplicadas
Departamento de Engenharia de Produção

UM MODELO DE PROGRAMAÇÃO INTEIRA MISTA PARA RETOMADA DE CARVÃO EM PÁTIOS DE ESTOCAGEM

ESTEFÂNIA DE SÁ MOURA

João Monlevade MG
2020

Estefânia de Sá Moura

Um modelo de programação inteira mista para retomada de carvão em pátios de estocagem

Monografia apresentada ao Curso de Engenharia de Produção da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau em Bacharel em Engenharia de Produção.

Orientador: Prof. Thiago Augusto de Oliveira Silva

João Monlevade - MG
24 de setembro de 2020

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

M929u Moura, Estefania de Sa .
Um modelo de programação inteira mista para retomada de carvão em pátios de estocagem. [manuscrito] / Estefania de Sa Moura. - 2020.
35 f.: il.: color., tab..

Orientador: Prof. Dr. Thiago Augusto de Oliveira Silva.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Engenharia de Produção .

1. Programação linear . 2. Simetria. 3. C++ (Linguagem de programação de computador). 4. Heurística. 5. Modelos matemáticos. I. Silva, Thiago Augusto de Oliveira. II. Universidade Federal de Ouro Preto. III. Título.

CDU 519.85

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE ENGENHARIA DE PRODUCAO - ICEA

FOLHA DE APROVAÇÃO

Estefânia de Sá Moura

Um modelo de programação inteira mista para retomada de carvão em pátios de estocagem.

Membros da banca

Thiago Augusto de Oliveira Silva - Doutor - Ufop
Alexandre Xavier Martins- Doutor - Ufop
Amanda de Oliveira Magalhães - Engenheira de Produção - Usiminas

Versão final
Aprovado em 05 de outubro de 2020

De acordo,

Thiago Augusto de Oliveira Silva
Professor (a) Orientador (a)

Documento assinado eletronicamente por **Thiago Augusto de Oliveira Silva**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 06/10/2020, às 10:04, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0090118** e o código CRC **17DB63FB**.

Referência: Caso responda este documento, indicar expressamente o Processo nº 23109.007403/2020-29 SEI nº 0090118

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: - www.ufop.br

Este trabalho é dedicado ao meu pai, à minha mãe, ao meu irmão e à minha irmã

Agradecimentos

Agradeço primeiramente à minha família, por garantir recursos e apoio para meus estudos. Gostaria também de agradecer a todos os professores da Universidade Federal de Ouro Preto, pelo profissionalismo e competência na área de educação. Um agradecimento especial aos professores da área de pesquisa operacional, Thiago Augusto, Alexandre Xavier e Mônica Amaral, sem eles não teria a capacidade de elaborar o presente trabalho. Agradeço também à Fundação CAPES (Coordenadoria de Aperfeiçoamento de Pessoas de Nível Superior) por fornecer suporte financeiro à minha graduação sanduíche, que me permitiu desenvolver na área da informática. A todos os professores do ISIMA (Institut Supérieur d'Informatique, de Modélisation et de leurs Applications - Clermont Ferrand, França), em especial ao professor Hervé Kerivin e ao Doutorando do ISIMA, Rafael Colares deixo meus sinceros agradecimentos pelo suporte e conhecimentos transmitidos nas atividades de projeto e estágios desenvolvidos durante o intercâmbio. Agradeço também aos meus colegas de estudo que me motivaram a continuar crescendo academicamente. Finalmente, agradeço novamente ao meu orientador, professor Thiago Augusto, por ter me acompanhado e ajudado durante toda elaboração do meu trabalho.

"É preciso que eu suporte duas ou três larvas se quiser conhecer as borboletas."

Antoine de Saint Exupéry.

Resumo

Este trabalho aborda o processo de Retomada de Carvão em Pátios de Estocagem considerando a simetria no problema. O objetivo foi possibilitar a gestão eficiente de maneira a minimizar o desvio negativo do material no sistema, respeitando certas restrições da planta de processamento de uma indústria específica. Para tanto, foi desenvolvido o modelo matemático e implementado em linguagem computacional C++ através do resolvedor ILOG CPLEX da IBM. Os dados indicam que o método exato foi efetivamente implementado e os resultados apresentados foram coerentes com o modelo inicial. Foram também utilizados o método de *Local Branching* e um método heurístico elaborado nessa pesquisa. Concluiu-se que tanto o método de *Local Branching* quanto o heurístico não alcançaram os objetivos esperados de remover a simetria.

Palavras-chave: Programação linear inteira mista, Simetria, $LocalBranching$, Heurística.

Abstract

This work addresses the Coal Resumption process in Stock Yards considering the symmetry in the problem. The objective was to enable efficient management in order to minimize the lack of the material in the system, respecting certain restrictions of the processing plant of a specific industry. To do this so, the mathematical model was developed and implemented in C ++ computational language through IBM's ILOG CPLEX solver. The results indicate that the exact method was effectively implemented and the results presented were consistent with the initial model. The *Local Branching* method and a heuristic developed method were also used. In conclusion, both *Local Branching* and heuristic methods did not achieve the expected objectives of removing the symmetry.

Keywords:Mixed integer linear programming, Symmetry, $LocalBranching$, Heuristics.

Lista de figuras

Lista de tabelas

Lista de algoritmos

Sumário

1 Introdução

O Brasil é um dos maiores produtores de aço do mundo e possui o maior parque industrial de aço da América do Sul, contando com quase 30 usinas administradas por 11 grupos industriais (PWC, 2018). Para acompanhar as necessidades do mercado e garantir padrões de qualidade, as indústrias siderurgias contam com processos de elevado custo de produção e alta demanda de material, sendo o carvão metalúrgico a matéria-prima essencial (SILVA, 2011). Em industrias integradas a coque, os carvões passam pela coqueificação e, para que isso ocorra, os mesmos são levados até os silos de $blending$, processo em que é realizada uma mistura, ou $blending$, de vários tipos de carvão para a formação de um produto intermediário do processo.

No processo de coqueificação considerado neste trabalho, a qualidade planejada para o coque é obtida através da mistura de diversos tipos de carvões disponíveis, todos com a mesma necessidade volumétrica mínima e capacidade volumétrica máxima. O processo de abastecimento dos silos se decompõe em duas etapas: (i) Remoção do material existente no pátio de estocagem da usina através de retomadoras (ii) Transporte do material até os silos através de correias. Tal processo exige uso eficiente dos recursos, principalmente da retomadora pelo seu elevado custo e difícil operação.

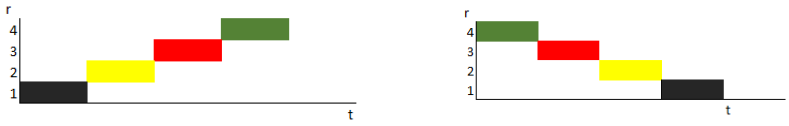
Com o intuito de melhorar os processos internos e reduzir custos, diversos autores elaboraram soluções visando tratar a programação de retomadoras. Suh, Lee e Ko (1997) subdividiu o problema de retomada em dois itens, sendo eles (i) a programação do retomador e (ii) a programação do transportador de correia. Os autores sugeriram uma arquitetura construída em dois níveis que se conectam: O *Higher-Level Scheduler* (HLS) gera programações dos transportadores de correia para retomar a matéria prima e leva-la até sua planta de processamento, atribuindo-as a vários *lower-level dispatchers* (LLDs). Nesse segundo nível, as retomadoras são programadas para retirar as matérias-primas do pátio equivalente.

Enquanto este ultimo autor construiu uma arquitetura hierárquica, Hu e Yao (2010) desenvolveram uma heurística para resolver um modelo de programação inteira mista do planejamento de empilhamento e remoção de materiais em pátios. A heurística proposta faz um associação ao modo de reprodução assexuado da biologia. Sendo assim, o algoritmo realiza iterações de troca, reversão e inserção ("crossover"e "mutação"). Em seus testes, os autores encontraram resultados satisfatórios para o método escolhido. Outros autores, como Hanoun et al. (2013) e Angelelli et al. (2016), estudam outras variações do problema.

Apesar da semelhança entre os problemas da literatura e o problema em estudo neste trabalho, ainda existem diferenças operacionais, tais como o modelo de planta de processamento. Logo, é necessária a descrição do problema considerando as características específicas do processo de retomada da indústria siderúrgica em questão.

Sendo assim, foi construído um modelo matemático para gerenciar o abastecimento dos silos, que visa solucionar e otimizar o problema de alocação de retomadoras ao longo do processo, garantindo o estoque ideal de carvão no silo para atender a mistura necessária. Tal problema é denominado de Problema de Retomada de Carvão em Pátios de Estocagem (PRCPE). Tendo em vista que temos variáveis de decisão binária e (quantidade de carvão estocado ao longo do período de tempo), trata-se de um Problema de Programação Linear Inteira Mista (PLIM). Além de ser um problema considerado como NP-Difícil (SUH; LEE; KO, 1997) e (HU; YAO, 2010), ele pode conter forte simetria. A simetria acontece quando conseguimos elaborar duas soluções diferentes mas que se assemelham em estrutura, implicando em um mesmo valor para função objetivo. A Figura 1 mostra uma simples comparação entre duas soluções simétricas, também chamadas de isomórficas, para uma alocação carvão/retomadora ao longo do tempo, em que o eixo vertical representa as retomadoras; o eixo horizontal, o horizonte de tempo e cada bloco colorido representa um tipo de carvão diferente.

Figura 1 – Comparação entres duas soluções do problema de retomada de carvão (simétricas)



(a) Retomada - solução 1 (b) Retomada - solução 2

De acordo com Margot (2009), a simetria aparece em diversos problemas de alocação em que os parâmetros são idênticos, como pode acontecer no caso do processo em estudo. Apesar dos carvões possuírem tipos diferentes, eles podem contar com parâmetros idênticos e, logo, se assemelharem em visão computacional. As propostas deste trabalho consistem em elaborar o modelo matemático e, a partir de sua implementação, tentar utilizar de outros métodos para reduzir a simetria e acelerar a resolução do problema.

1.1 Objetivos

1.1.1 Objetivo geral

O objetivo geral deste trabalho consiste em propor uma solução para o Problema de Retomada de Carvão em Pátios de Estocagem (PRCPE) considerando a existência de simetria no problema.

1.1.2 Objetivos específicos

- Descrever o PRCPE
- Modelar matematicamente o problema e implementá-lo

- Propor melhorias de resolução
- Testar instâncias de diferentes tamanhos
- Comparar resultados obtidos pelo modelo e pelas melhorias propostas

1.2 Justificativa

Tal trabalho se justifica através de dois aspectos: (a) Relevância do problema para as usinas que envolvem o processo de retomada de carvão e (b) Dificuldade a nível computacional de resolução do problema.

Em termos de custos, a gestão eficiente dos recursos envolvidos no processo, sobretudo daqueles que envolvem maiores custos e são mais difíceis de operar (como é o caso da retomadora), proporciona uma economia significativa para indústrias siderúrgicas. Existe neste processo um custo de mistura e de interrupção do processo, isto significa que é necessário garantir ininterruptividade e a quantidade ideal de carvão. Por esse motivo, é de extrema relevância a otimização de tal processo.

Diversos autores, como Suh, Lee e Ko (1997) e Hu e Yao (2010), mostraram que, em termos de complexidade computacional, o problema de retomada é um problema NP-difícil. Ainda, variações do problema podem gerar soluções isomórficas (i. e. soluções simétricas). A simetria traz uma maior dificuldade de resolução para os métodos tradicionais de resolução, como o *Branch and Cut* e *Branch and Bound*.

1.3 Organização do Texto

Este trabalho está organizado em 5 capítulos. O primeiro capítulo contém a introdução. No Capítulo 2, a revisão bibliográfica aborda a simetria no âmbito de problemas lineares, como é construída uma solução pelo método *Branch and Cut* e qual o impacto da simetria na execução deste algoritmo. Ainda nessa seção, será apresentado o método de Local *Branching* a ser utilizado no método exato. O Capítulo 3 descreve a formulação do Problema de Retomada de Carvão em Pátios de Estocagem e uma heurística específica que servirá posteriormente de solução inicial para o método exato. Já, o Capítulo 4 apresenta os resultados, análises e comparações de todas execuções realizadas ao longo do trabalho. Finalmente, o Capítulo 5 apresenta as considerações finais e perspectivas.

2 Revisão de Literatura

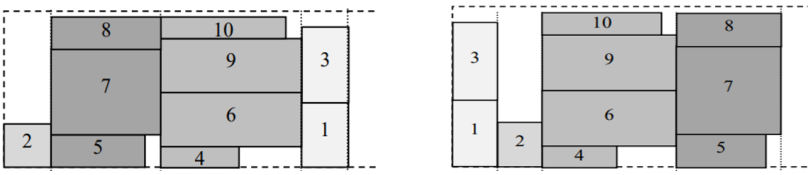
2.1 A Simetria na Programação Linear Inteira

A simetria acontece na Prgramação Linear Inteira (PLI) ou Programação Linear Inteira Mista (PLIM) quando diversas soluções isomórficas (i.e. soluções construídas de maneiras diferentes mas que apresentam a mesma estrutura final) podem ser geradas com a simples permutação entre alocações. Margot (2009) mostrou que problemas simétricos estão constantemente em problemas combinatórios clássicos, como Problemas de Alocação, $Timetabling$, Coloração de Graphos, dentre outros. Isso acontece, principalmente, quando recursos são idênticos (é importante ressaltar aqui que o conceito de "idêntico"está relacionado à invariação dos parâmetros). O autor também provou a dificuldade de resolver problemas simétricos através do método *Branch and Cut*. Uma vez que a simetria cria diversos resultados isomórficos, o algoritmo é forçado a explorar um considerável número de subproblemas na árvore de enumeração. Dessa forma, o objetivo desta seção é mostrar, de maneira didática, como a simetria pode aparecer em problemas lineares diversos.

Scalcon (2012) demostrou em seu estudo um problema de dimensionamento de lotes de produção com apenas uma máquina. Este problema constitui em determinar o melhor meio de atribuir lotes de produção à maquina de maneira a reduzir o tempo total de produção, respeitando a capacidade da máquina (quantidade máxima de produção da maquina em um intervalo de tempo). As soluções (a) e (b) apresentadas na Figura 2 mostram duas soluções simétricas para este problema, em que cada bloco representa um lote de produção, sendo o tempo de processamento representado pelo eixo horizontal e o consumo de recursos representado pelo eixo vertical.

Podemos observar que uma simples permutação da ordem de passagem dos lotes na máquina fornece uma nova solução com mesmo valor da função objetivo, isto é, mesmo tempo de processamento total na máquina.

Figura 2 – Comparação entres duas soluções de Lot Sizing (simétricas)



(a) Lot Sizing solução 1 (b) Lot Sizing solução 2

A simetria também é considerada no trabalho de Ostrowski, Anjos e Vannelli (2010) para resolver problemas de alocação de blocos cirúrgicos às salas de operação. Este problema consiste em decidir sobre abertura de salas, assim como a atribuir a essas salas determinados

blocos cirúrgicos. O objetivo é minimizar o custo de abertura das salas e o custo de horas por cirurgias atrasadas. Dado que as salas de operação são idênticas, existe uma simetria completa no que diz respeito à atribuição de cirurgias às salas.

Neste exemplo, os autores quebraram a simetria através da introdução de um ordem lexicográfica às decisões de alocação por meio de restrições. As restrições adicionais determinam que uma cirurgia deve ser afetada em uma sala que já está atribuída a outras cirurgias ou na primeira sala da lista de salas que ainda não foram abertas. Em resumo, tal ordem pode ser definida como

(i) O i-ésimo bloco cirúrgico será atribuído à primeira sala j de operação aberta

(ii) Um bloco cirúrgico não pode ser atribuído à uma sala de operação j vazia se existir uma outra sala vazia j' de tal modo que $j' \prec j$.

Para melhor entender a dificuldade de tratar tais tipos de problema através da resolução pelo método *Branch and Bound* (ou *Branch and Cut*), o leitor é convidado a ler a próxima seção para entender no detalhe o funcionamento do algoritmo. Assim, as próximas seções deste capítulo buscam justificar a utilização do método *Local Branching* para um problema simétrico NP-Difícil.

2.2 *Branch and Cut*

O método *Branch and Cut* é derivado do método *Branch and Bound*, que é um método muito eficaz para resolver problemas de otimização inteira linear. Wolsey (1998) descreveu em seu livro a partir de um problema de maximização Z.

$$Z = \max\{c(x) : x \in S \cap \mathbb{Z}^n\}$$

Seja $S \cap \mathbb{Z}^n$ o conjunto de pontos que satisfazem as restrições do problema, $c(x)$ a função objetivo e Z o valor de solução otimal. Também podemos dizer que a formulação Z_R

$$Z_R = \max\{c(x) : x \in R\}$$

é uma relaxação linear de Z se todo ponto em $S \cap \mathbb{Z}^n$ também está presente em R, quer dizer que $S \cap \mathbb{Z}^n \subseteq R$. Consequentemente, max $\{c(x) : x \in S\} \leq$ max $\{c(x) : x \in R\}$. Se R = S, então Z_R é chamado relaxação linear de Z.

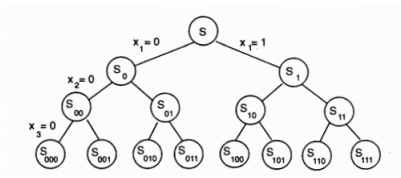
Dado que a relaxação linear de um PPLIM pode ser efetivamente resolvido pelo algoritmo simplex, dividir um problema em várias relaxações lineares se torna muito prático. Na realidade, essa separação em vários pequenos problemas é a base do processo do método de *Branch and Cut* para resolver o problema original.

Proposição 7.1 de (WOLSEY, 1998) página 91: *Seja* $S = S_1 ... S_k$ *a decomposição de* S *em pequenos grupos e seja* $Z^k = max\{c(x) : x \in S_k\}$ *para* $k = 1, ..., k$. *então* $Z = max_k Z^k$.

Com base nesse princípio, o algoritmo de *Branch and Cut* constrói uma árvore de enumeração composta por subproblemas do problema original. A descrição a seguir, explica como ocorre a construção de tal árvore.

O nó raiz da árvore representa a relaxação linear do problema original. Se a solução ótima x^* deste nó é inteira, isto é, $x^* \in S \cap \mathbb{Z}^n$, então essa solução também será a solução ótima do problema original. Caso contrário, existe um variável x_i^* nessa solução que é fracionária. Podemos assim decompor este nó em dois outros nós adicionando as restrições $x_i \leq floor(x_i^*)$ e $x_i \geq ceil(x_i^*)$, bordas inferiores e superiores. Esta operação é chamada de Branch (traduzido para o português como "ramificação"). A Figura 3 mostra um exemplo de construção dessa árvore.

Figura 3 – Árvore de exploração binária



Fonte: Wolsey (1998)

Nesta árvore, é possível observar a divisão sucessiva de S em diversos sub-conjuntos S_i, seja $S_0 = \{x \in S : x_1 = 0\}$, S_1 = $\{x \in S : x_1 = 1\}$, em seguida S_{00} = $\{x \in S_0 : x_2 = 0\}$ = $\{ x \in S : x_1 = x_2 = 0\}$.

Para um problema de maximização, por exemplo, a resolução da relaxação linear de cada nó fornece um limite superior e os limites inferiores são dados pelas soluções inteiras à medida que a árvore é construída. Portanto, uma nova proposição é definida por Wolsey (1998)

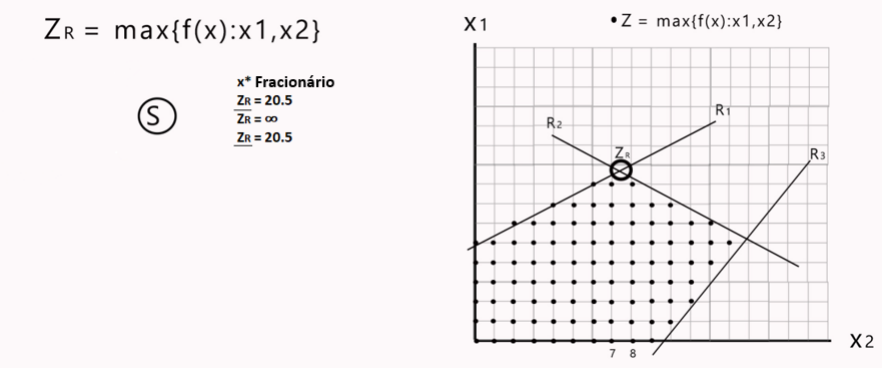
Proposição 7.2 de (WOLSEY, 1998) página 93: *Seja* $S = S_1 ... S_k$ *a decomposição de* S *em pequenos grupos e seja* $z^k = max\{c(x) : x \in S_k\}$ *para* $k = 1, ..., k$. $\overline{z}^k$ *a borda superior de* z^k *e* $\underline{z}^k$ *a borda inferior de* z^k. *Então,* $\overline{z}^k$. $= max_k \overline{z}^k$ *é uma borda superior de z e* $\underline{z}^k$. $= max_k \underline{z}^k$ *uma borda inferior de z.*

Se o valor da solução ótima de um nó da árvore for menor que seu limite inferior, esse nó nunca poderá fornecer uma solução ótima do problema original. Este nó é então cortado da árvore e nenhuma solução derivada desse nó será explorada - esta é a fase de *Bound/cut* do método.

Para exemplificar o método, consideraremos a Figura 4 como um problema inteiro de maximização com duas variáveis x_1 e x_2 e uma função objetivo $f(x)$. Nesta figura, é possível observar um primeiro vértice contendo o resultado Z_R, além do limite superior e do limite inferior. A imagem também contém um gráfico com as soluções candidatas, em que os pontos representam

as soluções candidatas cujas variáveis são inteiras.

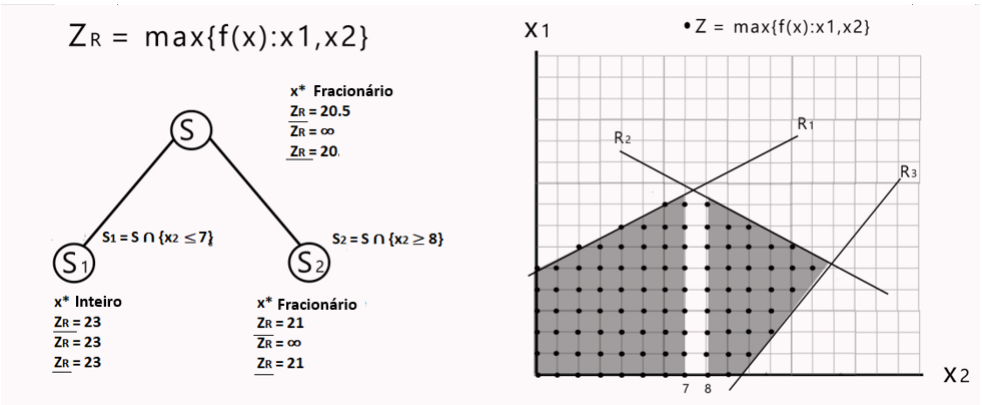
Figura 4 – Um exemplo de *Branch and Cut* - parte 1



Elaborado pela autora

Neste primeiro vértice, vê-se que a variável x_2 de relaxação linear não é inteira. Esse fato pode ser observado na Figura 5.

Figura 5 – Um exemplo de *Branch and Cut* - Parte 2

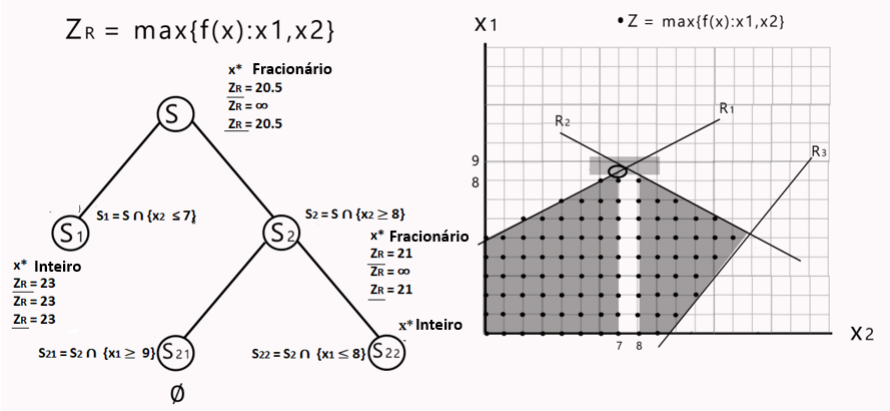


Elaborado pela autora

A região escura do gráfico representa uma nova região de soluções candidatas após a ramificação. Os vértices S_1 e S_2 contêm novas soluções relaxadas, oferecendo novos limites. Para o vértice S_1, a solução encontrada contém apenas variáveis inteiras. Além disso, seu valor é maior que o limite inferior antigo. Então os valores dos limites inferior e superior de S são iguais

ao valor de Z_R encontrado. Não precisamos continuar explorando o vértice S_1. No entanto, a solução encontrada em S_2 não está completa e, portanto, uma nova ramificação deve ser feita (mostrada pela Figura 6).

Figura 6 – Um exemplo de *Branch and Cut* - Parte 3



Elaborado pela autora

Por fim, o novo vértice S_{21} está vazio porque as restrições R1 e R2 impedem que x_1 seja maior que 9. Como a solução encontrada para S_{22} é inteira, nenhuma nova ramificação é necessária e a solução ideal do problema original está em S_{22} ou S_1.

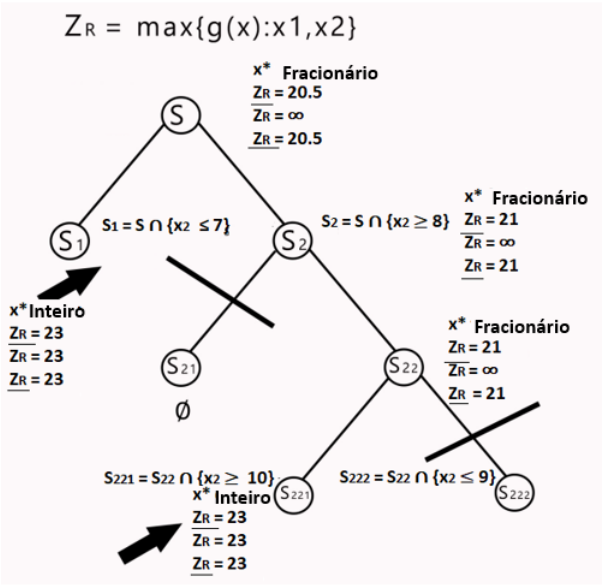
De acordo com Wolsey (1998), existem três cenários em que não se pode mais explorar um nó (não ramificar), ou seja, quando não há motivo para continuar destrinchando um nó e cortá-lo. O primeiro caso é pela otimalidade (i.e. $z_t = \{maxc(x) : x \in S_t\}$) e se dá quando encontramos uma solução inteira ótima para o problema; o segundo caso ocorre quando uma solução Z é encontrada pior que o limite inferior atual (i.e. $\overline{z_t} \leq \underline{z}$); finalmente, o terceiro caso ocorre se, para um vértice processado, não houver solução possível (i.e. $S_t = \emptyset$), como no vértice S_{21} no exemplo acima.

Posteriormente, outros tipos de cortes podem ser adicionados ao algoritmo dependendo das restrições impostas no problema. Para finalizar, é importante enfatizar que, quando os subconjuntos são separados, a união dos subconjuntos filhos tem menos soluções que o conjunto pai. Por outro lado, todas as soluções inteiras presentes no conjunto pai estão presentes em um desses subconjuntos.

2.2.1 Impacto da simetria no *Branch and Cut*

Para problemas em que existem casos de simetria, como os exemplos na Seção 2.1, o método de *Branch and Cut* realiza cálculos desnecessários. Isto ocorre porque a simetria atrapalhará a progressão dos limites no desenrolar do algoritmo. Se retomarmos o exemplo da seção 2.4 com uma dada função objetivo $g(x)$, diferente da original, e realizarmos as ramificações, podemos ver um novo desenvolvimento na Figura 7.

Figura 7 – Um caso de simetria no branching



Elaborado pela autora

Observando a imagem, notamos que S_1 e S_{221} são dois vértices simétricos, pois têm os mesmos resultados para diferentes valores de variáveis. Apesar de parecer evidente que não há como melhorar a ramificação direita, o algoritmo continua executando ramificações para tentar provar otimalidade. Neste caso não há progressão do limite superior e existe uma exploração desnecessária em S_2, que poderia ser substituída por um corte, evitando retrabalho.

Se retornarmos ao trabalho de Ostrowski, Anjos e Vannelli (2010), mencionado na seção 2.1, e considerarmos uma solução $S_1 = \{x_{1,2} = 1, x_{2,1} = 1\}$, sendo $x_{i,j}$ a variável binária de decisão de alocação de um bloco cirúrgico i à uma sala de operação j, uma simples permutação entre as alocações i - j resultaria em uma solução simétrica $S_2 = \{x_{1,1} = 1, x_{2,2} = 1\}$ com mesmo valor de S_1 (Lembrando que blocos cirúrgicos e salas de operações são qualitativamente idênticos, isto quer dizer de mesmo custos e parâmetros invariantes). Logo, para um problema de m blocos cirúrgicos, cada solução possui $m!$ soluções isomórficas. Em outras palavras, o algoritmo de *Branch and Cut* fará pelo menos $m!$ cálculos desnecessárias.

Trabalhar na simetria através da atuação na fase de ramificação é uma maneira de guiar o algoritmo em suas decisões, na tentativa de reduzir o espaço de procura, ou seja, a árvore exploratória. Espera-se que os cálculos sejam consideravelmente reduzidos e a solução ideal seja encontrada mais rapidamente.

Ostrowski, Anjos e Vannelli (2010) mostraram em seu trabalho que, de fato, quando tratamos a eliminação da simetria no ramo, os resultados são bastante satisfatórios e que, em comparação com outros métodos, há uma melhoria considerável. Outros autores, como (MARTINEZ; CUNHA, 2008), utilizaram um método de fixação flexível (*Local Branching*) junto com a adição de restrições durante a fase de ramificação e mostraram uma obtenção de resultados com bom desempenho. Este último método foi usado também neste trabalho em uma tentativa de

melhoria de resolução do PRCPE e sua definição é descrita na próxima seção deste capítulo.

2.3 O método de *Local Branching*

O método de *Local Branching* foi primeiramente descrito em Fischetti e Lodi (2003) como um método de fixação flexível para melhorar resolução de problemas lineares inteiros mistos. A partir de um modelo genérico (K), considere o conjunto de variáveis $K = \{i, ...n\}$, em que subconjuntos B, G e C armazenam os índices de variáveis, sendo $B \neq \emptyset$ constituído pelos índices de variáveis inteiras $(0-1)$ e B, G constituídos pelos índices de variáveis inteiras e contínuas. Dessa maneira, define-se que:

$$\min c^t x \tag{2.1}$$
$$Ax \geq b \tag{2.2}$$
$$x_j \in 0, 1 \forall j \in B \tag{2.3}$$
$$x_j \geq 0 \text{ inteiro } \forall j \in G \tag{2.4}$$
$$x_j \geq 0 \forall j \in C \tag{2.5}$$

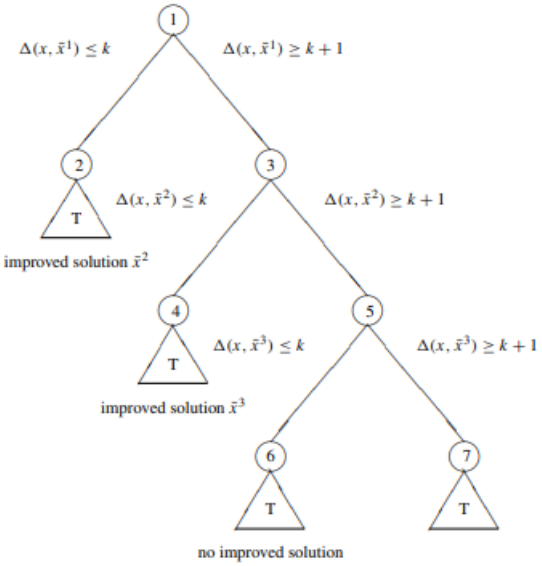
Uma vez que existe uma solução válida x', tem-se que o conjunto $S' = \{j \in B : x'_j = 1\}$ como suporte de x'. Para um valor qualquer p, sendo este inteiro e positivo, é possível definir uma vizinhança K-opt $N(x', p)$ para x' que contenham conjunto de soluções também válidas e que satisfaçam a seguinte restrição de *Local Branching*.

$$\Delta(x, x') = \sum_{j \in S'}(1 - x_j) + \sum j \in B \backslash S' x_j \leq p \tag{2.6}$$

Essa ultima restrição constrói uma diferença forçada entre as soluções x e x' de no máximo p posições. Dessa forma, o espaço de soluções associado ao branching de uma solução pode ser particionado tanto pela ramificação direita ($\Delta(x, x') \leq p$), quanto pela ramificação esquerda ($\Delta(x, x') \geq p+$) 1. Tais disjunções definem o tamanho da vizinhança através do parâmetro k.

A estrutura representada pela Figura 8 mostra a ideia básica do algoritmo, que é de forçar uma escolha de maneira que a solução não seja nem tão rápida a ponto de não encontrar soluções melhores que a incumbente (solução atual do nó), mas que se execute em um tempo razoável. Assim, os triângulos representam subárvores exploradas pelo método exato. A partir de uma primeira solução incumbente inicial x'^1 (no primeiro nó), o ramo esquerdo contem a vizinhança $N(x'^1, p)$ que será explorada pelo segundo nó e resultará em uma melhor solução x'^2 que será a nova solução incumbente ao próximo nó. O processo é repetido e o subespaço que se associa ao nó é fracionado em vista da nova solução incumbente. Para, por exemplo, no nó 4, o subespaço será fracionado como $N(x'2, p) \backslash (x'1, p)$ e será explorado produzindo uma nova

Figura 8 – Estrutura básica do *Local Branching*



Fonte: Fischetti e Lodi (2003)

solução incumbente. A resolução segue para o próximo nó até que, se nenhuma solução melhor seja encontrada no subespaço no nó 6, existe uma inserção da restrição ($\Delta(x, x') \geq p + 1$) no nó 7, para limitar a resolução dali em diante.

Para esta resolução não existe uma força para definir valores de variáveis, mas apenas um guia para a execução do modelo. Por esse motivo, entende-se que possa existir uma antecipação de uma solução incumbente em um nó anterior. Tendo em vista que este modelo busca melhorar a resolução por meio da fase de ramificação, e considerando o efeito da simetria nesta fase, acredita-se que o método possa compensar o esforço gasto quando tratamos problemas simétricos. Além disso, este é um método que pode ser aplicado a qualquer problema linear inteiro misto.

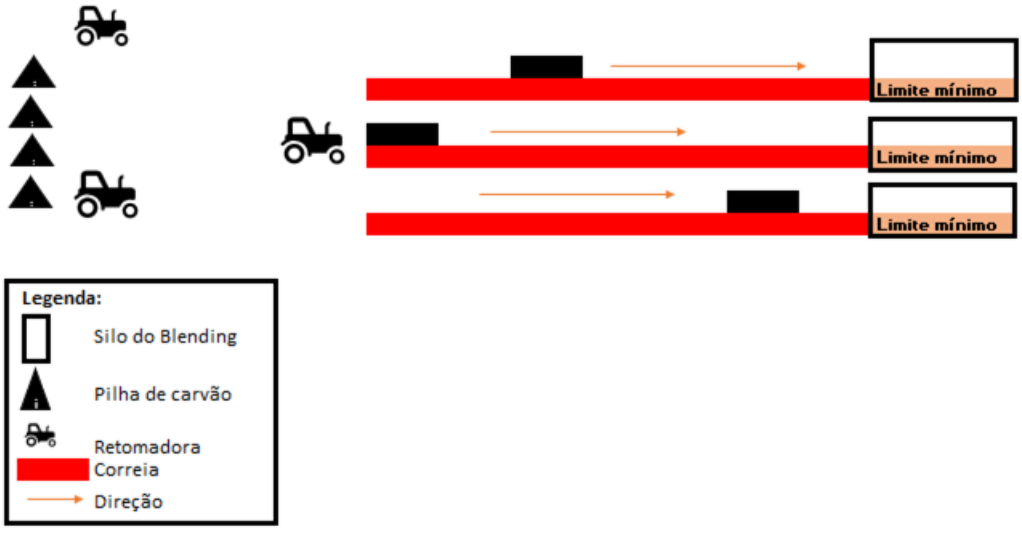
O próximo capítulo irá abordar o PRCPE, através de sua descrição e formulação. Será também proposta uma heurística usada como geradora de solução inicial para o problema com o objetivo de melhorar os resultados e realizar uma comparação ao método *Local Branching*.

3 Descrição do problema

3.1 O Problema de Retomada de Carvão em um Pátios de Estocagem

Os carvões chegam na usina através de um transporte locomotivo e são, em seguida, levados até as empilhadeiras por meio de correias. As empilhadeiras são operadas de modo que os carvões sejam alocados a determinadas pilhas no pátio de estocagem. Uma vez empilhados, eles são recuperados por máquinas retomadoras que os recuperam até outras correias, que servirão para levar o carvão até o silo de $blending$ - responsável pelos próximos processos de mixagem dos carvões. Este processo de recuperar os carvões das pilhas e levá-los até os silos é denominado Retomada de Carvão. Tendo em vista que é preciso garantir uma determinada quantidade de material nos silos para a qualidade do coque e o processo deve ser ininterrupto, existe um prejuízo a cada vez que uma falta de material é detectada. Isso significa que a resolução do problema deve estar na alocação ininterrupta de retomadora aos carvões de maneira a garantir uma diminuição desse desvio negativo. A Figura 9 mostra o funcionamento desse sistema.

Figura 9 – Esquema de representação do processo de Retomada de carvão



Elaborado pela autora

O sistema é descrito por um conjunto de pilhas de carvão I a serem retomados por retomadoras R em um horizonte de tempo T. Ele também conta com parâmetros de limites volumétricos mínimos e máximos de material no sistema, taxa de remoção de carvão tipo i por uma retomadora r, volume de carvão i empilhado ao longo do horizonte de planejamento e o período necessário para que o carvão i, removido pela retomadora r, chegue no silo, na correia e seja retirado do silo.

As variáveis do problema em questão são o volume do carvão i retirado no tempo t, a alocação da retomadora r ao carvão i no tempo t e a variável que indica o desvio negativo do carvão i no tempo t nos silos. (é o que buscamos minimizar).

3.1.1 Formulação do problema

A tabela a seguir contém a descrição do problema através dos símbolos equivalentes aos parâmetros e variáveis.

Tabela 1 – Descrição dos parâmetros e variáveis do problema

Símbolo	Descrição
Parâmetros	
α_{ri}	Períodos necessários para que o carvão i removido pela retomadora r chegue no silo
Δ_{ri}	Taxa de retomada do carvão i pela retomadora r da pilha
γ_i	Taxa de saída do carvão i do silo
Ω_{it}	Volume empilhado de carvão i no tempo t
Lm_i	Limite mínimo do carvão i no silo
Cap	Capacidade do silo
Variáveis	
k_{irt}	Alocação de retomadora r ao carvão i no tempo t: 1 se alocado, 0 caso contrário
$s_{it} \geq 0$	Desvio negativo do carvão i no tempo t (no silo)
$w_{it} \geq 0$	Volume de carvão i no tempo t (no silo)
$x_{it} \geq 0$	Volume de carvão i no tempo t (na pilha)

O modelo matemático genérico do problema pode ser descrito da seguinte forma:

$$\min \sum_{i \in I} \sum_{t \in T} s_{it} \tag{3.1}$$

$$w_{it} - w_{i(t-1)} - \sum_{r \in R} \Delta_{ri} k_{ir(t-\alpha_{ir})} + \gamma_i = 0, \qquad \forall i \in I, \forall t \in T \tag{3.2}$$

$$x_{it} - x_{i(t-1)} + \sum_{r \in R} \Delta_{ri} k_{irt} - \Omega_{ri} = 0, \qquad \forall i \in I, \forall t \in T \tag{3.3}$$

$$s_{it} - Lm_i + w_{it} \geq 0, \qquad \forall i \in I, \forall t \in T \tag{3.4}$$

$$\sum_{r \in R} k_{irt} \leq 1, \qquad \forall i \in I, \forall t \in T \tag{3.5}$$

$$\sum_{i \in I} \sum_{r \in R} \sum_{l=t-\alpha_{ri}}^{t-\alpha_{ri}} k_{irt} \leq 2, \qquad \forall t \in T, \tag{3.6}$$

$$\sum_{\substack{l \in I: \\ v \in R_i}} \sum_{v=t-\alpha_{ri}}^{t} k_{lrv} + k_{irt} - k_{ir(t-1)} \leq 1, \qquad \forall i \in I, \forall r \in R, \forall t \in T \tag{3.7}$$

$$w_{it} \leq Cap, \quad \forall i \in I, \forall t \in T \tag{3.8}$$

A função objetivo (3.1) visa minimizar o desvio negativo de carvão no silo, garantindo o mínimo de falta de material no sistema. As restrições (3.2) e (3.3) garantem o balanço de massa do sistema, tanto para a pilha quanto para o silo. A restrição (3.4) é responsável por impor o limite

mínimo de carvão no silo. A restrição (3.5) impede que uma mesma retomadora seja utilizada simultaneamente. A restrição (3.6) impõem que apenas 2 retomadoras possam ser utilizadas ao mesmo tempo (neste caso, consideramos duas correias, ou seja, a capacidade de utilização de uma retomadora no sistema durante o período de tempo de transferência e processamento do material deve ser igual a 2). Já a restrição (3.7) garante que uma retomadora termine seu processo antes de iniciar outro. Finalmente, a restrição (3.8) determina uma capacidade máxima de material no silo.

Como já mencionado nesse trabalho, a simetria aparece em diversos problemas de otimização linear quando recursos são idênticos. Instâncias simétricas foram abordadas como instâncias que possuem parâmetros invariantes. Considerando que exista, por exemplo, apenas um Δ_{ri} para qualquer alocação Retomadora-Carvão (ou seja, Δ), uma simples permutação entre carvões e retomadoras pode gerar soluções com uma mesma estrutura já que o tempo de alocação seria o mesmo independente da escolha realizada pelo programa.Logo, para garantir um modelo completamente simétrico e uma análise de resultados coerente, todos parâmetros do problema em estudo serão invariantes.

Na seção seguinte, será construída uma heurística que permitirá criar uma solução inicial do modelo, a ser usada pelo resolvedor de problema. A ideia dessa heurística é reduzir a simetria, excluindo soluções isomórficas.

3.2 Uma heurística para o PRCPE

Com o intuito de maiores análises e ganho de performance, foi desenvolvida neste trabalho uma heurística para o problema. Essa heurística servirá como uma solução inicial ao método exato.

O objetivo de tal heurística é construir uma solução que force um primeiro conjunto de alocação em uma tentativa de agilizar a execução do *Branch and Cut*. O algoritmo a seguir descreve como essa primeira solução é construída:

Algoritmo 1 Uma heurística para solução inicial

Entrada: Conjuntos $I = \{1, .., n\}$, $R = \{1, .., m\}$ e $T = \{1, .., p\}$
Vetor $a[r] = \emptyset$ **Saída:** Conjunto de variáveis $S = \{k_{1,1,1}, ..k_{n,k,v}\} = 1$;

```
begin
    for t = 1, ..., (p − 1) do
        int contador = 1;
        for r = 1, ..., m do
            for i = 1, ..., n do
                if k ≤ 3 then
                    if t = 0 then
                        k_{i,r,t} = 1;
                        a[r] = t;
                    else
                        a[r] = a[r] + Δ;
                        if aux ≤ (p − 1) then
                            k_{i,r,a[r]} = 1;
                    if r < (m − 1) then
                        contador + +;
                        r + +;
                    else
                        r = 0;
                else
                    if t < (p − 1) then
                        t + + contador = 0;
                    else
                        t = p;
```

Queremos construir uma alocação que siga um ordem crescente de índice em que haja o maio proveito de tempo possível. O vetor $a[r]$ estoca, a cada nova iteração, o ultimo instante em que a retomadora r foi alocada a um carvão i. Dessa forma, a decisão de alocação de r será sempre em $a[r]$ + Δ.

Considere um problema com $I = \{1, .., 4\}$, $R = \{1, .., 3\}$ e $T = \{1, .., 25\}$. Seguindo a lógica do algoritmo, os tempos de alocação de r seguirão o conjunto $\{p, p+\Delta, p+2\Delta, .., p+k\Delta\}$, tal que $p+k\Delta \leq 25$. Supondo Δ = 2, o conjunto de alocações esperado é de $\{k_{1,1,1} = 1, k_{2,2,1} = 1$, $k_{3,3,3} = 1$, $k_{4,1,3} = 1$,.., $k_{1,1,23} = 1$, $k_{2,2,23} = 1$, $k_{3,3,25} = 1$, $k_{4,1,25} = 1\}$. Em seguida, enviamos as alocações em uma tentativa de ajudar o resolvedor.

Como é possível observar no algoritmo, para o conjunto de variáveis $k_{i,r,t}$, existe uma alocação inicial de todas retomadoras aos carvões, até que todos carvões sejam alocados ao

máximo e a capacidade da retomadora seja respeitada. Espera-se que fazendo isso exista uma melhora do limite superior e evite uma árvore exploratória muito extensa (eliminando soluções isomórficas). Uma vez executada a solução através da utilização de uma heurística como solução inicial, será possível a comparação ao método exato e ao método *Local Branching*.

4 Resultados

4.1 Instanciação e implementação

Para análise de resultados, foram geradas 21 instâncias variando conforme os parâmetros α_{ri}, Δ_{ri}, γ_i, Ω_{it}, Lm_i, Cap_s e conforme os índices I, R e T. A Tabela 2 apresenta cada uma dessas instância nomeadas conforme o padrão $i\_T\_R\_\Delta_{ri}\_\gamma_i$.

Tabela 2 – Variação das instâncias

Instância	r	i	t	Δ_{ri}	α_{ri}	γ_i	Ω_{it}	Lm_i	Cap
i_25_3_40_8	3	5	25	40	3	8	50	100	300
i_25_3_80_10	3	5	25	80	3	10	50	200	6000
i_25_9_40_8	9	15	25	40	3	8	50	100	300
i_25_9_80_10	9	15	25	80	3	10	50	200	6000
i_30_3_40_8	3	5	30	40	3	8	50	100	300
i_30_3_80_10	3	5	30	80	3	10	50	200	6000
i_30_9_40_8	9	15	30	40	3	8	50	100	300
i_30_9_80_10	9	15	30	80	3	10	50	200	6000
i_35_3_40_8	3	5	35	40	3	8	50	100	300
i_35_3_80_10	3	5	35	80	3	10	50	200	6000
i_35_9_40_8	9	15	35	40	3	8	50	100	300
i_35_9_80_10	9	15	35	80	3	10	50	200	6000
i_40_3_40_8	3	5	40	40	3	8	50	100	300
i_40_3_80_10	3	5	40	80	3	10	50	200	6000
i_40_9_40_8	9	15	40	40	3	8	50	100	300
i_40_9_80_10	9	15	40	80	3	10	50	200	6000
i_45_3_40_8	3	5	45	40	3	8	50	100	300
i_45_9_40_8	9	15	45	40	3	8	50	100	300
i_50_3_40_8	3	5	50	40	3	8	50	100	300
i_50_9_40_8	9	15	50	40	3	8	50	100	300

O método exato foi desenvolvido na linguagem computacional c++, com a utilização do resolvedor ILOG CPLEX (IBM, 2020) versão 12.8 pelo método *Branch and Cut*. Todas instâncias foram rodadas com um tempo limite de 1800 segundos em sistema operacional Windows 10, com processador Intel(R) Core(TM) i5-3210M CPU @ 2.50GHZ 6 Gb de memória RAM.

A heurística foi implementada e utilizada como uma solução inicial para o problema através do método $IloCplex.addMIPStart$. Tal método possibilita aos usuários de fornecer ao CPLEX um ponto de avanço inicial para a otimização da programação ($MIPStart$). Um $MIpStart$ pode incluir valores para variáveis inteiras e fracionárias, e não necessariamente apresenta valores para todas as variáveis do problema (IBM, 2020).

Para o *Local Branching*, existe uma função da biblioteca padrão do CPLEX, $IloCplex :: Param :: MIP :: Strategy :: LBHeur$ que permite ativar de maneira simples a utilização do

método em uma tentativa de melhorar a procura por soluções incumbentes através do parâmetro CPX_ON. Os parâmetros k e o tempo gasto em cada nó são definidos pelo próprio CPLEX e o usuário não tem acesso eles.

A seção seguinte apresenta os resultados obtidos para diferentes cenários dentro das instâncias aqui apresentadas.

4.2 Resultados Computacionais

Como descrito nas seções anteriores, o objetivo deste trabalho é, além de implementar o modelo através do método exato, analisar seu comportamento quando adicionados métodos de melhoria de resolução. Tendo em vista que o problema contém bastante simetria, espera-se que os métodos acelerem a resolução do problema evitando cálculo de soluções isomórficas.

A Tabela 3 descreve os resultados através dos indicadores de tempo de melhor solução inteira encontrada, melhor limite inferior (em toneladas), gap (em %) e o tamanho da árvore de enumeração (em milhares). A Tabela também apresenta uma comparação entre 4 diferentes cenários. O primeiro cenário apresenta o resultado do método exato executado pelo método de *Branch and Cut* robusto. O segundo cenário apresenta uma melhoria de performance proposta pelo resolvedor ILOG CPLEX chamada *Dynamic Search* (de acordo com IBM (2020), este algoritmo tende a encontrar soluções ótimas de maneira mais rápida que o algoritmo convencional). Os dois últimos cenários apresentam resultados para a heurística e para o método de *Local Branching* estudados neste trabalho, respectivamente.

Tabela 3 – Resultados computacionais para o modelo do PRCPE

Instâncias	Melhor inteiro				Melhor Limite Inferior				GAP				Nós			
	B&C	LB	HE	DS	B&C	LB	HE	DS	B&C	LB	HE	DS	B&C	LB	HE	DS
i_25_3_40_8	414	414	414	414	261,3	368,6	369	368	36,8	10,9	10,96	10,95	4264,9	945,9	1077,2	1007,6
i_25_3_80_10	5550	5550	5550	5550	5372,03	5550	5550	5550	3,2	0	0	0,00	5794,0	46,8	0,9	2,5
i_25_9_40_8	1896	1896	1896	1896	1039,9	1670,6	1650,6	1669,7	45,18	11,8	12,9	11,90	113,5	64,8	83,7	194,1
i_25_9_80_10	20340	20340	20340	20340	19980	20010	20010	20010	1,76	1,6	1,6	1,60	770,7	263,9	174,5	393,1
i_30_3_40_8	864	864	864	864	643,6	763,4	780,1	759,2	25,5	11,6	9,7	12,10	2819,9	811,8	1004,9	806,7
i_30_3_80_10	6090	6090	6090	6090	5815	5993,7	5989,1	5999,1	4,5	1,6	1,7	1,49	4520,5	1205,2	2408,7	2908,7
i_30_9_40_8	3512	3474	3474	3474	2561,7	3152,4	3158,9	3166,7	27,1	9,2	9,1	8,80	79,2	30,1	33,8	61,7
i_35_3_40_8	1536	1536	1536	1536	1335,8	1463,3	1429,1	1442,1	13,1	4,7	6,9	6,10	4074,0	842,1	563,0	734,4
i_35_3_80_10	6510	6510	6510	6510	6100,8	6268,91	6299,6	6258,1	6,3	3,7	3,2	4,10	3958,0	702,6	737,2	1015,4
i_35_9_40_8	5850	5850	5850	5850	4934,7	5516,5	5496,1	5562,3	15,6	5,7	6,1	4,90	123,7	22,5	33,8	58,8
i_35_9_80_10	24030	24030	24030	24030	22830	23178	23193,2	23199,2	4,9	3,5	3,4	3,46	842,3	21,1	38,2	34,3
i_40_3_40_8	2370	2370	2370	2370	1253,6	2370	2286.8	2298.7	9.1	0	3.5	3.1	3675,5	231,9	1031,1	1937,3
i_40_3_80_10	6810	6810	6810	6810	6215,9	6394,16	6410,1	6399,5	8,7	6,1	5,9	6,03	3324,0	455,5	618,0	625,4
i_40_9_40_8	8940	8940	8942	8942	8040,76	8621,4	8592,16	8600.4	10.1	3.6	3.9	3.8	98,1	21,7	10,3	20,0
i_40_9_80_10	25350	25350	25350	25350	23361,6	24125,3	24170,1	24059,5	7,8	4,8	4,6	5,09	94,8	14,5	37,0	42,5
i_45_3_40_8	3480	3480	3480	3480	3272,3	3412,6	3452	3407,84	5,9	1,9	0,8	2,10	2870,8	866,3	1260,0	1756,0
i_45_9_40_8	12750	12750	12750	12750	11822,4	12426,7	12418,2	12429,5	7,3	2,5	2,6	2,50	57,0	20,9	21,8	46,3
i_50_3_40_8	4870	4870	4870	4870	4683,2	4799,1	4835,95	4799,7	3,8	1,5	0,7	1,40	2734,0	742,5	881,0	1517,9
i_50_9_40_8	17280	17280	17280	17280	16393,6	16944,5	16844,6	16844,8	5,1	1,9	1,9	1,70	41,3	10,9	11,8	32,1

A primeira coisa que se nota é que, de fato, o *framework Dynamic Search* cumpre o seu propósito e melhora todos indicadores para 100% das instâncias testadas. Em se tratando do *Local Branching*, percebe-se pouca melhora de performance, pois ao compará-lo ao método de *Branch and Cut* (com dynamic search ativado) não há progressos notáveis e, inclusive, algumas pioras de resolução. Evidentemente, o método apresentado não é apropriado para este tipo de problema. Uma explicação seria que o algoritmo tenha dificuldade de distinguir todas as variáveis do problema (levando em consideração que o PRCPE conta com 4 variáveis). Tal dificuldade pode limitar a criação de uma vizinhança eficiente para as instâncias. Além disso, os subproblemas criados pelo método são mistos, isto quer dizer que nem sempre são facilmente resolvidos, o que pode tornar o método inviável.

Um fator importante que não foi tratado neste trabalho diz respeito aos parâmetros para a execução do algoritmo de *Local Branching*. Os tempos de execução de cada nó, e o valor k foram definidos pelos parâmetros padrões do CPLEX. Seria interessante um estudo na alteração desses valores para tentar entender o comportamento do método.

Quando analisamos a heurística, também não notamos melhorias significativas nos resultados. Entretanto, o algoritmo em si foi capaz de construir melhores soluções iniciais que aquelas obtidas pelo CPLEX. Tal fato pode ser observado na Tabela 4, que mostra o melhor resultado inteiro do primeiro nó e o gap para cada instância com e sem a ativação da Heurística.

Tabela 4 – Resultados 1º nó: Método exato (com *Dynamic Search*) e Heurística

	Melhor inteiro		Gap	
Instâncias	Método Exato	Heurística	Método Exato	Heurística
i_25_3_40_8	3456	414	94,1	49,9
i_25_3_80_10	1290	5550	58,9	4,5
i_25_9_40_8	3546	2046	69,5	47,1
i_25_9_80_10	40820	21280	51,1	6,1
i_30_3_40_8	3114	900	80,4	32,2
i_30_3_80_10	17070	6090	66,8	7,1
i_30_9_40_8	6284	3802	59,6	33,5
i_30_9_80_10	50930	22490	57,4	3,6
i_35_3_40_8	3768	1536	66,2	17,2
i_35_3_80_10	17950	6510	66,9	8,7
i_35_9_40_8*	8930	8930	44,3	44,3
i_35_9_80_10*	56380	56380	59,1	59,1
i_40_3_40_8	4442	2370	52,4	11,1
i_40_3_80_10	17440	6930	65,5	13,1
i_40_9_40_8	60630	43996	61,4	46,9
i_40_9_80_10	6252	3480	48,6	7,7
i_45_3_40_8	17150	7010	64,5	13,2
i_45_9_40_8*	76770	76770	69,5	69,5
i_50_3_40_8	7452	4870	38,1	5,4

De fato, a heurística conseguiu trazer bons resultados para uma solução inicial. Porém, grande parte das soluções não foram suficientemente boas a ponto de trazer um alto ganho de performance e, consequentemente, não houve prova de otimalidade dentro do prazo limite.

Observando mais afundo o funcionamento da heurística, foi possível verificar que instâncias maiores demandam um maior tempo de construção da solução inicial. Alguma soluções também não foram construídas conforme o previsto pelo algoritmo 3.2 da seção 3.2.

Finalmente, comparando o método *Local Branching* à heurística, percebe-se um resultado sem padronização, em que cada método serviu melhor uma instância - e às vezes da mesma maneira - sem apresentar padrões. Dependendo das necessidades do processo, é interessante rodar apenas a heurística como solução final, já que seu resultado para a função objetivo foi satisfatório.

5 Considerações Finais

Muitos problemas reais quando implementados em modelos de programação são descritos como NP-difíceis, apresentando alta complexidade de resolução. Um outro fator associado à dificuldade de resolução de problemas é a simetria, responsável pela criação de soluções isomórficas redundantes. Dessa forma, alguns problemas não são capazes de provar otimalidade em um tempo razoável. Margot (2009) mostrou que as soluções isomórficas prejudicam de maneira significativa a resolução pelo método tradicional de *Branch and Cut*. Nesse sentido, alguns autores, como Ostrowski, Anjos e Vannelli (2010), buscaram meios de tratar problemas simétricos através de métodos que atuam nas fases internas de ramificação do algoritmo.

Dentre os problemas reais complexos, o PRCPE apresenta relevância por envolver elevados custos dentro de uma indústria siderúrgica. Portanto, este trabalho teve por objetivo modelar e implementar um modelo de programação que atenda a planta de planejamento de alocação de retomadoras a carvão de maneira a otimizar o processo e garantir o estoque necessário no silo de $blending$. Buscou-se aqui também entender a implicação da simetria no algoritmo de *Branch and Cut* e tentar solucionar esta dificuldade de resolução através de um método que atue na ramificação (*Local Branching*). Em uma segunda tentativa de melhoria de resolução, foi elaborada uma heurística que forneça uma solução inicial ao método exato.

O método *Local Branching* não se mostrou apropriado para o problema em questão. Uma teoria é que a quantidade de variáveis do problema possa ter influenciado o funcionamento do algoritmo. A escolha dos parâmetros k e tempo gasto por nó também podem ter influenciado o comportamento do resolvedor, visto que esses últimos foram escolhidos conforme padrão do CPLEX e não de acordo com as características do problema. Já a simetria, apresentou propostas satisfatórias de soluções iniciais, porém não foram suficientemente boas a ponto de auxiliar o CPLEX em provar otimalidade no tempo limite estabelecido. De maneira geral, os métodos não apresentaram melhoria de performance significativa para o método exato.

Um ponto importante que convém destacar aqui compreende as limitações durante a realização deste trabalho. As instabilidades e características da máquina podem ter influenciado negativamente na execução dos testes. Acredito que melhores características de CPU poderiam gerar resultados mais fáceis de ser analisados e com menor instabilidade. A quantidade de instâncias também foi uma limitação do problema pois, para que pudessem ser executadas em condições razoáveis, foram limitadas em tamanho e quantidade.

Mesmo que os objetivos de melhoria através de métodos não tenham sido atingidos, acredito que sejam, ainda, oportunidades a serem trabalhadas. Para o método de *Local Branching*, uma implementação do método manualmente (ou seja, sem utilizar a função fornecida pela biblioteca do CPLEX) pode ser uma maneira de adequá-lo ao problema, possibilitando variações

dos parâmetros. Já para a heurística, uma implementação de soluções incumbentes via callbacks pode trazer ao método exato um maior número de antecipação de soluções. Esta implementação pode ser feita através de Matheurísitcas (i. e. Heurísticas que se apoiam ao método exato para seu desenvolvimento).

5.1 Conclusão

Finalmente, espera-se que este trabalho sirva para instigar outros estudos nesse âmbito. Existe uma busca constante na área da pesquisa operacional por métodos que acelerem e resolvam em tempo razoável problemas NP-Difíceis. A simetria está presente em diversos problemas reais e é um assunto ainda pouco abordado na literatura, por este motivo tem sua relevância em novos estudos.

Referências

ANGELELLI, E. et al. A reclaimer scheduling problem arising in coal stockyard management. *Journal of Scheduling*, Springer, v. 19, n. 5, p. 563–582, 2016. 1

FISCHETTI, M.; LODI, A. Local branching. *Math. Program , Ser. B*, p. 26–28, 2003. 10, 11

HANOUN, S. et al. An effective heuristic for stockyard planning and machinery scheduling at a coal handling facility. In: IEEE. *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*. [S.l.], 2013. p. 206–211. 1

HU, D.; YAO, Z. Stacker-reclaimer scheduling for raw material yard operation. In: IEEE. *Third International Workshop on Advanced Computational Intelligence*. [S.l.], 2010. p. 432–436. 1, 2, 3

IBM. *Knowledge Center IBM*. 2020. <http://www.ibm.com>. Accessed: 2020-05-25. 17, 18

MARGOT, F. Symmetry in integer linear programming. *50 Years of Integer Programming, Springer*, p. 1–4;35, 2009. 2, 4, 22

MARTINEZ, L. C.; CUNHA, A. S. da. Um arcabouço local branching para problemas de otimização combinatória aplicado ao problema da arvore de custo mínimo com k arestas. *XL Simpósio Brasileiro de Pesquisa Operacional*, 2008. 9

OSTROWSKI, J.; ANJOS, M. F.; VANNELLI, A. Symmetry in scheduling problems. *Groupe d'études et de recherche en analyse des décisions*, 2010. 4, 9, 22

PWC, B. *Siderurgia no Brasil*. [S.l.: s.n.], 2018. 1

SCALCON, C. G. Um modelo de otimização para o problema de dimensionamento e programação de lotes de produção em maquina unica. 2012. 4

SILVA, G. L. R. da. Otimização da mistura de carvões na produção de coque metalúrgico. Programa de Pós-Graduação em Engenharia de Materiais. Rede Temática em . . . , 2011. 1

SUH, M. S.; LEE, Y. J.; KO, Y. K. A two-level hierarchical approach for raw material scheduling in steelworks. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 10, n. 5, p. 503–515, 1997. 1, 2, 3

WOLSEY, L. A. *Integer Programming*. [S.l.: s.n.], 1998. v. 1. chapitre 7 : 91 - 100. 5, 6, 8