O Problema de Sequenciamento de Caminhões em um Centro de *Crossdocking* com Duas Máquinas

Gabriela Braga Fonseca
Departamento de Engenharia de Produção – Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627, Pampulha – Belo Horizonte – MG – CEP 31270-901
gabrielabragafonseca@gmail.com

Martín Gómez Ravetti
Departamento de Engenharia de Produção – Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627, Pampulha – Belo Horizonte – MG – CEP 31270-901
martin.ravetti@dep.ufmg.br

Thiago Henrique Nogueira
Departamento de Engenharia de Produção – Universidade Federal de Viçosa
Rodovia MG-230 – Rio Paranaíba – MG – CEP 38810-000
thiagoh.nogueira@ufv.br

RESUMO

O presente trabalho visa desenvolver formas eficientes para resolver o problema de sequenciamento de caminhões em um centro de *crossdocking*, denotado por $F2|CD|C_{max}$, e formulado como um problema de sequenciamento do tipo *flowshop* com duas máquinas, com restrições de *crossdocking*, no qual a função objetivo busca minimizar o *makespan* (C_{max}). Considerou-se um modelo de programação linear inteira com variáveis indexadas no tempo. Para validar e avaliar as soluções foram realizados testes com 500 instâncias. Implementamos a Relaxação Lagrangeana com Algoritmo do Volume, de forma a obter boas soluções em tempo computacionalmente eficiente. Além disso, quatro heurísticas polinomiais foram propostas e analisadas com objetivo de encontrar um limite superior para o problema e possibilitar melhorias na Relaxação Lagrangeana. Os resultados obtidos mostraram a eficiência da Relaxação Lagrangeana juntamente com as heurísticas implementadas.

PALAVRAS CHAVE. Programação Linear Inteira, *Crossdocking*, Relaxação Lagrangeana.

L&T - Logística e Transportes, PM - Programação Matemática.

ABSTRACT

This work aims to develop effective ways to solve the truck scheduling problem at a crossdocking facility, denoted by $F2|CD|C_{max}$ and formulated as a scheduling problem of flowshop type with two-machines with crossdocking constraints, in which the objective function seeks to minimize the makespan (C_{max}). For this, an integer linear programming model with time-indexed variables is considered. To confirm and evaluate the solutions tests with 500 instances were performed. We implemented the Lagrangean Relaxation with Volume Algorithm, in order to obtain good solutions in time computationally efficient. Furthermore four polynomial heuristics have been proposed and analyzed in order to find an upper bound for problem and to improve the solution of the Lagrangean Relaxation. The results demonstrate the efficiency of Lagrangean relaxation proposed framework.

KEYWORDS. Integer Linear Programming, Crossdocking, Lagrangean Relaxation.

L&T - Logistics and Transport, PM - Mathematical Programming.

1. Introdução

Diante da grande valorização por diferenciais competitivos que agreguem valor para a empresa, é imprescindível que se valorize a agilidade, a excelência e a satisfação dos clientes, bem como a redução dos custos das empresas. Em meio a este cenário, buscam-se alternativas para que se consiga alcançar melhorias nos processos logísticos, sem que a qualidade seja afetada. Uma dessas alternativas pode ser o *crossdocking*.

Um Centro de *Crossdocking* (CCD) funciona como um ponto intermediário na rede de suprimentos em que se realiza o transbordo de cargas, sem intenção de estocagem, recebendo caminhões com cargas completas de diversos fornecedores. Dentro do CCD, as cargas são retiradas dos caminhões de chegada, separadas, combinadas e recarregadas em caminhões de saída, de acordo com os pedidos específicos dos clientes. O *Crossdocking* elimina ou reduz significativamente a estocagem e coleta dos produtos.

Este trabalho trata o problema de sequenciamento de caminhões em um centro de *crossdocking*, denotado por $F2|CD|C_{max}$, e interpretado como um problema de *flowshop* com duas máquinas, com restrições de *crossdocking*, no qual a função objetivo busca minimizar o *makespan* (C_{max}). O problema consiste em determinar uma sequência de descarregamento dos caminhões que chegam ao CCD e uma sequência de carregamento dos caminhões que saem do CCD de forma a minimizar o *makespan*.

Dessa forma, primeiramente estudamos o modelo de programação linear inteira com formulação baseada em indexação no tempo, posteriormente elaboramos a relaxação lagrangeana para o problema, implementamos o algoritmo do volume para resolvê-la, e demonstramos que os subproblemas podem ser resolvidos em tempo polinomial. A fim de obter bons limites superiores desenvolvemos quatro heurísticas polinomiais.

Neste artigo, propomos uma Relaxação Lagrangeana e quatro heurísticas polinomiais para o problema de sequenciamento *flowshop crossdocking* com duas máquinas. O modelo aqui considerado foi proposto inicialmente por [Chen e Lee, 2009], e posteriormente estudado por [Lima, 2014]. Este artigo baseia-se nos trabalhos desenvolvidos por esses autores, com o intuito de ampliar e melhorar os resultados presentes na literatura. Trabalhos relacionados com o problema de sequenciamento em centros de *crossdocking* com múltiplas máquinas podem ser estudados em [Chen e Song, 2009] e [Cota et al., 2016].

2. Trabalhos Relacionados ao tema *Crossdocking*

O aumento da utilização de centros de *crossdocking* e a consequente maior exposição do tema tem motivado diversos autores a pesquisar novas maneiras de melhorar as operações associadas ao *crossdocking*. Segundo os autores [Boysen, 2010] e [Belle et al., 2012], centros de *crossdocking*, assim como unidades de produção, lidam com problemas envolvendo tomada de decisão, desde o nível estratégico até o nível operacional, esses problemas podem ser alocados de acordo com a seguinte classificação: localização do CCD, *layout*, roteamento de veículos, atribuição de docas, sequenciamento de caminhões, sequenciamento interno de recursos.

Exemplos de decisões estratégicas em um CCD podem ser vistos em muitos trabalhos da literatura. Problemas que envolvem a localização dos centros de *crossdocking* podem ser estudados em: [Klose e Drexl, 2005], [Chen et al., 2006], e [Chen, 2007]. Já os estudos que tratam do *layout* do centro podem ser verificados em: [Bartholdi e Gue, 2002] e [Lee et al., 2008]. Alguns trabalhos lidam com o roteamento de veículos, tais como: [Oh et al., 2006] e [Clausen et al., 2009]. No que diz respeito as decisões operacionais, alguns trabalhos consideram o problema de atribuição de docas: [Belle et al., 2012], [Boysen e Fliedner, 2010] e [Bozer e Carlo, 2008].

Outros trabalhos tratam do sequenciamento de caminhões nos centros de *crossdocking*, nesse sentido, a Tabela 1, produzida por [Boysen e Fliedner, 2010] e adaptada pelos autores, resume e detalha alguns dos trabalhos nessa área de pesquisa.

[Chen e Lee, 2009] estudaram o problema de sequenciamento *flowshop crossdocking* com duas máquinas, cujo objetivo é sequenciar os caminhões de chegada e saída de forma a minimizar o

Tabela 1: Histórico de pesquisas relacionadas ao sequenciamento de caminhões em CCD. (Fonte: Adaptado de [Boysen e Fliedner, 2010]).

Publicação	Notação do Problema	Complexidade	Contribuição
[Miao et al., 2009]	$[M\|limit, t_{io}\|\star]$	NP-difícil	MM, HM, P
[Chen e Lee, 2009]	$[E2\|t_j = 0\|C_{max}]$	NP-difícil	B, ES, P
[Chen e Lee, 2009]	$[E2\|t_j = 0\|C_{max}]$	NP-difícil	P
[Chen e Song, 2009]	$[E\|t_j = 0\|C_{max}]$	NP-difícil	MM, HS, B
[Boysen, 2010]	$[E\|p_j = p, no - wait, t_j = 0\|\sum T_o]$	Desconhecida	MM, HM, ES
[Boysen e Fliedner, 2010]	$[E\|t_{io}, f_{ix}\|\sum W_s U_s]$	NP-difícil	MM, P
[Boysen e Fliedner, 2010]	$[E\|t_i = 0, f_{ix} \mid \sum W_s U_s]$	NP-difícil	P
[Boysen et al., 2010]	$[E2\|p_j = p, change\|C_{max}]$	NP-difícil	MM, HS, HI, ES, P
[McWilliams, 2010]	$[E\|no - wait\|*]$	Desconhecida	HM
[Vahdani e Zandieh, 2010]	$[E2\|change\|C_{max}]$	NP-difícil	MM, HM
[Araújo e Melo, 2010]	$[E\|t_j = 0\|C_{max}]$	NP-difícil	HM
[Arabani et al., 2011]	$[E2\|change\|C_{max}]$	Desconhecida	HM
[Larbi et al., 2011]	$[E2\|pmtn\|*]$	NP-difícil	MM, ES
[Lima, 2014]	$[E2\|t_j = 0\|\sum C_j^2]$	NP-difícil	MM, HS, HM
[Cota et al., 2016]	$[E\|t_j = 0\|C_{max}]$	NP-difícil	MM, HS

As siglas da coluna "Contribuição"correspondem a: MM (Modelo Matemático), B (Programação por *bound*), HI (Procedimento de Melhoria da Heurística), HS (Heurística para solução inicial), HM (Metaheurística), P (Propriedades e.g. complexidade do problema), e ES (Procedimento de resolução exato).

makespan. A modelagem foi feita como um problema de *flowshop* com duas máquinas, porém com restrições de precedência, a fim de garantir que o caminhão de saída seja processado depois que todas as tarefas de seus predecessores sejam concluídas. Os autores provam que esse problema é fortemente NP-difícil e apresentam uma heurística baseada no algoritmo de Johnson e um algoritmo de *branch-and-bound*. Os resultados mostram que o problema pode resolver instâncias com até 60 caminhões em tempo aceitável.

[Chen e Song, 2009] estendem o trabalho de [Chen e Lee, 2009] considerando múltiplas docas de entrada e saída, denominando problema de *crossdocking* híbrido de dois estágios. Nesse caso múltiplos veículos podem ser carregados ou descarregados ao mesmo tempo, considerando que as docas são como máquinas paralelas. Os autores apresentam um modelo de programação inteira mista e propõem heurísticas baseadas no algoritmo de Johnson para solucionar o problema.

3. Formulação do Problema

O modelo de programação inteira considerado neste trabalho adota a formulação de indexação no tempo proposta por [Lima, 2014] e trata a função objetivo como proposto por [Chen e Lee, 2009], cujo critério de otimização é minimizar o *makespan* (C_{max}).

3.1. Definição do Problema

Em um centro de *crossdocking* no qual chegam *n* caminhões, cada um carregado com produtos que são demandados por um ou vários clientes, deve-se descarregar os produtos desses caminhões e carregá-los em *m* caminhões de saída, estes são responsáveis por carregar vários tipos de produtos para destinos específicos. Cada caminhão de saída só pode deixar o centro de *crossdocking* se todos os produtos necessários para aquele caminhão já tiverem sido descarregados pelos caminhões de chegada correspondentes em algum momento. Com relação ao número de docas no centro de *crossdocking*, considera-se que existem duas, uma doca de entrada (máquina 1 - M1) e outra de saída (máquina 2 - M2). Dessa forma, o problema é sequenciar o descarregamento dos caminhões de chegada e o carregamento dos caminhões de saída de forma a minimizar o *makespan* (C_{max}), ou seja, minimizar a data de conclusão do último *job* (caminhão) processado pela máquina 2. Este problema é considerado como NP-difícil [Chen e Lee, 2009].

3.2. Modelo Matemático

Para a formulação apresentada a seguir os seguintes conjuntos, parâmetros e variáveis são considerados.

Conjuntos

- $J^1 = \{j_1^1, j_2^1, ..., j_n^1\}$ representa os caminhões de chegada, que devem ser processados em M1.
- $J^2 = \{j_1^2, j_2^2, ..., j_m^2\}$ representa os caminhões de saída, que devem ser processados em M2.

As restrições de *crossdocking* são representadas pela seguinte condição: para cada *job* $j_j^2 \in J^2$, existe um subconjunto S_j de *jobs* em J^1, tal que j_j^2 só pode ser processado em M2 se todos os *jobs* em S_j tiverem sido concluídos em M1. Considera-se que cada elemento do subconjunto S_j apresente pelo menos um elemento, ou seja, um determinado *job* $j \in J^2$ possui ao menos um *job* $j \in J^1$ como precedente.

Parâmetros de Entrada

- n: Número de *jobs* a serem processados na máquina 1.
- m: Número de *jobs* a serem processados na máquina 2.
- p_{ij}: Tempo de processamento do *job j* na máquina *i*.
- T: Horizonte de tempo considerado - Uma primeira estimativa para o horizonte de tempo é a soma dos tempos de processamento de todos os *jobs*.

Variáveis de Decisão

- A variável x_{jt} $(\forall j \in J^1, \forall t \in T)$, será igual a 1, se o *job j* começar no instante *t*. Caso contrário, o valor da variável será igual a 0.
- A variável y_{jt} $(\forall j \in J^2, \forall t \in T)$, será igual a 1, se o *job j* começar no instante *t*. Caso contrário, o valor da variável será igual a 0.

O modelo matemático completo (F) é apresentado a seguir.

$$(F) \qquad Min \;\; Z = C_{max} \tag{1}$$

sujeito à:

$$\sum_{t=0}^{T-p_{1j}} x_{jt} = 1, \qquad \forall j \in J^1, \tag{2}$$

$$\sum_{t=0}^{T-p_{2j}} y_{jt} = 1, \qquad \forall j \in J^2, \tag{3}$$

$$\sum_{j \in J^1} \sum_{s=max(0;t-p_{1j}+1)}^{t} x_{js} \leq 1, \qquad \forall t \in T, \tag{4}$$

$$\sum_{j \in J^2} \sum_{s=max(0;t-p_{2j}+1)}^{t} y_{js} \leq 1, \qquad \forall t \in T, \tag{5}$$

$$\sum_{t=0}^{T-p_{2j}} t y_{jt} - \sum_{t=0}^{T-p_{1k}} (t + p_{1k}) x_{kt} \geq 0, \qquad \forall j \in J^2 \wedge \forall k \in S_j, \tag{6}$$

$$C_{max} \geq p_{2j} + \sum_{t=0}^{T-p_{2j}} t y_{jt}, \qquad \forall j \in J^2, \tag{7}$$

$$x_{jt} \in \{0, 1\}, \qquad \forall j \in J^1 \wedge \forall t \in T, \tag{8}$$

$$y_{jt} \in \{0, 1\}, \qquad \forall j \in J^2 \wedge \forall t \in T, \tag{9}$$

$$C_{max} \geq 0. \tag{10}$$

A função objetivo 1 minimiza a data de conclusão do último *job* processado pela máquina 2. O conjunto de restrições 2 diz que cada *job* $j_j^1 \in J^1$ deve iniciar seu processamento em uma e somente uma data dentro do horizonte de planejamento *T*. O conjunto 3 trabalha com o mesmo raciocínio, porém aplicado aos *jobs* $j_j^2 \in J^2$. As restrições indicadas por 4 garantem que um *job* $j_j^1 \in J^1$ não inicia seu processamento enquanto outro *job* estiver sendo processado na máquina 1. O

conjunto de restrições 5 trabalha da mesma forma que o conjunto anterior, no entanto é aplicado aos *jobs* $j_j^2 \in J^2$. O conjunto 6 são as restrições de precedência do tipo *crossdocking*. Assim, para cada relação de precedência existente na instância, cria-se uma restrição desse conjunto, na qual a data de início do *job* $j_j^2 \in J^2$ deve ser maior ou igual à data de conclusão de seu precedente $j_k^1 \in S_j$. O conjunto de restrições 7 indicam que a variável C_{max}, *makespan*, deve ser a máxima data de conclusão dos *jobs* processados na máquina 2. Por fim os conjuntos 8 à 10 definem o domínio das variáveis do modelo.

4. Relaxação Lagrangeana

Seja L(λ) a relaxação Lagrangeana da formulação F, quando as restrições 6 são dualizadas e sua violação penalizada na função objetivo. Para cada restrição é associado um multiplicador de lagrange λ representando o peso dado à violação. O conjunto de multiplicadores de lagrange será representado por λ_{jk}, com $j \in J^2$ e $k \in S_j$. Observe que essas restrições são as restrições de precedência do tipo *crossdocking*, decisões de acoplamento em M1 com as decisões em M2. Assim, o subproblema L(λ) é definido como:

$$L(\lambda) = min\ \ C_{max} + \sum_{j \in J^2} \sum_{k \in S_j} \lambda_{jk} (\sum_{t=0}^{T-p_{1k}} (t + p_{1k}) x_{kt} - \sum_{t=0}^{T-p_{2j}} t y_{jt}) \tag{11}$$

sujeito à (2) – (5), (7) – (10), $\lambda_{jk} \geq 0, \forall j \in J^2 \wedge k \in S_j$.

Agora somos capazes de desacoplar o problema em dois novos problemas de sequenciamento, um para cada máquina. O subproblema X considera M1 enquanto o subproblema Y considera M2. O valor de L(λ), que será um limite inferior para o problema, é obtido agregando os subproblemas L$(\lambda)_x$ e L$(\lambda)_y$.

Subproblema em X

Isolando os termos da função objetivo que contêm x, chega-se ao subproblema em X:

$$L(\lambda)_x = min \sum_{j \in J^2} \sum_{k \in S_j} \lambda_{jk} \sum_{t=0}^{T-p_{1k}} (t + p_{1k}) x_{kt} \tag{12}$$

sujeito à (2), (4) e (8).

Reescrevendo a função objetivo acima sob a perspectiva dos *jobs* em J^1, temos:

$$min \sum_{j \in J^1} \sum_{t=0}^{T-p_{1j}} (t + p_{1j}) x_{jt} w_j^1 \tag{13}$$

onde $w_j^1 = \sum_{i \in J^2} \lambda_{ij}$. Se $j \notin S_i : \lambda_{ij} = 0$.

O problema citado acima é conhecido como Tempo Total de Conclusão Ponderado [Pinedo, 2008.], denotado por $1||\sum C_j W_j$. Esse problema pode ser resolvido por ordenação de acordo com a regra WSPT (*Weighted Shortest Processing Time First*) proporta por Smith em 1956 [Smith, 1956]. De acordo com essa regra, a solução ótima é obtida ordenando-se os *jobs* em ordem decrescente de w_j^1 / p_{1j}, e pode ser obtida em O($nlog(n)$).

Subproblema em Y

Com os termos que contém y isolados da função objetivo do subproblema lagrangeano, obtém-se o subproblema em Y:

$$L(\lambda)_y = min\ \ C_{max} - \sum_{j \in J^2} \sum_{k \in S_j} \lambda_{jk} \sum_{t=0}^{T-p_{2j}} t y_{jt} \tag{14}$$

sujeito à (3), (5), (7), (9), (10), $\lambda_{jk} \geq 0$.

Definindo $w_j^2 = -\sum_{k \in S_j} \lambda_{jk}, \forall j \in J^2$, podemos reescrever 14 da seguinte maneira:

$$L(\lambda)_y = min\ \ C_{max} + \sum_{j \in J^2} \sum_{t=0}^{T-p_{2j}} t y_{jt} w_j^2 \tag{15}$$

O primeiro termo da função objetivo acima representa a maior data de conclusão na máquina 2. O segundo termo representa o somatório das datas de início ponderadas dos *jobs* na máquina 2, este problema é conhecido como Tempo Total de Início Ponderado, denotado $\sum I_j W_j$. Dessa forma, o subproblema em Y pode ser definido como $1||C_{max} + \sum I_j W_j$. Para resolver este subproblema, criamos uma regra denominada WSPT-TRD capaz de alocar os *jobs* de maneira correta, conforme pseudocódigo abaixo.

A regra WSPT-TRD funciona da seguinte maneira, primeiramente aplica-se a regra proposta por Smith em 1956 [Smith, 1956] para gerar a sequência de *jobs* na máquina 2. O próximo passo consiste em definir a maneira correta de alocar os *jobs* no horizonte de tempo. Para isso foi necessário realizar uma avaliação da expansão do *makespan* C_{max} quando os pesos associados aos *jobs* são negativos. Quando os pesos são negativos, cria-se uma situação de *trade-off* na função objetivo, conforme explicado a seguir.

Analisando a situação em que os pesos dos *jobs* são positivos (no nosso caso $w_j^2 = 0$, dado que $\lambda_{jk} \geq 0$), tanto C_{max} quanto $\sum I_j W_j$ possuem comportamento similar, ou seja, dado que a função objetivo é de minimização, a solução ótima é encontrada alocando-se os *jobs* no início do horizonte de tempo de forma a obter o menor valor possível para a função objetivo.

Quando os pesos dos *jobs* são negativos ($w_j^2 < 0$), os termos da função objetivo possuem comportamento divergente, de um lado o *makespan* C_{max} tende a alocar os *jobs* no início do horizonte de tempo, e do outro o $\sum I_j W_j$ tende a alocar os *jobs* o mais tarde possível, uma vez que, para pesos negativos, isso faria com que a função objetivo decrescesse.

Para lidar com esse *trade-off* analisou-se a taxa de crescimento de C_{max} quando deslocamos os *jobs* com peso negativo mais para o final do horizonte. Verificamos que a taxa de crescimento/expansão de C_{max} é constante e igual a 1, ou seja, independentemente do tamanho do deslocamento dos *jobs* no horizonte de tempo, C_{max} sempre seria acrescido de uma unidade. Assim criamos uma variável denominada $Negsoma$ para o cálculo da soma dos pesos negativos, pois, se a soma dos pesos negativos é menor que -1, o deslocamento dos *jobs* mais para o fim do horizonte compensa o aumento de C_{max}, definindo a melhor maneira de alocar os *jobs*.

Pseudocódigo da regra WSPT-TRD:

1. Calcule os pesos na máquina 2, sendo $w_j^2 = -\sum_{k \in S_j} \lambda_{jk}$.
2. Obtenha a sequência ordenando-se os *jobs* em ordem decrescente de w_j^2 / p_{2j}, conforme a regra WSPT de Smith.
3. Se existir pesos negativos ($w_j^2 < 0$), calcule a soma dos pesos negativos através de $Negsoma = \sum_{j \in J^2} w_j^2$. O cálculo da soma dos pesos negativos é importante, pois define a forma adequada de alocação dos *jobs*, de forma a compensar o *trade-off* entre *makespan* e tempo total de início ponderado quando $w_j^2 < 0$.
4. Se $w_j^2 = 0$ ou $Negsoma \geq -1$, aloque os *jobs* a partir de $t = 0$, de frente para trás, conforme sequência gerada pelo WSPT. No caso estudado como $\lambda_{jk} \geq 0$, então não temos $w_j^2 > 0$, mas o método criado aplica-se também para $w_j^2 > 0$.
5. Se $w_j^2 < 0$, aloque os *jobs* a partir de t = T, de trás para frente, mantendo a sequência gerada pela regra WSPT.
6. Calcule C_{max} como a maior data de término.
7. Calcule a função objetivo igual a $C_{max} - \sum_{j \in J^2} \sum_{k \in S_j} \lambda_{jk} \sum_{t=0}^{T-p_{2j}} t y_{jt}$.

A regra WSPT-TRD é ótima para o problema $1||C_{max} + \sum I_j W_j$ [Fonseca, 2015].

5. Resolvendo o Dual Lagrangeano

Neste artigo, o dual lagrangeano foi resolvido pelo Algoritmo do Volume conforme apresentaremos a seguir.

5.1. Algoritmo do Volume

No algoritmo proposto as restrições de precedência são dualizadas. Os multiplicadores de Lagrange definem um peso para os *jobs* alocados em uma dada posição. Se o peso do *job* é um valor grande, significa que ele tem um maior impacto na função objetivo, pois é grande a violação sobre a restrição dualizada, ou seja, o *job* está alocado em uma posição desvantajosa. Essa informação é importante e pode ser usada para decidir quando sequenciar os *jobs*.

Seja x, y e z, a solução do subproblema em X, solução do subproblema em Y e o valor da função objetivo do subproblema lagrangeano, respectivamente. Seja λ os multiplicadores de Lagrange obtidos pelo Algoritmo do Volume e $\nu(\lambda, x, y)$ o subgradiente. Considere UB como limite superior ou solução viável. O UB é encontrado através das heurísticas construtivas polinomiais tal como proposto na seção 5.2. Os passos do algoritmo proposto podem ser resumidos conforme a seguir:

Passo 0: Inicialize um vetor λ_{jk}, onde $\lambda_{jk} = 0$. Resolva o subproblema lagrangeano e obtenha a solução inicial do Subproblema em X (x^0), do Subproblema em Y (y^0) e o valor da função objetivo (z^0). Obtenha um UB através da heurística polinomial e compute $\bar{UB}$ sendo a melhor solução viável obtida pelas heurísticas polinomiais. Faça $\bar{x} = x^0$, $\bar{y} = y^0$, $\bar{z} = z^0$ e k=1.

Passo 1: Compute o subgradiente $\nu(\lambda_{jk-1}, \bar{x}, \bar{y})$ e $\lambda_{jk} = \bar{\lambda_{jk}} + s\nu(\lambda_{jk-1})$, o cálculo do tamanho do passo s é dado pela equação 18. Resolva o subproblema lagrangeano com o novo λ_{jk} e obtenha as soluções x^k, y^k e z^k. Então faça $\bar{x}$=αx^k + (1-α)$\bar{x}$, $\bar{y}$=αy^k + (1-α)$\bar{y}$ e $\bar{\lambda}$=$\alpha\lambda^k$ + (1-α)$\bar{\lambda}$, onde α é um número entre 0 e 1, definido por combinação convexa tal como definido em 16.

Passo 2: Se $z^k > \bar{z}$ atualize $\bar{\lambda}$=λ_{jk} e $\bar{z}$=z^k. Se $\bar{z}$ é melhorado em 10% vá para o Passo 3. Senão vá para o Passo 4.

Passo 3: Heurísticas Construtivas Polinomiais. Nesse passo as quatro heurísticas são testadas e o melhor valor encontrado para o limite superior computado. Além disso, se o UB for melhorado, adicionamos cortes no horizonte, reduzindo o horizonte de tempo, que passa a receber o valor do melhor limite superior encontrado até o momento. Dessa forma conseguimos melhorar os limites, tanto superior quanto inferior, e aumentar o desempenho do algoritmo a cada iteração.

Passo 4: Critério de Parada: se satisfeito então pare. Senão k = k + 1 e volte ao Passo 1.

O tamanho do passo s e o parâmetro α usados para definir $\bar{x}$ e $\bar{y}$ são calculados tal como proposto em [Fukuda, 2007]. Primeiro definimos α_{opt} da seguinte forma:

$$\alpha_{opt} = argmin \parallel \alpha\nu'^{k}_{(\lambda_{jk-1},x^k,y^k)} + (1-\alpha)\nu^{k}_{(\lambda_{jk-1},\bar{x},\bar{y})} \parallel^2 \tag{16}$$

O melhor vetor de parâmetro para o algoritmo do volume é π=0.00679, MaxWaste=24, factor=0.87753, α_{max}=0.30337, st=1.41179, α=0.08300, amarela=0.48159 e verde=1.56487, valores definidos através da utilização do método SPOT (maiores detalhes em [Fonseca, 2015]). O parâmetro α é computado como:

$$\alpha = \alpha_{max} * \alpha \quad se \quad \alpha_{opt} < 0; \quad \alpha = min\{\alpha_{opt}, \alpha_{max}\} \quad se \quad \alpha_{opt} \geq 0 \tag{17}$$

Para definir o valor de π definimos três tipos de iterações baseado nos trabalhos de [Barahona e Ladányi, 2006]. Cada iteração sem melhoria é denominada iteração vermelha, dessa forma o número máximo de iterações sem melhoria do limite inferior é definido pelo parâmetro MaxWaste. Se $z^k > \bar{z}$ e $\nu'^{k}_{(\lambda_{jk-1},x^k,y^k)}\nu^{k}_{(\lambda_{jk-1},\bar{x},\bar{y})} < 0$, significa que um passo maior em direção a ν^k resultaria em um valor menor para z^k. Essas iterações são denominadas amarela, caso contrário, a iteração é denominada verde. A cada iteração amarela multiplicaremos π pelo parâmetro amarela. A cada iteração verde multiplicaremos π pelo parâmetro verde. Depois de 24 (MaxWaste) iterações vermelhas consecutivas, multiplique π pelo parâmetro factor. Assim, o tamanho do passo s na iteração k, é definida como:

$$s = \frac{\pi * (st * UB - \bar{z}))}{\parallel \nu_{(\lambda_{jk-1},\bar{x},\bar{y})} \parallel^2} \tag{18}$$

Finalmente o critério de parada do Algoritmo do Volume é determinado se um dos seguintes critérios é satisfeito:

1. Número máximo de iterações, neste caso 1000 iterações, ou;
2. Número de iterações sem melhoria de limite inferior, definido por: $\frac{z^k - \bar{z}}{\bar{z}} < 0.1$, ou;
3. Módulo do subgradiente nulo ou desprezível: $\| \nu_{(\lambda_{jk-1}, \bar{x}, \bar{y})} \|^2 \leq 0.000001$.

5.2. Heurísticas Construtivas Polinomiais

Nessa seção são detalhadas as heurísticas construtivas polinomiais implementadas para obtenção de um limite superior do problema $F2|CD|C_{max}$.

Heurística 1

1. Obtenha a sequência na máquina 1 através da resolução do subproblema em X, ou seja, ordenando-se os *jobs* dessa máquina de acordo com a regra WSPT.
2. Calcule, para cada *job* da máquina 2, sua *release date* (r_j, $j \in J^2$). A *release date* de um *job* j representa a data a partir da qual aquele *job* pode ser sequenciado. É calculada como $r_j = max_{k \in S_j} C_k$, ou seja, a maior data de conclusão dentre os precedentes de j na máquina 1 na sequência dada. A sequência obtida deve minimizar o *makespan*, que é a função objetivo do problema original.
3. Obtenha a sequência na máquina 2 ordenando-se os *jobs* em ordem não decrescente de *release dates* (r_j).

Heurística 2

1. Obtenha a sequência na máquina 2 através da resolução do subproblema em Y.
2. Sequencie os *jobs* na máquina 1 conforme a sequência obtida pelo subproblema em Y, respeitando as relações de precedência de *crossdocking*. Caso exista um *job* na máquina 1 sem relação de precedência na máquina 2, sequencie esse *job* por último.
3. Calcule as novas *release dates* dos *jobs* na máquina 2, a partir da sequência na máquina 1.
4. Obtenha a sequência na máquina 2 ordenando-se os *jobs* em ordem não decrescente de *release dates* (r_j).

Heurística 3

1. Para cada *job* na máquina 1 ($k \in J^1$), calcule um $\beta_k = \frac{\sum_{j \in J^2} \sum_{k \in J^1} \lambda_{jk}}{Nprec_k}$. Onde $Nprec_k$ corresponde ao número de vezes que o *job k* da máquina 1 é precedente na máquina 2.
2. Obtenha a sequência na máquina 1 ordenando-se os *jobs* em ordem decrescente de β_k.
3. Calcule as novas *release dates* dos *jobs* na máquina 2, a partir da sequência na máquina 1.
4. Obtenha a sequência na máquina 2 ordenando-se os *jobs* em ordem não decrescente de *release dates* (r_j).

Heurística 4

1. Para cada *job* na máquina 2 ($j \in J^2$), calcule um $\beta_j = \frac{\sum_{j \in J^2} \sum_{k \in S_j} \lambda_{jk}}{Nprec_j}$. Onde $Nprec_j$ corresponde ao número de precedentes que o *job j* possui.
2. Obtenha a sequência na máquina 2 ordenando-se os *jobs* em ordem crescente de β_j.
3. Sequencie os *jobs* na máquina 1 conforme a sequência na máquina 2, respeitando as relações de precedência de *crossdocking*. Caso exista um *job* na máquina 1 sem relação de precedência na máquina 2, sequencie esse *job* por último.
4. Calcule as novas *release dates* dos *jobs* na máquina 2, a partir da sequência na máquina 1.
5. Obtenha a sequência na máquina 2 ordenando-se os *jobs* em ordem não decrescente de *release dates* (r_j).

6. Experimentos Computacionais

Os experimentos realizados neste artigo foram executados utilizando-se uma máquina com processador Intel (R) Xeon (R) CPU X5690 @ 3.47GHz com 24 processadores, 132 GB de RAM, e sistema operacional Ubuntu Linux. Foi utilizada a linguagem de programação C++ e o software de otimização CPLEX 12.4.

6.1. Geração de Instâncias

As instâncias utilizadas neste trabalho foram geradas através do software MATLAB, baseando-se nas propriedades das instâncias utilizadas por [Chen e Lee, 2009]. A Tabela 2 apresenta um resumo das instâncias geradas e suas respectivas características. Os valores dos tempos de processamento dos *jobs*, presentes nos grupos J^1 ou J^2, foram definidos de forma aleatória, assim como o número de precedentes, de acordo com a respectiva distribuição uniforme definida na Tabela.

Tabela 2: Sumário das instâncias geradas para teste, separadas em grupo 1 e 2, e subgrupos, de acordo com o número de *jobs* na máquina 1 ($JobsM1(n)$). Além disso, a tabela informa o número de *jobs* na máquina 2 ($JobsM2(m)$), o número máximo de precedentes do *job* $j \in J^2$ (NP), o tempo de processamento dos *jobs* (TP) e o código da instância (*Código*).

Grupo	Jobs M1(n)	Jobs M2(m)	NP	TP	Código
1	5	3-4-5-6-7	U(1,4)	U(1,10)	5-m-np-1
	10	6-8-10-12-14	U(1,9)	U(1,10)	10-m-np-1
	20	12-16-20-24-28	U(1,19)	U(1,10)	20-m-np-1
	40	24-32-40-48-56	U(1,39)	U(1,10)	40-m-np-1
	60	36-48-60-72-84	U(1,59)	U(1,10)	60-m-np-1
2	5	3-4-5-6-7	U(1,4)	U(10,100)	5-m-np-2
	10	6-8-10-12-14	U(1,9)	U(10,100)	10-m-np-2
	20	12-16-20-24-28	U(1,19)	U(10,100)	20-m-np-2
	40	24-32-40-48-56	U(1,39)	U(10,100)	40-m-np-2
	60	36-48-60-72-84	U(1,59)	U(10,100)	60-m-np-2

O número de *jobs* em cada estágio (n e m) é definido a priori. Para cada n, cinco instâncias com diferentes valores de m são consideradas. E para cada par (n, m), geramos 10 problemas diferentes, então o *benchmark* consiste de 500 instâncias, 50 instâncias por linha da Tabela 2.

6.2. Resultados Computacionais

Os resultados dos testes computacionais realizados estão expostos nas Tabelas 3 e 4. Na Tabela 3 a coluna Código da Instância refere-se a instância trabalhada, os itens MC, RL e RLg significam modelo completo, relaxação linear e relaxação Lagrangeana, respectivamente. Já as siglas FO, Melhor_LI, LI, LS, GAP e T representam os resultados médios para a função objetivo, melhor limite inferior encontrado, limite inferior, limite superior, o GAP de otimalidade calulado por $\frac{LS - LI}{LS}$ e o tempo de processamento em segundos, respectivamente.

Para cada instância, 10 diferentes problemas foram resolvidos e a média dos resultados é apresentada na Tabela 3. O tempo de execução foi limitado em 1 hora. Se em menos de 1 hora é encontrada a solução ótima do problema, a execução é automaticamente interrompida, caso a solução não seja encontrada em menos de uma hora, a execução é interrompida e os valores arquivados. O traço (-) significa que o valor correspondente não foi encontrado dentro do limite de tempo.

6.3. Análise dos Resultados

Por meio dos resultados expostos na Tabela 3, é possível comparar o desempenho do modelo completo, da relaxação linear e da relaxação Lagrangeana, para todas as instâncias testadas. O experimento mostrou que o modelo proposto é eficiente para resolver instâncias com *n*=5 e *n*=10 *jobs* com tempos de processamento curtos, e em sua maioria, em baixo tempo computacional. Não foi identificada a solução ótima para as maiores instâncias do grupo 1 e no grupo 2 apenas duas instâncias tiveram sua solução ótima identificada. Esse fato expõe a dificuldade de resolução de modelos com indexação no tempo. Com a indexação no tempo, o número de variáveis é proporcional ao horizonte de tempo, e quanto maior o número de *jobs* da instância, maior o horizonte de tempo necessário para sequenciá-los. Estes fatores aumentam a complexidade do problema, sendo necessário maior esforço computacional para resolvê-lo.

Tabela 3: Comparação entre Modelo Completo, Relaxação Linear e Relaxação Lagrangeana.

Código da	MC				RL		RLg			
Instância	FO	Melhor_LI	GAP	T(seg)	FO	T(seg)	LI	LS	GAP	T(seg)
5-3-4-1	33	33	0%	0.2	23	0.0	26	35	22%	0.6
5-4-4-1	33	33	0%	0.3	23	0.0	25	33	22%	0.7
5-5-4-1	34	34	0%	0.5	23	0.0	28	35	19%	0.6
5-6-4-1	39	39	0%	1.3	24	0.0	31	41	24%	0.4
5-7-4-1	44	44	0%	3.7	25	0.0	36	48	23%	0.4
10-6-9-1	59	59	0%	20.2	36	0.0	40	67	39%	0.9
10-8-9-1	66	66	0%	167.5	39	0.1	44	73	39%	0.7
10-10-9-1	71	71	0%	369.6	39	0.1	52	81	35%	0.5
10-12-9-1	82	77	5%	2440.5	40	0.1	70	91	23%	0.2
10-14-9-1	90	86	5%	3141.6	44	0.1	80	104	23%	0.0
20-12-19-1	135	114	15%	3600	65	0.3	67	148	55%	1.1
20-16-19-1	153	102	32%	3600	66	0.7	90	165	46%	1.0
20-20-19-1	167	107	36%	3600	68	0.6	115	182	37%	0.8
20-24-19-1	190	115	38%	3600	72	0.9	139	203	31%	1.1
20-28-19-1	206	130	36%	3600	82	1.0	153	222	31%	0.7
40-24-39-1	306	164	46%	3600	122	2.0	140	309	55%	1.1
40-32-39-1	-	-	-	3600	126	3.5	179	350	48%	1.0
40-40-39-1	-	-	-	3600	124	4.9	220	377	42%	1.0
40-48-39-1	-	-	-	3600	135	12.7	262	413	37%	1.0
40-56-39-1	-	-	-	3600	156	26.2	306	447	31%	1.7
60-36-59-1	-	-	-	3600	176	34.5	198	464	57%	3.8
60-48-59-1	-	-	-	3600	176	43.0	269	521	48%	4.6
60-60-59-1	-	-	-	3600	181	133.1	337	578	42%	3.0
60-72-59-1	-	-	-	3600	203	112.7	396	632	37%	4.9
60-84-59-1	-	-	-	3600	241	122.8	463	696	33%	8.6
5-3-4-2	317	317	0%	21.0	225	0.2	167	338	50%	0.0
5-4-4-2	330	330	0%	354.9	229	0.3	194	352	43%	0.0
5-5-4-2	343	339	1%	1570.1	227	0.4	237	369	34%	0.3
5-6-4-2	393	340	12%	2388.3	237	0.6	287	425	32%	0.1
5-7-4-2	441	376	13%	2882.5	246	1.9	356	485	25%	0.1
10-6-9-2	625	438	29%	3600	355	10.4	366	688	46%	0.4
10-8-9-2	731	454	37%	3600	363	12.4	470	730	35%	0.6
10-10-9-2	820	443	45%	3600	375	12.5	582	830	30%	0.4
10-12-9-2	1014	435	56%	3600	393	31.9	698	911	23%	0.3
10-14-9-2	1125	415	62%	3600	409	68.3	803	1041	23%	0.1
20-12-19-2	-	-	-	3600	651	195.4	671	1481	55%	2.6
20-16-19-2	-	-	-	3600	655	237.7	898	1644	45%	3.2
20-20-19-2	-	-	-	3600	674	322.6	1125	1813	38%	2.3
20-24-19-2	-	-	-	3600	720	453.4	1367	2013	32%	0.9
20-28-19-2	-	-	-	3600	828	1697.9	1600	2186	27%	0.7
40-24-39-2	-	-	-	3600	1184	3600	1352	2751	51%	2.0
40-32-39-2	-	-	-	3600	1176	3600	1768	3405	48%	9.7
40-40-39-2	-	-	-	3600	1021	3600	2182	3746	42%	10.3
40-48-39-2	-	-	-	3600	1280	3600	2623	4131	36%	14.3
40-56-39-2	-	-	-	3600	-	3600	3066	4465	31%	17.1
60-36-59-2	-	-	-	3600	-	3600	1977	4631	57%	19.3
60-48-59-2	-	-	-	3600	-	3600	2675	5212	48%	25.9
60-60-59-2	-	-	-	3600	-	3600	3357	5776	42%	33.0
60-72-59-2	-	-	-	3600	-	3600	3953	6318	37%	47.8
60-84-59-2	-	-	-	3600	-	3600	4623	6950	33%	13.2

Tabela 4: Resultados das heurísticas: % de vezes em que as heurísticas obtiveram melhor limite superior.

Grupo	Heurística 1	Heurística 2	Heurística 3	Heurística 4
Grupo 1	30%	31%	13%	27%
Grupo 2	31%	30%	9%	30%

Ao se comparar as relaxações, verifica-se que a relaxação Lagrangeana fornece melhores limites inferiores do que os encontrados pela relaxação linear, além de ser capaz de resolver todas as instâncias em baixo tempo computacional. Isso foi possível graças a redução que realizamos no horizonte de tempo, sempre que o limite superior é melhorado. Quanto ao limite superior nota-

se que os valores encontrados não são muito distantes do valor fornecido pelo modelo completo, principalmente nas instâncias em que a solução ótima foi identificada. Dessa forma, trabalhar na melhora do cálculo dos limites, associando-os as heurísticas, pode ser uma boa opção na busca de soluções ótimas para instâncias maiores.

Conforme a Tabela 4, a heurística 1 assim como a heurística 2, apresentaram melhores desempenhos, uma vez que encontraram os melhores limites superiores na maioria das iterações. A heurística 4 também apresentou bom desempenho sendo capaz de encontrar uma solução próxima da solução ótima do problema. Já a heurística 3 apresentou desempenho inferior as demais heurísticas.

7. Conclusões e perspectivas

Neste trabalho, um modelo matemático para o problema de sequenciamento de caminhões em um centro de *crossdocking* com duas máquinas foi analisado. Implementamos a relaxação lagrageana através do algoritmo do volume e propomos quatro heurísticas construtivas polinomiais a fim de encontrar um limite superior para o problema.

O desempenho do modelo mostrou-se dependente do número de *jobs* e do tempo de processamento. A formulação matemática, por empregar indexação no tempo, apresentou dificuldades para resolver problemas de maiores dimensões. Os resultados apontaram que a relaxação linear apresentou dificuldade de resolução nas instâncias de grande porte, consumindo muito tempo. Já a Relaxação Lagrangeana se mostrou bastante eficiente uma vez que obteve limites mais próximos dos valores ótimos em tempos computacionais reduzidos. O resultado da relaxação lagrangeana superou o da relaxação linear devido a redução no horizonte de tempo durante a sua execução.

As heurísticas polinomiais implementadas apresentaram resultados bastante satisfatórios, não ultrapassando 2 segundos para as instâncias avaliadas. Duas das quatro heurísticas analisadas, a heurística 1 e a heurística 2, tiveram melhor desempenho na obtenção de um limite supeior, seguida pela heurística 4 que também apresentou resultados interessantes.

Diante dos fatos expostos conclui-se que a utilização da técnica de Relaxação Lagrangeana através do algoritmo do volume é uma forma eficiente para resolver o problema de sequenciamento de caminhões em um centro de *crossdocking*. Propõe-se como trabalhos futuros a aplicação de uma metaheurística híbrida que utilize a relaxação Lagrangeana integrada com heurísticas para resolver o problema de modo a obter boas soluções em menor tempo computacional e com garantia de performance.

Referências

Arabani, A. B., Ghomi, S. F., e Zandieh, M. (2011). Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage. *Expert systems with Applications.*, 38(3)::1964–1979.

Araújo, D. P. M. e Melo, B. M. R. (2010). Heurísticas construtivas para o sequenciamento de caminhões em centros de cross-docking. Master's thesis, Universidade Federal de Minas Gerais.

Barahona, F. e Ladányi, L. (2006). Branch and cut based on the volume algorithm: Steiner trees in graphs and max-cut. *RAIRO - Operations Research*, 40:53–73.

Bartholdi, J. e Gue, K. (2002). Reducing labor costs in an ltl crossdocking terminal. *Operations Research.*, 48(6)::823–832.

Belle, J. V., Valckenaers, P., e Cattrysse, D. (2012). Cross-docking: State of the art. *Omega.*, 40:: 827–846.

Boysen, N. (2010). Truck scheduling at zero-inventory cross docking terminals. *Computers & Operations Research.*, 37(1)::32–41.

Boysen, N. e Fliedner, M. (2010). Cross dock scheduling: Classification, literature review and research agenda. *Omega.*, 38(6)::413–422.

Boysen, N., Fliedner, M., e Scholl, A. (2010). Scheduling inbound and outbound trucks at cross-docking terminals. *OR spectrum.*, 32(1)::135–161.

Bozer, Y. e Carlo, H. (2008). Optimizing inbound and outbound door assignments in less-than-truckload crossdocks. *IIE Transactions.*, 40::1007–1018.

Chen, F. e Lee, C.-Y. (2009). Minimizing the makespan in a two-machine cross-docking flow shop problem. *European Journal of Operational Research.*, 193(1)::59–72.

Chen, F. e Song, K. (2009). Minimizing makespan in two-stage hybrid cross docking scheduling problem. *Computers & Operations Research.*, 36(6)::2066–2073.

Chen, J. (2007). A hybrid heuristic for the uncapacited single allocation hub location problem. *Omega.*, 35::211–220.

Chen, P., Guo, Y., Lim, A., e Rodrigues, B. (2006). Multiple crossdocks with inventory and time windows. *Computers and Operations Research.*, 33::43–46.

Clausen, J., Cordeau, J., Laporte, G., Wen, M., e J., L. (2009). Vehicle routing scheduling with cross-docking. *Journal of the Operational Research Society.*, 60::1708–1718.

Cota, P. M., Gimenez, B. M., Araújo, D. P., Nogueira, T. H., de Souza, M. C., e Ravetti, M. G. (2016). Time-indexed formulation and polynomial time heuristic for a multi-dock truck scheduling problem in a cross-docking centre. *Computers & Industrial Engineering.*, 95::135–143.

Fonseca, G. B. (2015). O problema de sequenciamento de caminhões em um centro de cross-docking com duas máquinas. Master's thesis, Universidade Federal de Minas Gerais.

Fukuda, E. H. (2007). Algoritmo de volume e otimização não diferenciável. Master's thesis, USP.

Klose, A. e Drexl, A. (2005). Facility location models for distribution system design. *European Journal of Operational Research.*, 162::4–29.

Larbi, R., Alpan, G., Baptiste, P., e Penz, B. (2011). Scheduling cross docking operations under full, partial and no information on inbound arrivals. *Computers & Operations Research.*, 38(6):: 889–900.

Lee, K., Lee, Y., e Jung, J. (2008). Positioning of goods in a cross-docking environment. *Computers & Industrial Engineering.*, 54(3)::677–689.

Lima, M. F. (2014). O problema de sequenciamento de caminhões numa estação de cross-docking com duas máquinas: Formulação indexada no tempo, relaxação lagrangeana e geração de colunas. Master's thesis, Universidade Federal de Minas Gerais.

McWilliams, D. L. (2010). Iterative improvement to solve the parcel hub scheduling problem. *Computers & Operations Research.*, 59(1)::136–144.

Miao, Z., Lim, A., e Ma, H. (2009). Truck dock assignment problem with operational time constraint within crossdocks. *European journal of operational research.*, 192(1)::105–115.

Oh, Y., Hwang, H., Chab, C., e Lee, S. (2006). A dock-door assignment problem for the korean mail distribution center. *Computers & Industrial Engineering.*, 51(2)::288–296.

Pinedo, M. (2008.). Scheduling: Theory, algorithms and systems. *Hall, Englewood Cliffs.*

Smith, W. E. (1956). Varius optimizers for single-stage production. *Naval Res. Logist.*, 3::59–66.

Vahdani, B. e Zandieh, M. (2010). Scheduling trucks in cross-docking systems: Robust metaheuristics. *Computers & Operations Research.*, 58(1)::12–24.