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

Um algoritmo *branch-price-and-cut* para o problema de roteamento de veículos com múltiplos entregadores e demanda incerta

Jonathan De La Vega, Pedro Munari, Reinaldo Morabito
Departamento de Engenharia de Produção, Universidade Federal de São Carlos
Rodovia Washington Luís, s/n, São Carlos - SP, 13565-905
jdlvmartinez@gmail.com, munari@dep.ufscar.br,
morabito@dep.ufscar.br

RESUMO

Neste artigo, abordado-se o problema de roteamento de veículos com janelas de tempo, múltiplos entregadores e demanda incerta. Além das decisões típicas de sequenciamento e programação, esse problema envolve também a decisão de alocação de entregadores a rotas, de modo a reduzir custos e tempos de serviço. As incertezas da demanda são tratadas via otimização robusta estática, em que apenas decisões aqui-e-agora são definidas. Uma formulação baseada em particionamento de conjuntos é proposta, cuja abordagem de solução se dá por um método exato do tipo *branch-price-and-cut*. Experimentos computacionais foram realizados com instâncias da literatura, visando avaliar o desempenho do método *branch-price-and-cut* em relação a abordagens de solução propostas em trabalhos prévios. Os resultados revelam que o algoritmo proposto supera claramente tanto em tempo computacional quanto em qualidade da solução as abordagens existentes no estado da arte do problema.

PALAVRAS CHAVE. Roteamento de veículos. Otimização robusta. Branch-price-and-cut.

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

ABSTRACT

In this paper, we address the vehicle routing problem with time windows, multiple deliverymen and uncertain demand. In addition to the typical sequencing and scheduling decisions, this problem also involves the allocation of deliverymen to routes in order to reduce costs and service times. Demand uncertainties are addressed via static robust optimization where only here-and-now decisions are defined. We propose a formulation based on set partitioning, which is solved using an exact method of branch-price-and-cut type. We perform computational experiments with instances from the literature to evaluate the performance of the branch-price-and-cut method with respect to approaches proposed in previous works. The results indicate that the proposed method clearly outperforms in both computational time and solution quality the existing approaches in the state-of-the-art of the problem.

KEYWORDS. Vehicle routing. Robust optimization. Branch-price-and-cut.

L&T - Logistic and Transportation. PM - Mathematical Programming

1. Introdução

Neste artigo, estuda-se o problema de roteamento de veículos com janelas de tempo e múltiplos entregadores (abrev. em Inglês VRPTWMD - *Vehicle Routing Problem with Time Windows and Multiple Deliverymen*). Esse problema surge em diferentes aplicações envolvendo a coleta e distribuição de produtos em configurações logísticas em que os tempos de serviços no local dos clientes são longos, comparados com os tempos de viagens, e dependem do número de entregadores no veículo. Essa variante é relevante, particularmente, nos casos em que os clientes devem ser servidos num mesmo dia, em áreas urbanas congestionadas, e não servi-los dentro de suas janelas de tempo é altamente indesejável [Pureza et al., 2012; Senarclens de Grancy e Reimann, 2016; Álvarez e Munari, 2017]. Nesse contexto, o VRPTWMD, além das decisões típicas dos problemas de roteamento, também permite atribuir a cada rota um número de entregadores para reduzir os tempos de serviço e minimizar os custos em relação ao número de veículos usados, número de entregadores totais alocados nas rotas da solução e distância [Munari e Morabito, 2018].

Em diversas situações práticas, as realizações da demanda dos clientes não são conhecidas no mesmo momento em que as decisões precisam ser tomadas. Uma prática comum nessas situações é usar os respectivos valores esperados e/ou nominais das demandas como suas realizações [Gendreau et al., 2016]. No entanto, variações da demanda dos clientes em torno de seu valor nominal podem fazer com que a solução obtida se torne infactível quando implementada. Nesse caso, faz-se necessário implementar ações de recurso como alternativa para recuperar a factibilidade da solução. Dependendo do montante no custo após a aplicação das ações de recurso, pode-se estar aumentando em demasia o custo operacional total. Por essa razão, neste artigo, usa-se a abordagem de otimização robusta (abrev. em Inglês RO - *Robust Optimization*) estática que busca fornecer soluções que sejam imunes ou pouco sensíveis às possíveis variações dos parâmetros incertos.

RO é uma técnica de programação matemática para modelar e resolver problemas de otimização com parâmetros incertos, a qual surge como resposta a contornar as dificuldades e/ou desvantagens envolvidas na programação estocástica (abrev. em Inglês SP - *Stochastic Programming*) [Ben-Tal et al., 2012; Bertsimas et al., 2011; Bertsimas e Goyal, 2012; Bertsimas et al., 2015]. Na SP, basicamente, duas possíveis dificuldades podem ser notadas. A primeira dificuldade reside em que as distribuições de probabilidade dos dados incertos devem ser conhecidas *a priori*. Em muitos casos, pode ser muito difícil, ou mesmo impossível, estimar distribuições de probabilidade bastante precisas dos dados. Mesmo quando há alguma informação sobre essa distribuição, a enumeração e a quantificação das realizações para capturar essas distribuições é raramente satisfeita na prática. A segunda dificuldade é que as abordagens baseadas em SP são tipicamente afetados pelo curso da dimensionalidade, impactando seu tratabilidade computacional [Gounaris et al., 2016].

Na RO, a suposição do conhecimento das distribuições de probabilidade dos parâmetros aleatórios é relaxada. Basta que esses parâmetros sejam representados como variáveis aleatórias limitadas, simétricas e, em alguns casos, independentes, cujas possíveis realizações estão contidas em um conjunto que, em geral, é chamado de conjunto de incerteza [Bertsimas e Sim, 2003; Sniedovich, 2012]. Nesse contexto, RO busca selecionar a melhor solução aqui-e-agora entre aquelas imunizadas contra os dados incertos, i.e., soluções candidatas que são factíveis para todas as possíveis realizações dos dados dentro do conjunto incerto. Neste trabalho, optou-se pelo uso do conjunto de incerteza *budgeted* introduzido em Bertsimas e Sim [2004], pois o modelo robusto resultante mantém-se como um problema de otimização linear inteira, bem como porque esse conjunto poliedral facilita lidar com o *trade-off* custo-risco.

Até o momento, o único trabalho estudando o VRPTWMD em contextos incertos corres-

ponde a De La Vega et al. [2017]. Os autores apresentaram uma nova formulação determinística do VRPTWMD, a qual foi usada como ponto de partida na determinação da formulação robusta. Também, propuseram uma abordagem de solução para resolver o programa robusto. Basicamente, a abordagem corresponde a uma tentativa de melhorar a convergência de um *solver* comercial de otimização de propósito geral por meio da inclusão de uma solução factível robusta inicial, a qual é obtida por uma adaptação da heurística I1 de Solomon [1987]. Experimentos numéricos revelaram que a abordagem proposta consegue superar claramente o uso do *solver* sem o fornecimento a solução inicial, porém não é capaz de provar a otimilidade das soluções para a maioria das instâncias consideradas (com 25 clientes).

Neste artigo, apresentam-se uma nova formulação de otimização robusta e um método exato customizado para lidar com o VRPTWMD. Não se tem conhecimento de outros trabalhos que exploram otimização robusta no VRPTWMD por meio do método *branch-price-and-cut* (BPC). O restante deste artigo está organizado como segue. A Seção 2 introduz o problema na versão determinística. A versão robusta do problema é descrita na Seção 3. O algoritmo BPC proposto é apresentado na Seção 4 e os experimentos computacionais são reportados na Seção 5. Finalmente, as considerações finais são descritas na Seção 6.

2. Definição do problema

No VRPTWMD, assume-se que n clientes requerem serviço de entrega ou de coleta, mas não ambos os serviços ao mesmo tempo. O VRPTWMD pode ser definido por meio de um grafo $\mathscr{G} = (\mathscr{V}, \mathscr{A})$, em que $\mathscr{V} = \{0, 1, \ldots, n, n+1\}$ e $\mathscr{A} = \{(i,j) : i,j \in \mathscr{V}, i \neq j\}$ representam, respectivamente, os conjuntos de nós e arcos do grafo. Os nós 0 e $n+1$ do conjunto $\mathscr{V}$ representam o depósito, e os nós restantes definem os clientes, sendo estes indexados por meio do conjunto $\mathscr{N} = \mathscr{V} \setminus \{0, n+1\}$. Adicionalmente, o escalar $\mathcal{L}$ define o número máximo permitido de entregadores que podem ser atribuídos a um único veículo ($\ell = 1, \ldots, \mathcal{L}$). Na prática, em geral, $\mathcal{L}$ é pequeno devido a limitações no tamanho da cabine do veículo. Se o número de entregadores atribuídos a um veículo for ℓ, diz-se que esse veículo opera no modo ℓ.

Para cada cliente i, atribui-se uma demanda nominal não negativa $\bar{q}_i$ e um tempo de serviço nominal não-negativo $\bar{s}_i^\ell$ dependente do número de entregadores ou modo ℓ. A cada nó i é dada uma janela de tempo $[w_i^a, w_i^b]$ que descreve o intervalo de tempo em que o nó i se encontra disponível para receber o início do serviço. Também, associam-se para cada arco (i,j) uma distância d_{ij} e um tempo de viagem nominal $\bar{t}_{ij}$ satisfazendo a desigualdade triangular.

Uma frota de $\mathcal{K}$ veículos idênticos encontra-se disponível no depósito, sendo estes indexados pelo conjunto $\mathscr{K}$, tal que $|\mathscr{K}| = \mathcal{K}$. Além dos $\mathcal{K}$ veículos no depósito, também há disponível um número máximo de entregadores, denotado por $\mathcal{D}$. O VRPTWMD consiste em desenhar um conjunto $\mathscr{R} = \{\mathscr{R}_1, \ldots, \mathscr{R}_{\mathcal{K}'}\}$, com $\mathcal{K}' \leq \mathcal{K}$, de rotas de custo mínimo com seu correspondente número de entregadores tal que: i) cada rota inicie e finalize no depósito; ii) cada cliente seja visitado uma única vez, e o início do serviço deve respeitar a sua janela de tempo; iii) o número de entregadores a bordo de um veículo não deve exceder a quantidade máxima permitida; iv) a demanda total de uma rota seguida por um veículo não deve exceder a capacidade Q deste e iv) o número total de veículos e entregadores usados não devem exceder as quantidades disponíveis no depósito.

A seguir, apresentam-se a forma como a demanda incerta dos clientes é modelada, bem como o conjunto de incerteza usado para encapsular suas possíveis realizações. Adicionalmente, uma expressão recursiva para avaliar a robustez de uma solução também é apresentada.

3. Parâmetro incerto e conjunto de incerteza

Neste estudo, a demanda $\tilde{q}_i$ é modelada como uma variável aleatória limitada, simétrica e independente, variando no intervalo $[\bar{q}_i - \hat{q}_i, \bar{q}_i + \hat{q}_i]$. O parâmetro $\hat{q}_i$ é o desvio máximo permitido de

seu correspondente valor nominal $\bar{q}_i$. Associa-se a cada variável aleatória outra variável aleatória ξ_i (chamada de variável aleatória primitiva), assumindo valores entre -1 e 1. Por essa razão, a variável aleatória original pode ser escrita em termos da primitiva como: $\tilde{q}_i = \bar{q}_i + \hat{q}_i\xi_i$. O conjunto de incerteza é do tipo *budgeted* [Bertsimas e Sim, 2004; Bertsimas et al., 2011], o qual é construído para as variáveis incertas do vetor aleatório primitivo $\boldsymbol{\xi}$ e simplesmente restringe o número de componentes desse vetor aleatório que podem variar. O conjunto de incerteza *budgeted* pode ser escrito da seguinte forma:

$$\mathcal{U}^q = \left\{ \boldsymbol{\xi} \in \mathbb{R}^{|\mathcal{V}|} : -1 \leq \xi_i \leq 1,\ i \in \mathcal{V};\ \sum_{i \in \mathcal{V}} \xi_i \leq \Gamma^q \right\}. \tag{1}$$

O conjunto $\mathcal{U}^q$ depende do parâmetro Γ^q, definindo o *budget* de incerteza para o recurso carga. Por simplicidade, considera-se que esse parâmetro é um inteiro não-negativo. Quanto maior for o valor atribuído a Γ^q, maior é o número de parâmetros incertos representados nas componentes do vetor $\boldsymbol{\xi}$ permitidos a variar e, portanto, maior é o conservadorismo da abordagem robusta. Dois casos precisam ser ressaltados aqui: i) Quando o mínimo valor (igual a 0) é atribuído ao parâmetro Γ^q, obtém-se o problema nominal/determinístico em que as decisões são definidas otimamente sob a premissa de que a realização dos parâmetros incertos será exatamente seu valor nominal/esperado, levando a soluções com altos níveis de risco; ii) Quando o máximo valor é atribuído ao parâmetro Γ^q (isto é, método de Soyster), as decisões correspondem a resolver uma instância do problema na sua versão determinística, assumindo que os parâmetros incertos atingem seu valor de pior caso. Assim, obtêm-se soluções totalmente conservadoras com níveis de risco nulos.

Note que, em ambos os casos, o VRPTWMD robusto (abrev. em Inglês RVRPTWMD - *Robust VRPTWMD*) resulta em uma instância desse mesmo problema, porém na sua versão determinística. No entanto, é pouco provável encontrar ambientes práticos onde esses dois casos mencionados ocorram. Por essa razão, deve-se atribuir valores adequados ao parâmetro Γ^q de modo a obter soluções com custos piores do que o custo ótimo do problema nominal/determinístico, porém com níveis de risco mais aceitáveis.

O conjunto de incerteza $\mathcal{U}^q$ apresenta uma estrutura especial que pode ser bem aproveitada para construir expressões para avaliar a robustez recursivamente de uma dada solução (veja Lee et al. [2012]; Agra et al. [2013]; Munari et al. [2018]; De La Vega et al. [2019]). Seja $\mathcal{R}$ o conjunto de rotas em uma dada solução, com seus respectivos modos de operação. Cada rota $r \in \mathcal{R}$, a qual visita $k+1$ nós e opera no modo ℓ, é representada por meio do vetor $(v_1, v_2, \ldots, v_k, v_{k+1})$, com $v_1 = 0$ e $v_{k+1} = n+1$. Definindo $u_{v_i\gamma}$ como a carga máxima no veículo saindo do cliente v_i, dado que a demanda de γ clientes, com $\gamma = 0, 1, \ldots, \Gamma^q$, na rota parcial $r_{v_i} = (v_1, v_2, \ldots, v_i)$ assume o valor de pior-caso, as equações de recursividade para a carga podem ser escritas como:

$$u_{v_{i+1}\gamma} = \begin{cases} u_{v_i\gamma} + \bar{q}_{v_{i+1}}, & \text{se } \gamma = 0, \\ \max\{u_{v_i\gamma} + \bar{q}_{v_{i+1}}, u_{v_i\gamma-1} + \bar{q}_{v_{i+1}} + \hat{q}_{v_{i+1}}\}, & \text{caso contrário,} \end{cases} \tag{2}$$

em que $u_{v_1\gamma} = 0$ para $\gamma = 0, 1, \ldots, \Gamma^q$. Observe que para $\gamma = 0$, têm-se o consumo determinístico da carga do veículo saindo do cliente v_{i+1}. Caso contrário, a equação (2) avalia o valor que assume a demanda do cliente v_{i+1} no cenário de pior-caso. Para que uma rota seja *factível robusta* em relação à capacidade do veículo, ela deve satisfazer $u_{v_{i+1}\gamma} \leq \mathcal{Q}$ para todo $\gamma = 0, \ldots, \Gamma^q$ e $i = 1, \ldots, k+1$.

4. Fomulação de particionamento de conjuntos

A formulação de particionamento de conjuntos (abrev. em Inglês SPF - *Set Partitioning Formulation*) para o RVRPTWMD envolve variáveis de decisão que descrevem rotas operando em

algum modo ℓ. Assim, define-se a variável de decisão binária λ_r^ℓ que assume o valor 1 se e somente se a rota r operando no modo ℓ for selecionada para visitar pelo menos um cliente. De acordo com a definição da variável de decisão λ_r^ℓ, faz-se necessário conhecer todas as rotas factíveis robustas r que operam no modo ℓ, pois as incertezas na demanda dos clientes são contempladas. Seja $\mathcal{R}^\ell$, para $\ell = 1, \ldots, \mathcal{L}$, o conjunto de rotas factíveis robustas operando no modo ℓ. Cada rota $r \in \mathcal{R}^\ell$ é representada como antes. Seja a_{ir}^ℓ o parâmetro que indica se o cliente i faz parte da rota r operando no modo ℓ. Então, a formulação matemática baseada na SPF que define as rotas factíveis robustas de custo mínimo para atender todos os clientes pode ser escrita como segue:

minimizar

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \mathcal{R}^\ell} c_r^\ell \lambda_r^\ell \tag{3}$$

sujeito a:

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \mathcal{R}^\ell} a_{ri}^\ell \lambda_r^\ell = 1,\ i \in \mathcal{N}. \tag{4}$$

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \mathcal{R}^\ell} \ell \lambda_r^\ell \leq \mathcal{D}, \tag{5}$$

$$\lambda_r^\ell \in \{0,1\},\ \ell = 1, \ldots, \mathcal{L},\ r \in \mathcal{R}^\ell. \tag{6}$$

A função objetivo (3) minimiza os custos associados às rotas selecionadas. O parâmetro c_r^ℓ exprime o custo de usar a rota r operando no modo ℓ. Esse custo pode ser determinado como:

$$c_r^\ell = p_1 + p_2 l + p_3 \sum_{i=0}^{k} d_{v_i, v_{i+1}}, \tag{7}$$

em que p_1 descreve o custo de alocar um veículo para percorrer a rota, p_2 o custo de alocar um entregador no veículo e p_3 o custo unitário de transporte associado à distância percorrida. As restrições (4) expressam que, das rotas selecionadas, o cliente i deve fazer parte de exatamente uma delas operando em um único modo. As restrições (5) garantem que o número total de entregadores usados não exceda a quantidade máxima disponível no depósito. Finalmente, as restrições (6) impõem que as variáveis de decisão da formulação são de natureza binária.

Uma desvantagem da formulação (3)-(6) reside no número exponencial de rotas em cada um dos conjuntos $\mathcal{R}^\ell$, para cada $\ell = 1, \ldots, \mathcal{L}$. Adicionalmente, as rotas em cada um desses conjuntos não são conhecidas *a priori* e qualquer estratégia para determiná-las pode ser tão difícil como resolver o problema de otimização. Por essa razão, recorre-se a um algoritmo do tipo BPC para resolver a SPF (3)-(6).

5. Algoritmo *branch-price-and-cut* para o RVRPTWMD

Nesta seção, descrevem-se as principais componentes do algoritmo BPC proposto para resolver o RVRPTWMD. Entre as componentes, encontram-se a técnica de geração de colunas, algoritmo *labeling* robusto, estratégia de planos de corte e as regras de ramificação. A seguir, cada uma dessas componentes é detalhada.

5.1. Geração de colunas

A técnica de geração de colunas (abrev. em Inglês CG - *Column Generation*) trabalha sobre uma relaxação da SPF (3)-(6), denominada de problema mestre restrito (abrev. em Inglês RMP - *Restricted Master Problem*). O RMP considera o conjunto $\tilde{\mathcal{R}}^\ell \subset \mathcal{R}^\ell$ que contém um subconjunto das rotas em $\mathcal{R}^\ell$ e desconsidera o requerimento de integralidade das variáveis de decisão λ_r^ℓ. Dados os conjuntos $\tilde{\mathcal{R}}^\ell$, para todo $\ell = 1, \ldots, \mathcal{L}$, o RMP pode ser escrito como:

minimizar

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \tilde{\mathcal{R}}^\ell} c_r^\ell \lambda_r^\ell \tag{8}$$

sujeito a:

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \tilde{\mathcal{R}}^\ell} a_{ri}^\ell \lambda_r^\ell = 1,\ i \in \mathcal{N}. \tag{9}$$

$$\sum_{\ell=1}^{\mathcal{L}} \sum_{r \in \tilde{\mathcal{R}}^\ell} \ell \lambda_r^\ell \leq \mathcal{D}, \tag{10}$$

$$\lambda_r^\ell \geq 0,\ \ell = 1, \ldots, \mathcal{L},\ r \in \tilde{\mathcal{R}}^\ell \tag{11}$$

As rotas iniciais dos conjuntos $\tilde{\mathcal{R}}^\ell$ podem ser determinadas por meio de procedimentos heurísticos, ou considerando apenas rotas triviais, isto é, uma rota para cada cliente. A técnica de CG pode ser vista como uma extensão do método *simplex* para resolver problemas de otimização com apenas variáveis contínuas. Para uma dada iteração do algoritmo de CG, as variáveis de decisão do RMP que assumem valores estritamente positivos ($\lambda_r^\ell > 0$) formam uma base. Para determinar se a base atual é ótima, recorre-se ao subproblema *pricing*, o qual determina se existem variáveis ou rotas em $\mathcal{R}^\ell \setminus \tilde{\mathcal{R}}^\ell$, para cada $\ell = 1, \ldots, \mathcal{L}$, com custos reduzidos negativos. Se o subproblema *pricing* retorna colunas ou rotas com essa característica, então as variáveis correspondentes devem entrar na base e, portanto, são adicionadas ao respectivo $\tilde{\mathcal{R}}^\ell$. Em seguida, o RMP resultante é resolvido e o processo descrito é repetido. Esse procedimento iterativo é realizado até que o subproblema *pricing* não proporcione colunas ou rotas com custos reduzidos negativos.

Similarmente ao método *simplex*, a etapa *pricing*, em uma dada iteração do algoritmo de CG, também requer a informação dual do RMP correspondente. Sejam $\bar{\pi} \in \mathbb{R}^n$ e $\bar{v} \in \mathbb{R}^+$ as soluções duais associadas às restrições (9) e (10) do RMP, respectivamente, em que $n = |\mathcal{N}|$ corresponde ao número de clientes. Então, para uma dada iteração do algoritmo de CG, o subproblema *pricing* correspondente ao modo ℓ, pode ser escrito como:

$$z_{SP}^\ell(\bar{\pi}, \bar{v}) = \min \left\{ p_1 \sum_{j \in \mathcal{N}} x_{r_0 j}^\ell + (p_2 - \bar{v}) \ell \sum_{j \in \mathcal{N}} x_{r_0 j}^\ell + \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} (p_3 c_{ij}^\ell - \bar{\pi}_i) x_{pij}^\ell | r \in \mathcal{R}^\ell \right\}. \tag{12}$$

No problema de otimização (12), $\boldsymbol{x}_r^\ell = \{x_{rij}^\ell\}_{i,j \in \mathcal{N}}$ corresponde a um vetor binário cujas componentes denotam a variável de fluxo binária do problema, assumindo o valor de 1 se e somente se o veículo destinado a percorrer a rota r no modo ℓ visita em sequência os clientes i e j.

O subproblema *pricing* (12) para cada modo ℓ corresponde ao problema do caminho mínimo robusto com restrições de recurso (abrev. em Inglês RSPPRC - *Robust Shortest Path Problems with Resource Constraints*), pois o conjunto $\mathcal{R}^\ell$ contém todas as rotas robustas operando

no modo ℓ. O problema do caminho mínimo com restrições de recurso (abrev. em Inglês SPPRC - *Shortest Path Problems with Resource Constraints*) já foi mostrado ser NP-difícil [Dror, 1994] e, portanto, a versão robusta, por ser uma extensão desse problema, também o é. Diferentes formulações e métodos de solução têm sido usados para resolver de forma exata o SPPRC [Feillet et al., 2004; Irnich e Villeneuve, 2006; Drexl e Irnich, 2014; Lozano et al., 2016]. Contudo, um algoritmo bastante usado, que tem se mostrado efetivo, é o algoritmo de *labeling* baseado em programação dinâmica. O algoritmo de *labeling* usado aqui é referenciado como algoritmo de *labeling* robusto, pois a RO estática é usada como abordagem de tratamento da incerteza. Portanto, o algoritmo de *labeling* precisa ser estendido e adaptado para a versão robusta do SPPRC.

5.2. Algoritmo *labeling* robusto

O algoritmo *labeling* padrão é usado para resolver vários tipos de problemas do caminho mínimo [Irnich e Desaulniers, 2005]. Esse algoritmo representa os caminhos parciais provenientes do depósito inicial 0 em uma rede usando vetores chamados *labels*. Iniciando com o *label* inicial $\mathcal{F}_0$ no depósito inicial, o algoritmo enumera rotas parciais propagando *labels* através da rede $\mathcal{G}$ usando funções de extensões. Os caminhos iniciando no depósito e finalizando no mesmo nó são comparados usando regras de dominância, de modo a eliminar caminhos para os quais pode ser comprovado que eles não podem produzir um caminho ou rota ótima iniciando no depósito inicial 0 e finalizando no depósito final $n+1$. O algoritmo *labeling* robusto funciona de forma similar ao padrão, com a diferença que a robustez deve ser levada em consideração na propagação dos caminhos parciais.

5.2.1. Representação do *label*

Uma rota parcial $r^{\ell}_{v_i}$ finalizando no nó v_i e operando no modo ℓ é representada por um *label* com $1+|\mathcal{N}|+(\Gamma^q+1)+1$ componentes, as quais são descritas a seguir:

- Uma componente para o custo reduzido $\tilde{C}^{\mathcal{F}}_{v_i}$;
- $n=|\mathcal{N}|$ componentes binárias $V^{\mathcal{F}}_{v_i,1},\ldots,V^{\mathcal{F}}_{v_i,j},\ldots,V^{\mathcal{F}}_{v_i,n}$ indicando com o valor 1 se o cliente $j\in\mathcal{N}$ faz parte da rota parcial $r^{\ell}_{v_i}$. A componente $V^{\mathcal{F}}_{v_i,j}$, para todo $j\in\mathcal{N}$, também pode assumir o valor 1 se o cliente j não é atingível, isto é, se uma extensão da rota parcial $r^{\ell}_{v_i}$ a esse cliente leva a uma infactibilidade de pelo menos um recurso;
- Γ^q+1 componentes $R^{\mathcal{F}}_{v_i,\gamma}$, para $\gamma=0,\ldots,\Gamma^q$, exprimindo a carga do veículo ao sair do cliente v_i, dado que a demanda de γ clientes na rota parcial $r^{\ell}_{v_i}=(v_0,v_1,\ldots,v_i)$ assumem seus respectivos valores de pior caso;
- Uma componente $T^{\mathcal{F}}_{v_i}$ para o consumo do recurso tempo.

As componentes $\tilde{C}^{\mathcal{F}}_{v_i}, V^{\mathcal{F}}_{v_i,1},\ldots,V^{\mathcal{F}}_{v_i,n}$ e $T^{\mathcal{F}}_{v_i}$ correspondem às componentes determinísticas e elas são equivalentes às do algoritmo *labeling* padrão [Irnich e Desaulniers, 2005]. Por outro lado, as componentes $R^{\mathcal{F}}_{v_i,\gamma}$ são definidas similarmente aos parâmetros usados para propagar as equações recursivas (2). Portanto, esses parâmetros são usados para levar em consideração a robustez do recurso carga.

5.2.2. Funções de extensão

Dado um *label* $\mathcal{F}^{\ell}_{v_i}$ associado à rota parcial $r^{\ell}_{v_i}$ e um nó v_j, com $V^{\mathcal{F}}_{v_iv_j}=0$, a função de extensão para as componentes $R^{\mathcal{F}}_{v_i\gamma}$ correspondem à Expressão (2). A componente do custo reduzido $\tilde{C}^{\mathcal{F}}_{v_i}$ é estendida como $\tilde{C}^{\mathcal{F}}_{v_j}=\tilde{C}^{\mathcal{F}}_{v_i}+p_3d_{v_iv_j}-\bar{\pi}_{v_i}$, em que $\bar{\pi}_{v_i}$ define o prêmio de visitar

o nó v_i. Os prêmios coincidem com as soluções duais associadas às restrições (9) do RMP. As componentes $V^{\mathcal{F}}_{v_i,k}$, para todo $k \in \mathcal{N}$, e $T^{\mathcal{F}}_{v_i}$ são estendidas similarmente como no algoritmo *labeling* padrão. Finalmente, após a extensão, aceita-se o *label* resultante $\mathcal{F}^{\ell}_{v_j}$ como uma extensão factível robusta se $V^{\mathcal{F}}_{v_i v_j} = 0$; $R^{\mathcal{F}}_{v_j\gamma} \leq Q$ para $\gamma = 0, \ldots, \Gamma^q$ e $T^{\mathcal{F}}_{v_j} \leq w^b_{v_j}$. Neste estudo, adaptou-se a regra de dominância robusta proposta em De La Vega et al. [2019], em que apenas a robustez do recurso carga é considerada.

5.3. Desigualdades válidas, regras de ramificação e heurística primal

No BPC proposto, usou-se as desigualdades válidas *Subset Row* (SR) preconizadas em Jepsen et al. [2008], baseadas em subconjuntos de três clientes, dado o ganho significativo em desempenho proporcionado por essas desigualdades em outras variantes próximas do problema de roteamento [Munari e Morabito, 2018; Munari et al., 2018].

Foram propostas três diferentes regras de ramificação, uma para cada termo que define os custos na função objetivo. Primeiramente, ramifica-se no número de veículos, seguido do número de total de entregadores e, finalmente, em termos das variáveis de fluxo associadas aos arcos. Também, usa-se uma heurística primal baseada em programação inteira-mista, impondo integralidade para variáveis do mestre e resolvendo o problema resultante por um *solver* de propósito geral. O objetivo é obter soluções factíveis do problema para melhorar o limitante superior, o que pode levar a um melhor desempenho do método BPC. Para mais detalhes sobre desigualdades SR, decisões de ramificação e heurística primal, veja Álvarez e Munari [2017]; Munari e Morabito [2018]; De La Vega et al. [2019].

6. Resultados Computacionais

Experimentos computacionais foram realizados visando comparar o desempenho computacional do algoritmo BPC proposto com as abordagens de solução desenvolvidas em De La Vega et al. [2017] para resolver o RVPTWMD com demanda incerta. O algoritmo BPC foi implementado em C++ usando a biblioteca PDCGM proposta por Gondzio et al. [2013, 2015], que oferece um método de geração de colunas estabilizado usando pontos interiores. A árvore de busca do BPC é administrada usando como base o código *branch-and-price* de pontos interiores descrito por Munari e Gondzio [2013]. Adicionalmente, a heurística primal do BPC usa o *solver IBM CPLEX Optimization Studio v.12.8* para resolver problemas de programação inteira-mista. Todos os experimentos foram executados em um PC Linux com um processador Intel Core i7-4790 3.6 GHz e 16 GB de memória RAM.

Para a realização dos experimentos foram usados exemplares baseados nas conhecidas instâncias de Solomon, retirados das classes C1 e R1, sendo as mesmas instâncias usadas em De La Vega et al. [2017]. A distribuição geográfica dos clientes nestas instâncias é classificada de acordo com: i) distribuição agrupadas (classe C1); ii) distribuição aleatória (classe R1). Do mesmo modo que em De La Vega et al. [2017], reduziu-se a capacidade dos veículos Q para 80 na classe C1 e 50 na R1. Os valores dos parâmetros $\bar{q}_i, \bar{t}_{ij}, d_{ij}, w^a_i$ e w^b_i correspondem aos 25-primeiros dados das classes C1 e R1 das instâncias de Solomon, exceto para o tempo de serviço, o qual foi determinado como em Pureza et al. [2012]. A classe C1 possui 9 instâncias, enquanto a R1 possui 12 instâncias.

Os custos unitários por rota gerada, por veículo usado e por distância percorrida foram de $p_1 = 1$, $p_2 = 0.1$ e $p_3 = 0.0001$, respectivamente, como em De La Vega et al. [2017]. Os números de veículos e entregadores disponíveis no depósito foram $|\mathcal{K}| = \mathcal{K} = 15$ e $\mathcal{D} = 20$, para o caso limitado. E, para o caso ilimitado, o número de veículos foi fixado em 25 e o número de entregadores em 75. O desvio $\hat{q}_i$, para todo $i \in \mathcal{N}$, foi determinado em termos do valor nominal $\bar{q}_i$, como $\hat{q}_i = \delta\bar{q}_i/100$, em que δ define o nível de incerteza da demanda. Experimentos computacionais

foram realizados com valores do parâmetro δ fixados em 15%, 20% e 30%, ao passo que os valores atribuídos ao parâmetro Γ^q foram 0, 2 e 5. Em total, são 378 instâncias para a realização dos experimentos numéricos, obtidas a partir da combinação dos valores atribuídos aos parâmetros δ, Γ^q e do número de problemas em cada uma das classes consideradas (C1 e R1) para cada um dos casos limitado e ilimitado.

6.0.1. Análise do algoritmo *branch-price-and-cut*

As Tabelas 1-2 e 3-4 apresentam os resultados médios tanto para o caso limitado ($\mathcal{K} = 15$ e $\mathcal{D} = 20$) como para o ilimitado ($\mathcal{K} = 25$ e $\mathcal{D} = 75$), respectivamente. Observe que essas tabelas além de apresentarem os resultados médios do algoritmo BPC proposto, também apresentam os resultados médios das abordagens de solução desenvolvidas em De La Vega et al. [2017]. Assim, tem-se quatro abordagens: CPLEX; RHI1 (abrev. em Inglês RHI1 - *Robust Heuristic I1*); RHI1 com CPLEX; e o BPC proposto. Para cada nível de incerteza da demanda δ e para cada estratégia de solução, as tabelas mostram o valor da função objetivo (z), o tempo de execução em segundos (t) e o *gap* de otimalidade estimado como a diferença relativa entre o valor da função objetivo de cada uma das abordagens de solução e o melhor limitante inferior estimado pelo algoritmo BPC após 3600 segundos de execução. Convém salientar que as abordagens CPLEX, RHI1 e RHI1 com CPLEX propostas em De La Vega et al. [2017] correspondem a: CPLEX – abordagem de solução baseada apenas no *branch-and-cut* do CPLEX v.12.8; RHI1 – adaptação da heurística construtiva proposta em Solomon [1987] para o RVRPTWMD com demanda incerta; e RHI1 com CPLEX – abordagem de solução combinada que toma a solução factível gerada pela RHI1 como ponto de partida do algoritmo *branch-and-cut* do CPLEX v.12.8, como tentativa de melhorar a convergência do algoritmo.

Tabela 1: Resultados para a classe C1.n25.Q80 com $\mathcal{K} = 15$ e $\mathcal{D} = 20$

	$\delta = 15\%$			$\delta = 20\%$			$\delta = 30\%$		
Estratégia	z	t (s)	gap (%)	z	t (s)	gap (%)	z	t (s)	gap (%)
CPLEX	7,7435	3112	4,83	7,7436	3221	4,63	8,1118	3113	5,45
RHI1	7,7478	< 1	4,78	7,7478	< 1	4,69	8,1168	< 1	5,53
RHI1 com CPLEX	7,7415	2944	4,65	7,7417	2930	4,60	8,1110	2950	5,43
BPC	7,6435	1110	0,02	7,6542	1560	0,03	8,0589	2000	0,03

Tabela 2: Resultados para a classe R1.n25.Q50 com $\mathcal{K} = 15$ e $\mathcal{D} = 20$

	$\delta = 15\%$			$\delta = 20\%$			$\delta = 30\%$		
Estratégia	z	t (s)	gap (%)	z	t (s)	gap (%)	z	t (s)	gap (%)
CPLEX	9,5726	3246	10,05	9,6925	3311	12,85	10,2610	3376	12,67
RHI1	9,7632	< 1	13,19	10,0869	< 1	19,53	10,5970	< 1	16,75
RHI1 com CPLEX	9,1677	3084	7,55	9,3813	3110	8,96	10,0049	3313	9,85
BPC	8,8536	25,60	0,00	8,9856	23,30	0,00	9,4254	25,10	0,00

Tabela 3: Resultados para a classe C1.n25.Q80 com $\mathcal{K} = 25$ e $\mathcal{D} = 75$

	$\delta = 15\%$			$\delta = 20\%$			$\delta = 30\%$		
Estratégia	z	t (s)	gap (%)	z	t (s)	gap (%)	z	t (s)	gap (%)
CPLEX	7,7507	3503	7,93	7,7564	3396	7,69	8,1535	3502	8,22
RHI1	7,7479	< 1	7,82	7,7478	< 1	7,53	8,1167	< 1	7,37
RHI1 com CPLEX	7,7434	3600	7,68	7,7452	3440	7,49	8,1137	3603	7,13
BPC	7,7240	1320	0,02	7,7356	1205	0,01	8,0954	1574	0,03

Tabela 4: Resultados para a classe R1.n25.Q50 com $\mathcal{K} = 25$ e $\mathcal{D} = 75$

	$\delta = 15\%$			$\delta = 20\%$			$\delta = 30\%$		
Estratégia	z	t (s)	gap (%)	z	t (s)	gap (%)	z	t (s)	gap (%)
CPLEX	9,7752	3499	21,07	9,9568	3502	10.67	10,4282	3496	18,13
RHI1	9,7632	< 1	8,94	10,0869	< 1	12,13	10,5970	< 1	16,65
RHI1 com CPLEX	9,4723	3506	5,46	9,6732	3503	7,02	10,2476	3504	12,43
BPC	9,1145	21,00	0,00	9,4521	19,5	0,00	9,9856	15,23	0,00

Os resultados correspondentes ao algoritmo BPC das Tabelas 1-2 e 3-4 ilustram que as instâncias da classe C1 representam as mais desafiadoras para o algoritmo BPC. Em média, o algoritmo BPC levou aproximadamente 1462 segundos para resolvê-las. De fato, há algumas instâncias em que o algoritmo BPC não provou otimalidade (11 instâncias das 162 na classe C1). Os clientes nas instâncias dessa classe estão agrupados, levando à existência de muitas rotas com custos similares. Adicionalmente, as janelas de tempo dos clientes são bastante amplas nesta classe, o que também leva a muitas rotas factíveis. Isso provoca o uso de mais *labels* no algoritmo *labeling* robusto e, portanto, bem mais comparações na regra de dominância robusta são feitas. Por outro lado, o algoritmo resolve otimamente todas as instâncias da classe R1 (216 instâncias) durante um tempo médio de execução de apenas 22 segundos. O tempo médio de resolução para as instâncias da classe R1 exprime que elas são as mais fáceis para o algoritmo BPC. Contrariamente às instâncias das classes com clientes agrupados, os clientes das instâncias da classe sendo analisada estão distribuídos aleatoriamente, induzindo a ter poucas rotas com custo similar e dado que as janelas de tempo dos clientes são bastante apertadas, então há poucas rotas factíveis. Dessa forma, poucos *labels* são usados e menos comparações na regra de dominância robusta são realizadas.

Por fim, o algoritmo BPC resolveu otimamente 367 instâncias com 25 clientes das 378 consideradas nos experimentos computacionais. Comparado aos 56 problemas resolvidos otimamente pela estrategia combinada (RHI1 com CPLEX) desenvolvida em De La Vega et al. [2017], pode-se concluir que o algoritmo BPC foi o algoritmo que teve o melhor desempenho em termos da qualidade da solução, bem como em eficiência computacional. Em média, o algoritmo BPC foi 20 vezes mais rápido do que a estrategia combinada para as instâncias resolvidas otimamente por ambas as abordagens de solução.

7. Considerações finais

Nesse artigo, abordou-se o RVRPTWMD com demanda incerta e pertencendo a um politopo de incerteza. As incertezas são levadas em conta usando uma abordagem de RO estática e utiliza-se um algoritmo BPC para resolver uma formulação baseada em particionamento de conjuntos. À luz dos resultados, o algoritmo BPC provou otimalidade para mais de 97% das instâncias consideradas na experimentação computacional e em tempos computacionais bem menores do que a estratégia combinada, desenvolvida em De La Vega et al. [2017]. Portanto, conclui-se que o algoritmo BPC resulta em uma estrategia efetiva para resolver de forma exata o RVRPTWMD em instâncias com 25 clientes. Como tópico de pesquisa futuro pretende-se avaliar o desempenho do algoritmo BPC proposto em instâncias de porte maior, com 50 e 100 clientes, por exemplo.

Agradecimentos

Os autores agradecem o apoio financeiro da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processos 2013/07375-0, 2015/14582-7 e 2016/01860-1; do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

Referências

Agra, A., Christiansen, M., Figueiredo, R., Hvattum, L. M., Poss, M., e Requejo, C. (2013). The robust vehicle routing problem with time windows. *Computers & Operations Research*, 40(3): 856 – 866.

Álvarez, A. e Munari, P. (2017). An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. *Computers & Operations Research*, 83:1 – 12.

Ben-Tal, A., Hertog, D. d., e Vial, J.-P. (2012). Deriving robust counterparts of nonlinear uncertain inequalities.

Bertsimas, D., Brown, D. B., e Caramanis, C. (2011). Theory and applications of robust optimization. *Society for Industrial and Applied Mathematics*, 53(3):446–501.

Bertsimas, D. e Goyal, V. (2012). On the power and limitations of affine policies in two-stage adaptive optimization. *Mathematical Programming*, 134(2):491–531.

Bertsimas, D., Goyal, V., e Lu, B. (2015). A tight characterization of the performance of static solutions in two-stage adjustable robust linear optimization. *Mathematical Programming*, 150 (2):281–319.

Bertsimas, D. e Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming*, 98(1-3):49–71.

Bertsimas, D. e Sim, M. (2004). The price of robustness. *Operations Research*, 52(1):35–53.

De La Vega, J., Munari, P., e Morabito, R. (2017). Robust optimization for the vehicle routing problem with multiple deliverymen. *Central European Journal of Operations Research*. ISSN 1613-9178. URL `https://doi.org/10.1007/s10100-017-0511-x`.

De La Vega, J., Munari, P., e Morabito, R. (2019). A compact formulation and a branch-price-and-cut algorithm for the robust vehicle routing problem with multiple deliverymen. Technical report, Production Engineering Dept., Federal University of São Carlos, Brazil.

Drexl, M. e Irnich, S. (2014). Solving elementary shortest-path problems as mixed-integer programs. *OR Spectrum*, 36(2):281–296.

Dror, M. (1994). Note on the complexity of the shortest path models for column generation in vrptw. *Operations Research*, 42(5):977–978.

Feillet, D., Dejax, P., Gendreau, M., e Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229. ISSN 1097-0037. URL `http://dx.doi.org/10.1002/net.20033`.

Gendreau, M., Jabali, O., e Rei, W. (2016). 50th anniversary invited article - future research directions in stochastic vehicle routing. *Transportation Science*, 50(4):1163–1173.

Gondzio, J., González-Brevis, P., e Munari, P. (2013). New developments in the primal-dual column generation technique. *European Journal of Operational Research*, 224(1):41 – 51.

Gondzio, J., González-Brevis, P., e Munari, P. (2015). Large-scale optimization with the primal-dual column generation method. *Mathematical Programming Computation*, 8(1):47–82.

Gounaris, C. E., Repoussis, P. P., Tarantilis, C. D., Wiesemann, W., e Floudas, C. A. (2016). An adaptive memory programming framework for the robust capacitated vehicle routing problem. *Transportation Science*, 50(4):1239–1260.

Irnich, S. e Desaulniers, G. (2005). *Shortest Path Problems with Resource Constraints*, p. 33–65. Springer US, Boston, MA.

Irnich, S. e Villeneuve, D. (2006). The shortest-path problem with resource constraints and k-cycle elimination for. *INFORMS Journal on Computing*, 18(3):391–406.

Jepsen, M., Petersen, B., Spoorendonk, S., e Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511.

Lee, C., Lee, K., e Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the operational research society*, 63(9):1294–1306.

Lozano, L., Duque, D., e Medaglia, A. L. (2016). An exact algorithm for the elementary shortest path problem with resource constraints. *Transportation Science*, 50(1):348–357.

Munari, P. e Gondzio, J. (2013). Using the primal-dual interior point algorithm within the branch-price-and-cut method. *Computers & Operations Research*, 40(8):2026 – 2036.

Munari, P. e Morabito, R. (2018). A branch-price-and-cut algorithm for the vehicle routing problem with time windows and multiple deliverymen. *TOP*, 26(3):437–464.

Munari, P., Moreno, A., De La Vega, J., Alem, D., Gondzio, J., e Morabito, R. The robust vehicle routing problem with time windows: compact formulation and branch-price-and-cut method. Working Paper (Accepted for publication in Transportation Science), 2018.

Poss, M. (2013). Robust combinatorial optimization with variable budgeted uncertainty. *4OR*, 11 (1):75–92.

Poss, M. (2014). Robust combinatorial optimization with variable cost uncertainty. *European Journal of Operational Research*, 237(3):836 – 845.

Pureza, V., Morabito, R., e Reimann, M. (2012). Vehicle routing with multiple deliverymen: Modeling and heuristic approaches for the VRPTW. *European Journal of Operational Research*, 218 (3):636 – 647.

Senarclens de Grancy, G. e Reimann, M. (2016). Vehicle routing problems with time windows and multiple service workers: a systematic comparison between ACO and GRASP. *Central European Journal of Operations Research*, 24(1):29–48.

Sniedovich, M. (2012). Robust optimization - the elephant in the robust-satisficing room. Technical report, University of Melbourne.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Soyster, A. L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operation Research*, 21(5):1154–1157.