LIII Simpósio Brasileiro de Pesquisa Operacional
João Pessoa - PB, 3 a 5 de novembro de 2021

Um Algoritmo Memético Aplicado ao Problema de Roteamento e Produção

Luiz Fernando Rodrigues
Instituto de Ciências Matemáticas e de Computação (ICMC) da Universidade de São Paulo (USP)
Av. Trabalhador Sao-carlense, 400, Centro, São Carlos - SP, Brasil, 13566-590
luiz.rodrigues@uftm.edu.br

Maristela Oliveira dos Santos
Instituto de Ciências Matemáticas e de Computação (ICMC) da Universidade de São Paulo (USP)
Av. Trabalhador Sao-carlense, 400, Centro, São Carlos - SP, Brasil, 13566-590
mari@icmc.usp.br

Bernardo Almada-Lobo
INESC TEC, Faculdade de Engenharia da Universidade do Porto
R. Dr. Roberto Frias, Porto 4200, Portugal
almada.lobo@fe.up.pt

RESUMO

Neste artigo abordamos o Problema de Roteamento e Produção (PRP) que consiste em determinar de maneira integrada o planejamento da produção, dos estoques e o roteamento de veículos de modo a minimizar os custos envolvidos. Uma fábrica é responsável por produzir um único produto visando atender a demanda conhecida de um conjunto de clientes, que são atendidos por uma frota homogênea de veículos com capacidade limitada ao longo do horizonte de planejamento. As abordagens evolutivas não foram exploradas em profundidade para resolver o PRP. Este trabalho mitiga esta lacuna da literatura, apresentando um Algoritmo Memético e testando sua eficácia em um conjunto de instâncias conhecido, comparando os resultados obtidos com um solver comercial de otimização. Experimentos computacionais foram massivamente executados com o objetivo de avaliar o impacto das inúmeras combinações de parâmetros envolvendo a meta-heurística e o solver. A partir de análises estatísticas, evidenciamos a robustez da técnica proposta.
PALAVRAS-CHAVE: Problema de Roteamento e Produção; Algoritmos Meméticos; Cadeia de Suprimentos. Tópicos: Meta-heurísticas; Otimização Combinatória.

ABSTRACT

This work approaches the Production Routing Problem (PRP), which integrates the planning of production, inventory management and vehicle routing in order to minimize the costs involved. More specifically, we consider a plant producing an item to face the known demand of customers within a finite planning horizon. A homogeneous fleet of capacitated vehicles is used. Evolutionary approaches have not been explored in depth to solve PRP. This work mitigates this literature gap, by presenting a Memetic Algorithm and testing its effectiveness on a known set of instances and compare its results against those obtained with a commercial solver. These instances are of small and medium-sized problems, with various levels of complexity. Computational experiments are massively performed with the aim of evaluating the impact of the numerous combinations of parameters of both the meta-heuristic and the solver. Based on statistical analysis of these results, we demonstrate the robustness of the proposed approach.
KEYWORDS: Production Routing Problem; Memetic Algorithm; Supply Chain. Paper topics: Metaheuristics; Combinatorial Optimization.

1. Introdução

Os problemas envolvendo o planejamento da produção, do estoque e da distribuição têm sido amplamente estudados nas últimas décadas, mas publicações que consideram estas decisões integradas em seus diferentes níveis ainda são recentes na literatura [Absi et al., 2018] e [Adulyasak et al., 2015]. O Problema de Roteamento e Produção engloba dois importantes problemas inerentes ao planejamento da cadeia de suprimentos, o problema de dimensionamento de lotes e o de roteamento dos veículos. Tradicionalmente, estes problemas são otimizados sequencialmente e, embora a complexidade do problema integrado seja maior, a integração proporciona importante e significativa redução dos custos da cadeia produtiva [Adulyasak et al., 2015]. Adotamos nesse trabalho sua nomenclatura mais conhecida que é o Problema de Roteamento e Produção (PRP, *Production Routing Problem*). Revisões podem ser encontradas em [Absi et al., 2018] e [Adulyasak et al., 2015].

Por se tratar de um problema NP-difícil [Adulyasak et al., 2015], métodos exatos tem sua aplicação restrita a instâncias de pequeno e médio porte. Considerando o problema com apenas uma fábrica, um produto e frotas homogêneas, destacamos os trabalhos de [Archetti et al., 2011], [Adulyasak et al., 2014a], [Darvish et al., 2019] e [Ramos et al., 2020]. Por outro lado, face à complexidade, temos muitos trabalhos que utilizaram métodos heurísticos com destaque para [Adulyasak et al., 2014b], [Solyalı e Süral, 2017] e [Roostaei e Kamal, 2018].

[Boudia e Prins, 2009] apresentaram um algoritmo memético com diferentes estratégias de gerenciamento populacional, para o mesmo problema que abordamos neste artigo. Outros métodos populacionais foram propostos, porém, considerando outras extensões do problema. [Kazemi et al., 2009], propuseram um método que utiliza um sistema com múltiplos agentes para o problema com múltiplas plantas e múltiplos produtos em uma estrutura de produção multi-estágio, porém, não consideram as decisões de roteamento dos veículos. [Abraham et al., 2015], apresentam um algoritmo genético para o problema considerando duas plantas com múltiplos itens e uma frota de veículos homogêneos. [Senoussi et al., 2018], consideraram uma fábrica e um produto para atender vários varejistas e, também, não consideram as decisões de roteamento, utilizando algoritmos genéticos híbridos combinados com estratégias de decomposição usando o CPLEX.

Embora muito promissoras, técnicas baseadas em algoritmos evolutivos foram pouco exploradas até o momento e, por esta razão, tornaram-se o foco deste artigo. Na Seção 2, definimos o problema e o modelo matemático utilizado. Na Seção 3, apresentamos os componentes do Algoritmo Memético proposto. Na Seção 4, os resultados computacionais obtidos e, por fim, na Seção 5, as conclusões do trabalho.

2. Definição do Problema e Modelo Matemático

O modelo proposto é uma adaptação dos trabalhos de [Armentano et al., 2011] e [Fumero e Vercellis, 1999]. Considerando uma fábrica com capacidade limitada de produção e armazenamento, produzindo um item que será distribuído a um conjunto de clientes para atender suas respectivas demandas em um horizonte finito de planejamento. O produto poderá ser armazenado na fábrica ou nos clientes com gerenciamento dos estoques feito pelo fornecedor (VMI, *Vendor Managed Inventory*). Além disso, adotamos as mesmas hipóteses da política de estoques de nível máximo (ML, *maximum level*), consideradas no trabalho original de [Fumero e Vercellis, 1999], permitindo que as quantidades entregues em cada cliente tenham qualquer valor positivo, desde que, o nível máximo de estoque não seja excedido. A distribuição do produto é realizada por uma frota de veículos homogênea com capacidade limitada, sendo considerados os custos fixos e variáveis pela utilização de cada veículo. Um cliente poderá ser visitado somente por um veículo em cada período e o veículo poderá realizar somente uma rota por período, partindo e voltando à fábrica ao final do

trajeto. O objetivo do problema é minimizar os custos de produção e preparação, custos de estoques na fábrica e nos clientes juntamente com os custos de roteamento dos veículos. A seguir, definimos a forma de representação, os parâmetros e as variáveis de decisão do modelo.

Considere um grafo completo $G = (W, E)$ que representa o problema, onde $W = \{0, 1, \ldots, N\}$ é um conjunto de nós representando a fábrica e os clientes e, $E = \{(i, k) : i, k \in W, i \neq k\}$ é o conjunto de arcos que são as rotas entre a fábrica e os clientes. A fábrica, representada pelo nó $i = 0$, gerencia os estoques dos clientes $i = 1, \ldots, N$ em um horizonte finito de planejamento, $t = 1, \ldots, T$. A entrega do produto será feita por uma frota limitada de veículos $v = 1, \ldots, V$ do fornecedor. Apresentamos a seguir um resumo desses parâmetros juntamente com as variáveis do modelo:

Parâmetros:

$B_t =$ Capacidade (tempo) de produção da fábrica no período t;
$b =$ Tempo de processamento do produto;
$c_t =$ Custo de produção do produto no período t;
$s_t =$ Custo de preparação do item p no período t;
$U_i =$ Capacidade máxima de estoque no local i;
$I_{i0} =$ Estoque do item no local i no período 0;
$h_{it} =$ Custo de estoque no cliente i no período t;
$C =$ Capacidade dos veículos;
$f =$ Custo fixo para utilização do veículo;
$a_{ik} =$ Custo de transporte para percorrer a aresta (i, k);
$d_{it} =$ Demanda do item no cliente i no período t.

Variáveis de decisão:

$$y_t = \begin{cases} 1, & \text{se houver produção no período } t; \\ 0, & \text{caso contrário.} \end{cases}$$

$x_t =$ Quantidade produzida no período t;
$I_{it} =$ Estoque no local i ao final do período t;

$$z_{vikt} = \begin{cases} 1, & \text{se o veículo } v \text{ percorre a aresta } (i, k) \text{ no período } t; \\ 0, & \text{caso contrário.} \end{cases}$$

$r_{vikt} =$ Quantidade transportada pelo veículo v na aresta (i, k) no período t;
$q_{vit} =$ Quantidade entregue pelo veículo v no cliente i no período t.

Assumimos que as quantidades produzidas no período t estão disponíveis para serem entregues neste mesmo período. Desse modo, temos o seguinte modelo para o PRP com apenas um produto:

Minimize

$$\sum_{i=0}^{N}\sum_{t=1}^{T} h_{it} I_{it} + \sum_{t=1}^{T}(s_t y_t + c_t x_t) + \sum_{v=1}^{V}\sum_{k=1}^{N}\sum_{t=1}^{T} f z_{v0kt} + \sum_{v=1}^{V}\sum_{\substack{i,k=0 \\ i\neq k}}^{N}\sum_{t=1}^{T} a_{ik} z_{vikt} \tag{1}$$

Sujeito a:

$$x_t + I_{0,t-1} - \sum_{v=1}^{V}\sum_{i=1}^{N} q_{vit} = I_{0t} \qquad 1 \le t \le T \tag{2}$$

$$\sum_{v=1}^{V} q_{vit} + I_{i,t-1} - d_{it} = I_{it} \qquad 1 \le i \le N \quad 1 \le t \le T \tag{3}$$

$$bx_t \le B_t y_t \qquad 1 \le t \le T \tag{4}$$

$$I_{it} \le U_i \qquad 0 \le i \le N \quad 1 \le t \le T \tag{5}$$

$$\sum_{\substack{i=0 \\ i\neq k}}^{N} r_{vikt} - \sum_{\substack{l=0 \\ l\neq k}}^{N} r_{vklt} = q_{vkt} \qquad 1 \le v \le V \quad 1 \le k \le N \quad 1 \le t \le T \tag{6}$$

$$\sum_{v=1}^{V}\sum_{k=1}^{N} r_{v0kt} - \sum_{v=1}^{V}\sum_{i=1}^{N} r_{vi0t} = \sum_{v=1}^{V}\sum_{l=1}^{N} q_{vlt} \qquad 1 \le t \le T \tag{7}$$

$$r_{vikt} \le C z_{vikt} \qquad 1 \le v \le V \quad 0 \le i \le N \quad 0 \le k \le N \quad 1 \le t \le T \quad i \neq k \tag{8}$$

$$\sum_{k=1}^{N} z_{v0kt} \le 1 \qquad 1 \le v \le V \quad 1 \le t \le T \tag{9}$$

$$\sum_{\substack{i=0 \\ i\neq k}}^{N} z_{vikt} - \sum_{\substack{l=0 \\ l\neq k}}^{N} z_{vklt} = 0 \qquad 1 \le v \le V \quad 0 \le k \le N \quad 1 \le t \le T \tag{10}$$

$$\sum_{v=1}^{V}\sum_{\substack{i=0 \\ i\neq k}}^{N} z_{vikt} \le 1 \qquad 1 \le k \le N \quad 1 \le t \le T \tag{11}$$

$$y_t, z_{vikt} \in \{0,1\} \quad 1 \le v \le V \quad 0 \le i \le N \quad 0 \le k \le N \quad 1 \le t \le T \quad i \neq k \tag{12}$$

$$x_t, I_{it}, r_{vikt}, q_{vit} \ge 0 \quad 1 \le v \le V \quad 0 \le i \le N \quad 0 \le k \le N \quad 1 \le t \le T \quad i \neq k \tag{13}$$

A função objetivo (1) minimiza os custos de produção, estoque e roteamento. As restrições (2) estabelecem o balanceamento de estoque na fábrica e as restrições (3), o balanceamento dos estoques nos clientes. As restrições (4) estabelecem a capacidade máxima de produção da fábrica em cada período e a relação entre as variáveis de produção e preparação. As restrições (5) delimitam a capacidade máxima de estoque na fábrica e nos clientes. As restrições (6) e (7) são as equações de conservação do fluxo de itens, assegurando o balanceamento em cada cliente, a coleta na fábrica e a eliminação de sub-rotas. As restrições (8) delimitam a capacidade máxima de carga dos veículos. As restrições (9) impõem no máximo uma rota para cada veículo por período. As restrições (10) asseguram que as rotas podem terminar somente na fábrica. As restrições (11) garantem que no máximo um veículo pode visitar um cliente em cada período. Por fim, as restrições (12) e (13) definem os tipos das variáveis.

3. Algoritmo Memético Proposto

Ao incorporar as estratégias de busca local nos Algoritmos Genéticos temos os chamados Algoritmos Meméticos ([Moscato e Norman, 1992]) ou ainda Algoritmos Genéticos Híbridos. Nesta seção, descrevemos os elementos básicos que definem o algoritmo proposto, assim como, alguns aspectos relevantes e necessários à sua implementação no contexto do PRP.

3.1. Representação da solução

Utilizamos uma representação completa da solução do problema, ou seja, um indivíduo armazenará as quantidades produzidas (x_t e y_t), os estoques (I_{it}) e as informações referentes as rotas percorridas pelos veículos (r_{vikt}, q_{vit} e z_{vikt}). Embora possam ser obtidas facilmente a partir das demais, optamos por manter, a princípio, as variáveis binárias na representação.

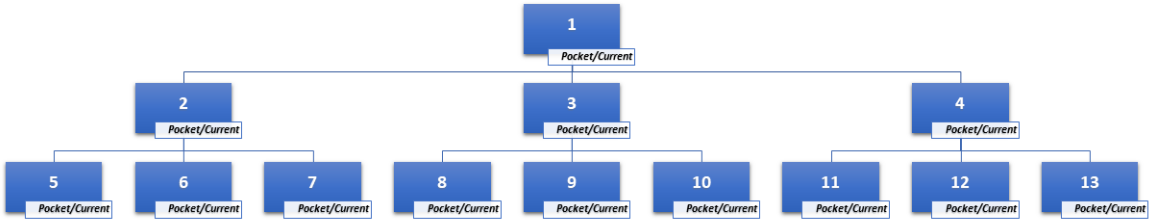
3.2. Função de avaliação

A função de avaliação armazena o custo e o grau de infactibilidade da solução que dependerá dos tipos de restrições que estão sendo violadas. Portanto, o valor de *fitness* será definido por um par ordenado, composto pelo custo e um número inteiro representando o grau de infactibilidade deste indivíduo. Obviamente, se o grau for igual a 0, então, a solução será factível.

3.3. Estrutura populacional

Adotamos uma população estruturada em árvore, proposta por [Berretta e Moscato, 1999], para organizar os indivíduos da população, conforme a Figura 1. A representação consiste em uma árvore ternária com três níveis e um total de 13 agentes. A população é organizada hierarquicamente por líderes e subordinados, classificados de acordo com o valor atribuído pela função de avaliação. Desse modo, temos uma estrutura ordenada de acordo com a qualidade das soluções que estes agentes representam para o problema. Analisando cada nível da árvore, podemos dividir a estrutura em 4 sub-populações. Assim, o agente 1 será o líder da sub-população que tem como subordinados os agentes 2, 3 e 4. O agente 2 será líder da sub-população que tem como subordinados os agentes 5, 6 e 7, e assim por diante. Cada agente armazena dois indivíduos, um denominado *pocket* e o outro denominado *current*, totalizando 26 indivíduos na população. O *pocket* terá sempre melhor valor do que o *current* e, quando isso não ocorre, os indivíduos são trocados de posição. O *pocket* de um líder possuirá melhor valor do que os seus subordinados. Sempre que modificações forem realizadas nos indivíduos um procedimento de atualização hierárquica será efetuado, garantindo assim, a verificação entre *pockets* e *currents* e, também, entre líderes e subordinados. Desse modo, teremos sempre armazenado, no agente 1, o melhor indivíduo da população.

Figura 1: Estrutura em árvore da população.



Fonte: Autoria própria.

3.4. População inicial

As soluções são construídas em duas etapas. A primeira consiste em determinar uma solução para o problema de roteamento dos veículos. Com as entregas definidas, o segundo procedimento determina um plano de produção para garantir o atendimento destas demandas assim como realizar o gerenciamento dos estoques. A ordem destas etapas pode ser invertida. Ao final, incorporando as decisões de roteamento com o plano de produção e os estoques, teremos uma solução para o problema integrado e, portanto, um indivíduo para a população inicial.

Utilizamos a heurística BFD (*Best Fit Decreasing*) para determinar uma solução para o problema de roteamento [Senoussi et al., 2018]. A heurística BFD resolve um problema de empacotamento binário, em cada período, para atribuir as cargas aos veículos de modo a atender a demanda dos clientes, mas desconsiderando as distâncias entre os clientes. Os pedidos dos clientes são carregados nos veículos, do maior para o menor, e respeitando-se as capacidades de carga. Se o número de veículos da frota for insuficiente para atender a demanda do período, então, um procedimento de ajuste será aplicado. Esse procedimento realiza tentativas de carregamento dos lotes remanescentes em períodos anteriores e priorizando veículos com maior capacidade ociosa. Implementamos também, mais duas versões desta técnica para garantir um número mínimo de indivíduos para compor a população inicial, mas mantendo a diversidade das soluções geradas. Na segunda versão, invertemos a ordem de carregamento dos pedidos nos veículos, ou seja, os pedidos serão carregados do menor para o maior pedido. Por fim, na terceira versão, a ordem de carregamento dos pedidos será aleatória, ou seja, obtido a partir do sorteio dos pedidos de cada período. Em todas as versões, o procedimento de ajuste, caso necessário, será o mesmo.

Implementamos também uma versão sequencial da heurística conhecida como Método das Economias. A principio, considera que existe um veículo atendendo cada cliente e voltando para a fábrica. Em seguida, determina a economia obtida com a inserção de outro cliente na rota ou mesmo da junção de duas rotas desde que satisfeitas as restrições de capacidade. Para o cálculo das economias em cada etapa, usamos as distâncias entre os clientes. O processo é continuado até que todos os clientes estejam inseridos em alguma rota e mais nenhuma economia seja possível. Caso o número de veículos não seja suficiente, novamente um procedimento de factibilização será necessário. Finalmente, para viabilizar a geração da população inicial, modificamos, aleatoriamente, os parâmetros para cálculo das economias em cada período de modo gerar número suficiente e diverso de soluções.

Na segunda etapa, utilizamos uma adaptação do algoritmo de [Wagner e Whitin, 1958], para determinar uma solução para o problema de dimensionamento de lotes. O algoritmo utiliza programação dinâmica para resolver de maneira ótima o problema para um único item. Se o plano de produção obtido violar as restrições de capacidade então um procedimento de ajuste deverá ser aplicado. Nesse procedimento, são realizadas tentativas de transferências de produção para períodos anteriores de modo a factibilizar as restrições violadas. Para garantir a geração não apenas de um, mas, de 26 indivíduos para compor a população inicial, introduzimos uma componente aleatória que modifica os custos de preparação dentro de limitantes estabelecidos e testados para garantir a diversidade das soluções obtidas.

Testes preliminares foram executados para avaliar tanto a diversidade quanto a qualidade das soluções obtidas pelas técnicas isoladamente ou por combinações das mesmas. A melhor combinação foi gerar metade da população usando primeiro o Wagner-Whitin e depois o método das economias e, a outra metade usando a heurística BFD com o Wagner-Whitin na sequencia. Cabe observar, ainda, que os procedimentos para ajuste das cargas e das restrições de capacidade podem falhar e, assim, teríamos a inserção de uma solução infactível na população inicial.

3.5. Mecanismos de seleção e substituição

A partir da estrutura populacional adotada, são selecionados apenas os indivíduos denominados de *pockets*. Em cada sub-população, o *pocket* do agente líder é selecionado para aplicação dos operadores de reprodução (cruzamento e mutação) com os indivíduos *pockets* dos agentes subordinados. Temos quatro sub-populações e, em cada sub-população, existem três subordinados. Portanto, serão realizados três cruzamentos em cada sub-população, gerando assim, doze novos indivíduos. A cada etapa deste processo denominamos de geração. A cada aplicação dos opera-

dores de reprodução será gerado um indivíduo filho que substituirá o indivíduo *current* do agente subordinado.

3.6. Estratégias de reprodução

Considerando os critérios de seleção definidos, denotamos o indivíduo pocket líder como sendo a mãe e o indivíduo pocket subordinado como pai para facilitar o entendimento dos operadores de reprodução propostos. Os operadores de cruzamento e mutação são descritos a seguir.

i) **crossXLF**: No operador de cruzamento de um ponto, o filho recebe o plano de produção da mãe e o roteamento do pai ou vice-versa. Essa escolha ocorre por meio de um sorteio. O filho será factível sempre que não houver ajustes no roteamento herdado dos pais, pois, durante o ajuste, podem ocorrer antecipações de entregas, o que poderia afetar o plano de produção original. De qualquer modo, se o filho for infactível, aplica-se um procedimento de factibilização. O operador de mutação permuta, conforme a taxa de mutação definida, dois clientes em uma rota escolhida ao acaso na solução filho.

ii) **crossOPR**: Operador de um ponto sobre o roteamento. O filho recebe todas as informações da mãe e um ponte de corte escolhido ao acaso entre $[T/2, T]$ que define quais informações serão herdadas do roteamento do pai. A opção pelos períodos finais tem por objetivo facilitar ajustes nas entregas em caso de infactibilidade. O operador de mutação não permitirá, para um período qualquer e, segundo uma taxa de probabilidade dentro do intervalo de tempo escolhido, a troca de informações do roteamento. Logo, para esse período escolhido, o filho permanecerá com as rotas da mãe.

iii) **crossOPP**: Operador de um ponto sobre a produção. Nesse caso, o filho receberá todas as informações da mãe. Um ponto de corte, selecionado ao acaso entre $[T/2, T]$, define quais informações do plano de produção do pai serão herdadas. Do mesmo modo, a escolha pelos períodos finais visa facilitar possíveis ajustes no plano de produção no caso de infactibilidade. Analogamente ao anterior, o operador de mutação não permitirá de acordo com uma taxa de probabilidade, para um período qualquer, a troca de informações e o filho permanece com as informações do plano de produção da mãe.

iv) **crossTWP**: Nesse operador de dois pontos, o filho herdará grande parte das informações da mãe e uma parte do plano de produção (crossOPP) e outra do roteamento (crossOPR) serão herdadas do pai. Logo, é equivalente dizer que, aplicam-se os dois operadores de um ponto definidos anteriormente, simultaneamente. O mesmo ocorre com o operador de mutação.

v) **crossOX1**: Adaptação do operador *Order crossover* (OX1) proposto para o problema do caixeiro viajante que utiliza uma estratégia baseada na ordem em que os clientes são visitados. O filho recebe o plano de produção da mãe e o roteamento é obtido pela aplicação do OX1. Para cada período, as rotas dos veículos são unidas em uma grande rota com múltiplos veículos. Em seguida, um segmento da rota mãe é copiado para o filho e o restante da rota é completado usando-se a ordem de visitas dos clientes presentes na rota do pai. Por fim, as rotas são distribuídas entre os veículos disponíveis a partir da rota ampliada e respeitando-se a capacidades dos veículos. Note que o número de veículos usados pode ser diferente das soluções pais e até mesmo infactível. Um ajuste poderá ser necessário para tentar corrigir esta infactibilidade.

A aplicação dos operadores pode provocar perda de factibilidade nos indivíduos, dependendo das informações que estão sendo alteradas. Dessa forma, procedimentos de factibilização efetuam deslocamentos das quantidades produzidas, estocadas ou entregues para períodos anteriores ou posteriores com o objetivo de reestabelecer as capacidades de produção, de estoque e de carga dos veículos. Embora sejam permitidas soluções infactíveis na população, estes procedimentos precisam ser aplicados para preservar minimamente a factibilidade da população.

Testes computacionais foram realizados com o intuito de ajustar os parâmetros dos operadores, tais como as taxas de cruzamento e de mutação. Testamos a eficiência dos operadores individualmente e combinados dentro do processo evolutivo. Os resultados obtidos indicaram melhor desempenho quando usamos os cinco operadores separadamente e sequencialmente ao longo das gerações.

3.7. Busca local

Cada novo indivíduo gerado representa uma nova solução factível, então, executa-se procedimentos com o objetivo de melhorar a qualidade desta solução. Quatro estratégias são utilizadas nesse trabalho e que, em sua maioria, são usadas em problemas de roteamento de veículos.

i) **Esvazia Veiculo**: Esvazia o veículo com maior capacidade de carga ociosa localizado a partir da segunda metade do horizonte de planejamento. A opção pela parte final do horizonte tem por objetivo facilitar possíveis remanejamentos dessa carga para períodos anteriores, ou seja, antecipar as entregas.

ii) **Antecipa Carga**: Tenta antecipar entregas de modo a garantir o atendimento da demanda. Sorteia-se uma rota localizada em um período da segunda metade do horizonte de planejamento e faz-se uma tentativa de transferir a carga do primeiro cliente desta rota para períodos anteriores. Se o movimento for bem sucedido e melhorar o custo total dessa solução, então uma nova tentativa será realizada.

iii) **2-Opt**: Consiste em eliminar duas arestas não adjacentes e reconectá-las de outro modo. Se houver melhoria na solução, a alteração é aceita e o processo se repete. As arestas para eliminação são escolhidas ao acaso, ou seja, não examinamos todas as opções para então escolher o melhor movimento. Ao menos uma tentativa é realizada para cada veículo em cada período.

iv) **3-Opt**: Nesse caso, a heurística consiste em eliminar três arestas não adjacentes e reconectá-las de um modo diferente. Nesse caso, existem $2^3 - 1$ possibilidades de religamento e a rota deve ter no mínimo seis nós. Novamente, pelo menos uma tentativa é realizada para cada veículo em cada período e o processo continua enquanto houver melhoria na solução.

Em resumo, no algoritmo proposto, inicialmente são gerados os vinte e seis indivíduos, conforme a Figura 1, que irão compor a população inicial, utilizando os procedimentos descritos. Caso uma solução seja infactível, um procedimento para factibilização é aplicado. Se o procedimento falhar, o indivíduo será inserido na população mesmo sendo infactível. Em seguida, a função de avaliação é aplicada em cada um dos indivíduos. Com os valores de *fitness* definidos, a população inicial é ordenada de acordo com a estrutura populacional. A partir do mecanismo de seleção, os operadores de cruzamento e mutação são aplicados. Os novos indivíduos gerados que são infactíveis passam novamente pelo processo de factibilização. Em seguida, a busca local é aplicada somente nos indivíduos factíveis com o objetivo de melhorar os descendentes que estão sendo inseridos na população. Depois de calculados os valores de *fitness*, a população é novamente organizada. Se o critério de parada for atingido, então, o algoritmo retornará a melhor solução encontrada que estará armazenada no indivíduo *pocket* do agente 1. Caso não, novos indivíduos são selecionados para aplicação dos operadores e o processo evolutivo continuará sendo repetido.

4. Experimentos Computacionais

Os algoritmos propostos foram implementados na linguagem Java, versão 14.0.2, com a utilização do solver comercial Gurobi, versão 9.1.1 e os testes executados em um computador Intel Core i7-10510U (1.8 GHz até 4.9 GHz, cache de 8MB, quad-core), 16Gb RAM DDR4 2666MHz, com sistema operacional Windows 10 Home. Com o objetivo de apurar a eficiência e a precisão do método proposto utilizamos as instâncias de pequeno e médio porte propostas por [Archetti et al., 2011], totalizando 960 instâncias conforme a Tabela 1. Na Tabela 2, temos as instâncias divididas em 4 classes com 120 problemas e diferentes níveis de complexidade associados aos custos de produção, estoques e transportes. Em relação aos dados originais, os estoques iniciais dos clientes, quando positivos, foram zerados utilizando-se as demandas iniciais.

Com o objetivo de determinar a melhor configuração de parâmetros para obter o máximo desempenho do *branch-and-cut* utilizado pelo Gurobi, empreendemos um grande esforço computacional para testar diversas possibilidades em um subconjunto com 96 instâncias (primeira semente de cada classe). Destacamos alguns desses parâmetros, como, por exemplo, *MIPFocus*, *Heuristics*,

Tabela 1: Características das instâncias de [Archetti et al., 2011].

Instâncias	A1	A2
Número de Instâncias	480	480
Número de Períodos	6	6
Número de Clientes	14	50
Número de Veículos	1	Ilimitado
Demanda	Constante	Constante
Capacidade de Produção	Ilimitada	Ilimitada
Capacidade de Estoque: Fábrica	Ilimitada	Ilimitada
Capacidade de Estoque: Clientes	Constante	Constante
Estoque Inicial: Fábrica e Clientes	0	0
Capacidade dos Veículos	Constante	Constante

Fonte: Adaptado de [Adulyasak et al., 2014b].

Tabela 2: Classificação das classes de instâncias de [Archetti et al., 2011].

Classe	Número	# Sementes	Total	Descrição
Classe I	1-24	5	120	Instâncias padrão
Classe II	25-48	5	120	Custos de Produção: alto (Classe I x 10)
Classe III	49-72	5	120	Custos de Transporte: alto (Classe I x 5)
Classe IV	73-96	5	120	Custos de Estoques dos Clientes: zerados

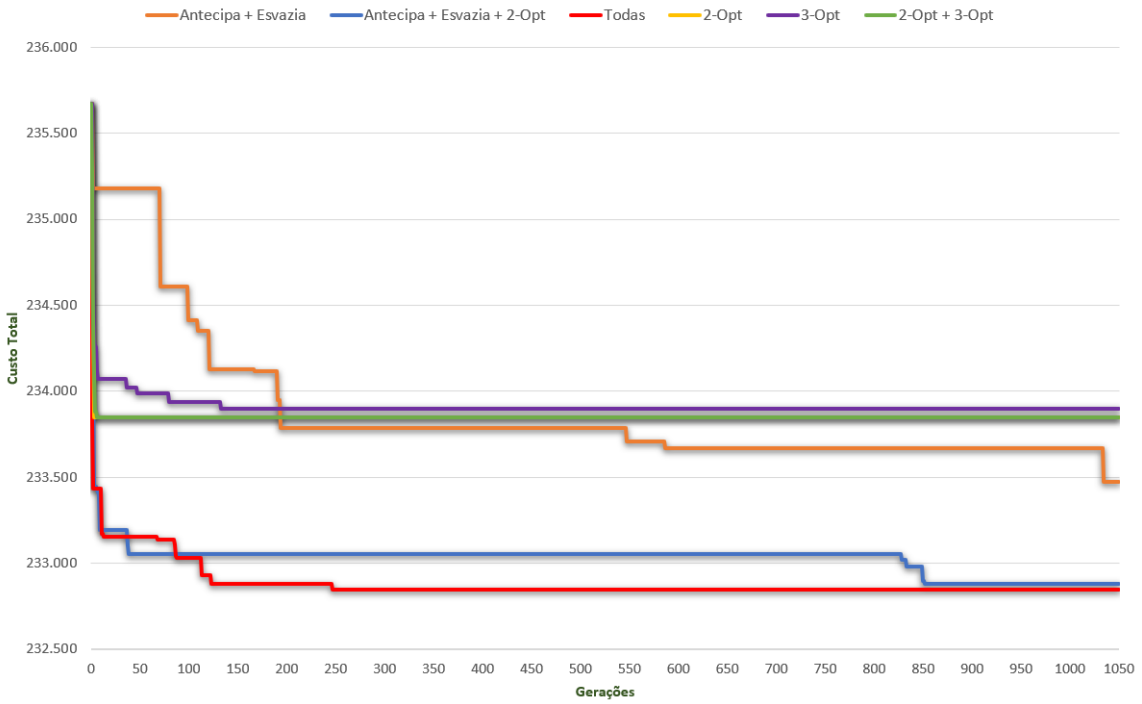
Fonte: Adaptado de [Adulyasak et al., 2014b].

Cuts, *PreSparsify* e *VarBranch*, perfazendo mais de 600 horas de testes computacionais. Em especial, testamos uma ferramenta disponibilizada pelo Gurobi que é o ajuste automático de parâmetros. Por conta do alto custo computacional, foi possível aplicar essa técnica somente para 4 instâncias, mais precisamente a primeira de cada classe. Definidos os parâmetros dessa instância estendemos esse ajuste para o restante de cada classe. Embora os resultados desse ajuste automático tenham sido excelentes, ainda são extremamente custosos do ponto de vista computacional.

Considerando o elevado número de parâmetros do MA, diversos testes foram necessários para encontrar a melhor configuração. Na Figura 2, podemos observar o desempenho, para uma instância com 14 clientes, das buscas locais utilizadas. Nesse exemplo, somente a combinação usando todas as buscas encontrou a solução ótima após 247 gerações. De maneira geral, nos testes de parametrização realizados em um conjunto com 96 instâncias a combinação com o melhor desempenho foi obtida com a aplicação de todas as buscas na mesma ordem mostrada no gráfico. Outro aspecto examinado refere-se a intensificação da busca local e diversos limitantes foram testados (0 a 100 iterações). As configurações com 25 repetições das buscas locais apresentaram melhor desempenho geral. Para o conjunto com 50 clientes as buscas elevam sobremaneira o esforço computacional do algoritmo, por isso, testamos limitar em apenas uma iteração, a aplicação de cada busca, independente de obter ou não melhoria da solução. Essa estratégia alcançou melhores resultados quando combinada com a intensificação.

Podemos observar que o Gurobi obteve excelente desempenho para as instâncias de pequeno porte, alcançando a solução ótima em 88% das instâncias. Para essas instâncias utilizamos a

Figura 2: Evolução da incumbente para diferentes buscas locais.



Fonte: Autoria própria.

configuração com foco em provar a otimalidade da solução (*MIPFocus* = 2) e o tempo limite de 300 segundos. Em média, o Gurobi precisou de um minuto e meio para determinar uma solução muito próxima da ótima, conforme dados na Tabela 3. Como era esperado, o MA encontrou dificuldades para obter a solução ótima em 5 minutos, mas o desempenho foi muito bom para as classes C2 e C4, conforme Figura 3. Na classe C1, obteve desempenho satisfatório, com GAP médio de 3%. O pior desempenho do MA nesse conjunto foi na classe C3, com altos custos de transporte associados, ou seja, para os problemas com maiores custos de transporte e, portanto, mais difíceis do ponto de vista do problema de roteamento dos veículos.

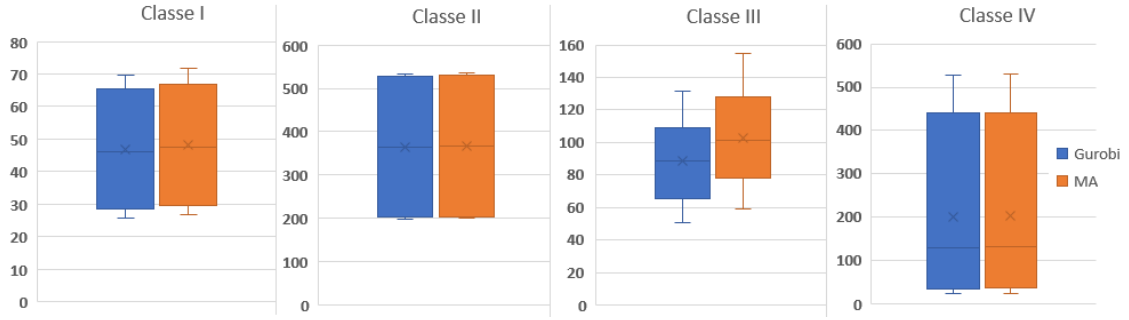
Tabela 3: Tabela geral de resultados (GAP por Classes).

Classes de Instâncias	14 Clientes								50 Clientes							
	Gurobi				MA				Gurobi				MA			
	Tempo (s)	# Soluções Ótimas	GAP Médio	GAP Mediano	Tempo (s)	# Soluções Ótimas	GAP Médio	GAP Mediano	Tempo (s)	# Soluções Factíveis	GAP Médio	GAP Mediano	# Soluções Factíveis	GAP Médio	GAP Mediano	
C1	102	101	0,09%	0,00%	300	1	2,98%	3,02%	300	56	32,72%	32,83%	120	17,39%	15,95%	
C2	52	113	0,01%	0,01%	300	14	0,49%	0,35%	300	60	8,29%	7,65%	120	4,70%	4,75%	
C3	111	100	0,53%	0,18%	300	0	14,17%	14,17%	300	63	47,88%	46,87%	120	31,41%	30,75%	
C4	79	107	0,07%	0,00%	300	11	2,18%	1,19%	300	83	14,52%	11,51%	120	12,83%	11,05%	
Totais	**86**	**421**	**0,18%**	**0,01%**	**300**	**26**	**4,96%**	**2,11%**	**300**	**262**	**25,85%**	**22,17%**	**480**	**16,58%**	**13,50%**	

Fonte: Autoria própria.

No segundo grupo com 50 clientes, o Gurobi encontrou dificuldades até para obter soluções factíveis (55% do total apenas). Utilizamos nesse conjunto a configuração com foco em encontrar soluções factíveis (*MIPFocus* = 1) juntamente com uma estratégia para tentar reduzir o número de valores não nulos na matriz de restrições (*PreSparsify* = 1), além do tempo limite de 300 segundos. Por outro lado, o MA atingiu ótimos resultados para a classe C2, com instâncias vinculadas a custos

Figura 3: Custos totais médios (em milhares) para instâncias com 14 clientes (300 segundos).



Fonte: Autoria própria.

de produção mais elevados, com GAP inferiores a 5%, enquanto o Gurobi ficou acima de 8%. Embora para as classes C1 e C3 os GAP obtidos pelo MA sejam mais elevados, ficaram muito abaixo dos valores encontrados pelo Gurobi. Para a classe C4, que não considera custos de estoques nos clientes, as duas técnicas alcançaram resultados mais próximos, mas com vantagem para o MA.

5. Conclusões Finais

Podemos concluir que as estratégias utilizadas nesse artigo apresentaram bons resultados para o Problema de Roteamento e Produção. Para as instâncias de pequeno porte, o Gurobi quando executado com a configuração adequada de parâmetros, resolveu muito rapidamente e de maneira ótima a grande maioria das instâncias. Por outro lado, para as instâncias de médio porte apresentou dificuldades inclusive do ponto de vista de factibilidade das soluções encontradas. Em relação ao Algoritmo Memético proposto, os resultados indicam tratar-se de uma técnica eficiente e robusta, especialmente pelo seu desempenho nas instâncias de médio porte. A partir de alguns testes realizados, percebemos que o desempenho melhora consideravelmente se usarmos limites de tempo superiores aos 5 minutos adotados nesse trabalho. Apesar disso, alguns ajustes visando otimizar o desempenho computacional serão necessários para ampliar sua eficácia em instâncias de grande porte, assim como, em problemas com múltiplos produtos e múltiplas fábricas.

Referências

Abraham, A., Kumar, K. R., Sridharan, R., e Singh, D. (2015). A Genetic Algorithm Approach for Integrated Production and Distribution Problem. *Procedia - Social and Behavioral Sciences*, 189:184–192.

Absi, N., Archetti, C., Dauzère-Pérès, S., Feillet, D., e Speranza, M. G. (2018). Comparing sequential and integrated approaches for the production routing problem. *European Journal of Operational Research*, 269(2):633–646.

Adulyasak, Y., Cordeau, J. F., e Jans, R. (2014a). Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing*, 26 (1):103–120.

Adulyasak, Y., Cordeau, J.-F., e Jans, R. (2014b). Optimization-Based Adaptive Large Neighborhood Search for the Production Routing Problem. *Transportation Science*, 48(1):20–45.

Adulyasak, Y., Cordeau, J. F., e Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms. *Computers and Operations Research*, 55:141–152.

Archetti, C., Bertazzi, L., Paletta, G., e Speranza, G. (2011). Analysis of the maximum level policy in a production-distribution system. *Computers and Operations Research*, 38(12):1731–46.

Armentano, V. A., Shiguemoto, A. L., e Løkketangen, A. (2011). Tabu search with path relinking for an integrated productiondistribution problem. *Computers and Operations Research*, 38(8): 1199–1209.

Berretta, R. e Moscato, P. (1999). The Number Partitioning Problem: An Open Challenge for Evolutionary Computational? In Corne, D.; Dorigo, M.; Glover, F.; Dasgupta, D.; Moscato, P.; Poli, R.; Price, K. V., editor, *New Ideas in Optimisation*, chapter 14, p. 261–278. McGraw-Hill.

Boudia, M. e Prins, C. (2009). A memetic algorithm with dynamic population management for an integrated production-distribution problem. *European Journal of Operational Research*, 195(3): 703–715.

Darvish, M., Archetti, C., e Coelho, L. C. (2019). Trade-offs between environmental and economic performance in production and inventory-routing problems. *International Journal of Production Economics*, 217(July 2017):269–280.

Fumero, F. e Vercellis, C. (1999). Synchronized Development of Production, Inventory, and Distribution Schedules. *Transportation Science*, 33(3):330–340.

Kazemi, A., Fazel Zarandi, M. H., e Moattar Husseini, S. M. (2009). A multi-agent system to solve the production-distribution planning problem for a supply chain: A genetic algorithm approach. *International Journal of Advanced Manufacturing Technology*, 44(1-2):180–193.

Moscato, P. e Norman, M. G. (1992). A Memetic Approach for the Traveling Salesman Problem Implementation of a Computational Ecology for Combinatorial Optimization on Message-Passing Systems. *International Conference on Parallel Computing and Transputer Applications*, (January):177–186.

Ramos, B., Alves, C., e Valério de Carvalho, J. (2020). An arc flow formulation to the multi-trip production, inventory, distribution, and routing problem with time windows. *International Transactions in Operational Research*, 00:1–28.

Roostaei, A. e Kamal, I. N. (2018). Considering Production Planning in the Multi-period Inventory Routing Problem with Transshipment between Retailers. *International Journal of Engineering*, 31(9):1568–1574.

Senoussi, A., Dauzère-Pérès, S., Brahimi, N., Penz, B., e Kinza Mouss, N. (2018). Heuristics Based on Genetic Algorithms for the Capacitated Multi Vehicle Production Distribution Problem. *Computers and Operations Research*, 96:108–119.

Solyalı, O. e Süral, H. (2017). A multi-phase heuristic for the production routing problem. *Computers and Operations Research*, 87:114–124.

Wagner, H. M. e Whitin, T. M. (1958). Dynamic Version of the Economic Lot Size Model. *Management Science*, 5(1):89–96.