Contribuições na logística de entrega urbana expressa de última milha usando grandes instâncias reais de cidades brasileiras

Thailsson Clementino[1], **Juan Rosas**[1], **Rosiane de Freitas**[1], **Eduardo Uchoa**[2]

[1]Instituto de Computação – Universidade Federal do Amazonas

[2]Departamento de Engenharia de Produção - Universidade Federal Fluminense

{thailsson.clementino, rosiane}@icomp.ufam.edu.br, uchoa@producao.uff.br

Abstract. *In this article, the Vehicle Routing Problem with Time Windows is addressed as a way of incorporating dynamic characteristics of the express delivery logistics chain, exploring theoretical aspects that best define the problem and its instances. The problem is to determine minimum cost routes that must be performed by a fleet respecting the capabilities of cars and the time windows associated with each delivery. A strategy that uses the Branch-Cut-and-Price method through the VRPsolver tool was applied to two sets of instances, the first being Solomon's artificial classical instances and, the second, real Brazilian instances adapted from the loggiBUD benchmark.*

Resumo. *Neste artigo, o problema de roteamento de veículos com janelas de tempo é abordado, como forma de incorporar características de dinamicidade da cadeia logística de entregas expressas, explorando aspectos teóricos que melhor definem o problema e suas instâncias. O problema consiste em determinar rotas de custo mínimo que devem ser executadas por uma frota de veículos respeitando suas capacidades e as janelas de tempo de entregas de cada cliente. Uma estratégia que utiliza o método de Branch-Cut-and-Price através da ferramenta VRPsolver foi aplicada em dois conjuntos de instâncias, sendo a primeira as instâncias clássicas artificiais de Solomon e, a segunda, instâncias reais brasileiras adaptadas do benchmark loggiBUD.*

1. Introdução

Os canais de vendas virtuais se tornaram cada vez mais relevantes para clientes e empresas na última década, mas, o advento da pandemia da COVID-19 e o consequente isolamento social praticado desde o início de 2020, levaram à mudanças ainda mais abrangentes e significativas nos hábitos de consumo, ocasionando uma migração massiva para as compras online [Cavalcanti and Doneux 2021] e gerando desafios significativos para a cadeia logística de entregas expressas. Nesse novo cenário, devido a competitividade do mercado, as empresas ampliaram a oferta de entregas no mesmo dia com baixo custo [Fonseca-Galindo et al. 2020].

Este trabalho aborda um desafio logístico em especial, a entrega de última milha. Esse é o ponto da cadeia logística no qual se realiza o transporte do objeto em questão entre o depósito e o cliente final (Figura 1). O problema de criar rotas de última milha com o menor custo associado possível é conhecido na literatura como o Problema de Roteamento

de Veículos (do inglês, *Vehicle Routing Problem* - VRP). Resolver o problema de roteamento de maneira eficiente implica em contribuir diretamente para o desenvolvimento de cidades inteligentes, pois pode-se observar esse problema em várias aplicações de uma cidade real como entrega de compras online, coletas de lixo, entregas bancárias, entregas postais, coleta de lixo industrial, entregas de supermercado, roteamento de ônibus escolares, etc.

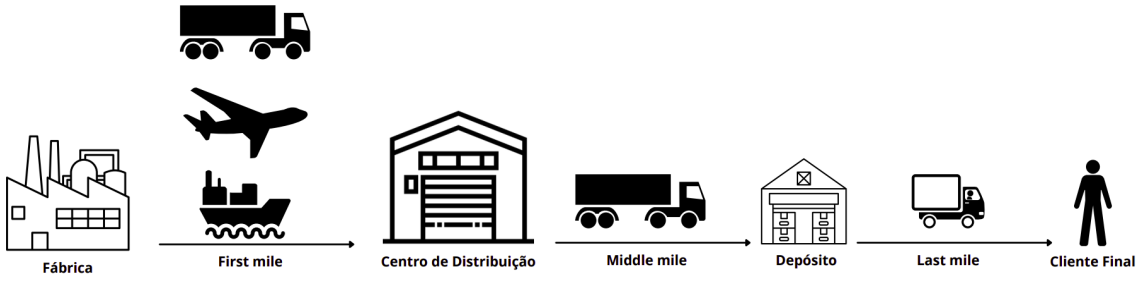

Figura 1. Representação simplificada da Cadeia Logística de entregas urbanas. Fonte: próprios autores (2022)

O VRP e suas variantes têm sido objeto de intensa investigação na otimização combinatória e na pesquisa operacional há pelo menos 60 anos [Dantzig and Ramser 1959, Toth and Vigo 2014]. Com o passar dos anos, cada vez mais são incorporados aos problemas de roteamento características que tentam, teoricamente, replicar restrições para lidar com problemas realistas [Vidal et al. 2020]. Essas restrições estão associadas a capacidade de armazenamento dos veículos, a integração de mais de um depósito como fonte de fornecimento dos produtos, o tempo em que as rotas podem ser executadas, e a ordem em que os clientes são visitados. Algumas variações ainda tratam a entrada do problema de maneira dinâmica, na qual não são conhecidos todas as entregas a serem realizadas a priori, tendo assim uma aproximação maior com algumas aplicações em tempo real. Uma restrição a ser destacada em alguns tipos de entregas feitas no mesmo dia é a disponibilidade do cliente para o recebimento, no qual não se pode realizar uma entrega fora da janela de tempo na qual o cliente esteja disponível. Essa restrição adicionada ao clássico VRP compõe a variante mais famosa na literatura do VRP, o problema de roteamento de veículos com janelas de tempo (em inglês, *Vehicle Routing Problem with Time Windows* - VRPTW) [Desaulniers et al. 2014].

O VRPTW é NP-Difícil, pois é uma generalização do VRP no sentido de que podemos obter o problema clássico quando usamos janelas de tempo grandes o suficiente. Os primeiros trabalhos sobre o problema são estudos de casos nos quais heurísticas simples foram utilizados como métodos de solução [Knight and Hofer 1968, Madsen 1976]. Em 1987, [Solomon 1987] introduziu um *benchmark* contendo instancias artificiais com 100 clientes cada, a partir de então a maioria dos trabalhos utilizou as instâncias como principal testes para seus métodos. Uma extensão dessas instancias para mais clientes foi proposta em [Gehring and Homberger 2001] e passou a ser mais utilizada quando os trabalhos começaram a alcançar resultados bons para as instancias conhecidas de Solomon. Em [Zhang et al. 2019], trabalha-se com instâncias adaptadas de dados reais da *China National Petroleum Corporation*.

Desde os primeiros trabalhos, muitos métodos foram propostos para solucionar o problema. Uma grande parte desses trabalhos tratam-se de Heurísticas

das mais variadas conhecidas pela literatura como algoritmos de busca local [Bräysy and Gendreau 2005], busca local iterativa [Lim and Zhang 2007], *Large Neighborhood Search* [Bent and Van Hentenryck 2004], além de algoritmos com busca baseada em população [Blocho and Czech 2013, Vidal et al. 2013].

Os primeiros métodos exatos foram algoritmos *branch-and-bound* [Christofides et al. 1981] construídos ainda nos anos 1980. Posteriormente outros tipos de algoritmos compõem o conjunto de soluções propostas para o problema. Alguns trabalhos, como [Bard et al. 2002, Lysgaard 2006], utilizam algoritmos *Branch-and-Cut*, esses algoritmos adicionam cortes aos nós da árvore de busca para apertar o relaxamento linear. [Baldacci et al. 2011] desenvolveu um método para reduzir o tamanho do modelo de particionamento de conjunto.

Algoritmos *Branch-and-Cut-and-Price* são um dos principais métodos utilizados para resolver os problemas de roteamento. Esse método surge da junção entre o método de geração de coluna para resolver as relaxações lineares (*Price*) quanto da adição de cortes para apertar a relaxação (*Cut*). Alguns trabalhos como [Jepsen et al. 2008, Pecin et al. 2017] utilizam dessa abordagem.

Neste trabalho procurou-se estudar o problema dentro de algumas grandes cidades brasileiras através de instancias baseadas em entregas urbanas reais. O método de solução utilizado foi um algoritmo Branch-Cut-and-Price implementado através da ferramenta VRPSolver [Pessoa et al. 2020]. O conteúdo apresentado é uma extensão do trabalho feito para o workshop do Programa de Bolsas de Pesquisa Loggi [Rosas et al. 2021].

O restante deste trabalho está organizado como segue. Na Seção 2 o VRPTW é formalmente descrito. A Seção 3, apresenta brevemente a ferramenta VRPSolver e a formulação aplicada a ferramenta para solucionar o VRPTW. Na Seção 4, são apresentadas as instâncias utilizadas, além dos resultados obtidos que são posteriormente discutidos. Por fim, na Seção 5 são tecidas as considerações finais sobre o trabalho.

2. Problema de Roteamento de Veículos com Janelas de Tempo

O problema de roteamento de veículos com janelas de tempo (*Vehicle Routing Problem with Time Windows* - VRPTW) é a variante do VRP mais abordada na literatura. Nesse problema são atribuídos a cada cliente uma janela de tempo em que as entregas podem ser realizadas, em sua versão mais clássica (chamada de *Hard Time Windows* [Solomon 1987]) é permitido que o veículo chegue ao cliente antes do inicio da janela, no entanto o tempo de espera é levado em consideração na hora de calcular o custo. Além disso, chegar atrasado em relação ao fim da janela é proibido. Em algumas variantes (chamadas de *Soft Time Windows*), quando o veículo chega antes ou depois da janela de tempo é aplicado uma penalização na função objetivo.

Formalmente definindo o problema básico de VRPTW [Baldacci et al. 2012], é dado um grafo direcionado completo $G = (V, A)$, onde o conjunto de vértices V é composto pelo vértice 0, que representa o depósito, mais um conjunto de vértices $N = \{1, ..., |N|\}$, que representa os clientes que serão visitados. A é o conjunto de arestas que liga cada par $(i, j) \in V \times V$, onde $i \neq j$. Associado a cada aresta existe um custo de viajem c_{ij} e um tempo de viagem $t_{ij} > 0$, onde t_{ij} inclui o tempo de serviço no vértice i. Em cada vértice $i \in V$ é associado uma demanda q_i e uma janela de tempo

$[e_i, d_i]$, onde e_i e d_i representam respectivamente o primeiro e o último momento para visitar o vértice i. O conjunto de veículos disponíveis é denotado por $K = \{1, .., |K|\}$. Os veículos são homogêneos e possuem capacidade Q. O objetivo é minimizar o custo total das rotas tais que essas rotas obedeçam as restrições de capacidade e das janelas de tempo.

As Figuras 2 e 3 ilustram, respectivamente, uma instância e sua solução para o problema do VRPTW.

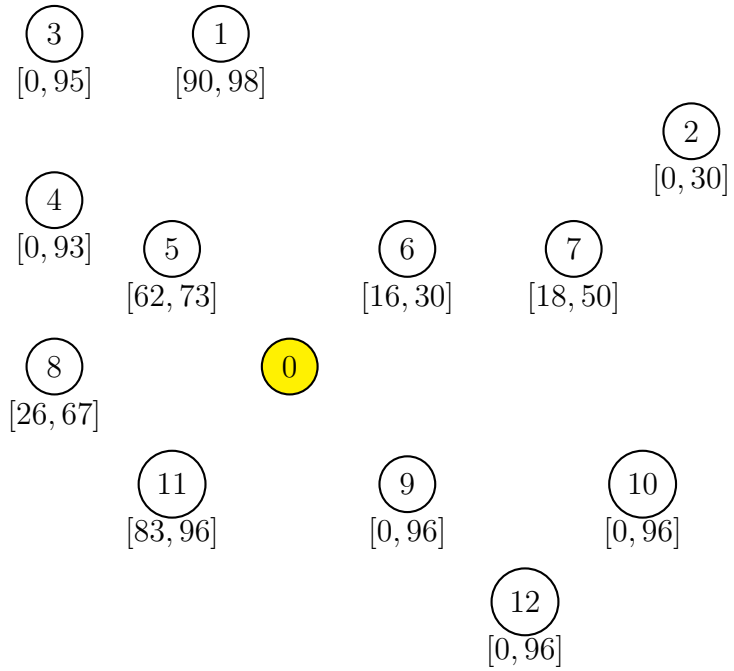

Figura 2. Representação de instância para o VRPTW. Fonte: próprios autores (2022).

3. Método de Resolução

Atualmente, métodos de Branch-Cut-and-Price são as abordagens dominantes no contexto de roteamento de veículos [Desaulniers et al. 2014]. Infelizmente, projetar e codificar um algoritmo Branch-Cut-and-Price complexo e sofisticado pode ser uma tarefa exigente com vários meses de trabalho de uma equipe qualificada. Por isso, neste trabalho resolveu-se utilizar um framework proposto para construção de modelos robustos sem que se use meses de trabalho, o VRPSolver [Pessoa et al. 2020].

A ideia do solver (VRPSolver) é definir um modelo genérico tal que, nas ocasiões em que se consegue encaixar um problema de otimização nesse modelo genérico ele pode ser resolvido por um algoritmo Branch-Cut-and-Price. No modelo, todos os subproblemas são modelados como um problema de caminho mínimo com restrição de recurso (do inglês *Resource Constrained Shortest Path* - RCSP [Pugliese and Guerriero 2013]). Para resolver os subproblemas, esses devem ser modelados como grafos com recursos associados as arestas (ou aos vértices), além disso deve ser definido a formulação mestre contendo a função objetivo e uma função de mapeamento que mapeia as variáveis de decisão aos grafos de subproblema.

O VRPSolver foi proposto inicialmente para problemas de roteamento

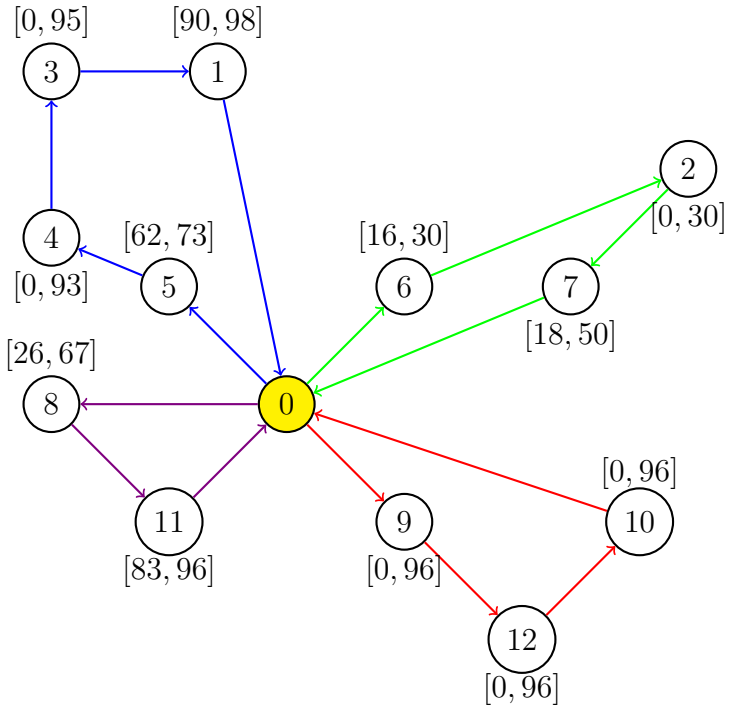

Figura 3. Representação de uma típica solução de instância. Fonte: próprios autores (2022).

[Pessoa et al. 2020], no trabalho onde é proposto o solver, ele é aplicado a vários problemas de roteamento, dentre eles CVRP, PDPTW, VRPTW, MDVRP, BWTSP, e CARP. O trabalho ainda aplica o VRPSolver em problemas que não são de roteamento como *Bin Packing Problem* e *Generalized Assigment Problem*. Na maior parte dos problemas o VRPSolver alcança soluções tão boas ou melhores do que o estado da arte até então. Em outro trabalho são abordados problemas de *Bin Packing* geral utilizando o VRPSolver [Pessoa et al. 2021]. Damião et al. aplicaram solver ao CCVRP resolvendo instancias em aberto na literatura [Damião et al. 2021].

3.1. Modelagem matemática

O modelo construído para resolver o VRPTW utilizando o VRPSolver é descrito nas seções 3.1.1 e 3.1.2:

3.1.1. Grafo para o Subproblema RCSP

- $G = (V, A), v_{source} = v_{sink} = 0$;
- $R = R_M = \{1, 2\}$;
- $q_{a,1} = d_j, a \in A$;
- $q_{a,2} = t_{ij}, a \in A$;
- $[l_{i,1}, u_{i,1}] = [0, Q], i \in V$;
- $[l_{i,2}, u_{i,2}] = [e_i, d_i], i \in V$;

No grafo G definido acima, foi adicionado dois recursos principais. O recurso de número 1 representa a restrição de capacidade, quando um veículo utiliza a aresta $a = (i, j)$ o seu consumo $q_{a,1}$ para essa aresta é dado pela demanda d_j do cliente j. Além

disso, para respeitar a restrição de capacidade do problema foi definido que em cada um dos vértices i o limite superior de consumo acumulado $u_{i,1} = Q$. O recurso de número 2 representa a restrição de tempo (janela), quando um veículo utiliza a aresta $a = (i, j)$ o seu consumo $q_a, 2$ para essa aresta é dado pelo tempo de deslocamento t_{ij} entre os vértices i e j. Seguindo a definição do problema foi definido como limites de consumo acumulado para o recurso de tempo $[l_{i,2}, u_{i,2}] = [e_i, d_i]$. Na Figura 4 é ilustrado um grafo para a instância descrita na Tabela 1.

Tabela 1. Tabela com o exemplo de uma instância.

i	q_i	e_i	d_i	s_i
0	0	0	30	0
1	3	4	8	1
2	4	1	10	1
3	10	2	6	1

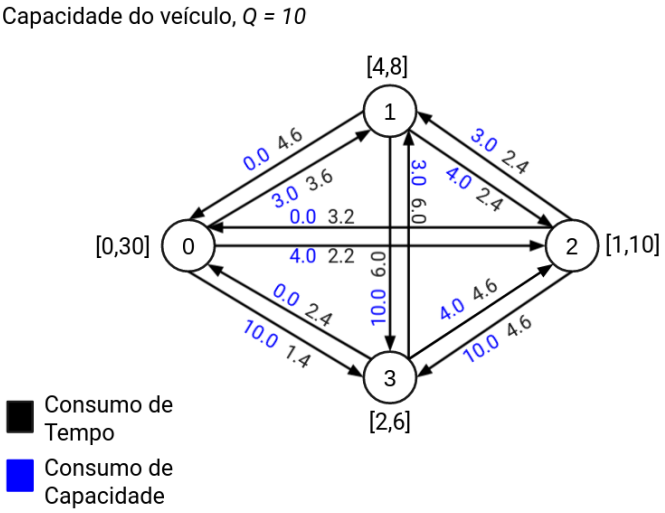

Figura 4. Exemplo de grafo com restrição de recursos. Fonte: próprios autores (2022).

3.1.2. Formulação

$$min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1a}$$

$$\sum_{(i,j)\in\delta^-(j)} x_{ij} = 1, \quad \forall j \in N \tag{1b}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall (i, j) \in A \tag{1c}$$

Na Formulação, a função objetivo (1a) quer minimizar o custo das rotas selecionadas para a solução. As Restrições (1b) definem que a solução só deve passar uma única vez por um cliente. As outras restrições são implicitamente definidas pela função de mapeamento definida M e pelo valor de L e U. A Restrição (1c) define o domínio inteiro binário para a variável de decisão x. As outras restrições para o problema são definidas implicitamente através do grafo descrito na seção 3.1.1. Para isso define-se uma função de mapeamento simples, no qual cada variável de decisão é mapeado para a respectiva aresta, formalmente $M(x_{ij}) = \{(i, j)\}, \forall (i, j) \in A$; e os limites de rotas no grafo são definidos por $L = 1, U = K$..

Além da modelagem básica para um algoritmo *Branch-Cut-and-Price* foram definidos *packing sets* (ou conjuntos de empacotamento), eles são utilizados pelo VRPSolver para melhorar o desempenho, passando para um algoritmo de *Branch-Cut-and-Price* avançado. Pode ser melhor entendido no artigo original [Pessoa et al. 2020] como os

packing sets são utilizados na implementação de elementos algorítmicos como *ng-paths*, enumeração de rotas, cortes *Rank-1* com memória limitada.

$$P^V = E^V = \cup_{i \in V}\{\{i\}\} \tag{2}$$

Os *Rounded Capacity Cuts* (RCCs) [Laporte and Nobert 1983], ainda são úteis em algoritmos *Branch-Cut-and-Price* modernos, através deles são adicionados cortes robustos.

$$(\cup_{i \in V}\{(\{i\}, q_i)\}, Q) \tag{3}$$

4. Resultados

O modelo elaborado no VRPSolver foi testado em dois conjuntos de instâncias diferentes. O primeiro conjunto é composto pelas tradicionais instâncias de Solomon [Solomon 1987], enquanto no segundo conjunto são utilizados instâncias adequadas a partir do *benchmark* de entregas urbanas expressas (loggiBUD) proposto pela empresa Loggi [Loggi 2021]. Essas instâncias são utilizadas afim de testar uma solução exata em instâncias de cidades brasileiras baseadas em entregas urbanas reais. Para se adequar ao VRPTW foram modificadas 60 instâncias disponíveis no *benchmark*. As modificações adicionaram janelas de tempo e tempo de serviço para cada um dos clientes pertencentes a uma instância. Essas instâncias foram escolhidas levando em conta a representatividade dos locais disponíveis no *benchmark*. As instâncias disponíveis no *benchmark* foram criadas sinteticamente com dados públicos, e simula um desafio real de uma grande empresa na etapa final da cadeia logística de entregas urbanas expressas. O *benchmark* é interessante por representar cidades brasileiras de dois diferentes estados (Pará - PA e Rio de Janeiro-RJ) e o Distrito Federal (DF), com demandas específicas para cada região. Sendo essa as principais contribuição deste trabalho: propor a adequação para incorporação de janelas de tempo em instâncias baseadas em entregas reais em cidades brasileiras utilizando distancias de ruas, visto que a maior parte dos trabalhos na literatura faz uso das instâncias artificiais utilizando como métrica a distância euclidiana; e a aplicação de um método exato nelas.

Na Seção 4.1 será explicado o processo para a adaptação das instâncias disponíveis no benchmark loggiBUD, de modo a suportarem janelas de tempo. Na Seção 4.2 são apresentados os resultados para os experimentos realizados.

4.1. Instâncias adaptadas

No repositório do *benchmark* loggiBUD são disponibilizados dois tipos de instâncias: *delivery-instances* e *cvrp-instances*. As instâncias trabalhadas neste artigo são as do segundo tipo, representando o clássico problema de roteamento de veículos capacitados (CVRP). Nessas instâncias são apresentadas informações sobre: a localização do depósito; a capacidade Q dos veículos da frota; as localizações de cada entrega a ser feita; e, as demandas de cada uma das entregas. O custo de deslocamento c_{ij} de um cliente i para um cliente j é calculado através de distâncias de mapas utilizando ruas reais (`http://project-osrm.org`), assim tem-se uma função de distância diferente da tradicional distância euclidiana utilizada na maior parte dos trabalhos da literatura.

O loggiBUD possui 1320 instâncias do tipo *cvrp-instances*, assim, foram selecionadas algumas instâncias representativas para serem adicionadas janela de tempo, com quatro tipos diferentes de intervalos:

- R - Instâncias com Janelas de Tempo aleatórias;
- F30m - Instâncias com Janelas de Tempo fixas de 30 min;
- F1h - Instâncias com Janelas de Tempo fixas de 1h;
- F2h - Instâncias com Janelas de Tempo fixas de 2h;

Para cada um dos tipos foram gerados 15 instâncias, sendo 5 geradas a partir de instâncias para o PA, e 5 a partir de instâncias para o DF e 5 para RJ. Em termos de número de entregas a serem feitas, as instâncias RJ são as maiores, com mais de 1000 entregas a serem feitas por instância, seguido pelas instâncias DF que variam entre 700 e 1000 por instância. As instâncias PA são as menores, contendo entre 200 e 400 entregas por instância.

Para construir as novas instâncias foram utilizadas as distâncias entre os clientes em metros disponibilizadas pelo servidor OSRM. Para definir as janelas fixas de tempo ficou definido que um veículo anda em uma velocidade constante de $30km/h$ então, $1h$ para as instâncias são equivalentes a 30000 metros de deslocamento. As instâncias foram construídas com um plano de horizonte $12h$, assim para todas as instâncias $[a_0, b_0] = [0, 360000]$.

As instâncias com janelas de tempo aleatórias foram definidas de acordo com o proposto para as instâncias clássicas do VRPTW na literatura [Solomon 1987], onde o centro da janela é definido aleatoriamente entre o intervalo $[a_0 + t_{0i}, b_0 - t_{0i} - s_i]$, e, a largura da janela também é definida aleatoriamente de maneira que a janela ainda respeite o intervalo. Nas instâncias com janela de tamanho fixo, o centro da janela é definido da mesma maneira, no entanto a largura passa de ser definida aleatoriamente para assumir um valor fixo dependendo do seu tipo (F30m, F1h e F2h).

4.2. Experimentos Computacionais

Os experimentos foram realizados utilizando um computador com 16Gb de memória disponível e processador Intel Core i7-8750H 2.20GHz, rodando Linux Ubuntu 20.04.4. Além disso a implementação base utilizada é a mesma da demo disponível no site do VRPSolver (`https://vrpsolver.math.u-bordeaux.fr`), apenas algumas mudanças relacionadas a pré e pós-processamento de dados foram realizados. Cada instância trabalha foi executada 3 vezes, nas quais foi definido um tempo máximo para cada execução de 1.5h (5400s).

Primeiro, foi refeito um experimento usando o BCP para o VRPTW [Pessoa et al. 2020]. Algumas instâncias de Solomon (cada uma com 100 entregas a serem feitas) foram testadas para comparar seus resultados com as instâncias geradas neste trabalho. Todas as instâncias testadas fazem parte do conjunto de instâncias com um grande horizonte de escalonamento, o que gera soluções com mais clientes atendidos em cada rota. No experimento, foram resolvidas quase todas as instâncias dentro do tempo limite definido (1.5h), exceto a instância R208 (A instância mais difícil do conjunto). Todas as instâncias resolvidas levaram menos de 15 minutos para encontrarem o ótimo, o que revela que o método funciona bem. Mesmo que os experimentos tenham sido realizados

em um computador pessoal, as instâncias não resolvidas antes de 2017 [Pecin et al. 2017] foram resolvidas. Na Tabela 2 estão dispostos os resultados para os experimentos em duas colunas, onde a primeira mostra o tempo médio de execução para alcançar o ótimo e a segunda exibe o custo da solução encontrada.

Tabela 2. Tabela com resultados de exeperimentos realizados nas instâncias Solomon.

Instância	Tempo (s)	Custo
C203	20.78	588.7
C204	46.61	588.1
R202	184.30	1029.6
R203	93.84	870.8
R204	470.02	731.3
R206	527.43	875.9
R207	305.83	794.0
R208	> 5400	
R209	549.01	854.8
R210	897.46	900.5
R211	656.16	746.7
RC204	339.39	783.5
RC207	217.88	962.9
RC208	344.38	776.09

O método de geração de colunas utilizado pelo VRPSolver funciona melhor para o problema do VRPTW quando suas janelas são mais justas (intervalos pequenos). Isso pôde ser constatado na resolução das instâncias usadas neste trabalho, onde para as instâncias geradas a partir do *benchmark* loggiBUD nenhuma das instâncias com janelas aleatórias, geralmente com intervalos grandes, conseguiu ser resolvida em menos de 2 horas de execução. Para as instâncias com janelas fixas algumas instâncias do estado do Pará (PA) causaram alguma dificuldade, mas, no geral foram resolvidas (obteve-se o ótimo) em menos de 1,5 horas (tempo limite definido).

Os resultados dos experimentos para as instâncias estão dispostos nas Tabelas 3 e 4 em duas colunas, onde a primeira mostra o tempo médio de execução para alcançar o ótimo e a segunda exibe o custo da solução encontrada. Pode-se observar que o método resolveu rapidamente algumas instâncias com menos de 300 entregas a serem feitas e janelas fixas. Quando o número de entregas passa de 300 o algoritmo acaba gastando bastante tempo de execução para resolver.

5. Considerações Finais

Instâncias reais de cidades brasileiras do *benchmark* loggiBUD [Loggi 2021], com informações de entregas urbanas expressas, foram adaptadas para incorporar diferentes intervalos representando janelas de tempo para a variação VRPTW. Foi utilizado também o clássico conjunto de instâncias artificiais de Solomon [Solomon 1987]. Um método de Branch-Cut-and-Price foi aplicado utilizando o VRPsolver, sendo validado de modo a se reproduzir resultados obtidos na literatura para um caso similar usando as instâncias de Solomon, mas também, incorporando resultados representativos envolvendo grandes instâncias urbanas reais de até 300 clientes. Uma análise empírica massiva, bem como

Tabela 3. Tabela com resultados para as instâncias com janelas fixas de 30 minutos.

Instância	Tempo (s)	Custo
F30m-pa-n289	1288.17	808037.0
F30m-pa-n266	273.04	895959.0
F30m-pa-n354	> 5400	
F30m-pa-n324	3081.95	875398.0
F30m-pa-n308	>5400	
F30m-df-n999	>5400	
F30m-df-n904	> 5400	
F30m-df-n974	> 5400	
F30m-df-n835	> 5400	
F30m-df-n939	> 5400	

Tabela 4. Tabela com resultados para as instâncias com janelas fixas de 1 hora.

Instância	Tempo (s)	Custo
F1h-pa-n269	486.05	880795.0
F1h-pa-n324	3474.46	1143354.0
F1h-pa-n322	> 5400	
F1h-pa-n212	958.37	645609.0
F1h-pa-n296	778.30	607808.0
F1h-df-n780	> 5400	
F1h-df-n815	> 5400	
F1h-df-n890	> 5400	
F1h-df-n957	> 5400	
F1h-df-n993	> 5400	

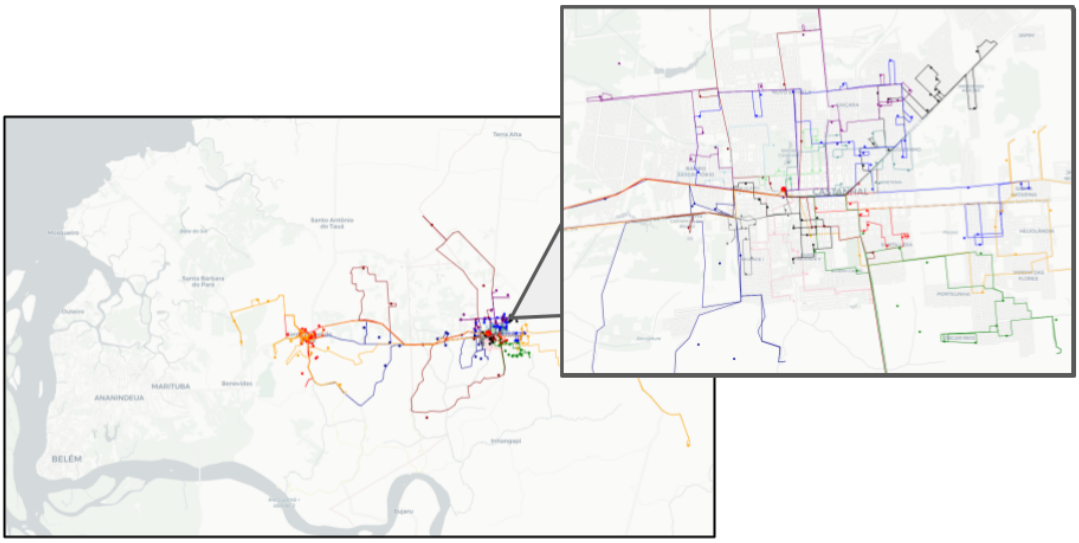

Figura 5. Solução para a instância F30m-pa-n266, traçada no mapa, contendo as ruas da cidade de Castanhal, no estado do Pará, e arredores. Fonte: [Loggi 2021].

a incorporação de características que representem melhor a dinamicidade das estregas expressas, através da modelagem como um VRP em conjunto com restrições de problemas de escalonamento para as entregas a serem feitas, incluindo datas de disponibilidade, data de término obrigatório e sugerida, fazem parte da investigação em andamento e seus próximos passos.

Agradecimentos

Este trabalho é um resultado obtido por meio do Programa de Bolsas de Pesquisa (PBP) da LOGGI, empresa brasileira de logística de entregas expressas. O trabalho também faz

parte do projeto de iniciação científica PIB-E/0186/2021 PIBIC UFAM/CNPq, do grupo de pesquisa do CNPq em Otimização, Algoritmos e Complexidade Computacional (ALGOX). Os autores também foram apoiados por agências de fomento brasileiras: CAPES, CNPq, FAPEAM e FAPERJ.

Referências

Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations research*, 59(5):1269–1283.

Baldacci, R., Mingozzi, A., and Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218(1):1–6.

Bard, J. F., Kontoravdis, G., and Yu, G. (2002). A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2):250–269.

Bent, R. and Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38(4):515–530.

Blocho, M. and Czech, Z. J. (2013). A parallel memetic algorithm for the vehicle routing problem with time windows. In *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pages 144–151. IEEE.

Bräysy, O. and Gendreau, M. (2005). Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation science*, 39(1):104–118.

Cavalcanti, L. J. E. and Doneux, N. F. (2021). Análise de fatores determinantes na decisão de compra online: reflexões sobre o impacto da pandemia no comportamento do consumidor brasileiro.

Christofides, N., Mingozzi, A., and Toth, P. (1981). State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11(2):145–164.

Damião, C. M., Silva, J. M. P., and Uchoa, E. (2021). A branch-cut-and-price algorithm for the cumulative capacitated vehicle routing problem. *4OR*, pages 1–25.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management science*, 6(1):80–91.

Desaulniers, G., Madsen, O. B., and Ropke, S. (2014). Chapter 5: The vehicle routing problem with time windows. In *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 119–159. SIAM.

Fonseca-Galindo, J. C., Surita, G. d. C., Neto, J. M., de Castro, C. L., and Lemos, A. P. (2020). A multi-agent system for solving the dynamic capacitated vehicle routing problem with stochastic customers using trajectory data mining. *arXiv preprint arXiv:2009.12691*.

Gehring, H. and Homberger, J. (2001). A parallel two-phase metaheuristic for routing problems with time-windows. *Asia Pacific Journal of Operational Research*, 18(1):35–48.

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

Knight, K. and Hofer, J. (1968). Vehicle scheduling with timed and connected calls: A case study. *Journal of the Operational Research Society*, 19(3):299–310.

Laporte, G. and Nobert, Y. (1983). A branch and bound algorithm for the capacitated vehicle routing problem. *Operations-Research-Spektrum*, 5(2):77–85.

Lim, A. and Zhang, X. (2007). A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 19(3):443–457.

Loggi (2021). loggibud: Loggi benchmark for urban deliveries. `https://github.com/loggi/loggibud`.

Lysgaard, J. (2006). Reachability cuts for the vehicle routing problem with time windows. *European Journal of Operational Research*, 175(1):210–223.

Madsen, O. (1976). Optimal scheduling of trucks-a routing problem with tight due times for delivery. *Optimization applied to transportation systems*, pages 126–136.

Pecin, D., Contardo, C., Desaulniers, G., and Uchoa, E. (2017). New enhancements for the exact solution of the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 29(3):489–502.

Pessoa, A., Sadykov, R., and Uchoa, E. (2021). Solving bin packing problems using vrpsolver models. In *Operations Research Forum*, volume 2, pages 1–25. Springer.

Pessoa, A., Sadykov, R., Uchoa, E., and Vanderbeck, F. (2020). A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183(1):483–523.

Pugliese, L. D. P. and Guerriero, F. (2013). A survey of resource constrained shortest path problems: Exact solution approaches. *Networks*, 62(3):183–200.

Rosas, J., Clementino, T., and de Freitas, R. (2021). Resolvendo instâncias de entregas urbanas reais com janelas de tempo baseadas no benchmark loggibud e aplicando um branch-cut-and-price via vrpsolver. In *Workshop PBP-Loggi*.

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

Toth, P. and Vigo, D. (2014). *Vehicle routing: problems, methods, and applications*. SIAM.

Vidal, T., Crainic, T. G., Gendreau, M., and Prins, C. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & operations research*, 40(1):475–489.

Vidal, T., Laporte, G., and Matl, P. (2020). A concise guide to existing and emerging vehicle routing problem variants. *European Journal of Operational Research*, 286(2):401–416.

Zhang, Z., Luo, Z., Qin, H., and Lim, A. (2019). Exact algorithms for the vehicle routing problem with time windows and combinatorial auction. *Transportation Science*, 53(2):427–441.