LI Simpósio Brasileiro de Pesquisa Operacional
Limeira–SP, 02 a 06 de setembro de 2019

HEURÍSTICAS PARA OBTENÇÃO DE LIMITES SUPERIORES PARA O PROBLEMA DE DIMENSIONAMENTO DE LOTES EM MÁQUINAS PARALELAS

Leonardo Andrade Apolinário
Universidade Federal da Paraíba
João Pessoa - PB - Brasil
leonardo@prac.ufpb.br

Hugo Harry Kramer
Departamento de Engenharia de Produção - Universidade Federal da Paraíba
João Pessoa - PB - Brasil
hkramer@ct.ufpb.br

RESUMO

No contexto do planejamento de produção, os problemas de dimensionamento de lotes são considerados como dos mais importantes na literatura. Neste artigo, consideramos o Problema de Dimensionamento de Lotes de Múltiplos Itens com Restrições de Capacidade em Máquinas Paralelas Distintas. Este problema é NP-difícil e consiste em determinar um plano ótimo de produção para os itens de modo que as demandas sejam atendidas, as capacidades das máquinas sejam respeitadas e o custo total seja minimizado. O objetivo do trabalho é a obtenção de soluções viáveis de qualidade em baixo tempo computacional. Para tanto, é proposta uma abordagem que combina as heurísticas *relax-and-fix* e *fix-and-optimize*. A abordagem proposta é comparada com a formulação compacta do problema sendo resolvida por um resolvedor comercial. Os resultados encontrados demonstram que as soluções obtidas são de qualidade, com *gaps* percentuais menores que 0,531 %.

PALAVRAS CHAVE. Dimensionamento de lotes, Relax-and-fix, Fix-and-optimize.

AD&GP – PO na Administração e Gestão da Produção, OC – Otimização Combinatória, PM – Programação Matemática

ABSTRACT

When dealing with production planning, lot-sizing problems are considered as very important to the literature. In this work, we consider the Multi Item Capacitated Lot-sizing Problem with Unrelated Parallel Machines. This problem is NP-hard and consists of determining an optimal production plan such that the demands are satisfied, the machine capacities are not exceeded and the total cost is minimized. Our objective is to obtain good feasible solutions with a reasonable computational effort. Therefore, we propose an approach combining the relax-and-fix and fix-and-optimize heuristics. The proposed approach is compared to a compact formulation running under a commercial MIP solver. The results show that the obtained solution are of good quality with percentual gaps under 0.531 %.

KEYWORDS. Lot-sizing. Relax-and-fix. Fix-and-optimize.

https://proceedings.science/p/106630?lang=pt-br

1. Introdução

Planejamento, Programação e Controle da Produção (PCP) são atividades que fazem parte da rotina da maioria das organizações que atuam no setor industrial e têm por objetivo gerenciar o processo de transformação de matérias-primas em produtos acabados, garantindo a disponibilidade das primeiras para garantir o funcionamento do processo de maneira eficiente e o atendimento das demandas dos clientes. Robert [1965] sugere que a estrutura hierárquica do PCP pode ser dividida em três níveis de planejamento distintos: estratégico, tático e operacional. Sendo assim, as decisões tomadas no contexto do PCP são fundamentais para que uma organização tenha sucesso em satisfazer as necessidades dos seus clientes e seja competitiva frente a seus concorrentes.

Este trabalho tem como proposta abordar um problema de planejamento da produção de nível tático, onde se deseja determinar planos de produção para itens de modo que suas demandas sejam atendidas ao longo de um horizonte de planejamento. O problema tratado se enquadra na classe dos Problemas de Dimensionamento de Lotes de Produção (*Lot-sizing Problems - LSP*).

O trabalho de Harris [1913] é o pioneiro na literatura a tratar sobre o dimensionamento de lotes. Neste trabalho o autor define o que hoje se conhece por Lote Econômico de Compra ou *Economic Order Quantity* (EOQ) para um único item considerando demandas constantes, tempo contínuo e horizonte de tempo infinito. O EOQ é a quantidade a ser adquirida do item que minimiza a soma do custos fixo e variável.

Quando as demandas são dinâmicas, o tempo é discretizado em períodos e o horizonte de planejamento é finito, o cálculo do EOQ de acordo com o modelo de Harris [1913] não resulta em uma solução de custo mínimo. Neste sentido surge o Problema de Dimensionamento de Lotes de um Único Item sem Capacidade (*Uncapacitated Lot-sizing Problem* - ULSP). No trabalho de Wagner e Whitin [1958] para o ULSP os autores propõem um algoritmo de programação dinâmica baseado em uma propriedade que reduz significativamente o espaço de busca por soluções ótimas. Uma versão computacionalmente mais eficiente deste algoritmo é apresentada por Evans [1985]. A partir do ULSP surgiram diversas variantes considerando múltiplos itens, restrições de capacidade, tempos de *setup* e produção, múltiplos níveis, múltiplas plantas, máquinas paralelas, *backlog*, horas-extras, deterioração dos itens, etc. Para mais detalhes sobre as diversas características presentes nos problemas de dimensionamento de lotes, o leitor pode consultar o trabalho de Karimi et al. [2003]. Mais recentemente, questões associadas à sustentabilidade passaram a ser incorporadas aos LSPs, a exemplo do retorno e remanufatura de itens defeituosos [Teunter et al., 2006; Sahling, 2013] e de restrições sobre emissões de gases de efeito estufa [Absi et al., 2016; Wu et al., 2018].

O Problema de Dimensionamento de Lotes com Múltiplos Itens e restrições de Capacidade (*Capacitated Lot-sizing Problem* - CLSP) é uma extensão do ULSP e foi proposto por Manne [1958]. O autor apresenta uma formulação matemática com um número exponencial de variáveis que posteriormente passou a ser utilizada como base em algoritmos de Geração de Colunas para a resolução do CLSP. Este problema é bastante estudado na literatura e diversas abordagens foram propostas como, por exemplo: formulações de fluxo em redes [Eppen e Martin, 1987], relaxação lagrangeana [Trigeiro et al., 1989], heurísticas [Maes e Van Wassenhove, 1988], algoritmos baseados em geração de colunas [Degraeve e Jans, 2007], *branch-and-cut*, entre outras.

Uma extensão do CLSP é o Problema de Dimensionamento de Lotes com Múlti-

https://proceedings.science/p/106630?lang=pt-br

plos Itens, restrições de Capacidade e Máquinas Paralelas (CLSP-PM). Este será o problema abordado neste trabalho em sua versão com máquinas distintas (*Unrelated Parallel Machines*). O primeiro trabalho de dimensionamento de lotes em um ambiente com máquinas paralelas idênticas foi proposto por Lasdon e Terjung [1971], porém envolvendo no mesmo problema o sequenciamento da produção. Carreno [1990] estendeu o trabalho anterior passando a considerar os tempos de preparação das máquinas. Tratando o CLSP-PM isoladamente pela primeira vez e considerando também os tempos de preparação e máquinas distintas, Sabbag Neto [1993] propôs sua resolução por meio de um procedimento heurístico e apresentou uma formulação matemática. Toledo [1998] estendeu a propriedade de Wagner e Whitin [1958] para o problema com um único item e máquinas paralelas distintas sem restrições de capacidade e apresentou algoritmos de programação dinâmica para resolvê-lo. Neste mesmo trabalho, a autora também propôs um algoritmo *branch-and-bound* com Relaxação Lagrangeana para o CLSP-PM e um outro algoritmo de *branch-and-bound* onde os limites duais são obtidos pela relaxação linear de uma formulação matemática. Heurísticas baseadas em Relaxação Lagrangeana foram propostas para o CLSP-PM por Toledo e Armentano [2006] e Fiorotto e de Araujo [2014]. Este último trabalho é melhorado por Fiorotto et al. [2015] onde os autores propõem métodos heurísitcos baseados na combinação de Relaxação Lagrangeana e decomposição de Dantzg-Wolfe [Dantzig e Wolfe, 1960].

Florian et al. [1980] mostram que o CLSP com um único item e sem considerar os tempos de preparação da máquina é NP-difícil. Ao se considerar os tempos de preparação, Maes et al. [1991] mostram que o problema de saber se há uma solução viável para o CLSP com um único item é NP-completo. Como o CLSP é um caso particular do CLSP-PM, é possível afirma que este último é um problema NP-difícil. Portanto, o CLSP e o CLSP-PM são frequentemente abordados na literatura por meio de métodos de resolução heurísticos capazes de encontrar soluções de boa qualidade com esforço computacional reduzido quando comparados aos métodos de resolução exatos. *Relax-and-fix* [Wolsey, 1998] e *Fix-and-optimize* [Pochet e Wolsey, 2006] são heurísticas baseadas em formulações matemáticas de Programação Inteira que são aplicadas frequentemente a problemas de dimensionamento de lotes. A primeira consiste em encontrar uma solução viável para o problema, podendo ser considerada uma heurística de construção. Já a segunda busca encontrar uma solução melhor a partir de uma solução inicial se assemelhando a um procedimento de busca local. Este trabalho tem por objetivo propor a aplicação destas duas heurísticas com o objetivo de encontrar limites primais para o CLSP-PM. Não é de conhecimento dos autores que estas abordagem tenham sido aplicadas ao problema em questão.

O restante deste artigo está organizado da seguinte forma: na Seção 2, o CLSP-PM é formalmente definido e uma formulação matemática é apresentada. A Seção 3 apresenta as abordagens *relax-and-fix* e *fix-ando-optimize* propostas para a resolução do problema. Em seguida, a Seção 4 apresenta os resultados obtidos pelas abordagens propostas e os compara com os resultados da formulação matemática sendo resolvida pelo Gurobi. Por fim, na Seção 5 são apresentadas as conclusões do trabalho e sugestões de trabalhos futuros.

2. Definição do Problema

O CLSP-PM pode ser definido da forma que segue. Seja $I = \{1, 2, 3, \ldots, n\}$ um conjunto de itens a serem produzidos durante um horizonte de planejamento definido por um conjunto $T = \{1, 2, \ldots, m\}$ de períodos, onde poderá ser utilizado um conjunto $J = \{1, 2, \ldots, r\}$ de máquinas para a produção dos itens. São dadas as demandas d_{it} para cada item $i \in I$ em cada período $t \in T$. Também são dados os custos unitários de produção, os

https://proceedings.science/p/106630?lang=pt-br

custos de *setup* e os custos unitários de estoque para cada item $i \in I$ em cada período $t \in T$ e em cada máquina $j \in J$, denotados por pc_{ijt}, sc_{ijt} e hc_{it}, respectivamente. Além disso, são dados os tempos de *setup*, os tempos unitários de produção para cada item $i \in I$ em cada período $t \in T$ e em cada máquina $j \in J$, e a capacidade de cada máquina $j \in J$ em cada período $t \in T$, definidos respectivamente como st_{ijt}, pt_{ijt} e Cap_{jt}. O objetivo do problema é encontrar planos de produção para cada um dos n itens de modo que suas demandas sejam atendidas sem atrasos e que as capacidades das r máquinas sejam respeitadas, de modo que os custos sejam minimizados. Para tanto, serão consideradas como variáveis de decisão: x_{ijt}, como a quantidade produzida do item i na máquina j no período t; y_{ijt}, variável binaria, indicando se haverá ou não produção do item i na máquina j no período t e s_{it}, como a quantidade em estoque do item i no final do período t.

Dessa forma, o CLSP-PM pode ser modelado como um problema de Programação Linear Inteira Mista e tem a seguinte formulação matemática $\mathcal{FC}$:

$$(\mathcal{FC}) \min \sum_{i=1}^{n}\sum_{j=1}^{r}\sum_{t=1}^{m}(sc_{ijt}\ y_{ijt} + pc_{ijt}\ x_{ijt}) \sum_{i=1}^{n}\sum_{t=1}^{m} hc_{it}\ s_{it} \tag{1}$$

sujeito a

$$\sum_{i-1}^{n}(st_{ijt}\ y_{ijt} + pt_{ijt}\ x_{ijt}) \leq Cap_{jt} \quad \forall j \in J, t \in T \tag{2}$$

$$s_{i,t-1} + \sum_{j=1}^{k} x_{ijt} = d_{it} + s_{it} \quad \forall i \in I, t \in T \tag{3}$$

$$x_{ijt} \leq sd_{itm}\ y_{ijt} \quad \forall i \in I, j \in J, t \in T \tag{4}$$

$$y_{ijt} \in \{0,1\} \quad \forall i \in I, j \in J, t \in T \tag{5}$$

$$x_{ijt} \geq 0 \quad \forall i \in I, j \in J, t \in T \tag{6}$$

$$s_{it} \geq 0 \quad \forall i \in I, t \in T \backslash \{0, m\} \tag{7}$$

A função objetivo (1) busca minimizar a soma dos custos de produção, *setup* e estoque. As restrições (2) garantem que as capacidades das máquinas sejam respeitadas, enquanto as restrições (3) garantem o balanço de estoque e o atendimento das demandas de cada item em cada período. As restrições (4) relacionam as variáveis x e y de modo que só haverá produção se houver *setup*, onde sd_{itm} é a soma das demandas de um item i a partir do período t até o fim do horizonte de planejamento. Por fim, as variáveis de decisão são definidas por (5), (6) e (7).

3. Heurísticas de horizonte deslizante

Esta seção apresenta duas heurísticas de horizonte desizante aplicadas ao CLSP-PM: (i) *relax-and-fix* e (ii) *fix-and-optimize*. A abordagem proposta neste trabalho consiste em utilizar a primeira para obter uma solução viável inicial para o problema. Tal solução será usada como ponto de partida para a heurística *fix-and-optimize* que atua como um procedimento de busca local com o objetivo de melhorar a solução inicial. Nas subseções a seguir as heurísticas serão descritas com mais detalhes.

Ao descrever os procedimentos, dados dois números inteiros a e b, denota-se que uma variável pertence ao intervalo $[a, b]$ se tiver índice $t \in T$ tal que $a \leq t \leq b$.

https://proceedings.science/p/106630?lang=pt-br

3.1. *Relax-And-Fix*

A heurística *relax-and-fix* consiste na decomposição de um modelo de Programação Inteira em subproblemas que são consideravelmente mais fáceis de resolver que o problema original. A decomposição se dá pelo particionamento das variáveis inteiras do modelo onde: (i) um subconjunto das variáveis estão fixadas; (ii) um subconjunto das variáveis estão restritas a a tomarem valores inteiros; e (iii) o restante das variáveis têm as restrições de integralidade relaxadas. A cada iteração, o subconjunto das variáveis fixadas aumenta, o subconjunto das variáveis inteiras é modificado, mas mantém o mesmo tamanho, e o subconjunto das variáveis relaxadas diminui. Quando o subconjunto das variáveis relaxadas fica vazio,uma solução viável inteira é encontrada e o procedimento termina.

Denota-se por $\mathcal{FC}_{RF}^{\alpha,\beta}$ como sendo uma decomposição da formulação $\mathcal{FC}$ em que as variáveis y (i) no intervalo $[1, \alpha - 1]$ estão fixadas, (ii) no itervalo $[\alpha, \beta]$ estão restritas a tomarem valores inteiros, e (iii) têm a restrição de integralidade relaxada no intervalo $[\beta+1, m]$. A heurística *relax-and-fix* proposta toma como entrada a formulação matemática $\mathcal{FC}$, um valor k que define a extensão do intervalo $[\alpha, \beta]$ em que as variáveis y são restritas a tomarem apenas valores inteiros, um valor k' que define o quanto o intervalo $[\alpha, \beta]$ se desloca a cada iteração, e valores para o tempo limite total da heurística ($tLimite$) e para o tempo limite de resolução de cada subproblema ($tLimiteSub$)

Algoritmo 1 Relax-and-fix ($\mathcal{FC}$, k, k', $tLimite$, $tLimiteSub$)

1: $\alpha \leftarrow 1,\ \beta \leftarrow \alpha + k - 1, S \leftarrow \emptyset$
2: **enquanto** S não for inteira and $tDecorrido \leq tLimite$ **faça**
3: $S \leftarrow \text{MIP}(\mathcal{FC}_{RF}^{\alpha,\beta}, tLimiteSub)$
4: **se** S é viável para $\mathcal{FC}_{RF}^{\alpha,\beta}$ **então**
5: Fixe as variáveis y no intervalo $[\alpha, \alpha + k' - 1]$ de acordo com seus valores em S
6: **senão**
7: Encontre o período mais tarde $0 \leq \alpha^{'} < \alpha$ para o qual $\mathcal{FC}_{RF}^{\alpha',\beta}$ é viável
8: $S \leftarrow \text{MIP}(\mathcal{FC}_{RF}^{\alpha',\beta}, tLimiteSub)$
9: Fixe as variáveis y no intervalo $[\alpha', \alpha + k' - 1]$ de acordo com seus valores em S
10: $\alpha \leftarrow \alpha + k',\ \beta \leftarrow \min\{\alpha + k - 1, m\}$
11: **retorne** S

O algoritmo 1 descreve a heurística *relax-and-fix* proposta para o CLSP-PM. Na linha 1 é inicializado o intervalo $[\alpha, \beta]$. Uma iteração do procedimento é descrita no laço das linhas 2–10, que é executado até que uma solução inteira viável seja encontrada para o problema ou o tempo limite seja excedido. A iteração inicia na linha 3 com a resolução do subproblema $\mathcal{FC}_{RF}^{\alpha,\beta}$ utilizando um resolvedor de problemas de Programação Inteira. Se a solução encontrada for viável, são realizadas as fixações das variáveis y (linha 9). Caso contrário (linhas 6–9), tenta-se aumentar o intervalo em que as variáveis y devem assumir valores inteiros até que uma solução viável seja encontrada. Ao fim da iteração, o intervalo $[\alpha, \beta]$ é atualizado na linha 10.

3.2. *Fix-and-optimize*

O comportamento da heuristica *fix-and-optimize* também se baseia na decomposição do problema original em subproblemas menores. As variáveis são particionadas em dois subconjuntos. Em um deles, as variáveis y terão seus valores fixados de acordo com uma solução viável, enquanto no outro as variáveis y estão restritas a tomarem apenas

https://proceedings.science/p/106630?lang=pt-br

valores inteiros. A cada iteração uma nova solução inteira é encontrada. Ao término do procedimento é retornada a melhor solução inteira encontrada.

Considere uma decomposição $\mathcal{FC}^{\alpha,\beta}_{FO}(S)$ da formulação $\mathcal{FC}$ onde as variáveis y estão fixadas nos intervalos $[1, \alpha - 1]$ e $[\beta + 1, m]$ de acordo com uma solução inteira viável S. No intervalo $[\alpha, \beta]$ (que se estende por k períodos) as variáveis y estão restritas a tomarem apenas valores inteiros. A cada iteração uma nova solução inteira é encontrada e o intervalo $[\alpha, \beta]$ é atualizado deslocando-se em k' períodos. Uma rodada é um conjunto de iterações em que o intervalo $[\alpha, \beta]$ percorre todo o horizonte de planejamento. O procedimento termina se o tempo limite ($tLimite$) for alcançado ou se ao término de uma rodada nenhuma solução melhor tenha sido encontrada. Se houver melhoria em uma rodada, ao seu término, o intervalo $[\alpha, \beta]$ é reinicializado.

Algoritmo 2 Fix-and-optimize ($\mathcal{FC}$, S, k, k', $tLimite$, $tLimiteSub$)

1: $S^* \leftarrow S$
2: **repita**
3: $\quad \alpha \leftarrow 1,\ \beta \leftarrow \alpha + k - 1$
4: $\quad$ **enquanto** $tDecorrido < tLimite$ ou $\beta < m$ **faça**
5: $\quad\quad S \leftarrow \text{MIP}(\mathcal{FC}^{\alpha,\beta}_{FO}(S^*), tLimiteSub)$
6: $\quad\quad$ **se** $f(S) < f(S^*)$ **então**
7: $\quad\quad\quad S^* \leftarrow S$
8: $\quad\quad \alpha \leftarrow \alpha + k',\ \beta \leftarrow \min\{\alpha + k - 1, m\}$
9: **até que** $tLimite$ seja alcançado ou S^* não seja melhorada na rodada atual
10: **retorne** S^*

O Algoritmo 2 recebe como entrada a formulação $\mathcal{FC}$, uma solução inteira viável de $\mathcal{FC}$ e os valores de k, k', $tLimite$ e $tLimiteSub$. Na linha 1 a solução S de entrada é considerada como sendo a melhor solução conhecida. O laço das linhas 2–9 descreve uma rodada do procedimento. Os valores de α e β são reinicializados a cada rodada na linha 3. A cada iteração (linhas 4–8) um subproblema é resolvido por meio de um resolvedor de Programação Inteira na linha 5. Se a solução encontrada for melhor que a melhor solução conhecida, esta última é atualizada na linha 7, e o intervalo $[\alpha, \beta]$ é atualizado na linha 8. Ao final, o procedimento retorna a melhor solução encontrada (linha 10).

4. Resultados Computacionais

Nesta seção são apresentados os resultados computacionais para o CLSP-PM obtidos pelas heurísticas propostas em comparação com resultados obtidos por um resolvedor comercial para a formulação $\mathcal{FC}$. Todos os experimentos foram realizados em um computador com processador Intel(R) Core(TM) i7-8550U @4.0GHz com 16 GB de memória RAM, sistema operacional Windows 10 e utilizando o Gurobi versão 8.10 como resolvedor de problemas de Programação Inteira. Os códigos foram escritos em linguagem Julia versão 0.6.3 (instalada em 28 de maio de 2018) em conjunto com o pacote de modelagem matemática JuMP versão 0.18.4. Todas as abordagens foram executadas utilizando apenas uma *thread* do processador. Todos os parâmetros padrão do resolvedor foram mantidos, exceto a tolerância do *gap* relativo de otimalidade (alterada para 10^{-8}) e a tolerância de integralidade das variáveis (alterada para 10^{-6}).

As instâncias utilizadas para realizar os experimentos computacionais foram pro-

https://proceedings.science/p/106630?lang=pt-br

postas no trabalho de Toledo e Armentano [2006][1]. Foi utilizado um subconjunto dessas instâncias com as seguintes características: $n = 25$ itens, $m = 18$ períodos e $r = \{2, 4, 6\}$ máquinas. Além disso, as instâncias são agrupadas em 8 classes de acordo com as características de capacidade, custo de *setup* e tempo de *setup*. As 8 classes de instâncias são representadas por siglas e são definidas a seguir:

- NCLSLT: Capacidade normal, custo de *setup* baixo, tempo de *setup* baixo;
- NCLSHT: Capacidade normal, custo de *setup* baixo, tempo de *setup* alto;
- NCHSLT: Capacidade normal, custo de *setup* alto, tempo de *setup* baixo;
- NCHSHT: Capacidade normal, custo de *setup* alto, tempo de *setup* alto;
- TCLSLT: Capacidade apertada, custo de *setup* baixo, tempo de *setup* baixo;
- TCLSHT: Capacidade apertada, custo de *setup* baixo, tempo de *setup* alto;
- TCHSLT: Capacidade apertada, custo de *setup* alto, tempo de *setup* baixo;
- TCHSHT: Capacidade apertada, custo de *setup* alto, tempo de *setup* alto;

Em cada uma das 8 classes tem-se 10 instâncias para cada uma das 3 combinações possíveis de n, m e r, totalizando um conjunto de 240 instâncias.

Para a formulação matemática $\mathcal{FC}$ foi estabelecido um tempo limite de 600 segundos. Para a heurística *relax-and-fix* foram estabelecidos os seguintes parâmetros: $tLimite = 600$ segundos, $tLimiteSub = 25$ segundos, $k = \frac{m}{3}$ e $k' = \left\lceil \frac{2}{3}\ k \right\rceil$. Já para a heurística *fix-and-optimize* os parâmetros de entrada são: $tLimite = 600$ segundos, $tLimiteSub = 25$ segundos, $k = \frac{m}{3}$ e $k' = \left\lceil \frac{1}{2}\ k \right\rceil$. Além disso, a solução inicial S^* de entrada é a obtida pelo procedimento *relax-and-fix*.

Tabela 1: Resultados comparativos entre as abordagens propostas e a formulação $\mathcal{FC}$

	$\mathcal{FC}$				*Relax-and-fix*				*Fix-and-optimize*			
	gap_{med}	gap_{max}		t_{med}	gap_{med}	gap_{max}		t_{med}	gap_{med}	gap_{max}		t_{med}
Classe	(%)	(%)	OPT	(s)	(%)	(%)	OPT	(s)	(%)	(%)	OPT	(s)
NCLSLT	0.023	0.196	22	224.920	0.244	0.724	0	8.551	0.058	0.252	12	10.868
NCLSHT	0.021	0.123	23	225.405	0.241	0.934	0	9.170	0.060	0.318	10	13.514
NCHSLT	0.023	0.197	22	223.881	0.244	0.725	0	8.479	0.058	0.253	12	10.688
NCHSHT	0.021	0.123	23	224.815	0.241	0.725	0	9.211	0.060	0.318	9	13.636
TCLSLT	0.087	0.377	15	366.322	0.337	1.014	1	15.469	0.132	0.465	7	24.430
TCLSHT	0.092	0.351	14	364.202	0.315	1.008	2	18.374	0.138	0.531	8	34.843
TCHSLT	0.088	0.378	15	353.862	0.337	1.015	1	15.416	0.132	0.467	7	24.321
TCHSHT	0.091	0.351	14	361.951	0.315	1.008	2	18.317	0.137	0.531	8	34.709

Os resultados estão apresentados de forma sumarizada na Tabela 1 para cada uma das 8 classes de instâncias. São comparados os resultados da formulação $\mathcal{FC}$ com os das heurísticas propostas. Na tabela, a primeira coluna contém as classes de instâncias testadas. Para cada uma das abordagens tem-se quatro colunas que reportam o *gap* percentual médio,

[1]Os autores agradecem ao Prof. Diego Fiorotto por disponibilizar os arquivos das instâncias.

https://proceedings.science/p/106630?lang=pt-br

o *gap* percentual máximo, o número de soluções ótimas encontradas e o tempo de execução médio em segundos, respectivamente. O *gap* percentual para uma dada instância é dado por $100 \times \frac{LS-LI}{LS}$, onde LS é o melhor limite superior obtido por uma dada abordagem e LI é o melhor limite inferior obtido pela formulação $\mathcal{FC}$.

É possível observar através dos resultados que as abordagens testadas são capazes de encontrar soluções de boa qualidade. A execução da formulação $\mathcal{FC}$ com o resolvedor Gurobi encontrou soluções com *gaps* percentuais médios menores que 0,1 % em cada classe de instâncias e, no pior dos casos o *gap* percentual foi de 0,378 %. Esta abordagem também foi capaz de encontrar 148 soluções ótimas entre as 240 instâncias testadas. A combinação das heurísticas *relax-and-fix* e *fix-and-optimize* também foi capaz de encontrar soluções de boa qualidade com *gaps* percentuais médios menores que 0,14 %, *gap* máximo de 0,531 %, e foi capaz de encontrar soluções ótimas para 73 instâncias. A abordagem se mostra eficiente em encotrar uma boa solução inicial com a heurística *relax-and-fix* que é consideravelmente melhorada pelo procedimento *fix-and-optimize*.

As duas abordagens são capazes de encontrar soluções de boa qualidade, com uma certa vantagem para a formulação $\mathcal{FC}$ em relação à combinação das heurísticas. No entanto, quando se comparam os tempos computacionais, observa-se que a heurística *relax-and-fix* seguida da *fix-and-optimize* apresenta tempos consideravelmente mais baixos, com uma média de 33,75 segundos, quando se compara à média de 293,17 segundos da formulação $\mathcal{FC}$.

5. Conclusões e Trabalhos Futuros

Este trabalho apresenta a aplicação das heurísticas *relax-and-fix* e *fix-and-optimize* ao Problema de Dimensionamento de Lotes de Múltiplos Itens com Restrições de Capacidade e Máquinas Paralelas Distintas (CLSP-PM). A proposta de combinação das heurísticas *relax-and-fix* e *fix-and-optimize* se mostrou eficiente no sentido de obter soluções de boa qualidade e competitivas quando comparadas com os resultados de um resolvedor comercial de problemas de Programação Inteira, com a vantagem de serem alcançadas em um tempo computacional razoavelmente menor.

Como trabalhos futuros, pretende-se a aplicação de um algoritmo de solução baseado em geração de colunas com o objetivo de obter melhores limites inferiores e encontrar as soluções ótimas das instâncias que não puderam ser resolvidas. Além disso, pode ser estudada uma forma de parametrização automática para as heurísticas propostas. Por fim, sugere-se também a comparação dos resultados obtidos com os do estado da arte da literatura do problema encontrados no trabalho de Fiorotto et al. [2015][2].

Referências

Absi, N., Dauzère-Pérès, S., Kedad-Sidhoum, S., Penz, B., e Rapine, C. (2016). The single-item green lot-sizing problem with fixed carbon emissions. *European Journal of Operational Research*, 248(3):849–855.

Carreno, J. J. (1990). Economic lot scheduling for multiple products on parallel identical processors. *Management Science*, 36(3):348–358.

[2]Os autores entraram em contato com o Prof. Diego Fiorotto para solicitar os resultados completos. Porém, problemas técnicos que não puderam ser resolvidos a tempo o impediram de disponibilizá-los.

https://proceedings.science/p/106630?lang=pt-br

Dantzig, G. B. e Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research*, 8(1):101–111.

Degraeve, Z. e Jans, R. (2007). A new dantzig-wolfe reformulation and branch-and-price algorithm for the capacitated lot-sizing problem with setup times. *Operations research*, 55(5):909–920.

Eppen, G. D. e Martin, R. K. (1987). Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848.

Evans, J. R. (1985). An efficient implementation of the wagner-whitin algorithm for dynamic lot-sizing. *Journal of Operations Management*, 5(2):229–235.

Fiorotto, D. J. e de Araujo, S. A. (2014). Reformulation and a lagrangian heuristic for lot sizing problem on parallel machines. *Annals of Operations Research*, 217(1):213–231.

Fiorotto, D. J., de Araujo, S. A., e Jans, R. (2015). Hybrid methods for lot sizing on parallel machines. *Computers & Operations Research*, 63:136–148.

Florian, M., Lenstra, J. K., e Rinnooy Kan, A. (1980). Deterministic production planning: Algorithms and complexity. *Management science*, 26(7):669–679.

Harris, F. W. (1913). How many parts to make at once. *Factory - The magazine of management*, 10(2).

Karimi, B., Ghomi, S. F., e Wilson, J. (2003). The capacitated lot sizing problem: a review of models and algorithms. *Omega*, 31(5):365–378.

Lasdon, L. S. e Terjung, R. (1971). An efficient algorithm for multi-item scheduling. *Operations research*, 19(4):946–969.

Maes, J., McClain, J. O., e Van Wassenhove, L. N. (1991). Multilevel capacitated lotsizing complexity and lp-based heuristics. *European Journal of Operational Research*, 53(2): 131–148.

Maes, J. e Van Wassenhove, L. (1988). Multi-item single-level capacitated dynamic lot-sizing heuristics: A general review. *Journal of the Operational Research Society*, 39(11): 991–1004.

Manne, A. S. (1958). Programming of economic lot sizes. *Management science*, 4(2):115–135.

Pochet, Y. e Wolsey, L. A. (2006). *Production planning by mixed integer programming*. Springer Science & Business Media.

Robert, N. A. (1965). Planning and control systems: a framework for analysis. *Harvard University Graduate School of Business Administration, Cambridge, MA*.

Sabbag Neto, Z. (1993). Planejamento da produção em máquinas paralelas sob restrições de capacidade. Master's thesis, Universidade Estadual de Campinas, Faculdade de Engenharia Eletrica e de Computação. URL `http://repositorio.unicamp.br/jspui/handle/REPOSIP/259934`.

https://proceedings.science/p/106630?lang=pt-br

Sahling, F. (2013). A column-generation approach for a short-term production planning problem in closed-loop supply chains. *Business Research*, 6(1):55–75.

Teunter, R. H., Bayindir, Z. P., e Den Heuvel, W. V. (2006). Dynamic lot sizing with product returns and remanufacturing. *International Journal of Production Research*, 44 (20):4377–4400.

Toledo, F. M. B. (1998). *Dimensionamento de lotes em máquinas paralelas*. Universidade Estadual de Campinas, Faculdade de Engenharia Eletrica e de Computação, http://www.repositorio.unicamp.br/handle/REPOSIP/261174.

Toledo, F. M. B. e Armentano, V. A. (2006). A lagrangian-based heuristic for the capacitated lot-sizing problem in parallel machines. *European Journal of Operational Research*, 175(2):1070–1083.

Trigeiro, W. W., Thomas, L. J., e McClain, J. O. (1989). Capacitated lot sizing with setup times. *Management science*, 35(3):353–366.

Wagner, H. M. e Whitin, T. M. (1958). Dynamic version of the economic lot size model. *Management science*, 5(1):89–96.

Wolsey, L. A. (1998). *Integer programming*. Wiley.

Wu, T., Xiao, F., Zhang, C., He, Y., e Liang, Z. (2018). The green capacitated multi-item lot sizing problem with parallel machines. *Computers & Operations Research*, 98:149–164.

https://proceedings.science/p/106630?lang=pt-br