

## Heurística GRASP-VND para o Problema de Roteamento de Veículos com *Cross-Docking*

**Lucas Abritta Costa, Fernando Afonso Santos**

Universidade Federal de Itajubá – Campus Itabira  
Rua São Paulo, 377 – Bairro Amazonas – Itabira/MG – CEP: 35900-373  
lucasabritta@yahoo.com.br, fernandosantos@unifei.edu.br

**Geraldo Robson Mateus**

Universidade Federal de Minas Gerais  
Av. Antônio Carlos, 6629 – Belo Horizonte/MG – CEP: 31270-010  
mateus@dcc.ufmg.br

### RESUMO

O Problema de Roteamento de Veículos com *Cross-Docking* surge na distribuição de mercadorias em sistemas de cadeias de suprimentos utilizando plataformas de *Cross-Docking*. Existem na literatura modelos e algoritmos para a solução exata deste problema, no entanto, tais soluções são de elevada complexidade computacional. Por este motivo, propõe-se neste trabalho a implementação de uma heurística capaz de obter resultados aproximados eficientemente para o problema, que se baseia na meta-heurística GRASP com o uso de uma busca local VND. Os resultados computacionais mostram que, quando comparada aos métodos exatos disponíveis, a heurística obtém soluções aproximadas de boa qualidade (em alguns casos ótimas) a um tempo computacional reduzido, servindo como alternativa a utilização de métodos exatos para a solução do problema.

**PALAVRAS CHAVE.** Problema de Roteamento de Veículos, *Cross-Docking*, GRASP.

### ABSTRACT

The Vehicle Routing Problem with Cross-Docking arises in the distribution of goods in supply chain systems using Cross-Docking platforms. Different models and algorithms are available in the literature for the exact solution of the problem. However, such solutions hold a high computational complexity. Therefore, this paper proposes implementing a heuristic algorithm able to approximate results for the problem efficiently, which is based on GRASP meta-heuristic with a VND local search. The computational results show that the heuristic algorithm achieves good quality approximated solutions (optimal solutions for some instances) with less computational time, compared to the exact methods available.

**KEYWORDS.** Vehicle Routing Problem, Cross-Docking, GRASP.

## 1. Introdução

A gestão das cadeias de suprimentos exige conhecimento sobre a oferta/demanda de produtos (ou subprodutos) além de decisões operacionais que envolvem a coleta, a entrega e eventualmente o estoque destes produtos de modo a satisfazer todas as requisições dos elos da cadeia. Dada a sua importância, diversas ferramentas tem sido propostas ao longo dos últimos anos para auxiliar a gestão das cadeias de suprimentos, em busca de reduzir custos de produção e estoque para diferentes tipos de negócios.

Entre os problemas tratados na gestão da cadeia de suprimentos, o Problema de Roteamento de Veículos (PRV) [Dantzig e Ramser (1959)] é um dos mais importantes, pois o problema de transportar mercadorias de fornecedores a consumidores pode surgir em diferentes elos da cadeia e seus custos são geralmente elevados.

Para flexibilizar as ações operacionais nas cadeias de suprimentos foram propostas as plataformas de *Cross-Docking* (CD) [Witt (1998)], que recebem produtos coletados nos pontos de oferta (fornecedores) e permite que estes sejam consolidados de forma a facilitar sua entrega aos pontos de demanda (consumidores). A consolidação consiste em receber os produtos que foram coletados e permitir sua manipulação e organização antes destes serem encaminhados para a entrega.

A fim de minimizar os custos de estoque na cadeia de suprimentos, não é possível o armazenamento de produtos a longo prazo nas plataformas de *Cross-Docking*. Estes podem permanecer nestas plataformas apenas por um curto período de tempo que normalmente não ultrapassa um dia. Embora o uso de plataformas de *Cross-Docking* seja recente na literatura, diversas companhias de mercado global já publicaram trabalhos sobre sua utilização, entre elas Wall Mart [Gue (2001)], Toyota [Witt (1998)] e UPS [Forger (1995)]. Esta estratégia de armazenamento vem crescendo entre as companhias de grande e médio porte ao longo dos últimos anos, justificando a importância de sua utilização em cadeias de suprimentos.

A utilização das plataformas de *Cross-Docking* torna os problemas de distribuição mais complexos, uma vez que para definir a coleta e a entrega dos produtos através do PRV deve-se também considerar as operações de consolidação realizadas *Cross-Docking*. No entanto, ao utilizar esta estratégia é possível minimizar custos com estoques e tornar o processo de distribuição das mercadorias centralizado e mais organizado.

Nos últimos anos tem ocorrido um aumento nos estudos da estratégia de *Cross-Docking* dada a sua relevância para a distribuição de mercadorias e sua utilização em empresas de médio e grande porte. Para otimizar a distribuição das mercadorias nas cadeias de suprimentos foi proposto o Problema de Roteamento de Veículos com *Cross-Docking* (PRVCD), que surge da integração das decisões de roteamento do PRV para coletar/entregar as mercadorias com as decisões de consolidação a serem feitas no *Cross-Docking*.

O PVRCD pode ser formalmente definido em um grafo direcionado  $G=(V,A)$ , onde o conjunto de vértices é particionado em  $V=\{0, F, C\}$ , sendo  $F=\{1, \dots, n\}$  o conjunto de fornecedores,  $C=\{1', \dots, n'\}$  o conjunto de consumidores e 0 representando o *Cross-Docking*. Os arcos do grafo são definidos para conectar fornecedores e o *Cross-Docking* e este aos consumidores. Não existem arcos conectando fornecedores diretamente a consumidores. Logo,  $A = \{A_F, A_C\}$ , onde  $A_F = \{(i, j) : i, j \in F \cup 0\}$  e  $A_C = \{(i, j) : i, j \in C \cup 0\}$ . É definido um conjunto  $R$  contendo  $n$  requisições no grafo, cada requisição consiste em uma mercadoria cujo peso é  $q_i > 0$  que deve ser coletada no fornecedor  $i$  e levada ao respectivo consumidor  $i'$ , portanto  $R = \{(i, i', q_i) :$

$i=\{1,\dots,n\}$ . Para o transporte das requisições do conjunto  $R$  tem-se disponível uma frota homogênea de  $K$  veículos, todos com capacidade  $Q$ . Ao associar custos  $c_{ij}$  às arestas de  $A$  e um custo  $c_i$  para trocar a mercadoria  $q_i$  no Cross-Docking durante a consolidação, o PRVCD consistem em definir  $K$  rotas para coletar as mercadorias nos fornecedores,  $K$  rotas para entregá-las aos respectivos consumidores e o arranjo das mercadorias no *Cross-Docking* que minimizem os custos de distribuição (roteamento + consolidação). Todas as  $2K$  rotas devem partir e retornar ao Cross-Docking. A Figura 1 mostra um exemplo do PRVCD com  $n=5$ . Note que, as mercadorias  $q_1, q_2$  e  $q_4$  mudam de caminhão durante a consolidação, e respectivos custos  $c_1, c_2$  e  $c_4$  devem ser incluídos na solução, além dos custos de roteamento para visitar fornecedores e consumidores.

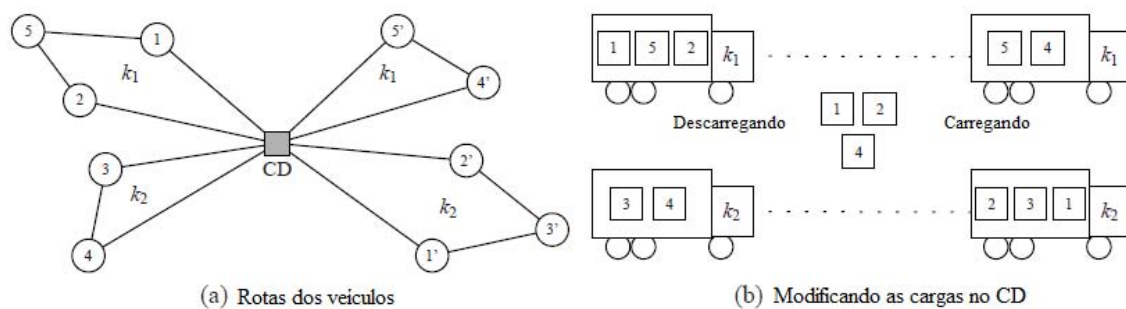


Figura 1 - Um exemplo de solução para VRPCD

Neste trabalho, propõe-se uma solução para o PRVCD utilizando a meta-heurística GRASP (*Greedy Randomized Adaptive Search Procedure*) [Feo e Resende (1995)]. O GRASP é uma meta-heurística sequencial iterativa. Para cada iteração são realizadas duas etapas: (i) construção de uma solução inicial para o problema; (ii) aplicação de um algoritmo de busca local na solução inicial em busca de uma solução mínimo local. É realizado um número finito de iterações e a melhor solução dentre todas as iterações (o melhor ótimo local obtido) é considerado a solução do problema. A construção de uma solução viável inicial é feita através do uso de uma função aleatorizada gulosa, utilizada com o intuito de conseguir alcançar diferentes mínimos locais ao longo do espaço de busca. Já para a busca local podem ser utilizadas diferentes estratégias e vizinhanças. Neste trabalho, propõe-se o uso de uma busca local VND (*Variable Neighborhood Decent*) [Mladenovic e Hansen(1997)].

O resto deste trabalho está estruturado da seguinte forma: na próxima seção, serão abordados os trabalhos relacionados ao PRVCD da literatura. Detalhes da implementação da heurística serão apresentados na Seção 3, enquanto seus resultados computacionais estão descritos na seção 4. Por fim, o trabalho é concluído na Seção 5, onde também são dadas direções futuras de pesquisa na área.

## 2. Trabalhos relacionados

O primeiro trabalho a integrar o PRV com a consolidação de mercadorias no *Cross-Docking* foi proposto por Lee et al. (2006). Segundo os autores, mesmo o *Cross-Docking* tendo

uma importância significativa na administração da cadeia de suprimentos a pesquisa de integração do roteamento de veículos com a consolidação de mercadorias no *Cross-Docking* ainda não haviam sido abordadas. Os autores apresentaram um modelo matemático para o PRVCD e uma heurística Busca Tabu para sua solução. O PRVCD apresentado por Lee et al. (2006) não considera custos para trocar mercadorias no *Cross-Docking*, no entanto, inclui janelas de tempo nos fornecedores e consumidores, dentro das quais os vértices devem ser visitados.

Logo após, Wen et al. (2009) apresentou uma abordagem para o PRVCD que se assemelha àquela proposta por Lee et al. (2006). Os autores consideraram o problema sem custos de consolidação e com janelas de tempo nos fornecedores e consumidores. No entanto, Wen et al. (2009) considera novas restrições operacionais obtidas de uma empresa de logística, e resolvem um PRVCD baseado num contexto real de distribuição de mercadorias. Para a solução, também foram propostas uma modelagem matemática e uma heurística Busca Tabu.

O PRVCD que considera custos de consolidação foi proposto por Santos et al. (2010, 2011a, 2011b, 2011c). Os autores propõem modelos matemáticos e algoritmos exatos baseados em estratégias de decomposição para solucionar o PRVCD na otimalidade. No entanto, as janelas de tempo para visitar fornecedores e consumidores são negligenciadas nestes trabalhos. São apresentados 4 modelos matemáticos e 3 algoritmos *Branch-and-price* para resolver o problema. Os resultados computacionais apresentados pelos autores permitem concluir que a inclusão das decisões de consolidação aumentam consideravelmente a complexidade do PRVCD. Mas ainda assim, foram obtidas soluções ótimas para instâncias com até 60 vértices e os gaps de dualidade foram pequenos para aquelas instâncias não solucionadas dentro do limite de tempo imposto.

### 3. Detalhes de implementação da heurística

Para a solução deste problema foi utilizada a meta-heurística GRASP, que tem como princípio dois passos para solucionar o problema: primeiro ele gera uma solução inicial gulosa-aleatorizada e então aplica uma busca local para melhorar o resultado e alcançar o mínimo local de uma dada vizinhança.

A construção da solução inicial gulosa-aleatorizada é controlada através do parâmetro  $\alpha = [0,1]$ . Quanto mais próximos de zero forem os valores de  $\alpha$  no intervalo, mais gulosa será a obtenção da solução inicial. Por outro lado, quanto mais próximos de um forem estes valores, mais aleatória será a construção da solução inicial.

A solução inicial é construída iterativamente partindo-se do depósito. A cada iteração, seleciona-se aleatoriamente um vértice adjacente ainda não visitado cujo custo da aresta esteja no intervalo  $[\min(c_{ij}), \min(c_{ij}) + \alpha * (\max(c_{ij}) - \min(c_{ij}))]$ , onde  $c_{ij}$  é o custo do arco que conecta o vértice  $i$  a seu adjacente  $j$ . Note que, quando  $\alpha = 0$ , a construção da solução inicial é totalmente gulosa, pois sempre escolhe-se aquele vizinho adjacente de menor custo. Já quando  $\alpha = 1$ , a construção é conduzida de forma aleatória, pois qualquer adjacente  $j$  pode ser escolhido a cada iteração. A seleção do adjacente  $j$  sempre leva em consideração a capacidade do veículo, que não pode ser excedida. A solução inicial é obtida quando o *Cross-Docking* é escolhido como vizinho adjacente.

A segunda etapa de implementação do GRASP consiste na utilização de vizinhanças e algoritmos de buscas locais. Nesta etapa foram utilizadas três vizinhanças, que foram integradas através da estratégia de busca local VND: (i) Trocar vértices da mesma rota, (ii) Trocar vértices entre duas rotas distintas e (iii) Retirar um vértice de uma rota e inserir em outra. Estas vizinhanças são explicadas em detalhes adiante.

Na vizinhança (i) o algoritmo avalia o custo de todas as possíveis trocas de ordem dos vértices para cada uma das rotas (tanto para as rotas que visitam fornecedores, quanto para aquelas que visitam consumidores). Quando existir uma troca de vértices que diminua o custo total da solução, a troca é realizada. Observe que, esta vizinhança não modifica os custos de consolidação no *Cross-Docking*, pois, os vértices a serem coletados/entregues pelo veículo continuam os mesmos, apenas altera-se a ordem de visita de tais vértices. A Figura 2 ilustra um exemplo de duas soluções vizinhas em (i).

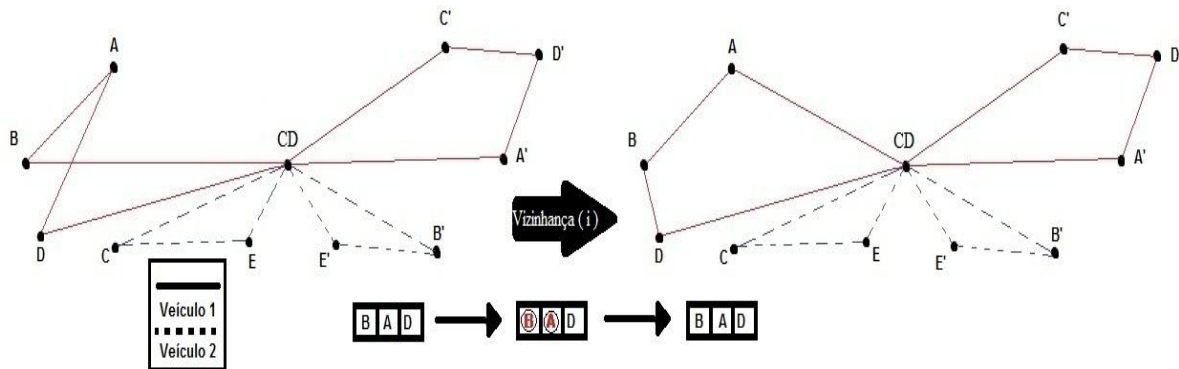


Figura 2 – Vizinhança que permite a troca de vértices na mesma rota

Na vizinhança (ii) serão selecionados um vértice em cada rota e ambos serão trocados. Deve ser avaliado o custo de cada possível troca e aquela que tiver a maior redução no custo da solução será efetuada. Neste caso, o custo de consolidação no *Cross-Docking* deve ser levado em consideração, pois nessa iteração os fornecedores/consumidores visitados pelos veículos são alterados, o que afeta o modo como a consolidação das mercadorias é realizada. Um exemplo de soluções vizinhas em (ii) é dado na Figura 3.

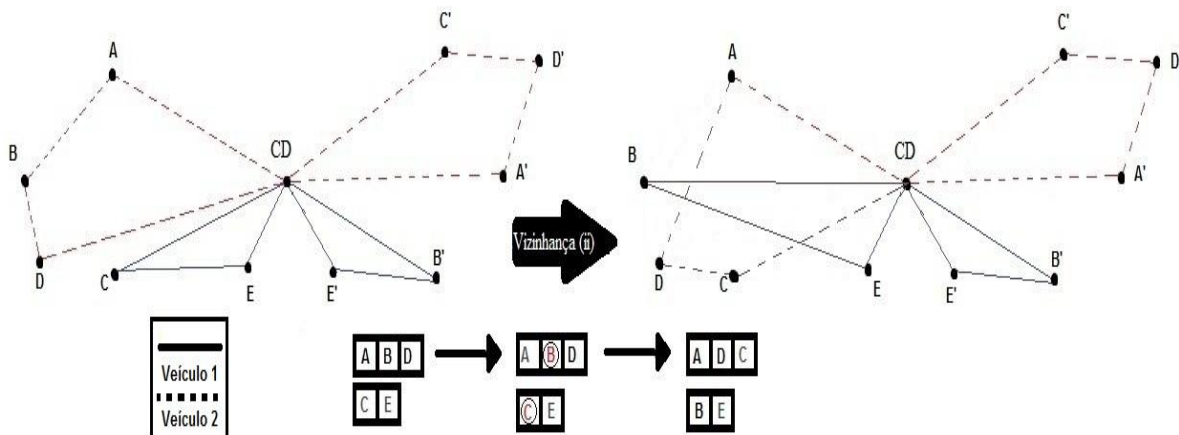


Figura 3 – Vizinhança para a troca de vértices entre rotas distintas

Por fim, a vizinhança (iii) consiste em retirar um fornecedor/consumidor de uma rota e incluí-lo em outra rota de modo que o custo da solução final seja reduzido. De forma análoga às vizinhanças anteriores, na vizinhança (iii) os movimentos de troca de fornecedor/consumidor devem respeitar a capacidade do veículo. Além disto, o custo de consolidação também pode ser alterado a cada movimento. A figura 4 ilustra um exemplo de movimento nesta vizinhança.

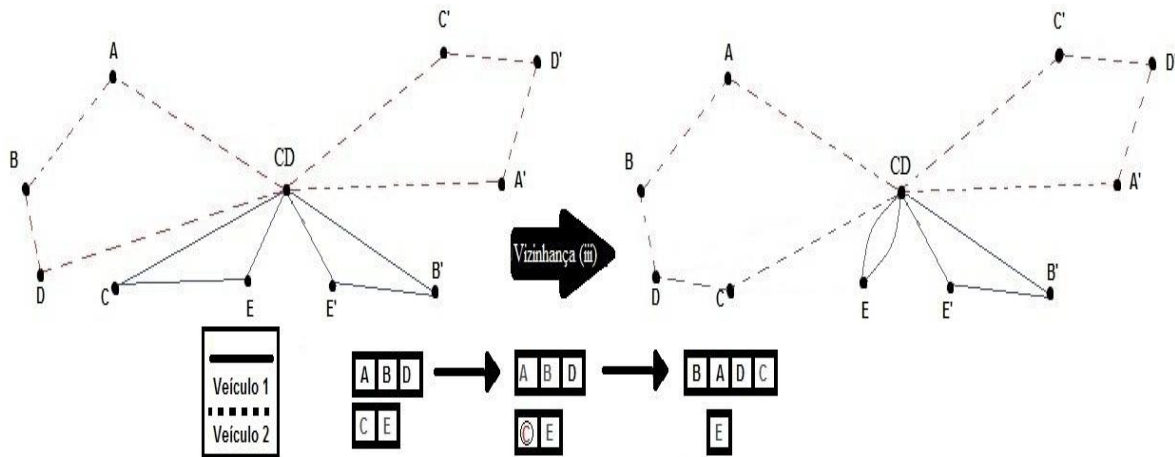


Figura 4 – Vizinhança para a remoção e inserção de um vértice em rotas distintas

As vizinhanças (i), (ii) e (iii) são integradas na busca local do GRASP usando o algoritmo VND. Primeiro, investiga-se todos os possíveis movimentos da vizinhança (i) que levam a uma solução que seja mínimo local. Quando esta solução for alcançada, parte-se para a investigação de soluções na vizinhança (ii) que sejam mínimo de (i) e (ii). Neste caso, quando uma solução for encontrada para a vizinhança (ii), deve-se retornar a vizinhança (i), para garantir também o seu mínimo local. O mesmo procedimento é realizado incluindo-se as vizinhanças (i), (ii) e (iii) até que não existam mais movimentos em nenhuma das vizinhanças que diminuam o custo da solução. Portanto, o VND assegura que o mínimo local obtido seja mínimo para as 3 vizinhanças, melhorando a qualidade das soluções obtidas.

Em busca de melhorar a eficiência da heurística foram implementadas também duas perturbações. A utilização de perturbações em soluções mínimos locais é um recurso utilizado para modificar as características da solução final obtida sem perder a sua qualidade, permitindo a investigação de novas direções de busca, mantendo a qualidade da solução que já se encontra no mínimo local. A primeira perturbação consiste em selecionar dois veículos quaisquer e trocar as rotas realizadas por eles, enquanto a segunda consiste em trocar um conjunto de mercadorias (cujas cardinalidade é aleatoriamente definida) entre dois veículos quaisquer. Estas perturbações foram propostas principalmente com o intuito de se encontrar soluções para as quais os custos de consolidação no *Cross-Docking* sejam minimizados. As Figuras 5 e 6 ilustram respectivamente exemplos de como a perturbação da troca de mercadorias e troca de rotas podem ser realizadas.



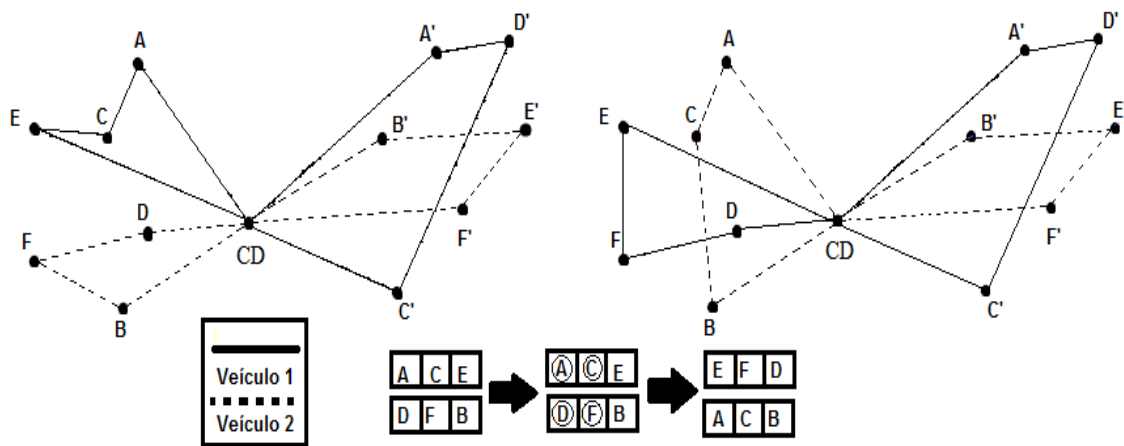


Figura 5 - Perturbação para a troca de mercadorias entre duas rotas.

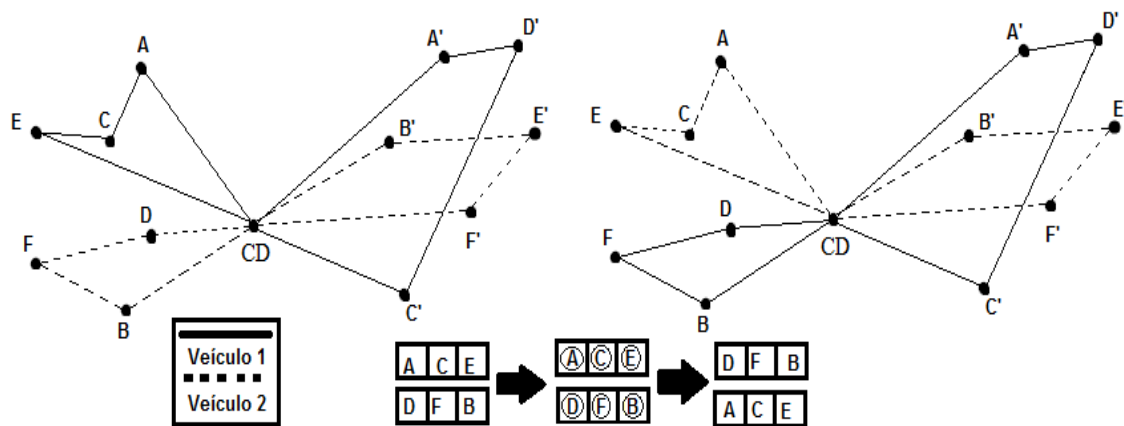


Figura 6 – Perturbação que realiza a troca das rotas entre dois veículos

### 5. Resultados Computacionais

Para testar a eficiência da heurística foram realizados experimentos computacionais utilizando o mesmo conjunto de instâncias utilizados em Santos et al. (2011a, 2011b). Este conjunto contém instâncias que variam o número de requisições  $n=\{10, 15, 20, 25, 30\}$ . Os fornecedores e consumidores são distribuídos geograficamente no plano euclidiano, com custos dos arcos dados pela distância euclidiana. Os experimentos foram conduzidos em uma máquina Intel i7-2630QM de 2.0 GHz e 8 Gb de memória RAM. A codificação da heurística foi realizada usando a linguagem C++.

No primeiro experimento será investigada a influência do parâmetro que mede o número de iterações da heurística com a qualidade das soluções obtidas. Para isto, foram selecionadas apenas aquelas instâncias com 30 requisições (61 vértices) e fixou-se o parâmetro  $\alpha = 0.5$ . O gráfico da Figura 7 mostra no eixo das abcissas a variação no número de iterações utilizado enquanto o eixo da ordenadas apresenta o custo das soluções em função da quantidade de iterações.

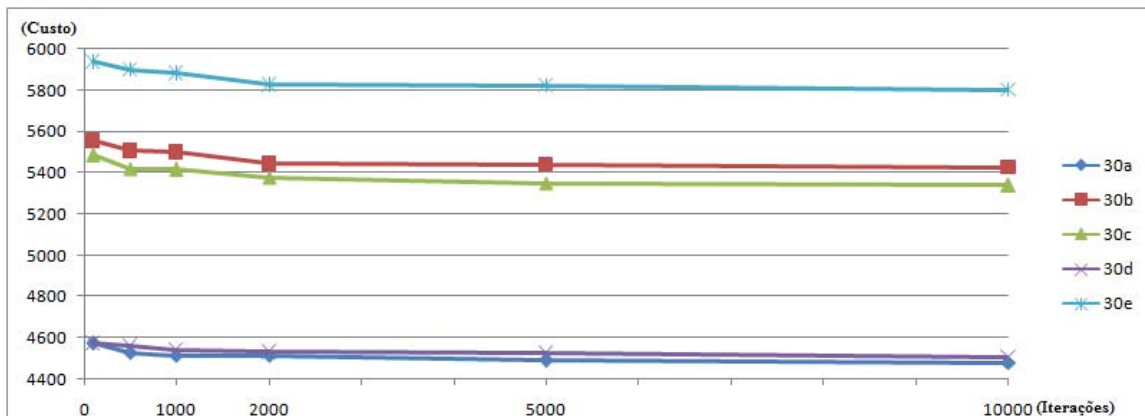


Figura 7 - Avaliação do parâmetro que mede o número de iterações na execução da heurística.

À medida em que o número de iterações aumenta, o custo das soluções tende a diminuir. No entanto, observa-se uma estabilização neste custo quando o número de iterações ultrapassa 2000. Por outro lado, o aumento no número de iterações aumenta também o tempo computacional necessário para obter a solução. Por este motivo, foi decidido que o valor mais adequado para este parâmetro a ser usado para a execução da heurística é 2000, pois é o valor com o qual soluções de boa qualidade são obtidas demandando menos tempo computacional.

O próximo experimento foi proposto para avaliar o comportamento da heurística de acordo com o parâmetro alpha, que conforme descrito anteriormente, controla o quão gulosa será a construção das soluções iniciais. Como não existe na literatura um consenso sobre qual o melhor valor para alpha foram feitos testes com 3 valores de  $\alpha = \{0, 0.5, 1\}$ . O número de iterações foi fixado em 2000 e foram feitos experimentos para todas as instâncias do conjunto, considerando o custo de troca  $c_i=20$  para todas as requisições. Os resultados estão descritos na Tabela 1, que mostra na primeira coluna a identificação da instância e então o tempo (em segundos) e o custo da solução obtidos para cada valor de alpha. Entradas na tabela identificadas com um traço, significa que o resultado não pode ser obtido em 2000 iterações da heurística.

Os melhores resultados foram obtidos ao utilizar o parâmetro alpha com valores 0.5 e 1. A construção gulosa da solução inicial ( $\alpha=0$ ) obteve os piores resultados, além de ter demandado maior esforço computacional e não ter sido capaz de encontrar soluções viáveis para 14 das 25 instâncias do conjunto. Decidiu-se adotar o valor  $\alpha=0.5$  para a execução da heurística, pois a média dos seus custos foram um pouco menores do que aqueles obtidos ao utilizar o valor  $\alpha=1$ .

Após ter verificado os valores mais adequados dos parâmetros para a execução da heurística GRASP-VND, o próximo experimento foi proposto para comparar o desempenho da heurística com os melhores valores obtidos pelos algoritmos exatos disponíveis na literatura de Santos et al (2011a, 2011b). Para isto, executa-se a heurística considerando-se todas as instâncias e compara-se o custo de suas soluções com aqueles obtidos através dos métodos exatos de Santos et al. (2011a, 2011b). Neste próximo experimento, diferentes custos de troca a serem incluídos na consolidação serão considerados:  $c_i=\{0, 20, 40\}$  para todas as requisições. Os resultados da comparação estão na Tabela 2. Na primeira coluna, está a identificação da instância, as próximas 3 colunas apresentam respectivamente para o custo de troca  $c_i=0$ , o tempo gasto pela heurística e a melhor solução após executar 2000 iterações, além do melhor resultado da literatura (MRL)



para a instância, obtido pelos algoritmos exatos de Santos et al (2011a, 2011b). O MRL pode ser dado pela solução exata da instância ou pelo melhor limite primal obtido, quando a solução ótima não pode ser avaliada no intervalo de 4 horas pelos algoritmos exatos. Informações similares são dadas nas colunas 5-7 para o custo de troca  $c_i=20$  e nas colunas 8-10 para o custo de troca  $c_i=40$ .

Instância	alpha = 0		alpha = 0.5		alpha = 1	
	Tempo(s)	Custo	Tempo(s)	Custo	Tempo(s)	Custo
10a	2.08	1840.37	1.85	1771.19	1.89	1766.61
10b	0.51	1817.86	0.84	1751.26	0.85	1751.26
10c	0.75	2086.66	0.74	1972.61	0.79	1971.93
10d	1.61	2070.74	1.51	1830.22	1.46	1831.53
10e	-----	-----	0.28	2170.43	0.25	2170.43
15a	14.96	2751.39	11.74	2539.41	11.53	2541.59
15b	4.37	2809.09	3.50	2682.07	3.57	2683.37
15c	6.49	3065.67	4.85	2964.44	4.72	2957.47
15d	9.42	2784.32	7.11	2641.89	7.26	2641.68
15e	-----	-----	1.74	2979.50	1.79	3000.92
20a	33.22	3468.74	28.34	3218.44	28.85	3222.99
20b	-----	-----	2.39	3544.66	2.35	3537.33
20c	-----	-----	6.31	3865.04	4.73	3826.32
20d	19.51	3352.53	18.46	3153.61	18.14	3154.49
20e	-----	-----	3.03	3897.48	1.44	3830.10
25a	54.10	4558.48	67.44	4056.93	67.00	4018.56
25b	-----	-----	10.76	4460.21	11.65	4449.93
25c	-----	-----	17.75	4744.08	16.87	4749.24
25d	52.12	4282.01	48.99	4019.39	39.62	3992.45
25e	-----	-----	8.63	4561.84	6.87	4641.99
30a	98.86	4827.75	105.47	4528.41	109.89	4518.79
30b	-----	-----	74.83	5524.56	12.93	5610.80
30c	-----	-----	25.24	5401.25	21.66	5466.43
30d	84.25	4798.24	94.96	4538.80	96.81	4571.53
30e	-----	-----	72.35	5806.23	12.49	5874.90

Tabela 1 – Avaliação do parâmetro alpha na execução da heurística.

De acordo com os resultados na Tabela 2, a heurística consegue resultados aproximados para as instâncias em um curto intervalo de tempo. Para as instâncias com 20 ou mais requisições, os algoritmos exatos levam até 4 horas para obter o MRL, enquanto a heurística consegue soluções próximas das ótimas (melhores) em poucos segundos. Isto nos permite concluir que a implementação da heurística tenha sido realizada com sucesso.

## 6. Considerações finais

Os estudos associados à utilização de plataformas de *Cross-Docking* são de grande relevância para empresas de logística, dada a relevância econômica envolvida às suas soluções. Em especial o Problema de Roteamento de Veículos com *Cross-Docking* (PRVCD) pode contribuir com soluções integradas de roteamento e consolidação de mercadorias nas plataformas, proporcionando a diminuição dos custos de distribuição na gestão das cadeias de suprimentos.

Instância	Custo de troca = 0				Custo de troca = 20				Custo de troca = 40			
	t(s)	Custo	MRL	GAP	t(s)	Custo	MRL	GAP	t(s)	Custo	MRL	GAP
10a	3.2	1696.1	1647.7	2.85%	4.0	1771.1	1703.9	3.78%	3.0	1790.4	1703.9	4.83%
10b	3.8	2130.1	2063.2	3.14%	1.7	2198.2	2103.2	4.32%	1.5	2238.2	2119.0	5.32%
10c	1.8	1892.3	1872.5	1.04%	1.5	1972.6	1918.1	2.76%	1.4	1981.3	1918.1	3.18%
10d	2.8	1759.9	1740.5	1.1%	3.1	1830.2	1804.1	1.42%	2.7	1831.5	1824.1	0.40%
10e	0.5	2079.0	1947.1	6.34%	0.5	2170.4	2027.1	6.60%	0.4	2212.4	2032.0	8.15%
15a	10.2	2776.9	2562.2	7.73%	23.7	2894.6	2664.8	7.93%	15.9	2932.6	2694.4	8.12%
15b	8.3	2937.6	2894.5	1.46%	7.5	3029.4	2955.1	2.45%	6.3	3144.5	2975.6	5.37%
15c	9.5	2903.8	2706.0	6.81%	8.8	2964.4	2752.6	7.14%	9.4	2977.0	2759.4	7.30%
15d	10.4	2707.1	2577.4	4.79%	13.1	2641.8	2617.4	0.92%	13.1	2673.4	2639.7	1.26%
15e	3.4	2833.5	2832.0	0.05%	2.7	2979.5	2922.5	1.91%	3.2	3058.7	2951.5	3.5%
20a	61.7	3254.4	3189.8	1.98%	59.3	3404.1	3256.4	4.33%	43.1	3321.6	3256.4	1.96%
20b	4.1	3308.1	3248.9	1.79%	4.3	3544.6	3325.9	6.17%	4.2	3560.4	3327.7	6.53%
20c	23.5	3173.5	2951.3	7.00%	13.3	3865.0	3019.2	21.88%	8.3	3972.0	3059.2	22.98%
20d	15.0	3463.0	3352.7	3.18%	34.4	3595.7	3375.0	6.13%	29.8	3645.1	3390.6	6.98%
20e	51.4	3078.1	3003.0	2.44%	2.0	3897.4	3084.9	20.84%	2.3	4065.1	3087.9	24.03%
25a	82.4	4248.7	4046.6	4.75%	131.4	4451.0	4135.2	7.09%	103.8	4217.1	4135.2	1.94%
25b	25.2	4195.1	4149.8	1.08%	20.7	4460.2	4167.1	6.57%	21.7	4573.2	4167.1	8.88%
25c	73.4	3747.5	3594.9	4.07%	34.7	4744.0	3691.7	22.18%	29.9	4864.5	3729.2	23.33%
25d	25.7	4159.8	3959.4	4.81%	68.9	4019.3	4000.1	0.47%	82.5	4385.2	4020.1	8.32%
25e	19.6	4385.5	4148.4	5.40%	19.6	4561.8	4275.9	6.26%	17.2	4823.4	4284.0	11.18%
30a	191.7	4843.9	4550.3	6.06%	209.4	4686.6	4635.7	1.08%	163.7	4636.3	4635.7	0.01%
30b	34.2	5153.0	4249.9	17.52%	38.8	5524.5	4329.4	21.63%	23.7	5969.6	4353.0	27.08%
30c	43.1	5051.7	4706.6	6.83%	51.2	5401.2	4706.6	12.86%	39.2	5533.3	4748.2	14.18%
30d	243.0	4244.7	4226.2	0.43%	174.9	4538.8	4246.0	6.45%	153.6	4626.4	4271.8	7.66%
30e	31.0	5503.7	4621.0	16.03%	142.3	5806.2	4649.8	19.91%	23.8	6407.6	4638.4	27.61%

Tabela 2 - Resultados para o PRVCD considerando custos de troca 0, 20 e 40.

Este trabalho apresenta a implementação de uma heurística GRASP para solucionar o PRVCD. A heurística GRASP é associada a um algoritmo de busca local VND que considera três vizinhanças distintas: (i) Trocar vértices da mesma rota, (ii) Trocar vértices entre duas rotas distintas e (iii) Retirar um vértice de uma rota e inserir em outra. Tais vizinhanças integradas em um VND permite a heurística alcançar mínimos locais de boa qualidade e soluções próximas das ótimas em um curto intervalo de tempo, diferente dos métodos exatos que encontram soluções ótimas (ou asseguram um certo gap de otimalidade) mas demandam um grande esforço computacional que pode levar horas ou até mesmo dias para instâncias de médio porte.

Como possibilidades de trabalhos futuros, diferentes vizinhanças e perturbações podem ser consideradas no GRASP-VND proposto ou diferentes tipos de heurísticas podem ser consideradas para a solução do PRVCD, como Algoritmos Genéticos ou Busca Tabu, que apresentam bons resultados na solução de problemas de roteamento.

## Referências

- G. B. Dantzig e R. H. Ramser**, The Truck Dispatching Problem, *Management Science*, 6: 80–91, 1959
- T. A. Feo e M. G. C. Resende**, Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.
- G. Forger**, UPS starts world’s premiere cross-docking operation, *Modern Materials Handling*, 36–38, 1995.
- R. K. Gue**, Cross-docking: Just-in-time for distribution, *Teaching Notes-Naval Postgraduate School* – Monterey, CA, May 2001.
- H. Y. Lee, J. W. Jung, K. M. Lee**, Vehicle Routing Scheduling for Cross-Docking in the supply chain, *Computers and Industrial Engineering*, 51: 247–256, 2006.
- N. Mladenovic e P. Hansen**, A variable neighborhood search. *Computers and Operations Research*, 24:1097–1100, 1997.
- F. A . Santos, A. S. da Cunha e G. R. Mateus**, Modelos de otimização para o Problema de Roteamento de Veículos com Cross-Docking, *Simpósio Brasileiro de Pesquisa Operacional - SBPO*, 2010.
- F. A . Santos, A. S. da Cunha e G. R. Mateus**, A branch-and-price algorithm for a Vehicle Routing problem with Cross-Docking, *Latin-American Algorithms, Graphs Optimization Symposium - LAGOS*, 2011a.
- F. A . Santos, A. S. da Cunha e G. R. Mateus**, A novel column generation algorithm for the Vehicle Routing Problem with Cross-Docking, *International Network Optimization Conference - INOC*, 2011b.
- F. A . Santos, A. S. da Cunha e G. R. Mateus**, Um Algoritmo Branch-and-price para o Problema de Roteamento de Veículos com Cross-Docking para Frotas Heterogêneas, *Simpósio Brasileiro de Pesquisa Operacional- SBPO*, 2011c.
- M. Wen, J. Larsen, J. Clausen, J-F. Cordeau e G. Laporte**, Vehicle Routing Problem with Cross-Docking. *Journal of Operational Research Society*, 60:1708–1718. 2009.
- C. E. Witt**, Cross-docking: Concepts demand choice, *Material Handling Engineering*, 53:44–49, 1998.