Universidade Federal de Minas Gerais
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

INVESTIGAÇÃO DE METAHEURÍSTICAS APLICADAS AO PROBLEMA DE ROTEAMENTO DE VEÍCULOS MULTIOBJETIVO COM COLETA OPCIONAL

LUCIANA PEREIRA DE ASSIS

Texto submetido à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Engenharia Elétrica.

Belo Horizonte
05 de julho de 2013

À minha filha Luísa

Agradecimentos

Agradeço primeiramente a Deus por me acompanhar em todos os momentos e por colocar verdadeiros anjos no meu caminho. Agradeço a minha filha Luísa por tranformar - como num toque de mágica - minhas tristezas em sorrisos e por cuidar de mim quando eu quem deveria cumprir este papel. Ao meu amor Alessandro por sempre acreditar em mim, me apoiar e incentivar em todos os momentos.

Não haveria palavras suficientes para descrever o quanto meus pais me ajudaram nesta jornada. Obrigada Mãe e obrigada Pai por tudo que fizeram por mim. Obrigada ao meu irmão Leandro e minha cunhada Marina pela paciência e carinho que tornam meus finais de semana sempre perfeitos.

Serei eternamente agradecida ao meu orientador Prof. Dr. Jaime Arturo Ramírez por ter me aceitado como orientanda, pelas sua presença constante em todas as fases deste trabalho, pela sua atenção, dedicação e paciência. Seus ensinamentos e conselhos eu levarei por toda a vida.

Por suas grandes contribuições, agradeço ao Prof. Dr. Felipe Campelo e André Maravilha.

Gostaria de agradecer também aos professores do Departamento de Engenharia Elétrica que também fizeram parte desta caminhada: Prof. Dr. Rodney Rezende Saldanha, Prof. Dr. João Antônio Vasconcelos e Prof. Dr. Oriane Magela Neto. Não poderia esquecer das secretárias Anete e Arlete e dos meus amigos do GOPAC, principalmente meus "filhos"do coração Bruno e Rodrigo.

Aos membros da banca, agradeço por suas importantes contribuições e sugestões.

Meu muito obrigada a todos do Departamento de Computação da UFVJM que me ajudaram nesta trajetória assumindo mes encargos didáticos. Em especial aos amigos Claudia, Cinthya, Cristiano, Oscar e Alan.

Não poderia deixar de mencionar minhas amigas irmãs queridas Chrys e Elaine por

estarem sempre comigo mesmo em pensamento.

Por fim, gostaria de agradecer ao Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão de bolsa de estudo por meio do Programa de Fomento à Pós-Graduação (PROF).

Resumo

Os problemas de roteamento de veículos com serviços de coleta e entrega constituem um problema básico de logística reversa. Em geral, este problema consiste em definir rotas de menor custo que satisfaçam a todas as demandas de coleta e entrega dos consumidores, respeitando a capacidade limitada dos veículos. Entretanto, em diversas situações reais, esses problemas não se limitam a apenas um único objetivo, havendo outros critérios de desempenho que precisam ser avaliados, tais como satisfação dos clientes, satisfação dos funcionários ou atendimento da legislação vigente.

É neste contexto que se inserem as contribuições deste trabalho, propondo uma abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega. No modelo matemático apresentado, as demandas de coleta não são mais obrigatórias e passam a ser desejáveis. Logo, os objetivos são a minimização do custo de transporte e a maximização do atendimento das demandas de coleta.

Devido à complexidade computacional do problema, quatro algoritmos heurísticos foram implementados: MOGVNS, NSGA-II, P_ϵ e IBMOLS que exploram as especificidades das metaheurísticas de busca local, de população e de métodos tradicionais de resolução de problemas multiobjetivo como o ϵ-Restrito.

A qualidade dos algoritmos foi avaliada em 2 conjuntos de instâncias de teste da literatura. O primeiro conjunto de instâncias é composto por 40 problemas teste contendo 50 consumidores. O outro é composto por 14 problemas teste no qual o número de consumidores varia entre 50 a 200. A análise em diferentes grupos de instâncias permitiu qualificar os algoritmos em problemas de tamanhos variados. As soluções obtidas foram avaliadas segundo as métricas hipervolume, cardinalidade, tempo de execução e cobertura.

Após análise estatística de cada uma das métricas, os testes de hipótese demostraram diferenças significativas entre os algoritmos, apontando uma superioridade do MOGVNS em relação aos algoritmos NSGA-II e IBMOLS. Analisando o hipervolume

e cobertura, as soluções retornadas pelo MOGVNS estão mais próximas da fronteira Pareto ótima e melhores distribuídas, para ambos conjuntos de instâncias.

Dessa forma, uma análise mais detalhada foi realizada em torno do MOGVNS, verificando a qualidade da solução em um dos pontos da fronteira, no qual é possível comparar com resultados da literatura. Assim, o MOGVNS alcançou bons resultados em relação a solução encontrada para o Problema de Roteamento de Veículos com Coleta e Entrega Simultânea que é um ponto extremo da fronteira, no qual todas as demandas de coletas são atendidas.

Além disso, a fronteira Pareto ótima foi encontrada para duas instâncias com menor número de consumidores e para as demais foi definido um limite inferior. Os resultados apresentados mostraram que o MOGVNS alcança a solução ótima em grande parte dos pontos da fronteira e pontos mais próximos do limite inferior.

Durante a geração dos limites inferiores, observou-se que a diferença entre os limites inferior e superior, apresentado pelo resolvedor, foi muito elevado mesmo com um tempo computacional limitado em 5 dias. Estes dados reforçam a necessidade de estudo e aplicação de métodos heurísticos eficientes para encontrar uma solução para o problema em um tempo viável.

Por fim, os resultados também comprovaram a importância de uma abordagem multiobjetivo para o problema tratado. De modo geral, os problemas nos quais a soma das demandas de coleta é menor que a soma das demanas de entrega, a diferença no custo de transporte entre os extremos da fronteira é muito pequena, ou seja, a diferença no custo de transporte gasto para efetuar todas as coletas e não efetuá-las é quase insignificante. De posse destas informações e de gráficos da fronteira que permitam visualizar todas as soluções passíveis de serem aplicadas, o tomador de decisões poderá adotar soluções mais eficientes e adequadas à realidade.

Abstract

The vehicle routing problem with pickup and delivery services is a basic problem of reverse logistics. In general, this problem consists in defining lowest-cost routes that satisfy all consumer demands of collection and delivery, and meet the limited capacity of the vehicles. However, in many real situations, this problem is not limited to just a single goal, but there are others performance criteria that need to be evaluated, such as customer satisfaction, employee satisfaction or compliance with current legislation.

In this context, the main contributions of this thesis is an approach for multiobjective vehicle routing problem with pickup and delivery services. In the mathematical model, the fulfillment of pickup demands are not mandatory, but highly desirable. Therefore, the objectives are the minimization of transportation costs and the maximization of pickup demands satisfied.

Due to the computational complexity of the problem, four heuristic algorithms are implemented: MOGVNS, NSGA-II, P_ϵ and IBMOLS. These algorithms exploit the particularities of the metaheuristics based on local search and population, and the traditional techniques for solving multiobjective problems such as ϵ-Restricted method.

The algorithms quality is assessed in 2 sets of known instances. The first one is composed by 40 test problems with 50 customers. The other one is composed by 14 test problems with 50 to 200 customers. The analysis in different groups of instances allow to qualified the algorithms for instances of different sizes. The solutions obtained is evaluate in terms of hypervolume, cardinality, execution time and coverage metrics.

After statistical analysis, the hypothesis tests demonstrated significant differences between the algorithms, indicating that the MOGVNS outperformed the NSGA-II and IBMOLS algorithms. Analysing the hypervolume and coverage metrics, the solutions found by MOGVNS are closer to the Pareto-optimal frontier and/or better distributed, for both sets of instances.

Thus, further analysis were performed with the MOGVNS to verify the solution qua-

lityin one end point, which can be compared with literature results. The MOGVNS algorithm achieved good results compared to the Vehicle Routing Problem with Simultaneous Pickup and Delivery solution, which is an approximation of the frontier extreme point where all pickup demands are fulfilled.

Furthermore, we have found the Pareto optimal frontier for the two instances with the smaller number of consumers while for the other instances a lower bound was defined. The results showed that MOGVNS returned the optimal solution in the most frontier points and points closer to the lower bound.

During the lower bound generation, it was observed that the difference between the lower and upper bounds presented by the solver was very high even limiting the computational time to 5 days. These data emphasize the need to use efficient heuristics to find a solution to the problem in a feasible time.

Finally, the results also confirm the importance of a multiobjective approach to the reported problem. In general, problems in which the sum of the demands collection is less than the sum of demands delivery, the difference in transport cost between the ends of the frontier is very small, i.e., the difference in transport cost to perform all pickup and not perform them is almost negligible. This information combined with the representation of the Pareto optimal frontier, that allows the visualization of all solutions that would be applied, the decision maker can adopt more efficient solutions and better suited to reality.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

Problemas

VRP	-	Vehicle Routing Problem
$CVRP$	-	Capacitated Vehicle Routing Problem
$VRPB$	-	Vehicle Routing Problem with Backhauls
$VRPPD$	-	Vehicle Routing Problem with Pickup and Delivery
$VRPCB$	-	Vehicle Routing Problem with Clustered Backhauls
$VRPMB$	-	Vehicle Routing Problem with Mixed Linehauls and Backhauls
$VRPDDP$	-	Vehicle Routing Problem with Divisible Delivery and Pickup
$VRPSPD$	-	Vehicle Routing Problem with Simultaneous Pickup and Delivery
$MOVRPOP$	-	Multiobjective Vehicle Routing Problem with Optional Pickup
$OVRP$	-	Open Vechicle Routing Problem
$DVRP$	-	Dinamic Vehicle Routing problem
$PVRP$	-	Periodic Vehicle Routing Problem
TSP	-	Traveling Salesman Problem
$TSPMB$	-	Traveling Salesman Problem with Mixed Linehauls and Backhauls
$TSPCB$	-	Traveling Salesman Problem with Clustered Backhauls
$TSPDDP$	-	Traveling Salesman Problem with Divisible Delivery and Pickup
$TSPSPD$	-	Traveling Salesman Problem with Simultaneous Pickup and Delivery
TTP	-	Traveling Purchaser Problem
$TSPP$	-	Traveling Salesman Problem with Profit
$GPDP$	-	General Pickup and Delivery Problem
$SPDP$	-	Selective Pickup and Delivery Problem
$DARP$	-	Dial-a-Ride Problem
CTP	-	Covering Tour Problem
ARP	-	Arc Routing Problem

Métodos

ILS	-	Iterated Local Search
$MOILS$	-	Multiobjective Iterated Local Search
VNS	-	Variable Neighborhood Search
VND	-	Variable Neighborhood Descent
$GVNS$	-	General Variable Neighborhood Search
$RandomVND$	-	Rondom Variable Neighborhood Descent
$MOGVNS$	-	Multiobjective General Variable Neighborhood Search
TS	-	Tabu Search
$GRASP$	-	Greedy Randomized Adaptive Search Procedure
SA	-	Simulated Annealing
SS	-	Scatter Search
ACO	-	Ant Colony Optimization
PSO	-	Particle Swarm Optimization
DE	-	Diferencial Evolution
$CLONALG$	-	Clonal Selection Algorithm
MA	-	Memetic Algorithms
GA	-	Genetic Algorithms
$VEGA$	-	Vector Evaluated Genetic Algorithm
$MOGA$	-	Multiobjective Genetic Algorithm
$NPGA$	-	Niched Pareto Genetic Algorithm
$SPEA$	-	Strength Pareto Evolutionary Algorithm
$NSGA$	-	Non-Dominated Sorting Genetic Algorithm
Heurística P_ϵ	-	Heurística baseada no método ϵ-Restrito

$GLPK$	-	GNU Linear Programming Kit

Lista de Símbolos

$\mathbf{x}$	-	Vetor de variáveis de decisão
$\mathbf{f}(.)$	-	Vetor de funções objetivo
$\mathbf{g}(.)$	-	Vetor de restrições de desigualdades
$\mathbf{h}(.)$	-	Vetor de restrições de igualdade
m	-	Número de funções objetivo
n	-	Número de variáveis de otimização
p	-	Número de restrições de desigualdade
q	-	Número de restrições de igualdade
$\mathcal{X}$	-	Espaço de decisões
$\mathcal{F}$	-	Conjunto factível
$\mathcal{Y}$	-	Espaço dos objetivos

n	-	Número total de consumidores
A	-	Conjunto de arestas ou conexões entre consumidores e entre consumidores e depósito
V	-	Conjunto de vértices ou conjunto dos consumidores e depósito
v_0	-	Vértice indicando o depósito
c_{ij}	-	Valor associado a aresta (i,j)
d_i	-	Demanda de entrega do consumidor i
p_i	-	Demanda de coleta do consumidor i
k_{max}	-	Número máximo de veículos disponível
Q	-	Capacidade máxima do veículos
x_{ij}	-	Variável de decisão indicando se aresta (v_i, v_j) pertence a solução do VRP
ℓ_j	-	Variável de decisão indicando se demanda de coleta do consumidor v_j é atendida
y_{ij}	-	Variável de fluxo indicando o somatório das cargas coletadas entre o depósito e o nó i (inclusive) dirigida ao nó j
z_{ij}	-	Variável de fluxo indicando o somatório das cargas entregues entre o nó i (exclusive) e o depósito dirigida ao nó j

Capítulo 1

Introdução

Os problemas de roteamento de veículos é um dos mais importantes e mais amplamente estudados no contexto da otimização combinatória por englobar grande parte dos problemas reais de logística de transporte. Em geral, este problema consiste em definir rotas entre um depósito e um conjunto de consumidores considerando uma frota finita de veículos capacitados. As rotas devem satisfazer as demandas dos consumidores sem extrapolar a capacidade dos veículos [Toth e Vigo, 2002].

Uma vez que existe um custo de transporte associado entre pares de consumidores e entre os consumidores e o depósito, o objetivo é definir rotas de menor custo. Este problema é caracterizado por uma natureza combinatória, ou seja, o domínio da função no qual se pretende otimizar é finito e a solução ótima é dada pela permutação do conjunto de arestas que conectam os pares de consumidores e também os consumidores ao depósito. Como o número de consumidores pode aumentar indefinidamente, acarretando também o aumento do número de arestas, encontrar soluções ótimas em algumas instâncias do problema pode ser extremamente difícil [Dantzig e Ramser, 1959]. Um dos ramos da ciência que estuda problemas com estes aspectos é a otimização combinatória [Papadimitriou e Steiglitz, 1982].

Este trabalho apresenta uma abordagem multiobjetivo para solucionar problemas de roteamento de veículos que possuem serviços de coleta e entrega [Deb, 2001]. Nesta variação do problema, a função objetivo é composta por duas ou mais funções e as demandas dos consumidores podem ser de coleta e/ou entrega. Como não se conhece um método eficiente para encontrar soluções ótimas para o problema abordado em tempo polinomial, alguns métodos heurísticos são adotados como as metaheurísticas baseadas em buscas locais e algoritmos evolucionários [Glover e Kochenberger, 2003].

Este capítulo tem por objetivo contextualizar o problema e se divide da seguinte

maneira. A motivação para o estudo é apresentada na seção 1.1. Em seguida, algumas aplicações para o problema são descritas na seção 1.2. Os objetivos alcançados na tese são apresentados na seção 1.3. Nas seções 1.4 e 1.5 encontram-se a metodologia adotada e as contribuições do trabalho, respectivamente. Por fim, a estrutura geral do texto é apresentada na seção 1.6.

1.1 Motivação

"*Sob qualquer ponto de vista: econômico, político e militar; o transporte é, inquestionavelmente, a indústria mais importante do mundo*"[Menchik, 2010]. As empresas e indústrias estão cada vez mais preocupadas em propiciar, com custo mínimo, melhor serviço e escoamento das suas mercadorias. O custo de escoamento tem influência direta no custo final de qualquer tipo de produto e pode ser a chave do sucesso ou fracasso de uma empresa. Em 1927, Henry Ford já afirmava que "*o limite real de uma empresa é o transporte*"[Ford, 1927].

A importância do transporte para as empresas está relacionada tanto com a satisfação do cliente, pois eles exigem um atendimento rápido e em prazos pré-determinados, quanto a redução de custos. Segundo os dados mais recentes apresentados pela Associação Brasileira de Logística [2008], os custos de transporte correspondem a 60% dos custos logísticos das empresas e, em 2008, os custos de transporte representaram 6,9% do Produto Interno Bruto (R$ 207,0 bilhões) do Brasil.

Boa parte destes números se deve ao modal de transporte que comumente é utilizado para transporte de carga no Brasil, o modal rodoviário. Em média, as grandes empresas transportam 88,3% de suas cargas por rodovia, segundo a Centro de Estudos em Logística - CEL/COPPEAD [2007]. Esse uso excessivo das rodovias encarece o transporte, já que o modal rodoviário é o mais caro, após o aéreo.

O elevado custo na utilização do modal rodoviário se deve principalmente: (i) ao aumento no preço dos combustíveis. No Brasil, o diesel é o combustível utilizado na maioria dos veículos de transporte de cargas rodoviário. O aumento do preço deste combustível acarreta no aumento dos fretes já que os gastos com diesel interfere diretamente nas tarifas de transporte, 35% de acordo com o Confederação Nacional de Transportes [2013a]. O ajuste acumulado do diesel nos últimos doze meses foi de aproximadamente 12%, segundo a Associação Nacional de Petróleo [2013]; (ii) às condições em que se encontram as rodovias. Segundo a Confederação Nacional de Transportes [2013b], a malha rodoviária brasileira tem atualmente uma extensão aproximada de 1,5 milhões de kilômetros, com pouco mais de 200 mil kilômetros de pistas pavimenta-

das (13,4% da extensão total). Um um estudo realizado pela CNT, em 2012, mostra que 60% das rodovias pavimentadas apresentam alguma deficiência no pavimento, na sinalização e/ou na geometria da via. Esse cenário compromete a qualidade e a segurança dos fluxos de carga e de pessoas. Um exemplo deste comprometimento está na perda da safra de grãos. A Empresa Brasileira de Pesquisa Agropecuária [2009] aponta que as condições das rodovias provocam uma perda de até 6% da safra e as rodovias em mal estado aumentam, em média, 46% o custo operacional dos veículos. Além do baixo índice de pavimentação da malha rodoviária do país, observa-se um elevado grau de deterioração das poucas estradas pavimentadas, o que compromete todo o sistema logístico.

Devido a tantos custos no setor de transporte, 74% das grandes empresas industriais têm priorizado a Tecnologia de Informação (TI) a fim de obter um maior controle, organização e integração das atividades relativas à área de transporte [Centro de Estudos em Logística - CEL/COPPEAD, 2007]. Atualmente existem diversas tecnologias para gerenciamento do setor de transporte dentre eles: rastreamento de veículos, gerenciamento de custos de transporte, informação do status da carga para o cliente, roteirização, montagem de carga e auditoria de frete. Pelo menos 89% das grandes empresas utilizam alguma destas tecnologias.

A roteirização, apesar de sua grande importância para redução dos custos logísticos, ainda é pouco utilizada, sendo que apenas 30% das empresas possuem algum tipo de roteirizador [Associação Brasileira de Logística, 2008]. Os sistemas de roteirizadores oferecem, dentre outros serviços, o trajeto a ser percorrido pelos veículos. Encontrar um trajeto ótimo entre um depósito e um conjunto de pontos de entrega, dada uma frota de veículos, pode ser denominado como um Problema de Roteamento de Veículos.

Alguns problemas de roteamento encontrados na literatura envolvem tanto serviço de entrega quanto coleta de mercadorias. Estudos envolvendo problemas de coleta e entrega têm crescido nos últimos anos por se tratar de um problema básico da logística reversa. As empresas vêm buscando formas eficientes de gerenciar o retorno de materiais após sua venda e consumo para reciclá-los, remanufaturá-los ou reaproveitá-los, agregando novos valores aos mesmos. As vantagens dessa prática são: redução de custo, redução dos danos ao meio ambiente e garantia do direito do consumidor de devolução e troca de mercadorias, entre outros.

Os materiais retornáveis podem ser divididos em: produto e embalagem. O fluxo de logística reversa do produto se dá pelo reparo ou simplesmente pela insatisfação do cliente. Em alguns casos como na venda por catálogo ou na venda pela internet, o

gerenciamento do fluxo reverso é fundamental para o negócio.

O fluxo da logística reversa nas embalagens acontece, em geral, para a sua reciclagem. Em alguns casos o retorno das embalagens é feito para atender a alguma legislação específica, como, por exemplo, nas indústrias de agrotóxicos que são obrigadas a coletar as embalagens dos produtos já utilizados.

Em diversos casos, a reciclagem está relacionada a fatores econômicos. Atualmente, o Brasil é o país que possui o maior índice de reciclagem de embalagens de alumínio do mundo. Segundo a Associação Brasileira do Alumínio, em 2009 foram reaproveitadas 92,2% de todas as latas consumidas. Aproximadamente 9 bilhões de unidades foram reaproveitadas pela indústria de alumínio, reduzindo os custos em R$ 850 milhões e gerando 152 mil empregos diretos e indiretos. Além da redução de capital, essa reciclagem proporciona uma economia de 1,7 mil GW hora/ano, correspondendo a 0,5% de toda a energia gerada no país e suficiente para abastecer a cidade de Campinas, com 1 milhão de habitantes. Isso leva a indústria de alumínio a efetuar tanto entrega de mercadorias quanto, também, a coleta das embalagens para reciclagem.

Além da reciclagem de alumínio diversos outros materiais tem sido reciclados como: papéis, plásticos, vidros, metais, borracha. Além dos ganhos financeiros, a vantagem de reaproveitamento das embalagens está ligada a imagem institucional da empresa. Diversas empresas estão valorizando a reciclagem como meio de atrair a atenção e preferência de clientes e consumidores finais por apresentar uma postura ecologicamente correta.

Existem situações reais onde a realização da coleta não é obrigatória, permitindo a escolha de quais coletas realizar. Este problema é conhecido como problema de roteamento de veículos com coletas opcionais. Um exemplo desta situação pode ser observado nas indústrias de agrotóxicos. No Brasil, os usuários têm um prazo de um ano para devolver as embalagens vazias para os estabelecimentos comerciais. As empresas produtoras e comercializadoras de agrotóxicos têm um prazo de um ano para coleta e destinação das embalagens [BRASIL, 2002]. Com isso, não é necessário atender a todas demandas, já que tem-se o prazo de um ano para efetuar a coleta. Apesar da realização das demandas de coleta não serem obrigatórias, a sua realização é desejada. Portanto, deve-se maximizar o número de coletas realizadas.

Em diversos trabalhos da literatura, onde a coleta não é obrigatória, o problema é tratado como mono-objetivo. Na maioria destes trabalhos a função objetivo é dada pelo custo das rotas menos o lucro gerado pelas coletas realizadas [Gribkovskaia et al., 2007, 2008; Bruck et al., 2012; Gutiérrez-Jarpa et al., 2010]. Entretanto, as características

do problema possibilitam a elaboração de uma modelagem multiobjetivo onde, além da minimização do custo da rota, deve-se maximizar o número de coletas realizadas.

Assim, considerando o exemplo das indústrias de agrotóxicos apresentado anteriormente, diversos cenários são retornados ao tomador de decisões que poderá optar por efetuar coletas de acordo com o prazo estabelecido pelo decreto que define o tempo máximo para realização desta tarefa.

A busca de um método eficiente para definir tais rotas viáveis e otimizadas que satisfaçam as demandas dos consumidores pode reduzir consideravelmente o custo das grandes empresas e grandes indústrias, além de ser um grande desafio para os pesquisadores por se tratar de um problema no qual os algoritmos exatos não fornecem uma solução em tempo computacional viável.

Assim sendo, o objetivo deste trabalho é apresentar uma abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega. Apesar de muitos problemas reais desta natureza possuírem características multiobjetivo, esta abordagem ainda é pouco estudada dentre os problemas de roteamento de veículos [Boffey et al., 1995; Jozefowiez et al., 2008a] .

1.2 Aplicações

No contexto da logística reversa, no Brasil, é fácil encontrar problemas reais de roteamento de veículos com serviço de coleta e entrega. As indústrias de bebidas, por exemplo, já utilizam há muitas décadas garrafas retornáveis. Somente a partir da década de 70 é que essas indústrias iniciaram a fabricação de garrafas descartáveis. Dessa forma, o leite, cerveja e refrigerantes eram somente comercializados em garrafas retornáveis, o que levava as indústrias à prática de entrega de mercadorias e coleta de embalagens.

Outro exemplo mais atual pode ser encontrado nas indústrias de agrotóxicos, como mencionado no item anterior. Estas devem, por lei, recolher os vasilhames de seus produtos após o consumo. No Brasil, segundo o Decreto Nº 4074, de 4 de janeiro de 2002 [BRASIL, 2002], os usuários de agrotóxicos e afins devem efetuar a devolução das embalagens vazias aos estabelecimentos comerciais em que foram adquiridos no prazo de um ano, contando da data de sua compra. O Decreto estabelece ainda que, as empresas produtoras e comercializadoras de agrotóxicos, seus componentes e afins, são responsáveis pelo recolhimento, transporte e destinação final das embalagens vazias, devolvidas pelos usuários aos estabelecimentos comerciais ou aos postos de recebimento.

Estas empresas tem um prazo máximo de um ano, a contar da data de devolução pelos usuários, para recolhimento e destinação final das embalagens.

Tanto no problema encontrado no setor de transporte de bebidas quanto nas indústrias de agrotóxicos, muitas vezes a empresa não precisa necessariamente efetuar todas as demandas de coleta. Ou seja, algumas delas podem ser efetuadas em outro momento mais oportuno. Isso caracteriza o problema de roteamento de veículos com coleta opcionais.

Na literatura, existem poucos trabalhos que abordam problemas reais com essas características. Em dois deles, os autores Süral e Bookbinder [2003] e Gribkovskaia et al. [2008] estudam o caso da empresa UPS (*United Parcel Service of America Inc.*), que oferece serviços de logística e de empresas que realizam a entrega de bebidas para lojas e recolhem os recipientes vazios e latas de alumínio para reutilização.

Uma outra situação particular é o comércio eletrônico. No Brasil, segundo o Art. 49 do código de defesa do consumidor [BRASIL, 1990]:

> "*O consumidor pode desistir do contrato, no prazo de 7 dias a contar de sua assinatura ou do ato de recebimento do produto ou serviço, sempre que a contratação de fornecimento de produtos e serviços ocorrer fora do estabelecimento comercial, especialmente por telefone ou a domicílio*".

Além disso, devido ao serviço de garantia, o produto, mesmo após a sua utilização, em algumas situações, podem ser devolvidos, substituídos por outro ou reparados. Em última análise, um produto após o seu uso poderá ser recolhido para ser remanufaturado, reciclado ou mesmo incinerado, descartando adequadamente o material, sem agredir o meio ambiente.

Existem outras aplicações envolvendo problemas de coleta e entrega que não estão relacionados com logística reversa. Os autores Toth e Vigo [1996] apresentam o caso real de uma empresa de transporte de passageiros em Bologna na Itália. Neste caso, a empresa coleta e entrega passageiros, sendo necessário atentar-se pela qualidade de serviço oferecido para que os passageiros não fiquem aguardando demasiado tempo para embarcar e também durante a sua viagem.

Outra aplicação é abordada por Ji e Chen [2007] que apresentam uma abordagem multiobjetivo para problemas de entregas postais em Hong-Kong. As empresas de entregas postais devem efetuar a coleta das correspondências/encomendas nas agências e entregá-las aos seus destinatários.

1.3 Objetivos

O objetivo principal deste trabalho é propor uma abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega e estudar ferramentas capazes de solucioná-lo de maneira satisfatória. O problemas abordado é composto por dois ou mais objetivos conflitantes como, por exemplo, a minimização dos custos de trasporte e maximização do número de demandas de coletas atendidas, considerando um problema no qual as coletas não são obrigatórias, mas desejadas.

É importante ressaltar que neste trabalho a resolução do problema abordado consiste em um conjunto de soluções não-dominadas. Estas soluções representam diversos cenários onde algumas demandas de coletas serão atendidas e outras não. Assim sendo, caberá ao tomador de decisões escolher qual solução melhor se ajusta a realidade atual.

Em resumo, os objetivos específicos do trabalho são descritos como:

- Estudar os problemas de roteamento de veículos com serviços de coleta e entrega mono-objetivo e problemas de roteamento de veículos multiobjetivo (capítulo 3).
- Propor uma formulação matemática para problemas de roteamento de veículos multi-objetivo que envolvem serviços de coleta e entrega (capítulo 4).
- Implementar algoritmos heurísticos multiobjetivo baseados em buscas locais e baseados em algoritmos evolutivos multiobjetivo (capítulo 5).
- Validar os resultados obtidos em ambas abordagens em instâncias que representem diferentes cenários de testes (capítulo 6).

1.4 Metodologia

O problema de roteamento de veículos com coleta e entrega e múltiplos objetivos é ainda pouco explorado na literatura e, na maioria das vezes, pondera-se os objetivos resultando em uma abordagem mono-objetivo para solucioná-lo. A metodologia adotada neste trabalho se baseia na definição de modelos matemáticos mais adequados e criteriosos, compostos por múltiplas funções-objetivo.

Devido à natureza combinatória dos problemas de roteamento de veículos, algoritmos exatos são inapropriados para solucionar problemas de grandes proporções em tempo computacional viável. Assim, para verificar qual técnica melhor se adequa ao problema tratado, um conjunto de heurísticas são propostas baseadas em diferentes metaheurísticas, tais como Busca Geral em Vizinhança Variável Multiobjetivo (MOGVNS, do inglês *Multiobjective General Variable Neighborhood Search*)[Assis et al., 2013a], Busca Local Multiobjetivo Baseada em Indicador (IBMOLS, do inglês *Indicator-Based Multiobjective Local Search*) [Basseur et al., 2011],

Algoritmo Genético de Ordenação Não-Dominante-II (NSGA-II, do inglês *Non-Dominated Sorting Genetic Algorithm*)[Deb et al., 2002], além de uma heurística que se baseia em um método tradicional de resolução de problemas multiobjetivo (ϵ-Restrito) [Chankong e Haimes, 1983]. Os algoritmos implementados foram escolhidos por terem características bem distintas sendo, assim possível avaliar qual ou quais métodos retornam um conjunto de soluções mais satisfatório.

Os algoritmos apresentados foram aplicados a um conjunto de 54 instâncias de testes encontradas na literatura que simulam diferentes cenários do problema de roteamento de veículos com serviços de coleta e entrega [Salhi e Nagy, 1999; Dethloff, 2001]. Os resultados obtidos foram avaliados considerando um conjunto de métricas capazes de mensurar a qualidade das soluções em termos de diversidade e proximidade da fronteira Pareto ótima [Takahashi, 2004].

Após simulação dos resultados e obtenção dos valores das métricas, realizou-se uma análise estatística a fim de conferir se há diferenças significativas entre os métodos assim como quantificar estas diferenças.

Assim sendo, após definido o melhor método heurístico, os resultados obtidos pelas instâncias com menor número de consumidores foram comparados aos resultados obtidos com um método exato, no qual foi utilizado o resolvedor Gurobi [Gurobi Optimization, 2013] para solucionar subproblemas gerados pelo método ϵ-Restrito [Chankong e Haimes, 1983] . Para as demais instâncias o limite inferior foi utilizado para aferir a qualidade dos algoritmos heurísticos propostos.

1.5 Contribuições

Os problemas de roteamento de veículos com serviços de coleta e entrega têm sido amplamente estudados nos últimos anos. Em geral, os autores têm focado no desenvolvimento de algoritmos mais rápidos e precisos para solucionar o problema. Este trabalho contribui no desenvolvimento de modelos mais realistas e algoritmos adequados para solucioná-los.

Após uma extensa revisão da literatura, no qual foram investigados diferentes problemas de roteamento de veículos com serviços de coleta e entrega, assim como os diferentes problemas de roteamento de veículo multiobjetivo, foi possível identificar uma escassez de trabalhos que confluem estas duas variações de problemas de roteamento. No estudo realizado, percebe-se que muitos desses problemas que envolvem serviços de coleta e entrega apresentam uma natureza multiobjetivo e requerem a avaliação de múltiplos critérios de desempenho e esta necessidade é ainda pouco explorada.

Esta tese, então, propõe um modelo matemático bi-objetivo que representa uma generalização do problema de roteamento de veículos coleta e entrega simultânea no qual uma das

restrições do modelo referente a obrigatoriedade do atendimento das demandas de coleta é transformada em um objetivo a ser alcançado. O modelo proposto pode melhor representar situações reais no qual o atendimento das demandas de coleta deixa de ser obrigatório e se torna desejável. Dessa forma, será possível analisar diferentes cenários: alguns com um número maior de demandas de coletas sendo atendidas, mas com um custo de transporte elevado e outras com um custo de transporte reduzido, mas contendo um número menor de demandas de coletas atendidas. Assim sendo, não há uma solução admissível que seja melhor em ambos os objetivos. Este conjunto de soluções, com diferentes características, permite, ao tomador de decisão, uma análise mais precisa dos resultados para então selecionar qual delas melhor se adequa ao cenário atual.

Devido à complexidade computacional dos problemas de roteamento de veículos, este trabalho investigou diversas metaheurísticas baseadas em busca local e em população, a fim de gerar algoritmos robustos capazes de solucionar o problema proposto. Os algoritmos foram comparados entre si a partir de um minunciosa análise estatística e comparando-os com a solução ótima de problemas de pequeno porte ou com o limite inferior dos demais problemas teste envolvidos na simulação.

1.6 Organização

O trabalho está estruturado da seguinte forma:

Capítulo 2: apresenta os principais conceitos, definições e notações dos problemas de otimização multiobjetivo utilizados neste trabalho, com a intenção de proporcionar um melhor entendimento do tema tratado. Este capítulo mostra ainda alguns métodos tradicionais e metaheurísticas utilizadas para solucionar estes problemas e as métricas utilizadas nesta tese para avaliar a qualidade das soluções encontradas também constituem este capítulo.

Capítulo 3: apresenta a definição de problemas de roteamento de veículos e algumas de suas variações. Este capítulo também mostra um estudo realizado dos diversos trabalhos da literatura envolvendo problemas de roteamento de veículos com serviços de coleta e entrega, e também problemas de roteamento de veículos com abordagem multiobjetivo. Este estudo possibilita a avaliação das abordagens encontradas na interseção dessas duas classes de problemas tornando possível identificar áreas pouco exploradas da literatura.

Capítulo 4: apresenta os principais trabalhos relacionados aos problemas de roteamento de veículos multiobjetivo com serviços de coleta e entrega. Após contextualizar o problema de roteamento de veículos multiobjetivo com coleta opcional, proposto nesta tese, este é formalmente definido e a formulação matemática proposta é apresentada.

Capítulo 5: apresenta as ferramentas e métodos utilizados para resolução de problemas de roteamento de veículos multiobjetivo com coleta opcional. Para solucionar o problema proposto foi necessário a utilização de algoritmos para solucionar problemas de roteamento de veículos mono-objetivo. Assim sendo, neste capítulo é descrito o algoritmo baseado na busca geral de vizinhança variável. Por fim, são apresentados os algoritmos desenvolvidos para solucionar o problema foco deste trabalho.

Capítulo 6: apresenta os resultados obtidos pelos algoritmos propostos para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, após simulação em um conjunto de 54 problemas da literatura. A avaliação da qualidade das soluções são apresentadas segundo as métricas hipervolume, cardinalidade, tempo e cobertura. Para avaliar as possíveis diferenças entre o desempenho dos métodos propostos, um planejamento experimental é apresentado possibilitando detectar diferenças significativa entre os métodos e estimar a magnitude destes. Por fim, este capítulo mostra uma comparação das soluções obtidas pelos algoritmos implementados com a fronteira Pareto ótima ou limite inferior das instâncias de teste.

Capítulo 7: apresenta uma avaliação geral do que foi desenvolvido ao longo do texto em relação ao problema tratado e as técnicas utilizadas para solucioná-lo. Alguns pontos que podem ser explorados em trabalhos futuros também são descritos.

Capítulo 2

Métodos de Otimização Multiobjetivo

Esta tese recai sobre o campo da otimização multiobjetivo que propõe métodos para solucionar problemas com dois ou mais objetivos conflitantes. A otimização multiobjetivo teve início na área da economia no século IX com os trabalhos de Edgeworth [1881] e Pareto [1896]. Posteriormente, houve um crescimento em meados da década de 80 [Steuer, 1986], mas, foi a partir da década de 90 que se encontra um maior volume de trabalhos publicados com o surgimento dos algoritmos evolucionários multiobjetivo [Srinivas e Deb, 1994; Zitzler, 1999; Deb, 2001; Coello et al., 2011].

Este capítulo aborda os principais conceitos, definições e notações utilizados neste trabalho, com a intenção de proporcionar um melhor entendimento do tema tratado. A seção 2.1 caracteriza os problemas de otimização multiobjetivo, apresentando os principais conceitos e definições destes problemas. As seções 2.2 e 2.3 apresentam métodos tradicionais e metaheurísticas, respectivamente, para solucionar problemas multiobjetivo. A seção 2.4 descreve as principais métricas utilizadas para avaliar as soluções dessa classe de problemas. Por fim, a seção 4.3 apresenta a conclusão do capítulo.

2.1 Conceitos Básicos

Em geral, os problemas multiobjetivo apresentam funções objetivo conflitantes [Deb, 2001]. Duas funções objetivo são conflitantes quando a melhora em uma só é possível através da piora na outra, assim, não existindo uma solução única que seja ótima para todos os objetivos simultaneamente.

Dado o seguinte problema de otimização multiobjetivo:

$$\begin{aligned} &\min_x \mathbf{f}(\mathbf{x}) \in \mathbb{R}^m, \mathbf{x} \in \mathcal{F} \\ &\text{sendo:} \\ &\mathbf{f} : \mathcal{X} \subset \mathbb{R}^n \mapsto \mathcal{Y} \subset \mathbb{R}^m \\ &\mathcal{F} = \begin{cases} g_i(\mathbf{x}) \geq 0, & i = 1, 2, ..., p \\ h_j(\mathbf{x}) = 0, & j = 1, 2, ..., q \\ \mathbf{x} \in \mathcal{X} \end{cases} \end{aligned} \tag{2.1}$$

O problema apresentado é composto por m funções-objetivo a serem minimizada (sem perda de generalidade considerou-se problemas de minimização). A região factível é definida por $\mathcal{F}$, composta por soluções que satisfazem p restrições de desigualdade e q restrições de igualdade. Os vetores $\mathbf{x}$ são denominados vetores de parâmetros do problema de otimização multiobjetivo (POV). Estes vetores compõem o espaço de parâmetros $\mathcal{X}$. Os vetores $\mathbf{f}(\mathbf{x})$ formam o espaço dos objetivos $\mathcal{Y}$, ou seja, a imagem da função $\mathbf{f}(\mathbf{x})$. O objetivo de um POV é determinar um conjunto de soluções que, de certa forma, otimize o vetor objetivo $\mathbf{f}(\mathbf{x})$. Logo, a dificuldade é definir quais soluções farão parte deste conjunto. Para isto, este trabalho utiliza o conceito de Dominância de Pareto (Definição 1) [Takahashi, 2004]. A figura 2.1 ilustra este conceito.

Definição 1 (**Dominância de Pareto**). *Uma solução $\boldsymbol{x}' \in \mathcal{F}$ domina outra solução $\boldsymbol{x}'' \in \mathcal{F}$ se $\boldsymbol{f}(\boldsymbol{x})' \leq \boldsymbol{f}(\boldsymbol{x})''$ e $\boldsymbol{f}(\boldsymbol{x})' \neq \boldsymbol{f}(\boldsymbol{x})''$. Esta relação de dominância é representada pela notação $\boldsymbol{x}' \prec \boldsymbol{x}''$.*

O conjunto de soluções será composto apenas por soluções não dominadas por nenhuma outra solução, conhecidas como soluções Pareto-Ótimas (Definição 2) [Takahashi, 2004]. Este conjunto de soluções Pareto-Ótimas recebe o nome de conjunto Pareto-Ótimo (Definição 3) [Takahashi, 2004]. Portanto, a solução de um problema de otimização multiobjetivo é um conjunto Pareto-Ótimo $\mathcal{Y}^* \subset \mathcal{Y}$. A figura 2.2 ilustra o mapeamento de soluções $\mathcal{X}$ em $\mathcal{Y}$. De posse deste conjunto, o tomador de decisões poderá escolher a solução mais efetiva à sua disposição para alcançar os objetivos pretendidos.

Definição 2 (**Solução Pareto-Ótima**). *$\boldsymbol{x}^* \in \mathcal{F}$ é uma solução Pareto-Ótima se não existe qualquer outra solução $\boldsymbol{x} \in \mathcal{F}$ tal que $\boldsymbol{x} \prec \boldsymbol{x}^*$, ou seja, $\boldsymbol{x}^*$ não é dominada por nenhuma outra solução viável.*

Definição 3 (**Conjunto Pareto-Ótimo**). *O conjunto $\mathcal{X}^* \subset \mathcal{X}$ é um conjunto Pareto-Ótimo se é composto por todas as soluções Pareto-Ótimas. O conjunto-imagem $\mathcal{Y}^* \subset \mathcal{Y}$ associado ao conjunto Pareto-Ótimo é denominado fronteira Pareto-Ótima.*

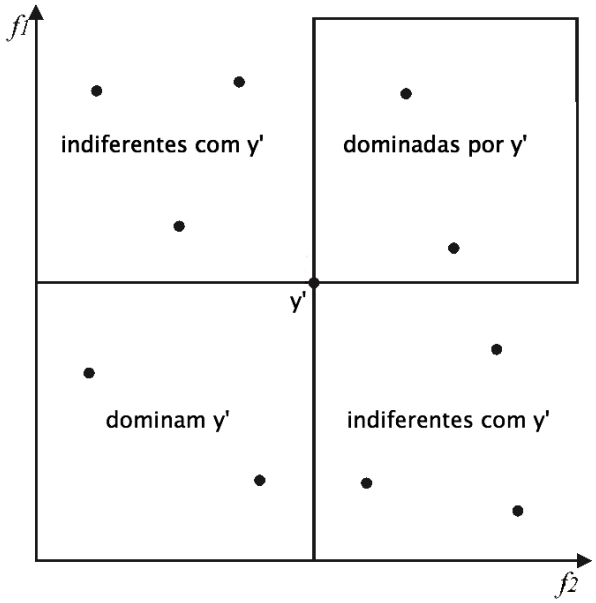


Figura 2.1: Relação de dominância: Dado duas funções objetivo para minimização f_1 e f_2 e um ponto no espaço objetivo obtido por uma solução **y'**. O espaço objetivo é dividido em quatro áreas a partir do ponto gerado por **y'**, onde pode ser observado a relação de dominância entre esta solução e as demais [Zitzler, 1999].

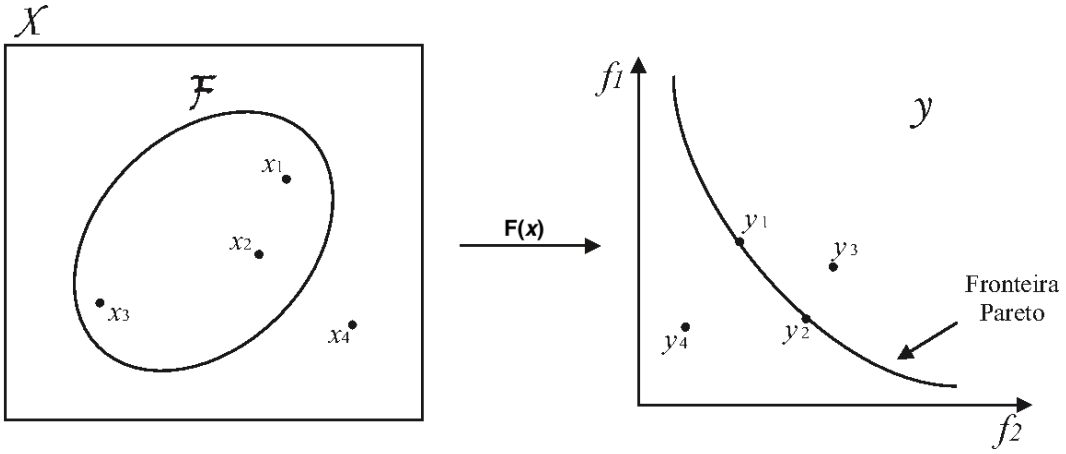


Figura 2.2: Mapeamento de soluções em um problema de otimização combinatória multiobjetivo: Mapeamento do espaço de decisões $\mathcal{X}$ no espaço objetivo (conjunto-imagem) $\mathcal{Y}$ feita pela função $\mathbf{f}(\mathbf{x})$. A região de soluções viáveis é dada por $\mathcal{F}$. Cada solução x_i tem seu respectivo ponto y_i no espaço objetivo. Apesar da solução x_4 resultar em um ponto y_4 aparentemente melhor que os obtidos pelas soluções x_1 e x_2, ela não pertence ao conjunto de soluções viáveis ($x_4 \notin \mathcal{F}$).

Encontrar o conjunto Pareto-Ótimo para alguns problema pode ser computacionalmente intratável [Deb, 2001]. Desta forma, a abordagem multiobjetivo é caracterizada pelo desenvolvimento de técnicas para tentar encontrar um conjunto de soluções viáveis que representam

pontos próximos à fronteira Pareto-Ótima ou um conjunto de soluções localmente pareto-ótima (Definição 4) [Takahashi, 2004]. A figura 2.3 ilustra uma fronteira Pareto aproximada.

Definição 4 (**Solução Localmente Pareto-Ótima**). *Dada uma estrutura de vizinhança $\mathcal{N}$ (Definição 5), uma solução factível $\boldsymbol{x}$ é localmente Pareto-Ótima de $\mathcal{N}$ se, e somente se, não existe nenhuma solução $\boldsymbol{x'} \in \mathcal{N}$ tal que $\boldsymbol{f}(\boldsymbol{x'}) \prec \boldsymbol{f}(\boldsymbol{x})$.*

Definição 5 (**Estrutura de Vizinhança**). *Dada uma solução viável x, uma estrutura de vizinhança $\mathcal{N}(x)$ gera um conjunto de soluções viáveis, denominadas vizinhos da solução x [Lust, 2010].*

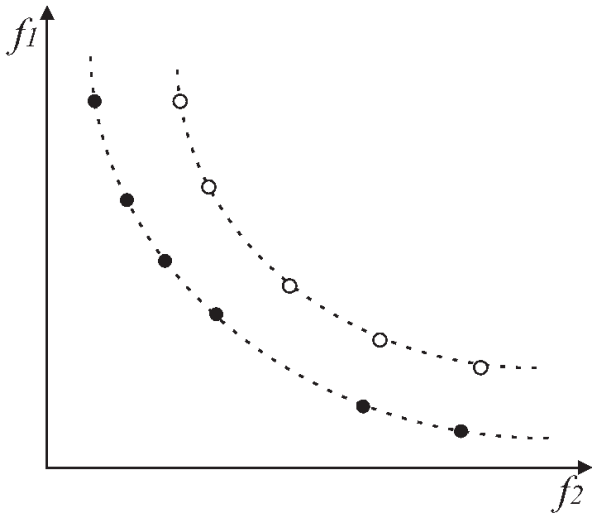


Figura 2.3: Fronteira Pareto: Os pontos preenchidos representam a fronteira Pareto-Ótima. Já os pontos não preenchidos formam uma aproximação à fronteira Pareto-Ótima. Este conjunto aproximado é formado por soluções não dominadas entre si, porém existem pontos que não fazem parte da fronteira Pareto-Ótima.

2.2 Metodologia Tradicional de Otimização Multiobjetivo

Existem diversas abordagens para geração de soluções eficientes em um problema de otimização multiobjetivo. Uma das formas de solucionar um problema multiobjetivo é através da sua redução para uma sequência de problemas de otimização mono-objetivo.

Um método bastante comum para solucionar problemas com múltiplos objetivos consiste na transformação deste problema em diversos problemas com um único objetivo, onde as soluções ótimas destes problemas representam as soluções da fronteira Pareto. Estes métodos definem um conjunto de problemas parametrizados de diferentes formas, sendo que cada

problema é composto por um único objetivo. Os problemas deste conjunto são, então, solucionados como sendo mono-objetivo. A escolha dos valores na qual se parametriza estes problemas irá determinar quais soluções do conjunto Pareto serão alcançadas. Dessa forma, a tarefa de escolher essa sequência de valores de parâmetros é extremamente difícil. Em muitos casos é impossível definir valores tais que toda a fronteira Pareto seja encontrada. Um método muito encontrado na literatura é baseado na agregação dos diferentes objetivos do problema através da soma ponderada [Takahashi, 2004].

Outro método tradicional para gerar a fronteira Pareto foi proposto por Chankong e Haimes [1983], denominado ϵ-Restrito. Este método é baseado na escalarização no qual um dos objetivos é minimizado enquanto os demais objetivos são limitados por meio de restrições. As soluções pertencentes à fronteira Pareto são obtidas através de uma variação sistemática do limite dessas restrições.

Teorema 1. *Seja k o número de objetivos para determinado problema multiobjetivo, se $\boldsymbol{x}^* \in \mathcal{F}$ é eficiente então existem um inteiro $i \in \{1, 2, ..., m\}$ e número reais ϵ_j, $j = 1, ..., m$ $(j \neq i)$ tais que $\boldsymbol{x}^*$ resolve [Takahashi, 2004]:*

$$
\begin{aligned}
& x* = arg\ min_{x \in \mathcal{F}}\ f_i(x) \\
& \textit{Sujeito a:} \\
& f_j(x) \leq \epsilon_j,\ \ j = 1, ..., e\ (j \neq i)
\end{aligned}
$$

Segundo o Teorema 1, o conjunto de soluções eficientes é constituído por cada solução $\mathbf{x}^* \in \mathcal{F}$ obtida para cada ϵ_j. Assim sendo, a principal vantagem do método ϵ-Restrito é que através dele é possível encontrar a fronteira Pareto independente da estrutura do problema multiobjetivo, podendo ele ser não-convexo e discreto. Este fato o distingue do método citado anteriormente no qual as soluções eficientes são obtidas através da ponderação dos objetivos. Por outro lado, a definição dos valores de ϵ_j que irão gerar o conjunto de soluções eficientes não é um tarefa trivial. O ponto obtido por um dado limite ϵ_j pode não ser eficiente e, além disso, um problema formulado para um dado valor de ϵ_j pode ser infactível [Takahashi, 2004].

As características principais deste método e os principais conceitos serão utilizados neste trabalho em heurísticas propostas para o problema de roteamento de veículos multiobjetivo com coleta opcional.

2.3 Metaheurísticas Multiobjetivo

Diversas técnicas heurísticas são encontradas na literatura para solução de problemas de otimização multiobjetivo. Dentre elas, os algoritmos evolucionários vem sendo bastante utilizados [Deb et al., 2002; Corberán et al., 2002; Dias e Vasconcelos, 2002; Jozefowiez et al., 2009]. Entretanto, estes métodos envolvem uma grande quantidade de parâmetros que precisam

ser ajustados, sendo que o seu desempenho depende destes ajustes. Isso pode tornar a sua aplicação muito complexa. Outros métodos, adaptados de metaheurísticas bastante conhecidas, como Busca Tabu [Pacheco e Marti, 2006; Lust, 2010] e ILS [Geiger, 2006], também são encontrados na literatura para solução de diversos problemas de otimização combinatória multiobjetivo.

As subseções a seguir descrevem duas metaheurísticas adaptadas neste trabalho: a busca local iterativa multiobjetivo (MOILS, *Multiobjective Iterated Local Search*) e o algoritmo genético de ordenação não dominante II (NSGA-II, *Non-Dominated Sorting Genetic Algorithm II.*

2.3.1 Busca Local Iterativa Multiobjetivo (MOILS)

A busca local iterativa multiobjetivo ou *Pareto Iterated Local Search*, como é definido por Geiger [2006], é um novo conceito para a resolução de problemas de otimização com múltiplos objetivos. O algoritmo combina intensificação e diversificação de forma simples e em um único algoritmo. Estes dois métodos fazem uso da busca local iterativa (ILS), introduzindo a ideia de perturbações para "escapar" de ótimos locais, e do método de descida em vizinhança variável(VND, *Variable Neighborhood Descent*) que busca encontrar o ótimo local de diversas vizinhanças exploradas sistematicamente. Na busca local iterativa multiobjetivo (MOILS, *Multiobjective Iterated Local Search*), os métodos ILS e VND são combinados e estendidos dentro de um algoritmo de busca no qual não só uma solução é retornada, mas um conjunto de alternativas. Neste trabalho o MOILS será referenciado como busca geral em vizinhança variável multiobjetivo (MOGVNS, *Multiobjective General Variable Neighborhood Search*), devido a semelhanças entre ILS/VND e GVNS.

O algoritmo 2.1 apresenta os procedimentos básicos do MOILS e a figura 2.4 ilustra estes procedimentos. A partir de uma solução inicial (x), a fase de intensificação é aplicada até que se alcance um conjunto de alternativas ótimas locais (Definição 4). Esse conjunto de alternativas é então armazenada em um conjunto que representa a fronteira Pareto aproximada. Nesta etapa inicial, o VND garante que todas as alternativas são localmente ótimas para o conjunto de vizinhanças exploradas [Geiger, 2006].

Após a identificação de um conjunto ótimo local, inicia-se a fase de diversificação. Assim, uma das soluções da fronteira é selecionada (x') e, a esta solução, é aplicada uma função de perturbação, gerando uma nova solução (x) onde, novamente, o VND será aplicado. A perturbação deve ser suficiente para que o método consiga alcançar outros ótimos locais, mas não deve ser exagerada para que o método não se torne completamente aleatório.

Para melhorar a diversidade da fronteira, a fase de seleção poderá implementar métodos que garantam que soluções presentes em regiões poucos exploradas sejam priorizadas como, por exemplo, utilizar distância de multidão [Deb et al., 2002].

Algoritmo 2.1 Busca Local Iterativa Multiobjetivo (MOILS) [Geiger, 2006]

1: Definir vizinhanças $N_1, ..., N_k$;
2: $i \leftarrow 1$;
3: $x \leftarrow$ Gerar Solução Inicial();
4: $Fronteira \leftarrow \{x\}$;
5: **repita**
6: **repita**
7: $Neighbor_x[\,] \leftarrow CalculaN_i(x)$;
8: Atualizar Fronteira(V_x);
9: **se** $\exists x' \in V_x | x' \preceq x$ **então**
10: $x \leftarrow x'$;
11: $i \leftarrow 1$;
12: Rearranjar o conjunto de vizinhanças $N_1, ..., N_k$ em alguma ordem aleatória;
13: **senão**
14: $i \leftarrow i + 1$;
15: **fim se**
16: **até** x ser um ótimo local em todas as vizinhanças $N_1, ..., N_k$
17: $i \leftarrow 1$;
18: **se** $\exists x' \in Fronteira | x'$ ainda não foi explorado **então**
19: $x \leftarrow x'$;
20: **senão**
21: $x' \leftarrow$ Selecionar($Fronteira$);
22: $x \leftarrow$ Perturbação(x');
23: **fim se**
24: **até** Critério de Parada
25: **retornar** $Fronteira$;

2.3.2 Algoritmo Genético de Ordenação Não-Dominante II (NSGA-II)

Dentre os diversos métodos utilizados para solucionar problemas de otimização multiobjetivo destacam-se os algoritmos evolucionários. Estes métodos permitem que um conjunto de soluções aproximadas da fronteira Pareto sejam encontradas mesmo para problemas discretos e não-convexos.

O primeiro algoritmo genético multiobjetivo, conhecido como VEGA (*Vector Evaluated Genetic Algorithm*), foi proposto por Schaffer [1984]. O algoritmo é constituído dos conceitos básicos do algoritmo genético simples e de otimização multiobjetivo. Neste algoritmo não há elitismo e mecanismo para melhorar a diversidade das soluções na fronteira. Além disso, outra característica do método é que a população é dividida em subconjuntos e os indivíduos de cada subconjunto são avaliados de acordo com um dos objetivos do problema.

Posteriormente, Goldberg [1989] apresentou um procedimento de ordenação das soluções por fronteiras não-dominadas e, a partir de então, diversos outros algoritmos genéticos multi-

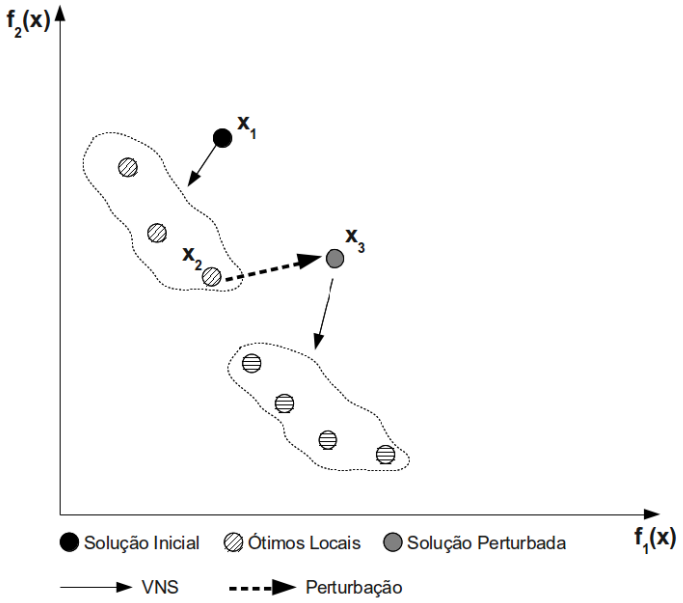


Figura 2.4: Busca Local Iterativa Multiobjetivo (MOILS) [Geiger, 2009]

objetivo foram propostos na literatura, como: MOGA (*Multiobjective Genetic Algorithm*) [Fonseca e Fleming, 1993], NPGA (*Niched Pareto Genetic Algorithm*) [Horn et al., 1994], NSGA (*Non-Dominated Sorting Genetic Algorithm*) [Srinivas e Deb, 1994], NSGA-II (*Fast Non-Dominated Sorting Genetic Algorithm*) [Deb et al., 2002], SPEA (*Strength Pareto Evolutionary Algorithm*) [Zitzler, 1999], SPEA-2 [Zitzler et al., 2002], PESA [Corne et al., 2000], PESA-II [Corne et al., 2001], dentre outros.

O que difere a maioria destes algoritmos é a forma como é feita a avaliação dos indivíduos, como é aplicado o elitismo ou, ainda, o método utilizado para garantir a diversidade das soluções na fronteira Pareto aproximada.

O NSGA-II, proposto por Deb et al. [2002], é um dos algoritmos mais utilizados para solucionar problemas de otimização com mais de um objetivo. A vantagem do NSGA-II se deve a sua baixa complexidade computacional, $O(MN^2)$, sendo M o número de objetivos e N o tamanho da população. Além disso, o método é capaz de gerar soluções de boa qualidade para a maioria dos problemas no qual é aplicado, sendo estas soluções bem distribuídas na fronteira de soluções não-dominadas.

O NSGA-II possui um módulo para classificação da população baseado no ordenamento de soluções proposto por Goldberg [1989], chamado de *fast-non-dominated-sort*. Este procedimento consiste em classificar os indivíduos de maneira elitista utilizando a dominância de Pareto como referência. A classificação é feita de acordo com a relação de dominância das soluções, no qual a população é dividida em subgrupos ou fronteiras de acordo com o seu grau de dominância. Em um mesmo subgrupo, nenhuma solução deve dominar outra solução.

A primeira fronteira contém os melhores indivíduos de toda a população, sendo eles soluções não dominadas por nenhuma outra solução. Seguindo esta ideia, a última fronteira contém os piores indivíduos da população. A figura 2.5 apresenta uma ilustração de um problema de minimização com duas funções objetivo e três fronteiras.

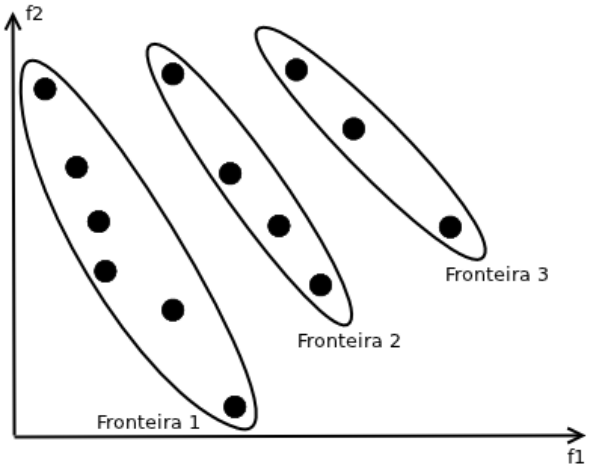


Figura 2.5: Ordenação por dominância

Para garantir a diversidade das soluções ao longo da fronteira Pareto, o NSGA-II utiliza um operador denominado distância de multidão (*Crowding Distance*) [Deb et al., 2002]. Este operador melhora a distribuição das fronteiras, evitando que as soluções fiquem próximas umas das outras e represente de forma adequada a fronteira Pareto.

Dado os subgrupos formados na ordenação por dominância, para avaliar soluções em um mesmo grupo e garantir maior diversidade, o operador distância de multidão indica o valor do perímetro do cubóide criado a partir de uma solução, sendo os vértices desse cubóide as soluções vizinhas a solução no subgrupo ou fronteira a qual pertencem. Os indivíduos contidos nas extremidades da fronteira recebem um valor arbitrariamente grande a fim de serem priorizados durante a seleção elitista. Os demais indivíduos i da fronteira recebem um valor correspondente à distância entre i e seus vizinhos mais próximos. Assim sendo, uma solução com a melhor classificação em uma fronteira é aquela que apresenta um maior valor da distância de multidão. A figura 2.6 ilustra o cálculo da distância de multidão. Na figura, a distância de multidão da solução i é dada pelo perímetro do retângulo formado pelos seus vizinhos mais próximos $i-1$ e $i+1$.

Dados os dois procedimentos básicos do NSGA-II, o algoritmo aplica o método de seleção por torneio da seguinte forma: uma solução i, pertencente a fronteira $Fronteira_i$ após a ordenação de dominância, é considerada melhor que a solução j, pertencente a fronteira $Fronteira_j$, se:

1. i possui um nível de não dominância melhor que j, ou seja, se $Fronteira_i < Fronteira_j$.

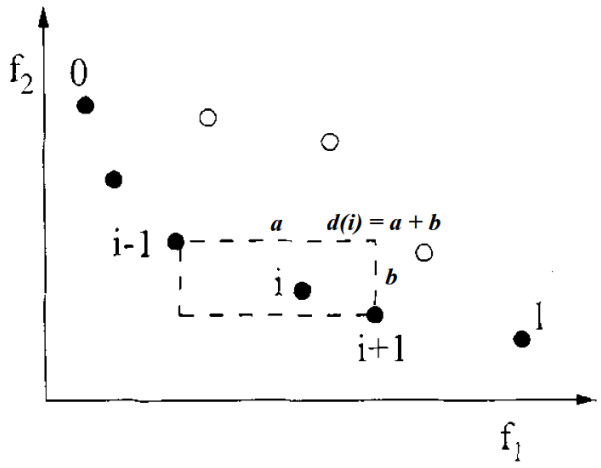


Figura 2.6: Distância de Multidão

2. $Fronteira_i = Fronteira_j$ e a distância de multidão de i é maior que distância de multidão de j.

O algoritmo 2.2 mostra todo o procedimento do NSGA-II. O algoritmo inicia com a geração aleatória de uma população inicial P de tamanho N (linhas 2 e 3). Uma nova população Q de tamanho N é gerada através de mecanismos de seleção, cruzamento e mutação aplicados a população inicial. As populações Q e P são então agrupadas em uma população R, de tamanho $2N$, que passará pela ordenação por não-dominância (linhas 5 e 6). A partir de então (linhas 9-11), N soluções de R serão copiadas para a população P da seguinte forma: enquanto não extrapolar o limite máximo de N soluções da população P, as soluções da primeira fronteira de R são inseridas em P. Em seguida, se o limite N ainda não foi alcançado, as soluções da segunda fronteira serão copiadas e assim sucessivamente. Este procedimento se repete até que não seja mais possível inserir inteiramente uma fronteira da população P. Neste instante, alguns indivíduos da fronteira serão selecionados de acordo com o operador distância de multidão, até completar os N indivíduos da população P (linhas 13-19). A figura 2.7 ilustra o procedimento de inserção das soluções da população R na população P.

Em seguida, aplica-se os operadores de seleção por torneio, conforme mencionado anteriormente, cruzamento e mutação, formando uma nova população Q. O procedimento então se repete até que um determinado critério de parada seja satisfeito. Ao final, o método retorna as soluções não dominadas contidas nas populações P e Q.

2.4 Métodos de Avaliação de Solução

A análise da qualidade dos resultados obtidos para problemas que envolvem múltiplos objetivos é mais complexa que para problemas com um único objetivo. Isto ocorre devido

Algoritmo 2.2 Pseudocódigo do NSGA-II

1: $n \leftarrow 0$; {contador de gerações}
2: $P_n \leftarrow$ Gerar População Inicial(); {População Pai}
3: $Q_n \leftarrow$ Aplicar operadores de Seleção, Cruzamento e Mutação(P_n); {População Filha}
4: **repita**
5: $R_n \leftarrow P_n \cup Q_n$;
6: $F[\,] \leftarrow$ Ordenação por dominância(R_n);
7: $P_{n+1} \leftarrow \varnothing$;
8: $i \leftarrow 1$
9: **enquanto** $|P_{n+1} + F_i| \leq N$ **faça**
10: $P_{n+1} \leftarrow P_{n+1} \cup F_i$;
11: $i \leftarrow i + 1$;
12: **fim enquanto**
13: **para todo** $s_j \in F_i$ **faça**
14: $d_j \leftarrow$ Distância de Multidão(s_j);
15: **fim para**
16: $s \leftarrow$ Ordenar as soluções $s \in F_i$ crescentemente de acordo com o valor de d_j;
17: **para** $j = 1$ até $j = N - |P_{n+1}|$ **faça**
18: $P_{n+1} \leftarrow s_j$;
19: **fim para**
20: $Q_{n+1} \leftarrow$ Aplicar operadores de Seleção, Cruzamento e Mutação(P_{n+1});
21: $n \leftarrow n + 1$;
22: **até** Critério de Parada
23: $R_n \leftarrow P_n \cup Q_n$;
24: $Fronteira \leftarrow$ Obter soluções não-dominadas(R_n);
25: **retornar** $Fronteira$;

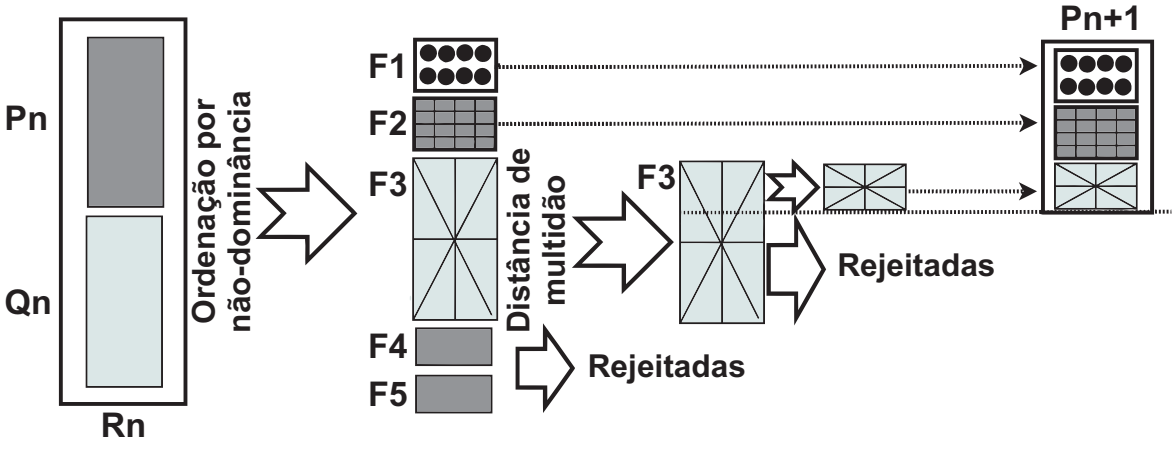


Figura 2.7: Procedimento do NSGA-II

à otimização multiobjetivo, em grande parte, envolver objetivos conflitantes, levando a um conjunto de soluções não dominadas.

As principais características avaliadas em soluções de problemas multiobjetivo são: (i) a proximidade à fronteira Pareto ótima; (ii) a cardinalidade, que é o número de soluções obtidas; (iii) a distribuição, isto é, se as soluções estão distribuídas de maneira uniforme ao longo da fronteira; e (iv) a extensão, que é o intervalo de valores coberto pela fronteira para cada objetivo.

Existem diversas métricas para avaliar uma solução para o problema de otimização com múltiplos objetivos, cada qual avalia uma determinada característica desejável a esta solução. As métricas aplicadas neste trabalho são: hipervolume (S) e Cobertura (C).

2.4.1 Hipervolume (S)

A proximidade das soluções em relação à fronteira Pareto ótima pode ser estimada pelo cálculo do hipervolume (S). O hipervolume é dado pela soma dos hipercubos formados pelas soluções não-dominadas obtidas pelos algoritmos. Ele calcula o volume da região coberta entre os pontos das soluções na fronteira encontrada e um determinado ponto de referência. Apesar da fronteira Pareto ótima não ser conhecida para o problema abordado neste trabalho, o hipervolume ajuda a mensurar a qualidade de uma solução em relação a outra, sendo que maiores hipervolumes indicam melhores soluções. A figura 2.8 apresenta uma ilustração do cálculo do hipervolume para um problema de minimização com dois objetivos e considerando W um ponto de referência Deb [2001].

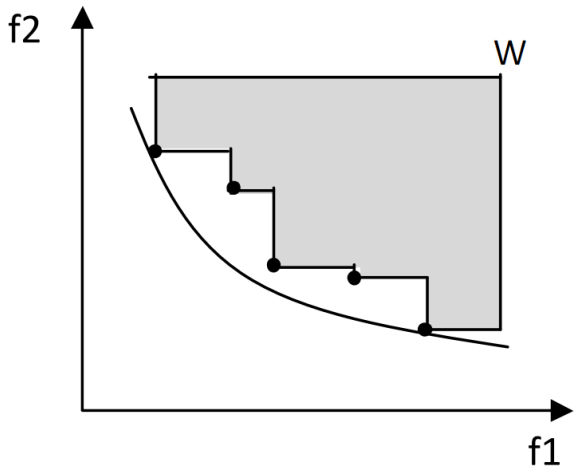


Figura 2.8: Cálculo do hipervolume para um problema de minimização com dois objetivos e considerando W o ponto de referência

2.4.2 Cobertura (C)

O propósito da métrica de cobertura é que um algoritmo com melhor desempenho retorne soluções com maior cobertura que o outro método utilizado na comparação. Assim sendo,

dadas duas fronteira Pareto aproximadas A e B, C(A,B) calcula o número de soluções em B que são fracamente dominadas por outra fronteira A, dada a cardinalidade de B. Se C(A,B) é igual a 1, então todas as soluções em B são dominadas por soluções da fronteira A. Por outro lado, se o valor retornado é igual a 0 isso aponta para uma situação oposta no qual nenhuma das soluções de B são dominadas por A. Para aplicação desta métricas é necessário avaliar tanto C(A,B) quanto C(B,A), pois estes valores não são complementares. A equação 2.2 apresenta como se obtém o valor da cobertura dadas duas fronteira Pareto aproximadas A e B [Deb, 2001].

$$C(A,B) = \frac{|\{y \in B : \exists x \in A, x \preceq y\}|}{|B|} \tag{2.2}$$

2.5 Conclusão

Neste capítulo foram apresentados os pricipais conceitos de otimização multiobjetivo que serão aplicados ao longo deste trabalho. Inicialmente foi apresentado um modelo geral de problema multiobjetivo e as notações empregadas neste trabalho. Posteriormente, seguindo o modelo geral, os conceitos de dominância, solução Pareto-ótima, conjunto Pareto-ótimo e Fronteira Pareto-ótima foram definidos.

Existem inúmeros métodos de resolução de problemas multiobjetivo, dentre eles metodologias tradicionais, capazes de encontrar a solução ótima, e as metaheurísticas. Neste capítulo foram apresentados duas metaheurísticas muito distintas: a busca local iterativa multiobjetivo (MOILS) e algoritmo genético de ordenação não-dominante II (NSGA-II). A primeira é caracterizada pelas buscas locais e a segunda pelo uso de uma população de indivíduos. A literatura aponta que ambos os métodos apresentam bons resultados quando aplicados a problemas de otimização combinatória [Geiger, 2006, 2003; Garcia-Najera e Bullinaria, 2009b] e, assim, são propícios de serem estudados e explorados para gerar algoritmos específicos para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional.

A qualidade de uma solução para um problema multiobjetivo envolve várias questões, dentre elas a diversidade das soluções geradas e a aproximação das soluções em relação a fronteira Pareto ótima. Assim sendo, este capítulo apresentou um conjunto de métricas comumente utilizadas para avaliar as soluções para um problema multiobjetivo. De posse das métricas, é possível distiguir qual o melhor método para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, que é o foco desta tese.

Capítulo 3

Problema de Roteamento de Veículos

O problema de roteamento de veículos é um dos mais importantes no âmbito da logística de transporte e distribuição. O objetivo do problema é determinar um conjunto de rotas de menor custo para uma determinada frota de veículos a fim de satisfazer as demandas de entrega de um conjunto de consumidores.

Em geral, o problema de roteamento de veículo apresenta um único objetivo referente a minimização dos custos de transporte. Este problema ignora o fato de que muitos problemas logísticos têm natureza multi-objetivo. As empresas têm grande interesse em manter a satisfação de seus clientes, minimizando o atraso das entregas dos produtos e maximizar a satisfação dos seus funcionários responsáveis pelo transporte, definindo rotas balanceadas de forma que os motoristas tenham carga de trabalho similar. Logo, o tomador de decisão, em diversas situações reais, necessita avaliar estes outros aspectos.

Por se tratar de um problema básico de logística reversa, houve um rápido crescimento no número de trabalhos referentes a problemas de roteamento de veículos com serviços de coleta e entrega. Portanto, este capítulo apresenta um estudo das duas variações de problemas de roteamento de veículos: com abordagem multiobjetivo e com serviços de coleta e entrega. Este estudo possibilitará a avaliação dos trabalhos que se situam na interseção dessas duas classes de problemas, e propiciará também a proposição de novas formulações.

Este capítulo está dividido em quatro seções: a seção 3.1 define o problema de roteamento de veículos e apresenta algumas de suas variações. A seção 3.2 apresenta uma revisão bibliográfica dos trabalhos referentes ao problema de roteamento de veículos com serviços de coleta e entrega e a seção 3.3 relata os diversos problemas de roteamento de veículos multiobjetivo. Por fim, a seção 3.4 apresenta a conclusão do capítulo.

3.1 Introdução

O problema de roteamento de veículos capacitado (CVRP, do inglês *Vehicle Routing Problem*), introduzido por Dantzig e Ramser [1959], tem como objetivo definir um conjunto de rotas entre um depósito e um conjunto de pontos de entrega que minimize os custos de transporte. As características básicas do problema consistem em:

- Cada cidade é visitada uma única vez por um único veículo;
- Cada rota é iniciada num depósito e finalizada no mesmo depósito;
- As demandas de todos os consumidores devem ser satisfeitas.

Assim sendo, o problema de roteamento de veículos capacitado pode ser definido como: dado um grafo completo $G = (V, A)$, onde $V = \{v_0, v_1, v_2, \ldots, v_n\}$ é o conjunto de vértices e $A = \{(v_i, v_j) : v_i, v_j \in V \ e \ v_i \neq v_j\}$ o conjunto de arestas. O vértice v_0 representa o depósito e os demais vértices representam os consumidores. Cada aresta (v_i, v_j) tem um valor $c_{ij} \geq 0$ associado que representa o custo de se alcançar o vértice v_j a partir do vértice v_i. Cada consumidor v_i tem uma demanda d_i de entrega. Tem-se disponível uma frota de k_{max} veículos homogêneos de capacidade Q para atendimento das demandas. A variável de fluxo z_{ij} indica o somatório das cargas entregues entre o nó v_i (exclusive) e o depósito dirigida ao nó v_j.

As variáveis de decisão do problema consistem em:

$$x_{ij}^k = \begin{cases} 1 & \text{, se arco } (v_i, v_j) \text{ faz parte da rota trafegada pelo veículo } k, \\ 0 & \text{, caso contrário.} \end{cases}$$

O problema básico de roteamento de veículo pode ser formulado como:

$$Min \quad f(\mathbf{x}) = \sum_{k=1}^{k_{max}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k \tag{3.1}$$

sujeito a

$$\sum_{i=0}^{n} \sum_{k=1}^{k_{max}} x_{ij}^k = 1, \qquad j = 1, \ldots, n \tag{3.2}$$

$$\sum_{i=0}^{n} x_{ij}^k - \sum_{i=0}^{n} x_{ji}^k = 0, \qquad j = 0, \ldots, n \text{ e } k = 0, \ldots, k_{max} \tag{3.3}$$

$$\sum_{j=1}^{n} x_{0j}^{k} \leq 1, \qquad k = 1, \ldots, k_{max} \tag{3.4}$$

$$\sum_{i=0}^{n} z_{ij} - \sum_{i=0}^{n} z_{ji} = d_j, \qquad \forall j \neq 0 \tag{3.5}$$

$$z_{ij} \leq Q \sum_{k=1}^{k_{max}} x_{ij}^{k}, \qquad i, j = 0, \ldots, n \tag{3.6}$$

$$x_{ij},\ \ell_j \in \{0, 1\}, \qquad i, j = 0, \ldots, n \tag{3.7}$$

$$z_{ij} \geq 0, \qquad i, j = 0, \ldots, n \tag{3.8}$$

A função objetivo, dada na equação 3.1, visa a minimização do custo (distância percorrida). A primeira restrição (equação 3.2) indica que cada ponto de demanda deve ser visitado por um único veículo. A equação 3.3 representa a restrição de conservação do fluxo. A equação 3.4 indica que k_{max} veículos, no máximo, podem ser utilizados. A equação 3.5 obriga a satisfação de todas as demandas de entrega. A equação 3.6 define que as demandas devem ser transportadas nos arcos incluídos na solução e ainda impõem um limite para a carga total transportada pelo veículo. A equação 3.7 representa a restrição de integralidade e a equação 3.8 representa a restrição de não-negatividade para demandas de entrega.

Além das restrições básicas, variações do problema de roteamento podem ser definidas dados os seguintes componentes [Jozefowiez et al., 2008a]:

Rede: pode ser representada por um grafo valorado onde os vértices representam os pontos de demanda e também os depósitos. As arestas representam as conexões entre os vértices e cada uma delas possui um valor referente ao custo de utilização da conexão. Assim, a rede pode ser simétrica, assimétrica ou mista. Poderão existir, também, janelas de tempo associadas a cada aresta e/ou vértice. Assim, os veículos devem percorrer uma determinada aresta ou atender a uma demanda em um intervalo de tempo estipulado.

Demandas: podem ser fixas ou dinâmicas, ou seja, podem ser determinadas previamente, antes da definição das rotas, ou durante o percurso do veículo. As demandas também podem estar associadas não só aos vértices, mas também às arestas e serem de um único ou vários produtos. Em geral, os veículos devem atender a um conjunto de demandas de entrega, mas existem problemas que requerem entrega e coleta de itens.

Frota: pode ser composta por um ou vários veículos homogêneos ou heterogêneos. O problema do caixeiro viajante (TSP, do inglês *Traveling Salesman Problem*), por exemplo, pode ser definido como um problema de roteamento com um único veículo. A utilização do veículo pode ser limitada em relação ao tempo, distância e capacidade. Alguns problemas podem apresentar outras restrições associadas a dependência entre veículos, produtos, arestas e vértices.

Custos: além dos custos associados às arestas (definidos na rede) que podem indicar distância percorrida, tempo de utilização, consumo de combustível, podem também existir outros custos concernentes ao atendimento com atraso ou atendimento incompleto de uma demanda. Por outro lado, pode haver um ganho (prêmio) associado às arestas ou vértices quando estes são percorridos ou visitados.

Objetivos: os mais utilizados são: minimização da distância percorrida ou tempo do percurso, minimização do número de veículos, maximização da qualidade do serviço, maximização dos lucros adquiridos. A função objetivo pode ser calculada em um único período ou em períodos diversos, dependendo se a demanda é fixa ou dinâmica, respectivamente. Em certos problemas existe a necessidade de avaliar mais de um objetivo simultaneamente, sendo estes conflitantes. Nestes casos é extremamente vantajosa a aplicação de uma abordagem multiobjetivo.

3.2 Problema Geral de Coleta e Entrega

Dentre as inúmeras variações apresentadas para o problema, nos últimos anos houve um rápido crescimento no estudo de diferentes problemas de roteamento de veículos com serviço de coleta e entrega de mercadorias ou passageiros. Em geral, o objetivo destes problemas é minimizar os custos de transporte, definindo rotas otimizadas que satisfaçam as demandas de coleta e/ou entrega. Parragh et al. [2008] definem este conjunto de problemas por: problema geral de coleta e entrega (GPDP, do inglês *General Pickup and Delivery Problem*).

O problema geral de coleta e entrega pode ser dividido em:

- *one-to-many-to-one*: todas as cargas que irão satisfazer as demandas de entrega devem partir de um ou vários depósitos e todas as cargas coletadas nos pontos de demanda devem ser transportadas para um dos depósitos.
- *one-to-one*: os veículos partem vazios de um ou vários depósitos, efetuam coletas e entregas, e retornam vazios ao depósito. As entregas são feitas com itens provenientes de coletas efetuadas anteriormente.

Parragh et al. [2008] apresentam uma revisão detalhada da literatura para os diferentes problemas de coleta e entrega. Segundo eles, o crescimento rápido de estudos na área acarretou

em inúmeras confusões na terminologia utilizada para descrever os vários problemas deste contexto. Assim, para facilitar o entendimento, os autores denominam os problemas da classe *one-to-many-to-one* como problema de roteamento de veículos com coletas de retorno (VRPB, do inglês *Vehicle Routing Problem with Backhauls*) e os problemas da classe *one-to-one* como problema de roteamento de veículos com coleta e entrega (VRPPD, do inglês *Vehicle Routing Problem with Pickup and Delivery*). Esta terminologia também será adotada neste trabalho.

O problema de roteamento de veículos com coleta e entrega pode ser subdividido em *Emparelhado* e *Desemparelhado*. No primeiro, às demandas dos consumidores são associados pares ordenados (v_i, v_j) indicando o vértice (v_i) onde a carga deve ser coletada e o vértice (v_j) onde a carga deve ser entregue. O problema de roteirização e programação de veículos (DARP, do inglês *Dial-a-Ride Problema*) é um problema emparelhado onde a carga transportada são passageiros e os vértices representam pontos de embarque e desembarque. No problema do tipo desemparelhado, uma carga coletada pode ser utilizada para satisfazer qualquer demanda de entrega de qualquer consumidor. Mais detalhes sobre problemas de roteamento de veículos com coleta e entrega podem ser encontrados em Parragh et al. [2008]. A figura 3.1 contém a classificação do problema geral de coleta e entrega.

Dentre os vários problemas que compõem o GPDP, este trabalho apresenta uma abordagem multiobjetivo para problemas do tipo *one-to-many-to-one* ou problema de roteamento de veículos com coleta de retorno (VRPB).

3.2.1 Problema de Roteamento de Veículos com Coletas de Retorno

O problema de roteamento de veículos com coletas de retorno, segundo Parragh et al. [2008], podem ser subdivididos em quatro subclasses, sendo que em duas delas os consumidores possuem demandas de coleta ou entrega, mas não ambas, e outras duas nos quais os consumidores podem requisitar coleta e entrega de mercadorias simultaneamente.

- Problema de Roteamento de Veículos com Coleta de Retorno Agrupada (VRPCB, do inglês *Vehicle Routing Problem with Clustered Backhauls*): é um problema de roteamento no qual os consumidores possuem demanda de coleta ou entrega, mas não ambas. Além disso, as demandas de entrega devem ser satisfeitas antes de iniciar o atendimento das demandas de coletas. Um problema com estas características e que possuem apenas um veículo é denominado problema do caixeiro viajante com coleta de retorno agrupada (TSPCB, do inglês *Traveling Salesman Problem with Clustered Backhauls*);
- Problema de Roteamento de Veículos com Coleta e Entrega de Retorno Mistas (VRPMB, do inglês *Vehicle Routing Problem with Mixed Linehauls and Backhauls*): os consumidores possuem demanda de coleta ou entrega, mas não ambos. No VRPMB

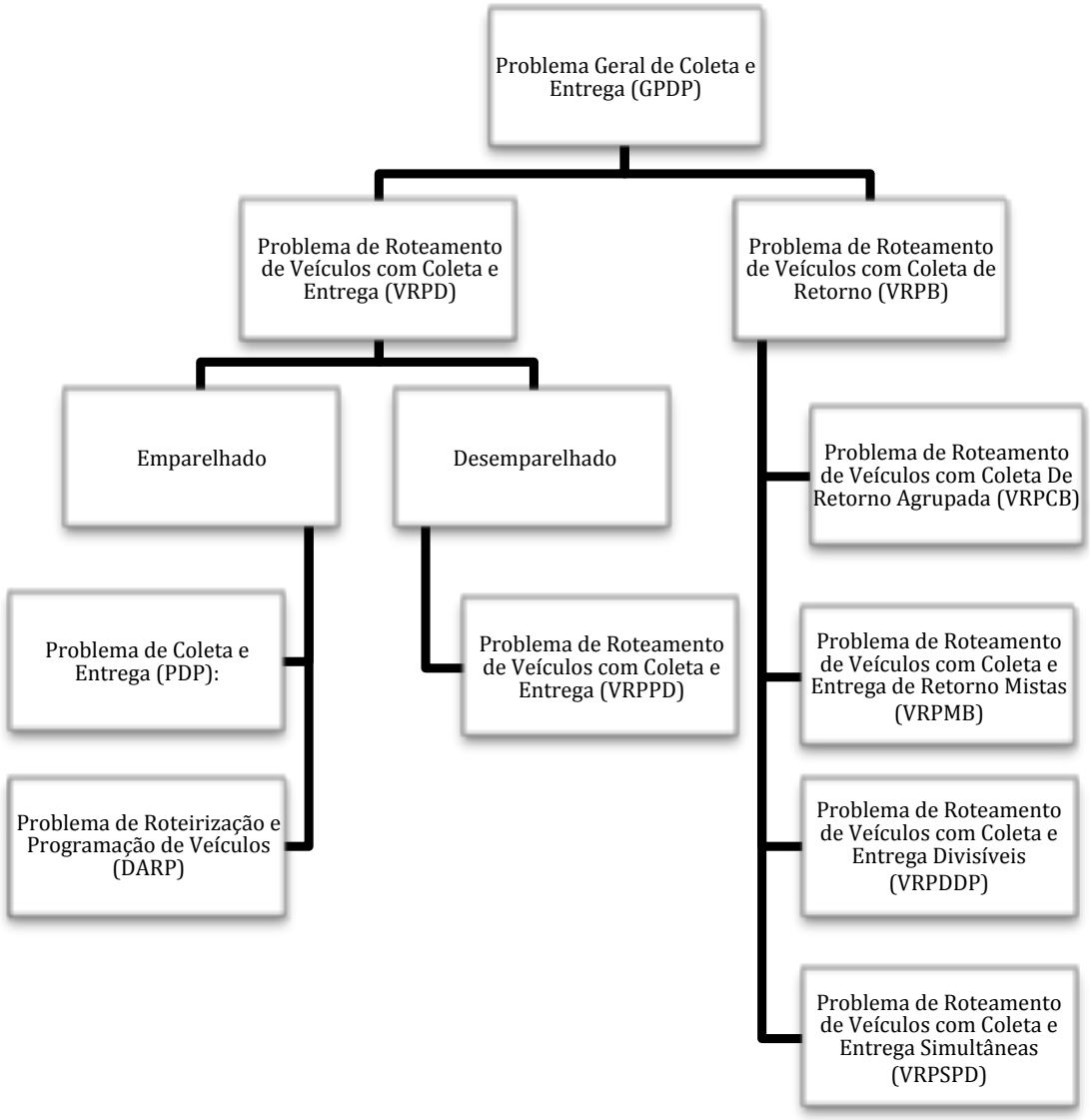


Figura 3.1: Variações do problema geral de coleta e entrega

não existe restrição na ordem de efetuar as coletas e entregas. Assim, uma demanda de coleta pode ser satisfeita antes de uma demanda de entrega e vice-versa. O VRPCB com um único veículo é denominado problema do caixeiro viajante com entregas e coletas de retorno Mistas(TSPMB, do inglês *Traveling Salesman Problem with Mixed Linehauls and Backhauls*);

- Problema de Roteamento de Veículos com Coleta e Entrega Divisíveis (VRPDDP, do inglês *Vehicle Routing Problem with Divisible Delivery and Pickup*): os consumidores possuem ambas demandas de coleta e entrega e podem ser visitados duas vezes, uma para atendimento da demanda de coleta e outra para entrega. O VRPDDP com um único veículo é denominado problema do caixeiro viajante com coleta e entrega divisíveis

(TSPDDP, do inglês *Traveling Salesman Problem with Divisible Delivery and Pickup*);

- Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas (VRPSPD, do inglês *Vehicle Routing Problem with Simultaneous Pickup and Delivery*): os consumidores possuem demandas de coleta e entrega, mas no VRPSPD o consumidor deve ser visitado uma única vez atendendo simultaneamente as demandas de coleta e entrega. O VRPSPD com um único veículo é denominado problema do caixeiro viajante com coleta e entrega simultânea (TSPSPD, do inglês *Traveling Salesman Problem with Simultaneous Pickup and Delivery*);

O problema de roteamento de veículos como coleta de retorno agrupada é abordado por Goetschalckx e Jacobs-Blecha [1993], Potvin et al. [1996], Reimann et al. [2002] e Gelogullari [2004]. Os dois primeiros trabalhos apresentam uma heurística para solucionar o VRPCB. Goetschalckx e Jacobs-Blecha [1993] propõem uma heurística baseada no problema de assinalamento generalizado, enquanto Potvin et al. [1996] apresenta uma heurística construtiva na qual cada consumidor é inserido na rota usando uma ordem de prioridade. Além da heurística construtiva, os autores desenvolveram um algoritmo genético para melhorar a qualidade das soluções. O trabalho de Reimann et al. [2002] também utiliza um método bioinspirado para solucionar o VRPCB. Nele os autores apresentam uma heurística baseada no método colônia de formigas. Uma abordagem exata pode ser encontrada em Gelogullari [2004].

Ghaziri e Osman [2003] propõem uma rede neural para resolução do TSPCB. Os resultados obtidos mostram que a abordagem proposta é comparável com as heurísticas encontradas na literatura em termos de qualidade da solução e recursos computacionais. Os testes foram aplicados em grandes instâncias superiores a 1000 consumidores.

O problema de roteamento de veículos com entregas e coletas de retorno mistas é abordado em Mosheiov [1998]. O autor apresenta uma heurística de particionamento de rota. Outra abordagem heurística envolvendo múltiplos depósitos pode ser encontrada em Salhi e Nagy [1999]. Ropke e Pisinger [2006] apresentam uma adaptação da heurística busca em grande vizinhança (LNS, do inglês *Large Neighborhood Search*). Este método mostrou bons resultados em um tempo computacional razoável em testes realizados em 350 redes distintas, com um número superior a 500 consumidores. O método conseguiu melhorar os resultados de 50% dos problemas em relação aos melhores resultados conhecidos na literatura.

O problema de roteamento de veículos com entregas e coletas de retorno mistas com um único veículo é apresentado por Süral e Bookbinder [2003]. Os autores propõem um novo modelo matemático utilizando o modelo Miller-Tucker-Zemlin (MTX) de restrição de eliminação de *subtour* para o problema do caixeiro viajante, assegurando a viabilidade da carga do veículo.

Mosheiov [1994] apresenta duas heurísticas para solucionar o TSPMB. A primeira consiste na busca por uma solução ótima para o problema do caixeiro viajante, considerando todos os consumidores do problema. Em seguida, para manter a viabilidade do problema e não extrapolar a capacidade do veículo, o depósito é inserido em diversos pontos. A segunda é baseada na heurística inserção mais barata também para solucionar o TSPMB. Primeiramente aplica-se um algoritmo exato para solucionar o problema do caixeiro viajante considerando apenas os consumidores com demandas de entrega. Em seguida, os demais consumidores são inseridos seguindo o critério de inserção mais barata e, ainda, verificando a capacidade do veículo.

O problema de roteamento de veículos com coleta e entrega divisíveis é uma mistura do VRPMB e VRPSPD. Ao contrário do VRPMB, cada consumidor pode possuir uma demanda de coleta e entrega, contudo estes consumidores não precisam ser visitados uma única vez. Poucas pesquisas têm abordado esse problema. Contudo, todos os métodos utilizados para solucionar o VRPMB podem ser aplicados ao VRPPD.

Diversos trabalhos sobre o problema de roteamento de veículos com coleta e entrega simultânea já foram publicados. Nestes trabalhos são propostos os mais variados métodos para solução dos problemas abordados, desde simples heurísticas construtivas a métodos exatos. Este problema foi proposto por Min [1989] que apresentou uma formulação matemática e um procedimento heurístico para sua solução. O método proposto para resolução consiste em uma heurística agrupar e rotear dividida em três fases: (i) agrupar os consumidores, (ii) atribuir os grupos aos veículos e (iii) realizar o roteamento dos grupos usando uma heurística para o problema do caixeiro viajante.

Abordagem exata para o VRPSPD pode ser encontrada em [Dell'Amico et al., 2006; Subramanian et al., 2011]. Dell'Amico et al. [2006] propõem uma abordagem exata para solução do VRPSPD. Neste trabalho, os autores apresentam um modelo matemático para o problema, e em seguida, utilizam o método *branch-and-price* para solucioná-lo. Para a fase de avaliação (*pricing*) são propostas duas técnicas: (i) programação dinâmica e (ii) relaxação do espaço de estados. Porém, o uso de técnicas exatas para solução deste problema demanda um grande custo computacional. Assim, a maior parte dos trabalhos propõem o uso de técnicas aproximadas para solução. Subramanian et al. [2011] apresentam um algoritmo *branch-and-cut* para o VRPSPD que controla a viabilidade da solução em relação a capacidade do veículo em pontos no interior da rota, através de uma restrição fraca. Os autores verificaram que na prática quando a restrição de capacidade é respeitada nas extremidades dificilmente não será respeitada no seu interior. O algoritmo foi testado em 87 instâncias com 5-20 clientes, encontrando limites inferiores melhores e várias novas soluções ótimas.

Diversas heurísticas construtivas para solucionar o VRPSPD podem ser encontrada na literatura [Salhi e Nagy, 1999; Dethloff, 2001; Assis, 2007; Chen, 2006]. Salhi e Nagy [1999]

propõem uma extensão da heurística de inserção clássica, a qual é denominada *cluster insertion heuristic*, sendo este método aplicado ao problema considerando um único depósito e múltiplos depósitos. Dethloff [2001] aborda vários aspectos da logística reversa e propõe uma técnica para solução do VRPSPD. Esta técnica é uma heurística construtiva baseada em diversos critérios para a inserção: distância trafegada, capacidade residual e distância entre o consumidor e o depósito. Em outro trabalho [Assis, 2007], diversas heurísticas baseadas no método dividir e rotear são propostas tais que conceitos de árvore geradora mínima são aplicados na fase de divisão dos consumidores. Após o procedimento de divisão, um algoritmo para solucionar o problema do caixeiro viajante é aplicado em cada grupo gerado na primeira fase. Outra heurística construtiva encontrada na literatura foi proposta por Chen [2006]. O algoritmo apresenta um procedimento de inserção paralela para o VRPSPD.

Algoritmos para solucionar o VRPSPD baseados em metaheurísticas como: busca tabu (TS, do inglês *Tabu Search*) [Chen, 2006; Montané e Galvão, 2006; Bianchessi e Righini, 2007; Wassan et al., 2007], busca local iterativa (ILS, do inglês *Iterated Local Search* [Subramanian et al., 2008, 2010; Morais et al., 2009], busca em vizinhança variável (VNS, do inglês *Variable Neighborhood Search*) [Freitas e Montané, 2008; Morais et al., 2009], procedimento de busca gulosa adaptativa aleatorizada (GRASP, do inglês *Greedy Randomized Adaptive Search Procedure*) [Freitas e Montané, 2008; Assis, 2007; Morais et al., 2009] são muito explorados na literatura. Dentre estes métodos, o algoritmo baseado na busca local iterativa, proposta por Subramanian et al. [2010], apresentam os melhores resultados da literatura.

No trabalho de Chen [2006], o autor apresenta um algoritmo híbrido baseado nas metaheurísticas recozimento simulado (SA, do inglês *Simulated Annealing*) e busca tabu, que faz uso de mecanismos para melhorar a qualidade das soluções geradas. A heurística híbrida permitiu reduzir a distância entre as soluções iniciais obtidas e as soluções ótimas. Para instâncias pequenas, ou seja, com poucos consumidores envolvidos, foi possível encontrar a solução ótima. Montané e Galvão [2006] também baseiam-se na busca tabu para solucionar o VRPSPD. Os autores apresentam heurísticas construtivas para encontrar uma solução inicial e diversas funções de vizinhança para o problema tratado. Bianchessi e Righini [2007] também propõem métodos baseados na metaheurística busca tabu. Eles realizam uma série de combinações entre heurísticas e buscas locais, permitindo verificar a qualidade e custo das soluções obtidas pelas combinações. Os resultados indicam que as buscas locais com vizinhanças mais complexas obtiveram resultados bons e mais robustas.

Wassan et al. [2007] mostram que a principal dificuldade do VRPSPD é a verificação da carga viável. Assim sendo, eles apresentam uma técnica que verifica a viabilidade dos movimentos de vizinhança sem aumentar sua complexidade computacional. Esta técnica é aplicada em um algoritmo também baseado na busca tabu.

Freitas e Montané [2008] apresentam dois algoritmos para solucionar o VRPSPD, um baseado no VNS e outro um método híbrido baseado no GRASP/VNS. Neste trabalho, os autores tratam o problema considerando a existência de frota homogênea e heterogênea. Ao comparar os algoritmos propostos, o algoritmo baseado no VNS apresentou melhores resultados que o método baseado no GRASP/VNS; no entanto, exige um maior esforço computacional. Outro trabalho que também apresenta um método híbrido é proposto por Morais et al. [2009]. Neste, é feito uma combinação das metaheurísticas GRASP, VND e ILS para solução do VRPSPD. A execução do método proposto ocorre da seguinte maneira: a etapa de busca local do GRASP é realizada pelo ILS que, por sua vez, utiliza o VND como procedimento para busca local. É utilizado a estratégia melhor-aprimorante para as estruturas de vizinhança utilizadas pelo VND, a fim de obter resultados de melhor qualidade.

Subramanian et al. [2008, 2010] propõem uma técnica que se mostrou bastante eficiente para solução do VRPSPD. Esta técnica é baseada na metaheurística ILS que utiliza o procedimento VND com ordenação da vizinhança aleatória (RVND, *Random Variable Neighborhood Descent*) na etapa de busca local. Os resultados obtidos foram comparados aos melhores resultados encontrados na literatura e a técnica proposta obteve soluções de melhor qualidade em muitos dos problemas teste utilizados. Este trabalho detém, até então, grande parte dos melhores resultados da literatura para o VRPSPD, mostrando-se um método eficiente para solução deste problema.

Algoritmos baseados em processos da natureza vêm apresentando bons resultados para os mais variados problemas. Gökçe [2004]; Zhang et al. [2008]; Gajpal e Abad [2009]; Maravilha et al. [2010] propõem heurísticas baseadas no processo de otimização por colônia de formigas para solução do VRPSPD, sendo que Zhang et al. [2008] consideram ainda a restrição de distância máxima, o que não ocorre nos demais trabalhos. Estas abordagens apresentam resultados de boa qualidade, entretanto, necessitam de um maior esforço computacional por trabalharem com diversas soluções.

Maravilha et al. [2010], que também fazem uso de técnicas bioinspiradas, além de um de algoritmo baseado em colônia de formigas, apresentam um algoritmo imunológico, adaptado do CLONALG (*Clonal Selection Algorithm*), para solução do VRPSPD. Ao comparar os resultados obtidos pelas duas técnicas, o algoritmo imunológico apresenta resultados de melhor qualidade, entretanto, necessita de um maior esforço computacional para execução. Em relação a outros métodos da literatura, o algoritmo imunológico apresentou resultados bastante próximos dos melhores conhecidos.

3.3 Problema de Roteamento de Veículos Multiobjetivo

As inúmeras variações para o problema de roteamento têm sido amplamente estudados por representar diversas situações reais de problema de transporte. Este problema é comumente modelados com um único objetivo: minimização dos custos. Neste trabalho, os custos de transporte são referentes a distância total percorrida pelos veículos. Contudo, vários problemas reais enfrentados pelas grandes empresas e indústrias não se limitam apenas aos aspectos de custo. Muitas vezes a solução destes problemas envolvem outros aspectos como equidade de trabalho e atrasos de entrega de produto. Assim sendo, o estudo de problemas de roteamento de veículos multiobjetivo e a busca por algoritmos eficientes para solucioná-lo tem atraído interesse de pesquisadores na área da Otimização Combinatória Multiobjetivo.

Segundo Jozefowiez et al. [2008b], as principais motivações para abordar problemas de roteamento com múltiplos objetivos se resumem em:

1. Estender problemas clássicos acadêmicos: quando a definição do problema permanece inalterada e novos objetivos são adicionados. Na maioria dos problemas de roteamento estudados na literatura a função objetivo é definida pela minimização dos custos de transporte. Ao estender estes problemas, busca-se então estudar outros objetivos. Dessa forma, as aplicações práticas do modelo inicialmente propostos são aperfeiçoadas, uma vez que diversos problemas de logísticas não estão unicamente ligados a questões de custo de transporte.

2. Generalizar problemas clássicos: quando uma ou mais restrições e/ou parâmetros são trocadas por funções objetivo.

3. Estudar casos reais: quando os objetivos foram claramente identificados pelo tomador de decisões e são dedicados a um problema real muito específico.

Uma vez que a redução dos custos é o principal objetivo almejado por empresas de qualquer natureza, então a minimização da distância percorrida pelos veículos é um objetivo comumente encontrado nos problemas abordados na literatura. Em alguns deles, além da minimização da distância percorrida, busca-se minimizar o número de veículos utilizados para satisfazer as demandas dos consumidores [Sousa et al., 2011; Ombuki et al., 2006; Tan et al., 2006b, 2003, 2006a]. Além dos objetivos citados, outros diferentes podem ser considerados de acordo com os interesses da empresa. Assim sendo, os objetivos podem ser classificados como [Garcia-Najera, 2010]:

Custo de Transporte: Este objetivo refere-se a economia dos gastos com transporte, como minimização da distância total percorrida pelo veículo, minimização do número de veí-

culos ou minimização do tempo de entrega [Garcia-Najera e Bullinaria, 2011, 2010, 2009a,b; Garcia-Najera, 2009, 2010; Ombuki et al., 2006; Chitty e Hernandez, 2004; Tan et al., 2007, 2006b, 2003; Barán e Schaerer, 2003; Tan et al., 2006a; Sousa et al., 2011].

Restrições: Assegura a minimização do número ou do grau de restrições violadas [Sessomboon et al., 1998; Chiang, 2008; Beham, 2007; Gutiérrez et al., 2008; Geiger, 2001, 2003; Rahoual et al., 2001].

Satisfação do Consumidor: Garante que o consumidor receba prontamente seu produto. Um exemplo de representação deste objetivo é a minimização do percurso da maior rota (rota com maior trajeto) [Sessomboon et al., 1998; Ribeiro e Lourenco, 2001; Murata e Itai, 2005, 2007, 2008; Geiger, 2001, 2003; Hong e Park, 1999; El-Sherbeny et al., 2002; Corberán et al., 2002; Pacheco e Marti, 2006; Bowerman et al., 1995; Jaw et al., 1986].

Satisfação dos Motoristas: Promove equidade da carga de trabalho dos motoristas, podendo ser representada pela minimização da diferença entre a rota mais longa e rota mais curta [Jozefowiez et al., 2002, 2005, 2007, 2009; Borgulya, 2008; Pasia et al., 2007; Lee e Ueng, 1998; Ribeiro e Lourenco, 2001; El-Sherbeny et al., 2002; Lacomme et al., 2006; Mourgaya e Vanderbeck, 2007].

Segurança: Busca minimizar os riscos de acidentes [Giannikos, 1998; Zografos e Androutsopoulos, 2004; Meng et al., 2005].

Acessibilidade: Objetivo duplo que corresponde a maximização da cobertura geográfica e minimização do número de facilidades móveis [Bowerman et al., 1995; Doerner et al., 2007; Doerner e Hartl, 2008].

Geografia: Requer que consumidores em uma mesma região sejam atendidos pelo mesmo veículo [Mourgaya e Vanderbeck, 2007; Doerner et al., 2007; Doerner e Hartl, 2008; Bowerman et al., 1995; Watanabe e Sakakibara, 2007].

Na literatura podem ser encontrados diversos trabalhos envolvendo problemas de roteamento de veículo multiobjetivo que podem ser generalizações, extensões ou estudo de casos reais de algum problema clássico, dentre eles: problema de roteamento de veículos capacitado, problema de roteamento de veículos com janela de tempo, problema de roteamento de veículos com múltiplos períodos, problema de roteamento de veículos dinâmico, problema de roteamento de veículos com demanda estocástica, problema de roteamento de veículos com coleta e entrega, problema de roteamento de veículos aberto, dentre outros. As subseções a seguir apresentam uma revisão de cada problema clássico que apresenta uma abordagem multiobjetivo.

3.3.1 Problema de Roteamento de Veículos com Janela de Tempo Multiobjetivo

O problema de roteamento de veículos com janela de tempo multiobjetivo é um dos mais abordados na literatura. Garcia-Najera e Bullinaria [2009a,b]; Garcia-Najera [2009, 2010] apresentam uma extensão deste problema, sendo que os objetivos são minimizar o custo de transporte, minimizar o número de veículos e Garcia-Najera e Bullinaria [2010, 2011] ainda incluem a minimização do tempo de entrega. O método utilizado para solucionar o problema trata-se de um algoritmo evolucionário multiobjetivo que incorpora uma medida de similaridade, baseada no coeficiente de *Jaccard*, para selecionar os indivíduos para o procedimento cruzamento. O método proporcionou uma maior diversidade da população, gerando soluções de melhor qualidade.

Outras extensões do problema com os mesmos objetivos são apresentadas por Ombuki et al. [2006]; Tan et al. [2003, 2006b]. Ombuki et al. [2006] ponderam as duas funções objetivo (minimizar distância percorrida pelos veículos e minimizar o número de veículos utilizados) e aplica um algoritmo genético mono-objetivo. Tan et al. [2003, 2006b] apresentam duas versões: uma modelagem com função objetivo ponderada e também utiliza um algoritmo genético mono-objetivo e outra na qual utilizam um algoritmo genético multiobjetivo.

Generalizações do problema de roteamento de veículos com janela de tempo geralmente transformam as restrições referentes a janela de tempo em um objetivo. Sessomboon et al. [1998] utilizam um algoritmo genético híbrido para solucioná-lo cujos objetivos considerados são: minimizar o número de violações da janela de tempo, o atraso ou adiantamento no atendimento a uma demanda, o número de veículos e a distância percorrida pelos veículos. Os objetivos do problema estão relacionados a satisfação do consumidor, restrições violadas e custos de transporte.

Geiger [2001, 2003] propõe um modelo similar ao de Sessomboon et al. [1998] composto por quatro objetivos: minimização da distância total trafegada, minimização do número de veículos, minimização das violações da janela de tempo e minimização do número de violações da janela de tempo. O autor também faz uso do algoritmo genético para solucionar o problema proposto.

Beham [2007] também apresenta uma generalização do problema, buscando, além da redução dos custos de transporte, a minimização do grau de violação da janela de tempo. O método adotado pelo autor é baseado na busca tabu no qual outras duas memórias são utilizadas além da lista tabu. Uma delas armazena as soluções não-dominadas de vizinhanças anteriores que serão utilizadas para reiniciar o processo de busca e a outra armazena as soluções não-dominadas encontradas durante todo o procedimento.

Alguns trabalhos transformam outras restrições em objetivos, além das restrições referen-

tes a janela de tempo. O trabalho de Gutiérrez et al. [2008] apresenta um algoritmo baseado no enxame de partícula (PSO, do inglês *Particle Swarm Optimization*) para solucionar o problema. O autor considera os seguintes objetivos na modelagem matemática proposta: minimizar a distância total percorrida, o número de veículos, o número de violações da janela de tempo e o número de violações em relação a capacidade do veículo. Já o trabalho de Rahoual et al. [2001] transforma a maior parte das restrições em objetivos do problema, sendo eles: minimizar a distância trafegada pelos veículos, minimizar o número de veículos, minimizar o número de violações da restrição de distância máxima, minimizar o número de violações da restrição de capacidade máxima do veículo, minimizar o número de violações referente a restrição de tempo máximo de tráfego dos veículos, minimizar o número de violações referente a restrição de janela de tempo. O método proposto para solucionar o problema pondera a função objetivo e aplica um algoritmo genético mono-objetivo. Outro método proposto aplica um algoritmo genético multiobjetivo.

Outra modelagem matemática diferente encontrada na literatura para o problema de roteamento de veículos com janela de tempo multiobjetivo é apresentado por Barán e Schaerer [2003]. Neste trabalho os autores apresentam um modelo composto pelos objetivos: minimizar o número de veículos, a distância trafegada e o tempo total de entrega. O tempo de entrega de determinada demanda é dado pelo tempo para percorrer a rota até o ponto de entrega da respectiva demanda (consumidor) acrescido do tempo de espera do veículo no consumidor caso o veículo chegue antes do início da janela de tempo determinada.

Hong e Park [1999] utilizam o método de programação linear de metas pra solucionar o problema modelado com dois objetivos: minimização do tempo total do trajeto percorrido pelos veículos e minimização do tempo total de espera dos consumidores a serem atendidos. Assim como nos demais trabalhos, a minimização do tempo total de espera dos consumidores ou minimização das violações da janela de tempo buscam garantir uma maior satisfação do consumidor.

Alguns estudos de caso que recaem ao problema de roteamento de veículos com janela de tempo podem ser encontrados na literatura. Sousa et al. [2011] apresenta um estudo de caso real de uma empresa de entregas de Portugal. O objetivo do problema é minimizar os custos de transporte, incluindo distância percorrida pelos veículos e número de veículos utilizados para satisfazer as demandas dos consumidores. O autor utiliza GLPK para solucionar o modelo proposto no trabalho.

3.3.2 Problema de Roteamento de Veículo Capacitado Multiobjetivo

Uma das extensões do problema de roteamento de veículo capacitado mais comumente encontrada na literatura é o problema de roteamento de veículos com balanceamento de rota.

Uma das questões que motivam o estudo do problema refere-se a satisfação do motorista, uma vez que a carga de trabalho destes devem ser aproximadas. Lee e Ueng [1998] introduziram o problema e apresentaram um modelo de programação inteira, no qual um problema multi-objetivo é transformado em mono-objetivo e então uma heurística é proposta solucioná-lo. Jozefowiez et al. [2002, 2005, 2007, 2009] também incluíram o objetivo balanceamento de rota ao problema de roteamento de veículo capacitado. Diversos métodos são abordados pelos autores dentre eles uma variação do algoritmo evolucionário multiobjetivo, NSGAII, busca tabu. Pasia et al. [2007]; Borgulya [2008] também estendem o problema de roteamento de veículos capacitado inserindo o balanceamento de rota como um dos objetivos do problema.

Outra extensão para o problema de roteamento de veículo capacitado é proposta por Murata e Itai [2005]. Os objetivos referem-se a minimização do número de veículos utilizados e minimização da maior rota, ou seja, a rota cujo o tempo gasto pelo veículo para concluí-la é o maior dentre todas as rotas.

Watanabe e Sakakibara [2007] incluíram novos objetivos ao problema a fim de obter soluções para o problema de roteamento de veículos capacitado de melhor qualidade. Geralmente as heurísticas utilizadas para solucionar este problema são compostas pelas fases: divisão dos consumidores e roteamento. Os autores adicionaram dois novos objetivos referentes a divisão dos consumidores: compactação dos grupos e conexão dos grupos. O primeiro destes objetivos avalia a densidade total dos grupos e o segundo avalia o grau em que consumidores vizinhos foram colocados no mesmo grupo. Assim, consumidores da mesma região geográfica tendem a ser atendidos pelo mesmo veículo, reduzindo os custos de transporte.

3.3.3 Problema de Roteamento de Veículo Aberto Multiobjetivo

O problema de roteamento de veículos aberto multiobjetivo com serviços de coleta e entrega de passageiros (*School Buses Routing Problem*) é encontrado em diversos estudos de casos reais. Muitos destes problemas referem-se ao transporte escolar na zona rural. Um exemplo dessa abordagem pode ser encontrada em [Corberán et al., 2002]. Os autores apresentam uma modelagem composta por dois objetivos conflitantes: minimização do número de veículos (ônibus escolar) e minimização do tempo máximo que um estudante trafega no interior do veículo. O segundo objetivo pode ser calculado pelo tamanho da rota com maior distância percorrida pelo veículo. Outro trabalho que aborda problema similar é apresentado por Pacheco e Marti [2006].

Bowerman et al. [1995] apresenta um estudo de caso real do problema de roteamento de ônibus escolar na zona urbana (município de Wellington, Ontário). Os objetivos do problema consistem em reduzir o número de veículos, reduzir a distância total percorrida pelos veículos, balanceamento de rotas, reduzir a distância percorrida pelos alunos de suas residências aos

pontos de ônibus e balanceamento de carga. Este último é obtido pela variação do número de alunos transportados ao longo de cada rota. O autor aplica uma heurística do tipo Dividir e Rotear para solucionar o problema tratado.

Outros trabalhos da literatura também abordam o problema de ônibus escolar na zona urbana. M. Spada e Liebling [2005] apresentam um estudo de caso em diversas cidades na Suíça e Euchi e Mraihi [2012] estudam problemas em cidades da Tunísia. Uma revisão da literatura detalhada dos trabalhos envolvendo problemas de roteamento de ônibus escolar é apresentado por Park e Kim [2010].

3.3.4 Problema de Roteamento de Veículo Periódico Multiobjetivo

Ribeiro e Lourenco [2001] aplicaram conceitos de otimização multiobjetivo a fim de melhorar os aspectos práticos do modelo. Como afirmado por Lee e Ueng [1998], o planejamento de rotas deve considerar o ganho de todos os envolvidos no processo, ou seja, organização, empregados e clientes. Os interesses de uma organização estão sempre voltados à redução de custo, assim, um dos objetivos do problema é redução dos custos de transporte representado pela distância total percorrida pelos veículos. Outro objetivo inserido ao modelo é balanceamento de rota. Neste caso, a fim de garantir maior satisfação dos motoristas. O terceiro objetivo incorporado refere-se a política de marketing. Em um ambiente cada vez mais competitivo muitas empresas adotam estratégias de relacionamento com seus clientes onde a lealdade e a amizade desempenham papel fundamental. A fim de satisfazer o consumidor, cada motorista deve atender sempre os mesmos clientes, enfatizando a relação pessoal entre motoristas e clientes como uma forma de melhorar os serviços prestados pela empresa. Uma vez que temos outras restrições envolvidas no problema, muitas vezes esta exigência precisa ser sacrificada, mas é importante tentar impor essa condição para os melhores clientes.

Chiang [2008] apresenta um algoritmo genético para solucionar o problema de roteamento de veículos com demandas flutuantes. A modelagem é composta por três objetivos: minimização da distância total percorrida pelos veículos, minimização da variação das cargas dos veículos e minimização do número de entregas divididas. Entregas divididas são aquelas que não puderam ser atendidas por um único veículo, em uma única visita.

Mourgaya e Vanderbeck [2007] apresenta um estudo de caso real do problema de roteamento de veículos periódico multiobjetivo no qual os autores buscam uma maior satisfação do motorista e melhor agrupamento dos consumidores nas rotas de acordo com sua posição geográfica. A satisfação do motorista é representada no balanceamento de rota incluído no conjunto de funções objetivo. Os autores apresentam um algoritmo exato baseado na geração de colunas para solucionar o modelo proposto por eles.

3.3.5 Outros Problemas de Roteamento de Veículo Multiobjetivo

Uma extensão do problema de roteamento dinâmico é apresentado por Chitty e Hernandez [2004]. Os objetivos do problema são minimizar o tempo médio gasto pelos veículos para atendimento das demandas e a variância do tempo médio. Os autores apresentam um algoritmo híbrido baseado no método colônia de formiga e programação dinâmica.

Tan et al. [2007] propõem uma extensão do problema de roteamento de veículos com demanda estocástica. Os autores propuseram um algoritmo evolutivo multiobjetivo para solucionar o problema cujos objetivos são minimizar a distância total percorrida pelos veículos, minimizar o número de veículos e minimizar a remuneração dos motoristas. A remuneração dos motoristas é baseada no tempo gasto para percorrer a rota.

O problema de cobertura de trajeto (*Covering Tour Problem*) é apresentado por Doerner et al. [2007]; Doerner e Hartl [2008]. Os autores apresentam um estudo de caso real do problema de roteamento de unidades de saúde móveis da região de Thiés no Senegal. Os métodos adotados pelos autores para solucionar o problema consistem em VEGA, MOGA e colônia de formiga. Os objetivos do problema consistem em aumentar a eficiência da implantação da força de trabalho medida pela razão entre o tempo gasto em procedimentos médicos e tempo gasto pela viagem acrescido do tempo de instalação da facilidade móvel, melhorar a acessibilidade média dada pela distância média que os habitantes precisam caminhar até alcançar uma facilidade, aumentar a cobertura medida pela porcentagem de habitantes que residem dentro de um raio de abrangência do ponto de atendimento.

Giannikos [1998]; Zografos e Androutsopoulos [2004]; Meng et al. [2005] apresentam um problema de localização e roteamento a fim de estudar casos reais de transporte de produtos perigosos. O objetivo do problema é minimizar os custos de transporte e propiciar maior segurança no transporte do produto. Para garantir segurança alguns objetivos são adicionados ao problema como: minimizar o risco total de acidentes, minimizar o total da população exposta ao produto.

Lacomme et al. [2006] abordam um estudo de caso real de coleta de lixo da cidade de Troyes, na França, no qual as arestas representam as ruas da cidade e as demandas se encontram nas arestas (*Arc Routing Problem*). Os objetivos do problema são a redução do tempo total gasto no conjunto de rotas e balanceamento de rota que pode ser obtido através da minimização da maior rota.

3.4 Conclusão

Este capítulo apresentou uma revisão dos trabalhos que envolvem problemas de roteamento de veículos com serviços de coleta e entrega e, também, problemas de roteamento de veículos com abordagem multiobjetivo.

Os problemas de roteamento de veículo com serviço de coleta e entrega são classificados como: problema de roteamento de veículos com coleta de retorno e problema de roteamento de veículos com coleta e entrega. Na primeira classe de problemas os veículos partem do depósito com os produtos a serem entregues aos consumidores e retornam ao depósito com a carga coletada. No segundo, os veículos partem do depósito vazio e os itens coletados devem ser utilizados para atender às demandas de entrega.

Os problemas de roteamento de veículo multiobjetivo foram classificados de acordo com as suas principais características e, dessa forma, foram definidos como: generalizações, extensões ou estudo de caso de problemas clássicos da literatura. As tabelas 3.1, 3.2 e 3.3 apresentam um resumo dos principais trabalhos envolvendo as três classes de problemas de roteamento de veículos multiobjetivo. Nos trabalhos que implementam algoritmos exatos é apresentado o número máximo de nós (ou consumidores) que o algoritmo foi capaz de solucionar. Com estes dados é possível concluir que os métodos exatos são capazes de solucionar problemas de pequena magnitude. Assim sendo, os métodos heurísticos são geralmente mais apropriados para solucionar os problemas reais encontrados nas grandes empresas de transporte.

Tabela 3.1: MOVRP - Extensão de problemas clássicos

Autor	**Ano**	**Problema**	**Método**
Lee e Ueng	1998	CVRP	Heurística
Ribeiro e Lourenço	2001	CVRP	LINGO (5 nós)
Jozefowiez et al.	2002,2005,2007,2009	CVRP	MOEA, NSGA-II, TS
Tan et al.	2003,2006	VRPTW	GA
Chitty e Hernandez	2004	DVRP	ACO
Murata e Itai	2005, 2007	CVRP	MOEA
Ombuki et al.	2006	VRPTW	MOGA
Tan et al.	2007	VRPSD	MOEA
Watanabe e Sakakibara	2007	CVRP	Heurística
Pasia et al.	2007	CVRP	Busca local/população
Borgulya	2008	CVRP	Heurística
Garcia-Najera e Bullinaria	2009; 2010; 2011	VRPTW	Algoritmo evolucionário
Garcia-Najera	2009; 2010	VRPTW	Algoritmo evolucionário

A revisão da literatura retrata uma crescente expansão de trabalhos envolvendo as duas variantes de problemas de roteamento de veículos: problemas com abordagens multiobjetivo e problemas com serviços de coleta e entrega.

Tabela 3.2: MOVRP - Generalização de problemas clássicos

Autor	**Ano**	**Problema**	**Método**
Sessomboon et al.	1998	VRPTW	GA
Hong e Park	1999	VRPTW	Heurística
Geiger	2001, 2008	VRPTW	MOGA
Rahoual et al.	2001	VRPTW	MOGA
Barán e Schaerer	2003	VRPTW	ACO
Tan et al.	2003,2006a,2006b	VRPTW	MOEA
Jozefowiez et al.	2007	CTP	MOEA e ϵ-restrito/Branch-and-Cut (120 nós)
Beham	2007	VRPTW	TS
Gutiérrez et al.	2008	VRPTW	PSO
Chiang	2008	PVRP	GA

Tabela 3.3: MOVRP - Estudo de casos reais

Autor	**Ano**	**Problema**	**Método**
Bowerman et al.	1995	OVRP	Heurística
Giannikos	1998	Localização e Roteamento	Programação por Metas(40 nós)
El-Sherbeny	2001,2002	VRP	SA
Corberan et al.	2002	OVRP	SS
M. Spada e Liebling	2005	OVRP	TS, SA
Pacheco e Marti	2006	OVRP	TS
Zografos e Androutsopoulos	2004	Localização e Roteamento	Heurística
Meng et al.	2005	Localização e Roteamento	Programação Dinâmica (22 nós)
Lacomme et al.	2006	ARP	MOGA
Mourgaya e Vanderbeck	2007	VRP	Geração de Colunas (50 nós)
Doerner et al.; Doerner e Hartl	2007; 2008	CTP	VEGA, MOGA, ACO
Sousa et al.	2011	VRPTW	GLPK (6 nós)

Capítulo 4

Problema de Roteamento de Veículos Multiobjetivo com Coleta Opcional

Este capítulo apresenta os principais trabalhos referentes aos problemas de roteamento de veículos multiobjetivo com serviços de coleta e entrega. Assim, é possível contextualizar em qual variante do problema geral de coleta e entrega o problema proposto nesta tese se enquadra. Após situar o problema abordado, este é formalmente definido e a formulação matemática proposta é apresentada.

O capítulo está organizado da seguinte forma: a seção 4.1 apresenta os principais trabalhos da literatura relacionados com o problema abordado. A seção 4.2 apresenta a definição formal do problema e a formulação matemática, mostrando que este pode ser considerado uma generalização do problema de roteamento de veículos com coleta e entrega simultânea. Por fim, o capítulo é concluído na seção 4.3.

4.1 Problema Geral de Coleta e Entrega Multiobjetivo

Ainda existem poucos trabalhos envolvendo problemas de roteamento de veículos com coleta e entrega e abordagem multiobjetivo. Os primeiros trabalhos tratam problemas reais de transporte de passageiros e de cargas entre duas localidades especificadas pelo consumidor [Jaw et al., 1986].

As generalizações do problema de roteamento de veículos com coleta e entrega encontradas na literatura, em geral, transformam a restrição de atendimento de todas as demandas de coletas em um objetivo. Um dos primeiros trabalhos que estuda esta abordagem foi apresentado por Süral e Bookbinder [2003], o qual foi denominado como *Single-Vehicle Routing Problem with Unrestricted Backhauls*. Os autores associam um lucro para cada demanda de

coleta caso ela seja atendida. Apesar da natureza multiobjetivo do problema, a modelagem proposta apresenta um objetivo único sendo ele o custo total de trasporte menos o lucro gerado pelas coletas efetuadas. Os autores apresentam um modelo de programação inteira mista, solucionado com o resolverdor CPLEX. A solução ótima é encontrada para instâncias com até 30 consumidores. Posteriormente, Gutierrez-Jarpa et al. [2009] apresentaram um algoritmo *branch-and-cut* para solucionar o mesmo problema, utilizando as mesmas instâncias para avaliar o algoritmo proposto.

Outra abordagem exata para esta classe de problemas pode ser encontrada em [Gutiérrez-Jarpa et al., 2010], no qual os autores apresentam um algoritmo *branch-and-price* para solucionar cinco variantes do problema de roteamento de veículos com entrega obrigatória e coleta opcional e, ainda, considerando janela de tempo para atendimento das demandas. Foram utilizadas instâncias com no máximo 50 consumidores.

O problema proposto por Gribkovskaia et al. [2008] apresenta uma modelagem diferenciada, permitindo que os clientes possam ser visitados duas vezes, uma para atender a demanda de entrega e outra para atendimento da demanda de coleta. Assim como nos trabalhos anteriores, a função objetivo é única, sendo composta pelo custo de transporte decrescido do lucro adquirido com as coletas efetuadas. Recentemente, Bruck et al. [2012] apresentou uma metaheurística híbrida para solucionar o mesmo problema. Ambos trabalhos utilizaram um conjunto de instâncias com até 101 consumidores.

Liao e Ting [2010] apresentam uma generalização do problema de coleta e entrega, onde a restrição de que todos os pontos de coleta devem ser visitados é relaxada, assim, podendo atender apenas alguns deles. Eles denominam esta variação de problema de entrega e coleta seletiva (SPDP, do inglês *Selective Pickup and Delivery Problem*). O objetivo do SPDP é estabelecer rotas de custo mínimo que recolham mercadorias em alguns pontos de coleta de forma que sejam suficientes para abastecer as demandas dos pontos de entrega. Os autores apresentam um algoritmo genético no qual foi avaliado em um conjunto de instâncias com até 350 consumidores.

Garcia-Najera [2012] mostra uma generalização do problema de roteamento de veículos com coleta de retorno agrupada (VRPCB). As demandas de coletas dos consumidores deixam de ser obrigatórias e passam a ser um dos objetivos do problema. Os autores apresentam uma abordagem multiobjetivo no qual consideram o número de veículos utilizados, o custo de transporte e o atendimento das demandas de coletas com igual importância. Algoritmos evolucionários foram utilizados para solucionar o problema proposto. O algoritmo foi aplicado a um conjunto de instâncias com até 200 consumidores.

Um estudo de caso real envolvendo 100 pontos de demanda de um problema de roteamento de veículo com coleta e entrega do tipo *one-to-one* de um empresa de transporte belga é feito por El-Sherbeny [2001]. Na empresa estudada, os consumidores requisitam que uma determi-

nada quantidade de carga seja trasportada de uma localidade a outra. As coletas efetuadas são destinadas a um ponto de entrega especificado pelo consumidor, sendo que estas demandas devem, ainda, ser atendidas de acordo com uma janela de tempo pré-definida. A empresa possui dois tipos de veículos que são utilizados para transporte de cargas. Dependendo do tipo de material transportado, deve-se utilizar caminhão coberto, caso contrário qualquer dos caminhões (coberto ou aberto) poderá ser utilizado. Os objetivos do problema são: minimizar o número de veículos, minimizar a duração total das rotas, minimizar o número total de caminhões cobertos, minimizar o número de caminhões abertos, balanceamento de rota, minimizar o tempo de carregamento e descarregamento, minimizar o tempo de espera devido às restrições de janela de tempo e minimizar o tempo de transporte total.

Outro problema de roteamento de veículos com coleta e entrega do tipo *one-to-one* é apresentado por Jaw et al. [1986]. O problema abordado trata-se da roteirização e programação de veículos (DARP, do inglês *Dial-a-Ride Problem*) no qual as cargas são pessoas (passageiros) e o tamanho das cargas são todas unitárias. Os passageiros devem ser recolhidos em locais pré-definidos e transportados para pontos conhecidos. O objetivo é minimizar o número de veículos, a distância percorrida, a quantidade de tempo que o passageiro permanece no veículo e a quantidade de tempo pelo qual a coleta ou entrega de um passageiro pode se desviar do desejado. O autor utiliza um banco de dados real contendo 2600 clientes e cerca de 20 veículos ativos simultaneamente. Uma heurística baseada em inserção foi aplicada na resolução das instâncias.

A figura 4.1 apresenta a classificação do problema geral de coleta e entrega com os trabalhos relacionados cujos problemas apresentados remetem a abordagem multiobjetivo. Os trabalhos destacados com campos na cor mais escura são trabalhos que apresentam modelagem de problemas com múltiplos objetivos e os destacados com campos na cor mais clara são trabalhos no qual o problema exibe caractetísticas multiobjetivo mas a formulação matemática é composta por uma agregação de objetivos. Em geral, a função objetivo deste problema consiste em minimizar os custos de transporte subtraídos do lucro adquirido pelas demandas de coletas atendidas durante o trajeto. É importante ressaltar que os trabalhos relacionados que são mostrados na figura têm algumas características no qual se pode classificá-los como uma variante do problema geral de coleta e entrega, mas muitos deles apresentam também outras características tais como janela de tempo, problemas com um único veículo, entre outros. Assim sendo, na figura há um destaque apenas nas características que envolvem o GPDP.

Na figura mencionada é possível concluir que existem poucos trabalhos que exploram a abordagem multiobjetivo das variantes do problema geral de coleta e entrega, em especial o problema de roteamento de veículos com coleta e entrega simultânea. Portanto, na figura, o campo com bordas destacadas e fundo escuro apresenta o problema abordado nesta tese intitulado: problema de roteamento de veículos multiobjetivo com coleta opcional, que se baseia na generalizacão do VRPSPD.

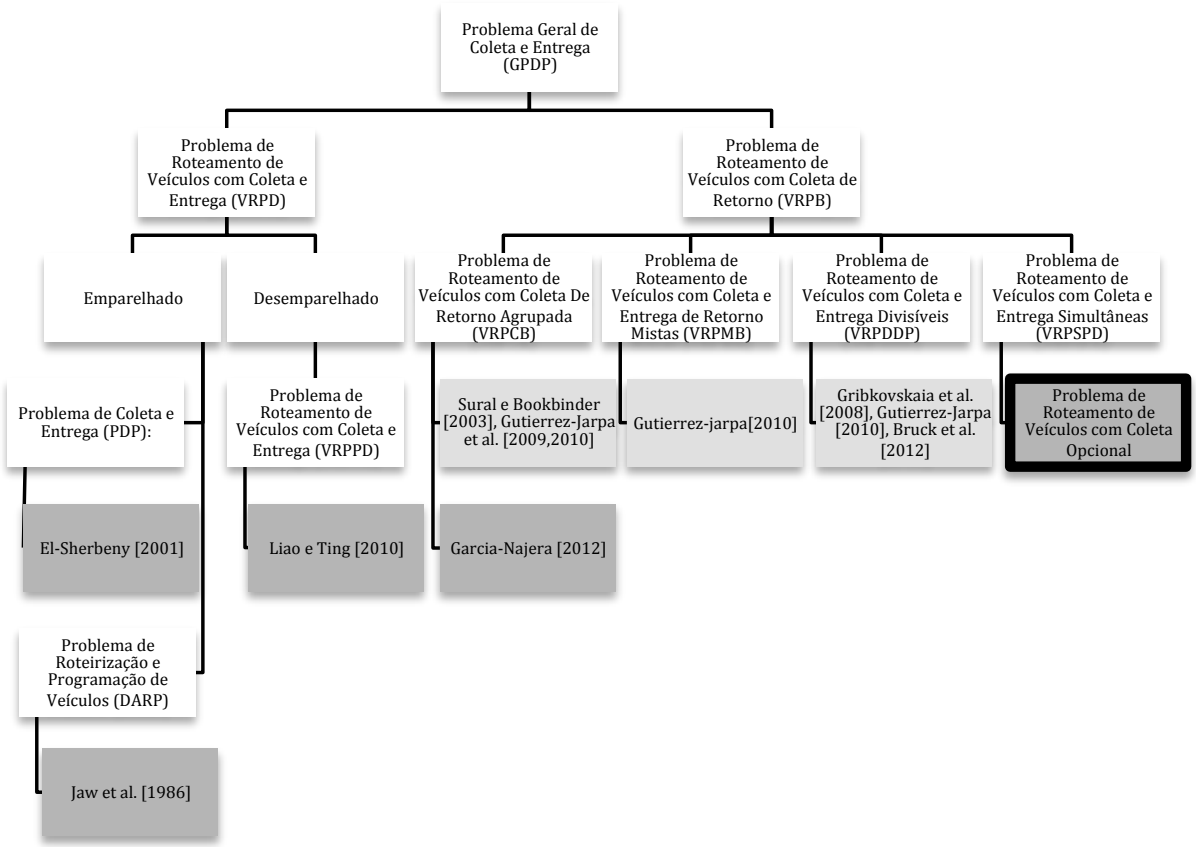


Figura 4.1: Variações do problema geral de coleta e entrega e os trabalhos que exploram uma abordagem multiobjetivo destes problemas.

4.2 Problema de Roteamento de Veículos Multiobjetivo com Coleta Opcional

Conforme apresentado na seção anterior, existem poucos trabalhos na literatura que apresentam uma abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega. Nestes trabalhos, apesar da característica multiobjetivo, o problema é solucionado de forma mono-objetivo, no qual associa-se um lucro às coletas efetuadas e o objetivo passa a ser, então, minimizar o custo de transporte subtraído do lucro gerado pelas coletas efetuadas.

O problema abordado neste trabalho é denominado: problema de roteamento de veículos multiobjetivo com coleta opcional (MOVRPOP, do inglês *Multiobjective Vehicle Routing Problem with Optional Pickup*). Seguindo a classificação proposta por Jozefowiez et al. [2008b] para problemas de roteamento multiobjetivo, este problema é uma generalização do problema de roteamento de veículos com coleta e entrega simultânea (VRPSPD). A generalização se dá pela transformação de uma restrição em uma função objetivo, assim sendo, neste problema

a restrição de atendimento de todas as demandas de coletas é transformada em uma função objetivo. Para facilitar a resolução do problema, este objetivo é transformado em uma função de minimização no qual deseja-se minimizar o número total de itens não coletados. Assim, o problema de roteamento de veículos multiobjetivo com coleta opcional apresenta duas funções objetivo: (i) minimização do custo total das rotas ou distância total percorrida e (ii) minimização do número total de itens não coletados.

No MOVRPOP todas as demandas de entrega devem ser satisfeitas, porém, o atendimento das demandas de coleta é opcional, mas desejável. Então, o objetivo é construir rotas que minimizem o custo de se realizar o trajeto, atendendo a todas as demandas de entrega, e ao mesmo tempo, minimizar o número de demandas de coleta não realizadas.

Para atendimento das demandas, existe uma frota de veículos homogêneos e de capacidade limitada, no qual esta capacidade não deve ser extrapolada em nenhum momento do trajeto. Outra restrição é que um consumidor deve ser visitado uma única vez e por um único veículo, ou seja, se este consumidor tiver a demanda de coleta atendida, esta deverá ser realizada simultaneamente com a demanda de entrega.

Os dois objetivos a serem otimizados são conflitantes, ou seja, a melhora em um provoca a piora do outro. Portanto, não existe uma solução que seja ótima para ambos os objetivos simultaneamente. Isso leva ao conceito de dominância de Pareto, gerando um conjunto de soluções não dominadas.

O MOVRPOP pode ser reduzido em tempo polinomial ao VRPSPD [Ziviani, 2004]. Utilizando o método tradicional de resolução de problemas multiobjetivo, por exemplo o método ϵ-Restrito, cada problema modelado seguindo o método citado recai ao VRPSPD, considerando que a função objetivo referente ao atendimento das demandas de coleta seja transformada em restrição. Sendo o MOVRPOP reduzido polinomialmente ao VRPSPD e sendo o VRPSPD um problema NP-Difícil [Dell'Amico et al., 2006], logo o MOVRPOP é também NP-Difícil por ser um problema de otimização.

4.2.1 Modelagem matemática

A modelagem matemática para o problema proposto é uma adaptação da modelagem proposta por Montané e Galvão [2006] para o problema de roteamento de veículos com coleta e entrega simultânea. Esta adaptação se diferencia da modelagem original dado que: (i) uma segunda função objetivo, com a finalidade minimizar o número de coletas não realizadas, é adicionada e; (ii) é removida a restrição que todas as demandas de coleta devem ser realizadas.

O problema de roteamento de veículos multiobjetivo com coleta opcional pode ser definido como: dado um grafo completo $G = (V, A)$, onde $V = \{v_0, v_1, v_2, \ldots, v_n\}$ é o conjunto de vértices e $A = \{(v_i, v_j) : i, j \in V \ e \ i \neq j\}$ o conjunto de arestas. O vértice v_0 representa o

depósito e os demais vértices representam os consumidores. Cada aresta (v_i, v_j) tem um valor $c_{ij} \geq 0$ associado que representa o custo de se alcançar o vértice v_j a partir do vértice v_i. Cada consumidor v_i tem uma demanda d_i de entrega e uma demanda p_i de coleta. Tem-se disponível uma frota de k_{max} veículos homogêneos de capacidade Q para atendimento das demandas. O parâmetro y_{ij} é o somatório das cargas coletadas entre o depósito e o nó v_i (inclusive) dirigida ao nó v_j, enquanto z_{ij} é o somatório das cargas entregues entre o nó v_i (exclusive) e o depósito dirigida ao nó v_j.

As variáveis de decisão do problema consistem em:

$$x_{ij}^k = \begin{cases} 1 & \text{, se arco } (v_i, v_j) \text{ faz parte da rota trafegada pelo veículo } k, \\ 0 & \text{, caso contrário.} \end{cases}$$

$$\ell_j = \begin{cases} 1 & \text{, se a demanda de coleta do consumidor } v_j \text{ é satisfeita,} \\ 0 & \text{, caso contrário.} \end{cases}$$

Dada a definição formal e as variáveis de decisão do problema, a modelagem matemática para o MOVRPOP é definida a seguir.

$$Min\ f(\mathbf{x}) \ = \ \sum_{k=1}^{k_{max}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k \tag{4.1}$$

$$Min\ f(\boldsymbol{\ell}) \ = \ \sum_{j=1}^{n} p_j (1 - \ell_j) \tag{4.2}$$

sujeito a

$$\sum_{i=0}^{n} \sum_{k=1}^{k_{max}} x_{ij}^k = 1, \qquad j = 1, \ldots, n \tag{4.3}$$

$$\sum_{i=0}^{n} x_{ij}^k - \sum_{i=0}^{n} x_{ji}^k = 0, \qquad j = 0, \ldots, n \text{ e } k = 0, \ldots, k_{max} \tag{4.4}$$

$$\sum_{j=1}^{n} x_{0j}^k \leq 1, \qquad k = 1, \ldots, k_{max} \tag{4.5}$$

$$\sum_{i=0}^{n} z_{ij} - \sum_{i=0}^{n} z_{ji} = d_j, \qquad \forall j \neq 0 \tag{4.6}$$

$$\sum_{i=0}^{n} y_{ji} - \sum_{i=0}^{n} y_{ij} = p_j \ell_j, \qquad \forall j \neq 0 \tag{4.7}$$

$$y_{ij} + z_{ij} \leq Q \sum_{k=1}^{k_{max}} x_{ij}^{k}, \qquad i, j = 0, \ldots, n \tag{4.8}$$

$$x_{ij}, \ell_j \in \{0, 1\}, \qquad i, j = 0, \ldots, n \tag{4.9}$$

$$y_{ij} \geq 0, \qquad i, j = 0, \ldots, n \tag{4.10}$$

$$z_{ij} \geq 0, \qquad i, j = 0, \ldots, n \tag{4.11}$$

A função objetivo, dada na equação 6.1, visa a minimização do custo (distância percorrida), já a equação 6.2, apresenta a segunda função objetivo, que visa a minimização do somatório das coletas não realizadas.

Estas funções objetivo devem satisfazer a um conjunto de restrições. A primeira restrição (equação 4.3) indica que cada ponto de demanda deve ser visitado por um único veículo. A equação 4.4 representa a restrição de conservação do fluxo. A equação 4.5 indica que k_{max} veículos, no máximo, podem ser utilizados. A equação 4.6 obriga a satisfação de todas as demandas de entrega. A equação 4.7 garante o atendimento da demanda de coleta do consumidor j quando a variável de decisão ℓ_j assumir valor unitário. A equação 4.8 define que as demandas devem ser transportadas nos arcos incluídos na solução e ainda impõem um limite para a carga total transportada pelo veículo. A equação 4.9 representa a restrição de integralidade e as equações 4.10 e 4.11 representam as restrições de não-negatividade para demandas de coleta e entrega, respectivamente.

4.3 Conclusão

Neste capítulo foram apresentados os principais problemas de roteamento de veículos que apresentam uma abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega. Pode-se concluir que existem poucas abordagens multiobjetivo e alguns dos trabalhos encontrados consideram múltiplos critérios de desempenho, mas a modelagem matemática é constituída por uma única função-objetivo, sendo ela a minimização do custo de transporte reduzido do lucro adquirido com as coletas efetuadas.

Para explorar a interseção entre problemas de roteamento de veículos multiobjetivo e os problemas gerais de coleta e entrega, este capítulo propõe o problema de roteamento de veículos multiobjetivo com coleta opcional e apresenta a sua definição e formulação matemática, tornando-o uma generalização do problema de roteamento de veículos com coleta e entrega simultânea. A formulação proposta apresenta duas funções objetivo referentes ao custo total de transporte e ao número de coletas atendidas. Este último objetivo é tipicamente tratado como uma restrição no problema de roteamento de veículos com coleta e entrega simultânea.

Em algumas situações reais a entrega é obrigatória, enquanto as demandas de coletas podem ser postergadas. Dessa forma, o tomador de decisões poderá escolher quais coletas devem ser efetuadas dado o cenário atual. No problema proposto todas as demandas de coleta têm igual importância para serem atendidas, não havendo um lucro específico para cada coleta efetuada como é feito nos trabalhos encontrados na literatura. Dessa forma, o problema de roteamento de veículos multiobjetivo com coleta opcional considera apenas o volume da demanda de coleta devido a capacidade limitada do veículo.

Capítulo 5

Estratégias de Solução

Neste capítulo são apresentadas as ferramentas e métodos utilizados para resolução de problemas de roteamento de veículos multiobjetivo com coleta opcional. Na seção 5.1 é apresentada a estrutura de dados utilizada para representação das informações do problema e algumas técnicas para melhorar o desempenho dos algoritmos. O conjunto de estruturas de vizinhança e os mecanismos de perturbação utilizados pelos algoritmos são detalhado nas seções 5.2 e 5.3.

Para solucionar problemas de roteamento de veículos mono-objetivo, muito comum nos algoritmos implementados para solucionar o problema bi-objetivo, este trabalho propõe um algoritmo baseado na busca geral de vizinhança variável, descrito na seção 5.4.

Nas subseções seguintes, 5.5, 5.6, 5.7 e 5.8, são apresentados os algoritmos propostos para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, foco deste trabalho.

5.1 Estrutura de dados

A maneira como os dados de entrada de um problema são armazenados interfere diretamente na eficiência dos algoritmos implementados. Os problemas de roteamento de veículos em geral podem ser representados através de um grafo, onde os consumidores e o depósito compõem o conjunto de vértices e as arestas representam as conexões entre consumidores e entre consumidores e depósito.

Os dados de entrada do problema de roteamento de veículo tratado são representados por um grafo completo, ou seja, existe uma aresta conectando cada par de vértices. Existem diversas formas de representar um grafo e, neste trabalho, uma nova estrutura é proposta a fim de unir as vantagens de duas estruturas bem tradicionais: matriz de adjacência e lista de adjacência.

Além da estrutura de dados proposta para armazenar os dados de entrada do problema, esta seção descreve, também, a estrutura utilizada para armazenar a solução para o problema.

5.1.1 Dados do problema

Conforme mencionado no início da seção, a estrutura de dados grafo é utilizada para armazenamento e manipulação dos dados de entrada do problema estudado. Um grafo é definido como $G = (V, A)$, onde $V = \{v_1, v_2, \ldots, v_n\}$ é um conjunto finito de vértices e A é o conjunto de arestas (v_i, v_j) que ligam pares de vértices.

Um grafo é uma estrutura matemática usada para representar as relações entre objetos discretos. Estas estruturas são muito utilizadas no âmbito da otimização combinatória, pois permitem a representação de diversos problemas, dentre eles, os problemas de roteamento de veículos, foco deste trabalho. Neste tipo de problema, os vértices representam os consumidores e o depósito, e as arestas representam a conexão entre pares de consumidores e a conexão entre consumidores e depósito, tratando-se, em geral, de um grafo completo. Para cada aresta existe um valor associado indicando o seu custo. Logo, o problema é representado por um grafo completo ponderado. A representação é ilustrada na figura 5.1.

As estruturas de dados mais tradicionais para representar um grafo computacionalmente são através da matriz ou lista de adjacência [Cormen et al., 2001]. Estas estruturas, apresentadas na figura 5.1, permitem uma fácil manipulação dos dados e possuem algumas vantagens e desvantagens.

A matriz de adjacência é vantajosa por permitir buscar informações de uma determinada aresta a uma complexidade computacional constante ($O(1)$). No exemplo apresentado na figura 5.1(a), a partir de um acesso a matriz ($matriz[1][6]$) pode-se afirmar que existe uma aresta que conecta os vértices 1 e 6 de custo 8. Por outro lado, dado o vértice 1, para verificar qual o vértice mais próximo a ele é necessário percorrer toda a linha 1 da matriz, sendo a complexidade computacional dessa busca $O(n)$, sendo n o número de vértices do grafo.

A lista de adjacência permite que os nós adjacentes possam ser ordenados pelo custo possibilitando encontrar o vértice adjacente mais próximo ou mais distante (se a lista for duplamente encadeada) a um custo constante ($O(1)$). No exemplo apresentado na figura 5.1 (b), a partir de um acesso a lista de adjacência do vértice 1, pode-se afirmar que o vértice mais próximo a ele é o vértice 2. A desvantagem dessa estrutura é que verificar o custo de uma determinada aresta é, no pior caso, $O(n)$. Por exemplo, para verificar o custo da aresta $(1, 4)$ é necessário percorrer toda a lista de adjacência do vértice 1.

Para aproveitar as vantagens de cada uma destas representações, a implementação utilizada neste trabalho encapsula ambas estruturas: matriz de adjacência e lista de adjacência. A lista de adjacência mantém os vértices ordenados em relação ao custo. Esta ordenação permite

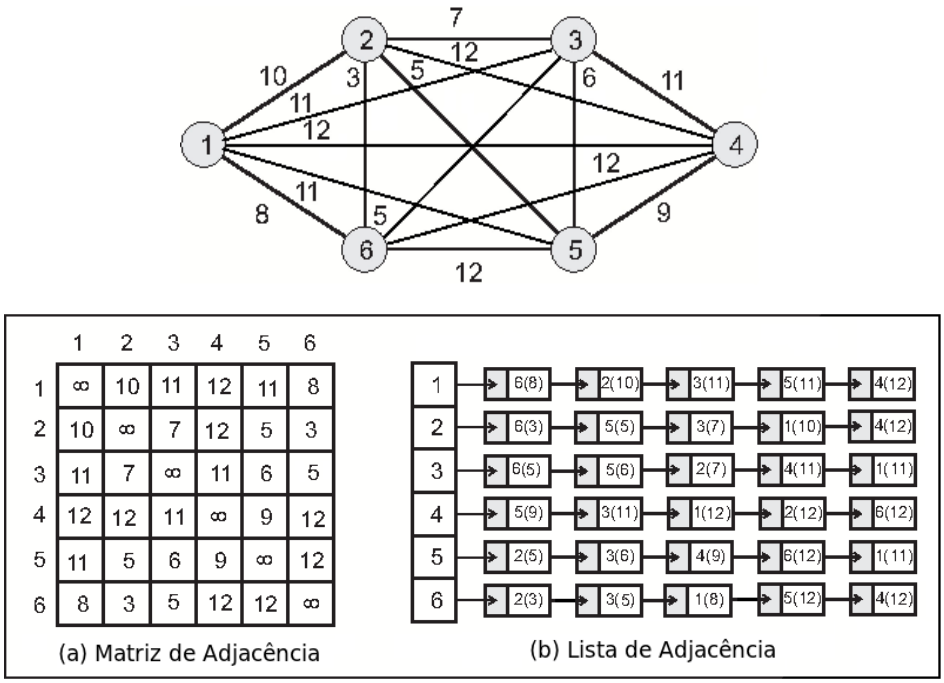


Figura 5.1: Estruturas de dados grafo: dado um grafo, a matriz de adjacência (a) e lista de adjacência (b) são exemplos de estrutura utilizadas para manipular um grafo. A lista de adjacência está ordenada de acordo com o custo dos vértices.

que a busca pelo vértice mais próximo a outro seja realizada em menor custo computacional, O(1). Diversas heurísticas efetuam esta buscas inúmeras vezes, demostrando a importância da redução computacional deste procedimento. Além disso, não há perda de desempenho para efetuar a ordenação, já que existem algoritmos com complexidade logarítmica, e a leitura dos dados do problema para construção do grafo é feita uma única vez no início da execução.

Uma desvantagem de utilizar estas duas representações simultaneamente é o aumento do consumo de recursos para armazenamento, porém, como os computadores hoje possuem grandes quantidades de memória, isso já não é mais um limitador.

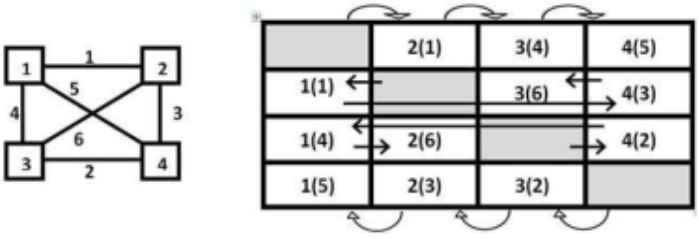


Figura 5.2: Estrutura proposta para um grafo: dado um grafo, a estrutura de dados proposta encapsula tanto a matriz de adjacência quanto a lista de adjacência (ordenada pelo custo dos vértices adjacentes).

Para a nova estrutura, foi implementada uma matriz de adjacência onde cada elemento (i, j) da matriz é uma célula de uma lista duplamente encadeada composta por: um vértice adjacente a v_i, o custo da aresta (v_i, v_j) e dois ponteiros. Partindo da diagonal principal da matriz, os ponteiros criam uma lista duplamente encadeada ordenada pelo custos das arestas. Os ponteiros da direita e esquerda ordenam de maneira crescente e decrescente, respectivamente, os vértices adjacentes a v_i pelo custo da aresta (v_i, v_j). Na figura 5.2 a forma como os dados são mantidos na estrutura de dados grafo proposta é apresentada. Nesta figura foi representado apenas um dos ponteiros para facilitar a visualização.

Além da estrutura de dados utilizada para representar o grafo, outra estrutura auxiliar foi empregada para armazenar somente as demandas dos consumidores. Neste trabalho, as demandas de coletas e entrega do consumidor v_i é armazenada em um vetor de tamanho n na posição i.

5.1.2 Representação de uma solução

A representação de uma solução para problemas de roteamento de veículos pode ser feita através de um conjunto de vetores, cada um representando uma rota e armazenando os consumidores (vértices) na ordem em que devem ser visitados.

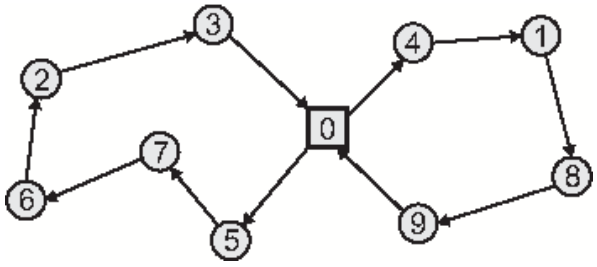
Porém, nos problemas de roteamento multiobjetivo com coleta opcional, além da ordem em que os consumidores são visitados, deve-se representar também quais deles terão a demanda de coleta atendida. Para isso é utilizado um vetor binário, informando, para cada consumidor, se a demanda de coleta será atendida ou não. A forma como uma solução para o problema de roteamento de veículos multiobjetivo com coleta opcional é representada é mostrada na figura 5.3.

5.1.3 Análise da Viabilidade das Rotas

Durante a construção ou na alteração de uma solução de um problema de roteamento de veículos com serviços de coleta e entrega é necessário verificar a viabilidade desta solução. A restrição do problema referente a capacidade máxima do veículo deve ser verificada sempre que um consumidor é inserido em alguma rota, uma vez que as demandas de coleta e entrega desta rota não devem extrapolar a capacidade máxima do veículo.

Dada a rota na qual um consumidor será inserido, uma implementação simples para este procedimento consiste em calcular a carga do veículo após o atendimento de cada demanda dos consumidores desta rota, verificando, assim, se a capacidade do veículo não será extrapolada com a inserção das demandas de entrega e coleta do novo consumidor. O algoritmo 5.1 apresenta o pseudocódigo para este procedimento. No algoritmo a seguinte notação é utilizada:

- R: rota na qual um consumidor será inserido.



Rota 1: [0 - 4 - 1 - 8 - 9 - 0]
Rota 2: [0 - 5 - 7 - 6 - 2 - 3 - 0]
Estado Coleta: [1 - 1 - 0 - 1 - 0 - 0 - 1 - 1 - 1]

Figura 5.3: Representação de uma solução para o problema de roteamento de veículos multiobjetivo com coleta opcional: os dois primeiros vetores representam as rotas que compõem a solução, indicando a ordem em que os consumidores devem ser visitados, sempre começando e terminado no depósito, representado pelo número 0. O último vetor informa quais deles terão a demanda de coleta atendida, representada pelo valor 1, ou 0 caso contrário. A primeira posição se refere ao consumidor 1, a segunda posição ao consumidor 2, e assim sucessivamente.

- r_j: consumidor pertencente a rota R, tal que $j = \{v_0, r_1, r_2, r_3, ..., v_0\}$. O primeiro e o último consumidor representa o depósito (v_0) e os demais elementos estão associados a um determinado consumidor $v_i \in V$
- $|R|$: o comprimento da rota, que é o número de consumidores presentes nela. Sendo as posições da rota numeradas a partir do zero, então, para uma rota com $|R| = 7$, as suas posições variam no intervalo $[0 - 6]$.
- v_{new}: novo consumidor a ser inserido na rota R, tal que $v_{new} \in V$.
- p: a posição na rota R em que o consumidor v_{new} será inserido, tal que $0 < p < |R|$.
- $E[i]$: a demanda de entrega do consumidor $v_i \in V$.
- $C[i]$: a demanda de coleta do consumidor $v_i \in V$.
- Q: a capacidade máxima do veículo.
- $SomaEntregas[j]$: é a soma das demandas de entrega do j-ésimo consumidor (r_j), exclusive, até o último consumidor presente na rota (v_0). Por exemplo, dada uma rota com 7 consumidores, se $SomaEntregas[3]$ é igual a 50, significa que a soma das demandas de entrega dos consumidores presentes no intervalo $(3 - 0]$ da rota é igual a 50.
- $SomaColetas[j]$: é a soma das demandas de coleta do primeiro consumidor (v_0) até o j-ésimo consumidor (r_j), inclusive, presente na rota. Por exemplo, dada uma rota com

7 consumidores, se $SomaColetas[4]$ é igual a 74, significa que a soma das demandas de coleta dos consumidores presentes no intervalo $[0-4]$ da rota é igual a 74.

A verificação de viabilidade referente a capacidade máxima do veículo deve ser feita tanto no depósito, considerando as demandas de entregas dos consumidores que partem do depósito e as demandas de coleta dos consumidores que retornam ao depósito, e em cada consumidor considerando a demanda líquida (demanda de coleta - demanda de entrega) dos consumidores acrescida a carga do veículo.

Algoritmo 5.1 Verificar a viabilidade de inserção de um consumidor em uma rota

1: {Verifica a carga do veículo ao partir e retornar ao depósito}
2: **se** $(SomaEntregas[0] + E[v_{new}]) > Q$ **então**
3: **retornar falso**;
4: **fim se**
5: **se** $(SomaColetas[|R|-1] + C[v_{new}]) > Q$ **então**
6: **retornar falso**;
7: **fim se**
{Inicia a verificação ponto a ponto}
8: carga $\leftarrow (somaEntregas[0] + E[v_{new}])$;
9: **para** $j = 0$ **até** $p-1$ **faça**
10: demanda liquida $\leftarrow (C[r_j] - E[r_j])$;
11: **se** (carga + demanda liquida) $> Q$ **então**
12: **retornar falso**;
13: **fim se**
14: carga $\leftarrow$ (carga + demanda liquida);
15: **fim para**
16: demanda liquida $\leftarrow (C[v_{new}] - E[v_{new}])$;
17: **se** (carga + demanda liquida) $> Q$ **então**
18: **retornar falso**;
19: **fim se**
20: carga $\leftarrow$ (carga + demanda liquida);
21: **para** $j = p$ **até** $|R| - 1$ **faça**
22: demanda liquida $\leftarrow (C[r_j] - E[r_j])$;
23: **se** (carga + demanda liquida) $> Q$ **então**
24: **retornar falso**;
25: **fim se**
26: carga $\leftarrow$ (carga + demanda liquida);
27: **fim para**
{Então satisfaz a restrição de capacidade máxima do veículo}
28: **retornar verdadeiro**;

O algoritmo 5.1 inicialmente confirma se o somatório das demandas de entrega e coletas dos consumidores $r_j \in R$ acrescido da demanda de entrega $E[v_{new}]$ e coleta $C[v_{new}]$ do consumidor v_{new} a ser inserido em R não extrapola a capacidade Q do veículo (linhas 1–7) . Na etapa seguinte, verifica-se a viabilidade em cada consumidor $r_j \in R$ até o ponto p de inserção do

novo consumidor v_{new}, calculando a carga atual do veículo acrescido da demanda líquida em cada consumidor r_j (linhas 8–15). No ponto p de inserção do novo consumidor v_{new}, faz-se a mesma verificação, calculando a carga do veículo (carga atual + demanda liquida) caso v_{new} seja inserido na posição p (linhas 16–19). Seguindo esta mesma ideia, a viabilidade é analisada nos consumidores r_j seguintes da rota R até o retorno ao depósito v_0 (linhas 20–27). Se, em nenhum dos pontos da rota a restrição de capacidade máxima do veículo é violada, então é retornado *verdadeiro*, indicando que a inserção do consumidor v_{new} na posição p da rota é viável.

Apesar de sua fácil implementação, este algoritmo tem uma complexidade computacional $O(|R|)$. Como a avaliação da viabilidade é realizada inúmeras vezes, qualquer otimização realizada melhora o desempenho dos algoritmos que o utilizam. Assim, é proposto o armazenamento de algumas informações adicionais sobre a carga do veículo ao longo da rota, tornando mais eficiente esta análise de viabilidade.

As informações que devem ser armazenadas são: um vetor $infoCarga$ que indica a carga presente no veículo em cada posição da rota e uma variável $maxPosicao$ que indica em qual ponto da rota o veículo apresenta a maior carga. Com estas informações, o algoritmo 5.2 apresenta o pseudo-código de como pode ser feito a verificação de viabilidade das rotas.

Inicialmente o algoritmo 5.2 analisa se a carga do veículo não extrapola a capacidade máxima quando o veículo parte do depósito e quando ele retorna ao depósito (linhas 2–7). Em seguida, verifica se a posição onde o veículo apresenta maior carga está antes ou depois da posição que se deseja inserir o consumidor (linha 9).

Caso o consumidor seja inserido antes da posição na qual o veículo atinge o ponto de maior carga, então a carga máxima do veículo será acrescida apenas da demanda de coleta do consumidor. Assim sendo, apenas esta averiguação será efetuada (linha 10) uma vez que não será preciso calcular a carga do veículo em todas as posições após p, pois se a capacidade máxima do veículo não for extrapolada no ponto em que o veículo apresenta a maior carga transportada, ela não será extrapolada nas demais posições após p. A carga do veículo no interior da rota só será calculada se a demanda de entrega do consumidor v_{new} a ser inserido for maior que a sua demanda de coleta (linha 13). Caso isso ocorra, será necessário calcular a carga do veículo nas posições anteriores a posição p onde ocorrerá a inserção do consumidor (linhas 14–18). Neste caso, não se pode prever se em algum ponto intermediário entre o depósito e o ponto p de inserção do consumidor v_{new} haverá alguma inviabilidade.

Quando p estiver após o ponto de maior carga, o procedimento é análogo (linhas 20–32). No fim, se o veículo não extrapola sua capacidade máxima, então é retornado *verdadeiro*, indicando que a inserção do consumidor na posição p é viável (linha 33).

Neste procedimento, quando $p \leq maxPosicao$ e $E[v_{new}] \leq C[v_{new}]$ ou $p > maxPosicao$

Algoritmo 5.2 Verificar a viabilidade de inserção de um consumidor em uma rota (otimizado)

1: {Verifica a carga do veículo ao partir e retornar ao depósito}
2: **se** $(SomaEntregas[0] + E[v_{new}]) > Q$ **então**
3: **retornar falso**;
4: **fim se**
5: **se** $(SomaColetas[|R| - 1] + C[v_{new}]) > Q$ **então**
6: **retornar falso**;
7: **fim se**
8: {Verifica se o local onde o consumidor será inserido está antes do ponto de maior carga}
9: **se** $p \leq maxPosicao$ **então**
10: **se** $(infoCarga[maxPosicao] + C[v_{new}]) > Q$ **então**
11: **retornar falso**;
12: **fim se**
13: **se** $E[v_{new}] > C[v_{new}]$ **então**
14: **para** $j = 0$ **até** $p - 1$ **faça**
15: **se** $(infoCarga[r_j] + E[v_{new}]) > Q$ **então**
16: **retornar falso**;
17: **fim se**
18: **fim para**
19: **fim se**
20: **senão**
21: {Consumidor será inserido depois do ponto de maior carga}
22: **se** $(infoCarga[maxPosicao] + E[v_{new}]) > Q$ **então**
23: **retornar falso**;
24: **fim se**
25: **se** $E[v_{new}] < C[v_{new}]$ **então**
26: **para** $j = p$ **até** $|R| - 1$ **faça**
27: **se** $(infoCarga[r_j] + C[v_{new}]) > Q$ **então**
28: **retornar falso**;
29: **fim se**
30: **fim para**
31: **fim se**
32: **fim se**
33: **retornar verdadeiro**;

e $C[v_{new}] \leq E[v_{new}]$, tem-se o melhor caso, com complexidade $O(1)$, pois ele não executa nenhum bloco de repetição. Apenas no pior caso o algoritmo 5.2 apresenta complexidade $O(|R|)$.

O pior caso ocorre em duas situações: (i) quando se trata da inserção de um consumidor v_{new} imediatamente anterior à posição de maior carga, e a posição de maior carga ocorre no final da rota, e a demanda de entrega do consumidor v_{new} é maior que a demanda de coleta v_{new}; (ii) quando ocorre a inserção de um consumidor v_{new} logo após a posição de maior carga, e a posição de maior carga ocorre no início da rota, e a demanda de coleta do consumidor v_{new} é maior que a demanda de coleta de x. Em ambos casos, as linhas 14–18 ou 26–30 serão executadas aproximadamente $|R|$ vezes.

A análise de viabilidade é ilustrada na figura 5.4. Neste exemplo, é avaliado se a inserção do consumidor $v_{new} = 6$ entre os consumidores 7 e 2 ($p = 3$), considerando que o veículo para o transporte tem uma capacidade máxima de $Q = 15$. Primeiramente, foi verificado se a capacidade do veículo é extrapolada considerando o somatório das demandas de entrega que partem do depósito e o somatório das demandas de coleta que retornam ao depósito. Em seguida, verifica-se a carga do veículo no ponto de maior carga, indicado por *maxPosicao*. Este é o ponto mais crítico da rota e onde geralmente ocorrem as inviabilidades. Como a inserção do consumidor 6 é realizada após este ponto, então a carga do veículo (9) é acrescida apenas da demanda de entrega do consumidor 6 ($E[6] = 3$). Assim, esta soma ($9 + 3 = 12$) é inferior a capacidade máxima do veículo ($Q = 15$), garantindo a viabilidade da rota até o ponto de inserção do consumidor. No restante da rota, a carga do veículo será acrescida apenas da demanda de coleta do consumidor 6 ($C[6] = 2$). Sendo a demanda de coleta ($C[6] = 2$) menor que a demanda de entrega ($E[6] = 3$), então garante-se a viabiliade no restante da rota. Em outras palavras, se no ponto de maior carga for acrescido um valor maior que nos demais pontos da rota, então certamente a carga do veículo nos demais pontos será inferior a este valor calculado.

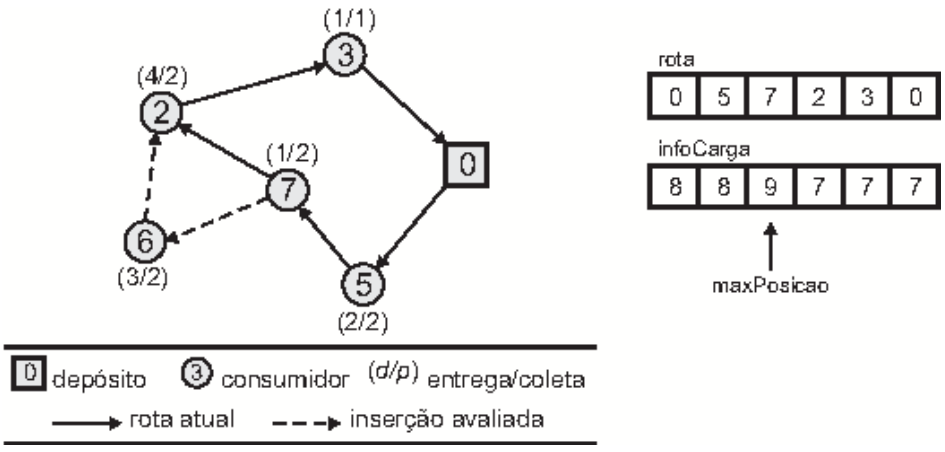


Figura 5.4: Análise de viabilidade de rotas

5.2 Estruturas de Vizinhança

Para explorar o espaço de busca e encontrar soluções que apresentem uma melhor avaliação para problemas de roteamento de veículos com serviços de coleta e entrega, são utilizadas um conjunto de estruturas de vizinhança. Estas estruturas podem ser divididas em dois grupos: inter-rotas e intra-rota.

As estruturas de vizinhança inter-rotas são aquelas que envolvem duas ou mais rotas no movimento. Já as intra-rotas envolvem apenas uma única rota no movimento. Para todas as estruturas foi utilizada a estratégia de exploração melhor-aprimorante, que avalia todas as soluções vizinhas antes do movimento ser aplicado. As subseções 5.2.1 e 5.2.2 descrevem as estruturas de vizinhança utilizadas.

5.2.1 Estruturas de Vizinhança Inter-Rotas

Neste trabalho foram implementadas dois tipos de estruturas de vizinhança inter-rotas: que exploram soluções levando em consideração o objetivo custo de transporte e outras que consideram, além do custo de transporte, o objetivo de minimização das demandas de coleta não atendidas.

As funções de vizinhança relatadas a seguir visam a redução do custo de transporte, por este ser considerado o objetivo mais difícil de ser alcançado. O estado das demandas de coleta dos consumidores (atendido e não atendido) permanece inalterado e somente a posição dos mesmos na rota é modificada. Assim sendo, neste trabalho foi explorado um grande número de funções que mantém a quantidade de itens não coletados inalterados e busca a redução do custo de transporte. Estas funções de vizinhanças são comumente encontradas na literatura para solucionar diversos problemas de roteamento de veículos mono-objetivo [Subramanian et al., 2008; Penna et al., 2013].

- **Shift(1,0):** um consumidor é transferido de uma rota para outra rota. Na figura 5.5b o consumidor 5 é removido de sua rota e inserido em outra rota.
- **Shift(2,0):** dois consumidores adjacentes são transferidos de uma rota para outra rota, continuando adjacentes, mas não necessariamente na mesma ordem. Na figura 5.5c os consumidores 5 e 6 são removidos e inseridos em outra rota.
- **Swap(1,1):** permutação entre dois consumidores de duas rotas distintas. Na figura 5.5d os consumidores 3 e 5 são permutados.
- **Swap(2,1):** permutação de dois consumidores adjacentes de uma rota com um consumidor de outra rota. Os dois consumidores adjacentes removidos devem permanecer adjacentes quando inseridos na outra rota, porém não necessariamente na mesma ordem. Na figura 5.5e o consumidor 3 é trocado com os consumidores 4 e 5.

- **Swap(2,2):** permutação de dois consumidores adjacentes de uma rota com outros dois consumidores, também adjacentes, de outra rota. Assim como na Swap(2,1), os consumidores adjacentes devem permanecer adjacentes mas não necessariamente na mesma ordem. Na figura 5.5f os consumidores 4 e 5 são trocados com os consumidores 2 e 3.

- **Crossover:** uma aresta (i, j) de uma rota r_1 e outra aresta (i', j') de uma rota r_2 são removidas. Em seguida as arestas (i, j') e (i', j) são inseridas. Na figura 5.5g as arestas $(3, 0)$ e $(5, 6)$ são removidas e as arestas $(3, 6)$ e $(5, 0)$ são inseridas.

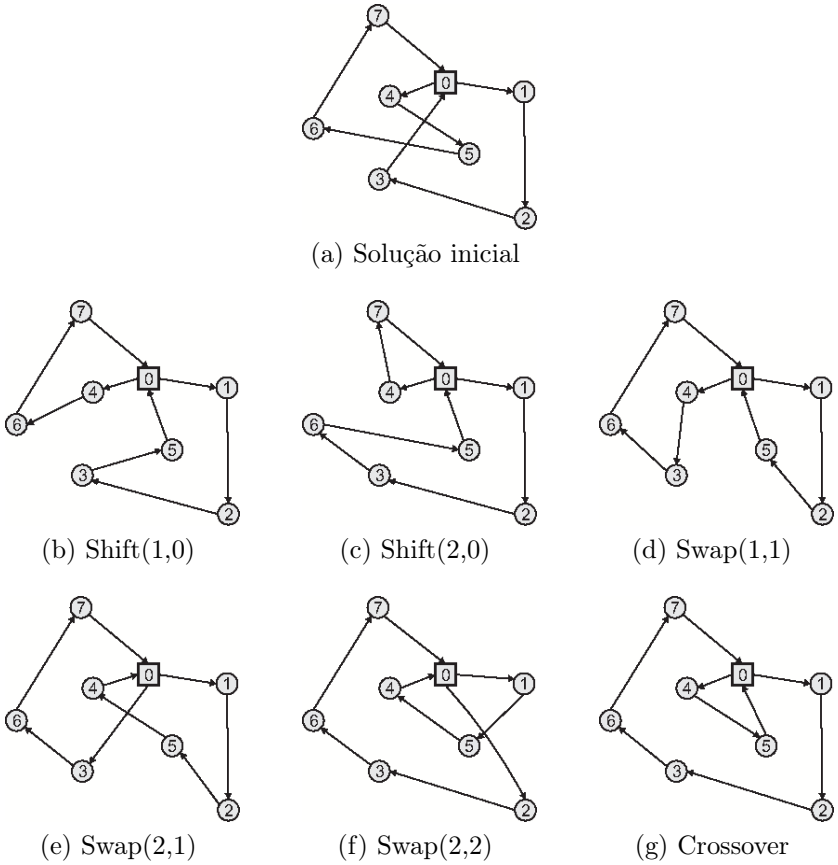
- **TryEmptyRoute:** remove os consumidores de uma determinada rota e tenta reinseri-los nas demais rotas pertencentes a solução a fim de reduzir o número de veículos utilizados para atender as demandas dos consumidores.

Devido a natureza multiobjetivo do problema de roteamento de veículos com coleta opcional, faz-se necessário a utilização de outras funções de vizinhança que também atuem no outro objetivo do problema: minimização das demandas de coletas não atendidas. Neste trabalho foram utilizadas duas funções de vizinhança inter-rotas (Shift(1,0) e Swap(1,1)) para atuar tanto na redução do custo de transporte quanto na minimização das demandas de coletas não atendidas. Para diferenciá-las das demais estruturas de vizinhança inter-rotas, as funções propostas serão intituladas estruturas de vizinhança inter-rotas multiobjetivo, apenas para distinguir que estas atuam nos dois objetivos do problema.

- **MOShift(1,0):** um consumidor é transferido de uma rota para outra rota. A transferência é feita considerando o estado de coleta atual do consumidor e, em seguida, a função de vizinhança é aplicada novamente alterando o estado de coleta do consumidor envolvido no movimento. Logo, a função retorna duas novas soluções, uma para cada movimento realizado, se nenhuma das soluções dominar a outra. Caso contrário, apenas a solução não-dominada é retornada.

- **MOSwap(1,1):** permutação entre dois consumidores de duas rotas distintas. A permutação é feita considerando os estados atuais das demandas de coleta dos consumidores e alternando estes estados. Logo, serão efetuados quatro movimentos, gerando quatro novas soluções: uma considerando os estados atuais das demandas de coleta de ambos consumidores, a segunda alterando o estado da demanda de coleta de um dos consumidores, a terceira alterando o estado do outro consumidor e, por fim, alterando os estados das demandas de coletas dos dois consumidores envolvidos no movimento. As quatro soluções são analisadas e somente as não-dominadas serão retornadas.

Para evitar que movimentos inter-rota sejam executados desnecessariamente, uma estrutura de dados auxiliar, proposta por Penna et al. [2013], foi utilizada indicando se uma



(a) Solução inicial

(b) Shift(1,0) (c) Shift(2,0) (d) Swap(1,1)

(e) Swap(2,1) (f) Swap(2,2) (g) Crossover

Figura 5.5: Movimentos das estruturas de vizinhança inter-rotas

determinada função de vizinhança poderá ser aplicada a uma rota específica de maneira a gerar soluções de melhor qualidade. Se durante a busca local, em uma iteração passada, uma estrutura de vizinhança i foi aplicada à rota j e não houve nenhuma melhoria da solução, então este procedimento fica desabilitado para a rota j até que esta rota sofra alguma modificação. A estrutura de dados, denominada estado de vizinhança, consiste de uma matriz cujas linhas referem-se às estruturas de vizinhança inter-rota e as colunas relacionadas as rotas contidas na solução. Assim, a posição (i, j) da matriz armazena um valor verdadeiro se a busca local i puder ser aplicada a rota j e armazena um valor falso, caso contrário.

5.2.2 Estruturas de Vizinhança Intra-Rota

Neste trabalho, o objetivo principal das estruturas de vizinhanças intra-rota é reduzir os custos de transporte das rotas, removendo, principalmente, os cruzamentos entre as arestas da solução. Isto se deve ao fato que a fase de busca local apresenta dificuldade em explorar o espaço do objetivos em direção ao objetivo mais difícil de ser otimizado. Assim, quando as estruturas de vizinhança inter-rotas não puderem gerar soluções na fronteira Pareto aproximada, aplica-se, então, as estruturas de vizinhança intra-rota a fim de reduzir ainda mais os custos de transporte e aproximá-la ainda mais da fronteira Pareto ótima. O objetivo referente

ao atendimento das demandas de coletas do problema de roteamento de veículos multiobjetivo com coleta opcional foi considerado em apenas uma estrutura de vizinhança.

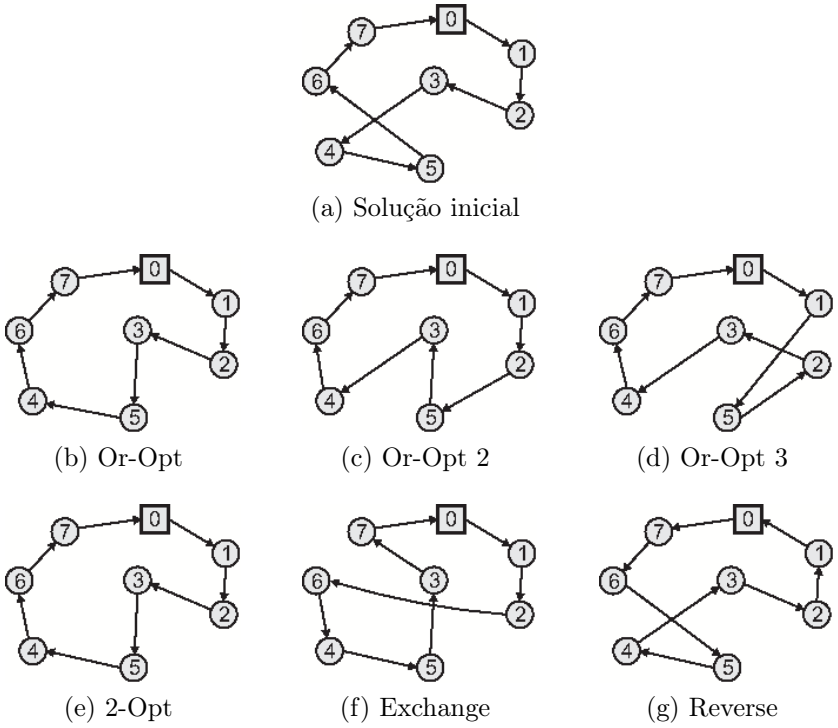
As funções de vizinhança intra-rota implementadas são descritas a seguir.

- **Or-Opt:** um consumidor é inserido em outra posição da mesma rota. Na figura 5.6b o consumidor 5 é alterado de posição.
- **Or-Opt 2:** dois consumidores adjacentes são removidos e inseridos em outra posição da mesma rota. Na figura 5.6c os consumidores 4 e 5 são removidos e inseridos em outra posição.
- **Or-Opt 3:** três consumidores adjacentes são removidos e inseridos em outra posição da mesma rota. Na figura 5.6d os consumidores adjacentes 2, 3 e 4 são removidos e inseridos em outra posição.
- **2-Opt:** duas arestas (i, j) e (i', j') não consecutivas da mesma rota são removidas. Em seguida as arestas (i, i') e (j, j') são inseridas. Na figura 5.6e as arestas $(3, 4)$ e $(5, 6)$ são removidas e as arestas $(3, 5)$ e $(4, 6)$ são inseridas.
- **Exchange:** realiza a permutação entre dois consumidores de uma mesma rota. Na figura 5.6f os consumidores 3 e 6 são permutados.
- **Reverse:** inverte a direção da rota. Neste movimento, pretende-se reduzir a carga máxima no veículo e não o custo da rota. Na figura 5.6g todas as arestas tem a direção alterada, invertendo o sentido da rota.

Assim como nos movimentos inter-rota, foi criada uma nova função de vizinhança intra-rota a fim de reduzir as demandas de coletas não atendidas da solução de um problema de roteamento de veículos multiobjetivo com coleta opcional. A função de vizinhança denominada *AddPickupDemand*, verifica se a demanda de coleta de um consumidor pode ser satisfeita uma vez que o atendimento desta demanda não extrapola a capacidade do veículo e, assim, o consumidor não precisa ser realocado em outra posição ou rota.

5.3 Mecanismos de Perturbação

Na etapa de perturbação dos algoritmos são utilizados mecanismos que realizam modificações em uma solução, escapando de ótimos locais, permitindo a exploração de outras áreas do espaço de busca. As modificações realizadas não podem ser muito pequenas, pois não permitiriam a exploração de novas soluções e o algoritmo retornaria para o mesmo mínimo local. assim como as modificações não poderiam ser muito grandes, pois assim, as buscas locais seriam realizadas a partir de pontos completamente aleatórios.



(a) Solução inicial

(b) Or-Opt (c) Or-Opt 2 (d) Or-Opt 3

(e) 2-Opt (f) Exchange (g) Reverse

Figura 5.6: Movimentos das estruturas de vizinhança intra-rotas

Neste trabalho foram criadas perturbações que modificam as soluções em relação a ambos objetivos do problema de roteamento multiobjetivo com coleta opcional. Quatro mecanismos de perturbação alteram simplesmente a posição do consumidor nas rotas e um mecanismo altera o estado de coleta dos consumidores.

Os quatro mecanismos de perturbação que modificam a posição dos consumidores na rota são:

- **Multiple Swap:** realiza n trocas entre consumidores de rotas distintas escolhidos aleatoriamente, sendo n um número aleatório;
- **Multiple Shift:** realoca n consumidores escolhidos aleatoriamente de uma rota para outra rota, sendo n um número aleatório;
- **Ejection Chain:** realoca um consumidor de uma rota R_1 em uma rota R_2, outro da rota R_2 em uma rota R_3, e assim sucessivamente até que um consumidor da última rota seja inserido na primeira rota. Os consumidores utilizados no movimento são escolhidos aleatoriamente.
- **Or-Opt:** mover 1 a 3 consumidores consecutivos de uma rota para outra [Or, 1976].

Em todas as quatro perturbações citadas anteriormente, a inserção dos consumidores é feita na primeira posição viável encontrada.

O outro mecanismo de perturbação, denominado "Alterar Coleta", visa a modificação da solução em relação ao atendimento da demanda de coleta. Alguns consumidores são selecionados para alteração do estado da demanda de coleta. A quantidade e quais consumidores serão selecionados são definidos aleatoriamente. Se o consumidor tiver a sua coleta efetuada, então ela não será mais realizada, e vice-versa. Quando a demanda de coleta de um consumidor deixar de ser atendida durante a perturbação, o consumidor permanece na mesma posição na rota. Caso contrário, ao passar a atender uma demanda coleta que não era atendida, a capacidade do veículo pode ser extrapolada, inviabilizando a solução. Nestes casos, procura-se uma posição viável que acarrete no menor aumento no custo da rota para inserir o consumidor que extrapolou a capacidade do veículo. Caso ainda não seja encontrada nenhuma posição viável, uma nova rota é criada para que este consumidor possa ser inserido.

5.4 Busca Geral em Vizinhança Variável (GVNS)

Os algoritmos propostos neste trabalho para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional (MOVRPOP) requerem a resolução de problemas mono-objetivo, tais como o problema de roteamento de veículo capacitado (CVRP) e problema de roteamento de veículos com coleta e entrega simultâneas (VRPSPD). Ao fixar que todas as demandas de coleta devem ser atendidas, o problema de roteamento de veículos multiobjetivo, proposto neste trabalho, é reduzido ao VRPSPD. Ao contrário, quando nenhuma demanda de coleta é satisfeita, o MOVRPOP é reduzido ao CVRP. Neste trabalho, as soluções para o VRPSPD e CVRP são obtidas utilizando o algoritmo proposto por Penna et al. [2013], baseado no GVNS.

A metaheurística GVNS, apresentada no algoritmo 5.3, pode ser dividida em duas fases: perturbação e busca local. Na fase de perturbação, o algoritmo gera uma solução aleatória em uma dada vizinhança (exploração estocástica das estruturas de vizinhança). A fase de busca local, consiste na aplicação do método VND (exploração determinística das estruturas de vizinhança).

Neste trabalho, na fase de busca local, foram utilizadas as estruturas de vizinhanças inter-rotas *Shift*(1,0), *Shift*(2,0), *Swap*(1,1), *Swap*(2,1), *Swap*(2,2), *Crossover* e *TryEmptyRoute* e as estruturas de vizinhança intra-rota *Or-Opt*, *Or-Opt2*, *Or-Opt3*, *2-Opt*, *Exchange* e *Reverse*, apresentadas na seção 5.2. Na fase de perturbação, os movimentos *Multiple Swap*, *Multiple Shift*, *Ejection Chain* e *Or-Opt*, apresentadas na seção5.3, foram aplicadas de forma aleatorizada.

Em alguns problemas de roteamento de veículos, Penna et al. [2013] constataram que a execução das funções de vizinhança de forma estocástica pode acarretar em soluções de melhor qualidade. Assim sendo, a metaheurística VND foi modificada de forma que a seleção da estrutura de vizinhança que será aplicada à solução corrente é definida aleatoriamente,

Algoritmo 5.3 Busca Geral em Vizinhança Variável (GVNS)

Entrada: Um conjunto de estrutura de vizinhanças $\mathcal{N}_k$, para $k = 1, ..., k_{max}$ a serem utilizadas na busca local; um conjunto de estrutura de vizinhanças $\mathcal{N}_l$, para $l = 1, ..., l_{max}$ a serem utilizadas na fase de perturbação; solução inicial x; escolher critério de parada.

1: **repita**
2: $l \leftarrow 1$;
3: **repita**
4: *Perturbação:* Selecionar x' aleatoriamente, tal que $x' \in \mathcal{N}_l(x)$; {Busca Local utilizando VND}
5: $k \leftarrow 1$;
6: **repita**
7: $x'' \leftarrow \mathcal{N}_k(x')$;
8: **se** $f(x'') < f(x')$ **então**
9: $x' \leftarrow x''$;
10: $k \leftarrow 1$;
11: **senão**
12: $k \leftarrow k + 1$;
13: **fim se**
14: **até** $k = k_max$
15: **se** $f(x') < f(x)$ **então**
16: $x \leftarrow x'$;
17: $l \leftarrow 1$;
18: **senão**
19: $l \leftarrow l + 1$;
20: **fim se**
21: **até** $l = l_{max}$
22: **até** Critério de Parada
23: **retornar** x

dadas as estruturas disponíveis. Os autores intitularam o novo método Random VND.

No algoritmo 5.4 é apresentado o pseudocódigo do Random VND. Dado um conjunto de funções de vizinhança ($\mathcal{N}$) e uma solução inicial (x), sendo $\mathcal{N}_k$ a função de vizinhança a ser aplicada, o algoritmo seleciona aleatoriamente uma função a ser utilizada (linha 2). Então, repetidamente, o algoritmo busca o melhor vizinho de x que esteja na vizinhança $\mathcal{N}_k$. Se esta solução x' é melhor que x, então x é atualizado com x', a busca é mantida na vizinhança $\mathcal{N}_k$ e todas as funções de vizinhanças passam a ser passíveis de serem selecionadas novamente. Caso contrário, outra função de vizinhança ainda não explorada é selecionada aleatoriamente. Estes passos são executados até que todas vizinhanças (k_{max}) tenham sido exploradas sem êxito, ou seja, sem que uma solução melhor que x tenha sido encontrada. Quando esta estrutura de repetição é encerrada, todas as estruturas de vizinhanças podem ser escolhidas novamente e o algoritmo seleciona aleatoriamente uma função de vizinhança. Este procedimento é repetido até que um ótimo local para todas as estruturas de vizinhanças seja encontrado. Ao término

do procedimento, retorna-se a melhor solução encontrada.

Algoritmo 5.4 Busca de Decida em Vizinhança Variável Aleatória (Random VND)

Entrada: Selecionar um conjunto de estrutura de vizinhanças $\mathcal{N}_k$, para k = 1,..., k_{max}, a serem utilizadas na busca; encontrar uma solução inicial x.

1: **repita**
2: $k \leftarrow$ Selecionar uma estrutura de vizinhança aleatoriamente;
3: **repita**
4: Encontrar x', tal que x' seja o melhor vizinho de x na vizinhança $\mathcal{N}_k(x)$;
5: *Critério de Aceitação:*
6: **se** $f(x') < f(x)$ **então**
7: $x \leftarrow x'$;
8: Todas as funções de vizinhanças podem ser escolhidas novamente;
9: **senão**
10: $k \leftarrow$ Selecionar uma estrutura de vizinhança ainda não explorada aleatoriamente;
11: **fim se**
12: **até** Todas as k_{max} vizinhanças tenham sido exploradas
13: Todas as funções de vizinhanças podem ser escolhidas novamente;
14: **até** todas as estruturas de vizinhança tenham sido exploradas
15: **retornar** x

5.4.1 Solução Inicial CVRP e VRPSPD

O método de construção de uma solução para o problema de roteamento de veículo com serviços de coleta e entrega é baseado no algoritmo proposto por Dethloff [2001]. A heurística baseada em inserção (IBH, do inglês *Insertion-Based Heuristic*), utiliza o conceito "Inserção Mais Barata". Inicialmente, o algoritmo deve escolher um consumidor (semente) e define uma nova rota: depósito $\longrightarrow$ semente $\longrightarrow$ depósito. Posteriormente, é analisada a inserção de cada consumidor restante em cada posição viável da rota atual. Cada uma dessas alternativas é avaliada de acordo com alguns critérios de inserção.

Neste trabalho foram utilizados dois critérios de inserção: critério de inserção viável mais barato (MB) e critério de inserção viável mais próximo (MP). Para cada critério de inserção, foram utilizadas duas estratégias de inserção: sequencial (S) e paralela (P).

O algoritmo 5.5 apresenta o pseudocódigo do algoritmo proposto por Penna et al. [2013]. O algoritmo primeiramente estima o número de veículos a ser utilizado para satisfazer as demandas dos consumidores (linha 1). Posteriormente, dada esta estimativa, são criadas rotas vazias (linhas 6–9). Cada rota vazia é inicializada com um consumidor selecionado aleatoriamente de uma lista de candidatos (linhas 10–14). Em seguida, seleciona-se aleatoriamente um critério de inserção e uma estratégia de inserção para geração de uma solução (linhas 15–21). Se a solução retornada é inviável o procedimento é repetido até que um número máximo de repetições ($MaxNumTentativas$) pré-estabelecido seja alcançado. Quando isso ocorre,

o procedimento é repetido considerando um veículo a mais para geração da solução (linhas 22–28). Quando uma solução viável é encontrada, esta solução é retornada.

Nas subseções 5.4.1.1 5.4.1.2, 5.4.1.3 são apresentados os detalhes dos procedimentos de estimativa do número de veículos, estratégia e critério de inserção, respectivamente, aplicados ao algoritmo de geração da solução inicial.

Algoritmo 5.5 Algoritmo para gerar uma solução inicial [Penna et al., 2013]

```
1:  v ← Estimar Número Veículos();
2:  repita
3:     NumTentativas ← 0;
4:     ConsumidoresCandidatos ← Criar Lista Consumidores Candidatos();
5:     Rotas ← Criar Conjunto de Rotas();
       {Criar v rotas vazias};
6:     para i = 1 até v faça
7:        Rota_i ← Criar Rota Vazia();
8:        Rotas ← Rota_i;
9:     fim para
       {Selecionar aleatoriamente v consumidores da lista de candidatos};
10:    para i = 1 até v faça
11:       c_i ← Selecionar Aleatoriamente(ConsumidoresCandidatos);
12:       Rota_i ← c_i;
13:       ConsumidoresCandidatos ← ConsumidoresCandidatos − c_i;
14:    fim para
       {Selecionar aleatoriamente um critério de inserção e uma estratégia de inserção}
15:    CriterioInsercao ← Selecionar aleatoriamente({MB, MP});
16:    EstrategiaInsercao ← Selecionar aleatoriamente({S, P});
17:    se EstrategiaInsercao = S então
18:       solucao ← Sequencial(v, ConsumidoresCandidatos, CriterioInsercao);
19:    senão
20:       solucao ← Paralela(v, ConsumidoresCandidatos, CriterioInsercao);
21:    fim se
22:    se solucao é inviável então
23:       NumTentativas ← NumTentativas + 1;
24:       se NumTentativas = MaxNumTentativas então
25:          v ← v + 1;
26:       fim se
27:    fim se
28: até solucao viável
29: retornar solucao;
```

5.4.1.1 Estimativa do Número de Veículos

A estimativa do número de veículos permite que o algoritmo de geração de solução inicial alcance uma solução de melhor qualidade, onde poucos veículos são utilizados. Pode-se verificar nos trabalhos da literatura que, ao reduzir o número de veículos, o custo de transporte

também é reduzido para problemas de roteamento de veículos capacitado, bem como para os problemas de roteamento de veículos com serviços de coleta e entrega. A proposta dessa estimativa é que um menor número de veículos seja utilizado de forma que eles possam atender as demandas dos consumidores.

O algoritmo 5.6 apresenta o pseudocódigo do procedimento de estimativa do número de veículos. Primeiramente considera-se a utilização de um único veículo e, então, um consumidor é selecionado aleatoriamente, dado uma lista de consumidores candidatos que ainda não foram adicionados a solução parcial, para ser incluído na rota deste veículo.

A inclusão do consumidor na rota é dada pela inserção mais barata, onde todas as posições da rota são avaliadas e a inserção na posição que retornar menor custo é executada. Enquanto houver consumidores na lista de consumidores candidatos este procedimento é repetido. Quando nenhum consumidor da lista puder ser inserido na rota atual, uma nova rota é gerada e o procedimento continua.

Algoritmo 5.6 Algoritmo para estimar número de veículos

1: $ConsumidoresCandidatos \leftarrow$ Criar Lista Consumidores Candidatos();
2: $Rota \leftarrow CriarRotaVazia()$;
3: $c \leftarrow$ Selecionar Aleatoriamente($ConsumidoresCandidatos$);
4: Remover c da lista $ConsumidoresCandidatos$;
5: **enquanto** $|ConsumidoresCandidatos| > 0$ **faça**
6: **para** cada consumidor $c \in ConsumidoresCandidatos$ **faça**
7: **para** cada posição $p \in Rota$ **faça**
8: $custo_{c,p} \leftarrow$ Calcula custo da rota caso o consumidor c seja inserido na posição p da $Rota$;
9: **fim para**
10: **fim para**
11: **se** não nenhum consumidor $c \in ConsumidoresCandidatos$ puder ser inserido a $Rota$ sem que a capacidade do veículo seja extrapolada **então**
12: $Rotas \leftarrow Rota$;
13: $Rota \leftarrow CriarRotaVazia()$;
14: **senão**
15: $menorCusto_{c_{min},p_{min}} \leftarrow MIN\{custo_{c,p}\}$;
16: Inserir consumidor c_{min} na posição p_{min} da $Rota$;
17: Remover c da lista $ConsumidoresCandidatos$;
18: **fim se**
19: **fim enquanto**
20: **retornar** $Rotas$;

5.4.1.2 Critério de Inserção

Os critérios de inserção utilizados neste trabalho baseiam-se na heurística de inserção, a qual foi proposta por Dethloff [2001] para solucionar o problema de roteamento de veículos com coleta e entrega simultâneas.

Segundo Dethloff [2001], dada uma lista de consumidores candidatos a serem inseridos na solução, uma simples abordagem gulosa (g_c) irá analisar o custo da inserção de um consumidor c, pertencente a esta lista, considerando apenas o aumento do custo de transporte da rota R após a inserção do consumidor c entre os consumidores i e j.

$$g_c = C_{ic} + C_{cj} - C_{ij}, \forall c \in \text{Lista de Consumidores Candidatos} \wedge \forall (i,j) \in R \tag{5.1}$$

Quando o procedimento de inserção é baseado apenas no custo de transporte (g_c), o critério de inserção é denominado: critério de inserção viável mais barato (MB).

Assim, calculado o custo de transporte, aquela inserção que retorne o menor custo é aplicada. Este procedimento é feito para todos os consumidores da lista de consumidores candidatos e em todas as posições viáveis da rota.

Esse critério de inserção é extremamente simples e não verifica alguns aspectos, como:

1. O efeito de uma inserção viável na carga de um veículo, ou seja, o grau de liberdade para futuras inserções.
2. Consumidores localizados remotamente podem ser deixados para serem inseridos em estágios de inserção tardios, resultando em um distância trafegada desfavorável.

Dessa forma, um outro critério de inserção é utilizado neste trabalho: critério de inserção viável mais próximo (MP). Este critério calcula a distância do consumidor para o depósito, evitando que consumidores localizados remotamente sejam inseridos de forma tardia. Assim sendo, o custo final de uma inserção de um consumidor na rota é dado pela soma ponderada do critério guloso e do critério de inserção viável mais próximo.

$$g_c = (C_{ic} + C_{cj} - C_{ij}) - \gamma(C_{0c} + C_{c0}), \forall c \in \text{Lista de Consumidores Candidatos} \wedge \forall (i,j) \in R \tag{5.2}$$

O critério referente ao efeito de inserção de um consumidor na carga do veículo não foi implementado neste trabalho devido ao seu custo computacional.

5.4.1.3 Estratégia de Inserção

Duas estratégias de inserção foram implementadas neste trabalho, sendo que o foco de uma delas é na rota e o foco da outra é nos consumidores candidatos. Dada uma rota, a primeira estratégia, denominada estratégia sequencial, avalia o custo de inserir cada consumidor da lista

de consumidores candidatos na rota dado o critério de inserção selecionado (seção 5.4.1.2). Em seguida, ele seleciona outra rota e repete o procedimento.

A outra estratégia, denominada estratégia paralela, avalia o custo de inserção de cada consumidor da lista de consumidores candidatos em cada posição viável de cada rota. A inserção que retornar menor custo é, então, aplicada. O custo é obtido de acordo com os critérios de inserção mencionados anteriormente. Este procedimento é repetido até que todos os consumidores do problema estejam inseridos na solução.

A primeira estratégia considera uma única rota a cada inserção de consumidor da lista de candidatos, enquanto a segunda considera todas as rotas para inserir os consumidores. Assim sendo, as soluções retornadas pela estratégia sequencial são mais balanceadas.

Os algoritmos 5.7 e 5.8 apresentam o pseudocódigo das duas estratégias de seleção.

Algoritmo 5.7 Algoritmo de Inserção Sequencial

Entrada: $CriterioInsercao$, conjunto $Rotas$

$\gamma \leftarrow 0$;
se $CriterioInsercao = MP$ **então**
  $\gamma \leftarrow$ Definir valor da ponderação;
fim se
enquanto $|ConsumidoresCandidatos| > 0 \wedge \exists c \in ConsumidoresCandidatos$ que pode ser adicionado ao conjunto $Rotas$ **faça**
  para cada rota $i \in Rotas$ **faça**
    para cada consumidor $c \in ConsumidoresCandidatos$ **faça**
      para cada posição viável $p \in Rota_i$ **faça**
        $custo_{c,p} \leftarrow$ Calcular o custo de inserção do consumidor c na posição p da rota i dado o critério de inserção selecionado;
      fim para
    fim para
    $menorCusto_{c_{min},p_{min}} \leftarrow MIN\{custo_{c,p}\}$;
    Inserir consumidor c_{min} na posição p_{min} da rota i;
    Remover consumidor c_{min} da lista $ConsumidoresCandidatos$;
  fim para
fim enquanto

5.5 Heuristica P_ϵ

Diversas técnicas heurísticas são encontradas na literatura para solucionar problemas de otimização multiobjetivo, dentre elas, os algoritmos evolucionários vem sendo bastante utilizados [Deb et al., 2002; Corberán et al., 2002; Dias e Vasconcelos, 2002; Jozefowiez et al., 2009]. Entretanto, estes métodos envolvem uma grande quantidade de parâmetros que precisam ser ajustados, sendo que o seu desempenho depende destes ajustes. Isso pode tornar a sua aplicação muito complexa. Outros métodos, adaptados de metaheurísticas bastante conhecidas,

Algoritmo 5.8 Algoritmo de Inserção Paralela

Entrada: $CriterioInsercao$, conjunto $Rotas$

1: $\gamma \leftarrow 0$;
2: **se** $CriterioInsercao = MP$ **então**
3: $\quad\gamma \leftarrow$ Definir valor da ponderação;
4: **fim se**
5: **enquanto** $|ConsumidoresCandidatos| > 0 \wedge \exists c \in ConsumidoresCandidatos$ que pode ser adicionado ao conjunto $Rotas$ **faça**
6: $\quad$**para** cada consumidor $c \in ConsumidoresCandidatos$ **faça**
7: $\quad\quad$**para** cada rota $i \in Rotas$ **faça**
8: $\quad\quad\quad$**para** cada posição viável $p \in Rota_i$ **faça**
9: $\quad\quad\quad\quad$$custo_{c,p,i} \leftarrow$ Calcular o custo de inserção do consumidor c na posição p da rota i dado o critério de inserção selecionado;
10: $\quad\quad\quad$**fim para**
11: $\quad\quad$**fim para**
12: $\quad$**fim para**
13: $\quad$$menorCusto_{c_{min},p_{min},i_{min}} \leftarrow MIN\{custo_{c,p,i}\}$;
14: $\quad$Inserir consumidor c_{min} na posição p_{min} da rota i_{min};
15: $\quad$Remover consumidor c_{min} da lista $ConsumidoresCandidatos$;
16: **fim enquanto**

como busca tabu [Pacheco e Marti, 2006; Lust, 2010] e busca local iterativa (ILS) [Geiger, 2006], também são encontrados na literatura para solução de diversos problemas de otimização combinatória multiobjetivo.

Este trabalho apresenta uma nova heurística para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, baseada no método ϵ-Restrito (seção 2.2) para o caso de dois objetivos. A heurística intitulada P_ϵ, busca minimizar o custo de transporte e transforma o outro objetivo em uma restrição do problema, sendo que, no algoritmo, isso significa que a cada iteração um número de consumidores terá obrigatoriamente sua demanda de coleta atendida [Assis et al., 2013b].

O cenário gerado a cada iteração da heurística proposta torna-se um problema mono-objetivo a ser solucionado. Assim, o presente trabalho aplica o algoritmo baseado no GVNS, apresentado na seção 5.4, para solucionar cada um dos cenários gerados pela heurística P_ϵ. Os passos básicos da heurística proposta são mostrados no algoritmo 5.9.

O passo inicial define os pontos de partida do algoritmo (linhas 1 e 2). Estes pontos consistem nas soluções dos problemas de roteamento de veículo capacitado (CVRP) e roteamento de veículos com coleta e entrega simultâneas (VRPSPD), nos quais nenhuma demanda de coleta é atendida e todas as demandas de coletas são atendidas, respectivamente. O algoritmo baseado no GVNS, apresentado na seção 5.4, foi utilizado para resolver ambos problemas. As duas soluções são, então, incluídas na fronteira Pareto aproximada.

Algoritmo 5.9 Heurística P_ϵ

1: $s_{CVRP} \leftarrow$ gerar solução para o CVRP utilizando GVNS( );
2: $s_{VRPSPD} \leftarrow$ gerar solução para o VRPSPD utilizando GVNS( );
3: atualizar($Front$, s_{CVRP});
4: atualizar($Front$, s_{VRPSPD});
5: $cont \leftarrow 0$;
6: **enquanto** $cont < maxCont$ **faça**
7: $s'_{CVRP} \leftarrow$ AddPickupDemand(s_{CVRP});
8: Atualizar($Front$, s'_{CVRP});
9: **enquanto** $\exists\ consumidor_j \in s'_{CVRP} | \ell_j = 0$ **faça**
10: Definir j aleatoriamente;
11: $\ell_j \leftarrow 1$
12: $s'_{CVRP} \leftarrow$ Insere uma coleta($consumidor_j$,s'_{CVRP});
13: $s'_{CVRP} \leftarrow$ GVNS(s'_{CVRP});
14: atualizar($Front$, s'_{CVRP});
15: **fim enquanto**
16: **enquanto** $\exists\ consumidor_j \in s'_{VRPSPD} | \ell_j = 1$ **faça**
17: Definir j aleatoriamente;
18: $\ell_j \leftarrow 0$
19: $s'_{VRPSPD} \leftarrow$ GVNS(s'_{VRPSPD});
20: atualizar($Front$, s'_{VRPSPD});
21: **fim enquanto**
22: $cont \leftarrow cont + 1$;
23: **fim enquanto**
24: **retornar** $Front$;

As soluções do CVRP e VRPSPD, encontrados nesta fase inicial, representam as extremidades da fronteira Pareto aproximada, como mostrado na figura 5.7. Uma vez encontrados e sendo estes de boa qualidade, os passos restantes do algoritmo consistem em encontrar as soluções entre os pontos extremos já definidos.

Figura 5.7: Solução do CVRP e VRPSPD, pontos extremos da fronteira Pareto aproximada

Dados os pontos extremos da fronteira, o algoritmo, então, irá executar $maxCont$ iterações

(linhas 6-23), nos quais a solução do CVRP será modificada gradativamente até que todas as demandas de coleta sejam satisfeitas. Por outro lado, a solução do VRPSPD será também modificada gradativamente até que nenhuma demanda de coleta seja atendida.

Iniciando pela solução s'_{CVRP}, na linha 7, para cada rota aplica-se a estrutura de vizinhança intra-rota *AddPickupDemand*, definida na subseção 5.2.2. Dessa forma, sem alterar a posição dos consumidores na rota, busca-se atender o máximo de demandas de coleta possível. A solução encontrada é inserida na fronteira utilizando a função $atualizar()$, apresentada na subseção 5.5.1.

Em seguida, enquanto houver demandas de coleta não atendidas na solução s'_{CVRP}, as demandas de coleta não atendidas serão selecionadas aleatoriamente, uma a uma, para serem satisfeitas. Este procedimento resulta na realocação do consumidor, uma vez que o atendimento de uma demanda de coleta pode acarretar na extrapolação da capacidade do veículo. Assim, este consumidor é realocado na melhor posição encontrada dentre todas as rotas, considerando o custo de transporte. Quando nenhuma posição viável é encontrada, uma nova rota é criada para a inclusão deste consumidor. Para refinar a solução obtida, aplica-se, então, o método GVNS (seção 5.4) à esta solução. Finalmente, atualiza-se o conjunto de soluções não-dominadas $Front$.

O procedimento descrito para a solução s'_{CVRP} é repetido para a solução s'_{VRPSPD} com algumas modificações. Neste caso, na solução de partida s'_{VRPSPD}, todas as demandas de coleta estão sendo atendidas. Assim, a cada passo um consumidor é selecionado aleatoriamente e o estado da demanda de coleta é alterado para "*não atendido*". Este procedimento não requer a realocação do consumidor, uma vez que o mesmo não acarreta em nenhuma inviabilidade da solução. Da mesma forma o método GVNS é utilizado para refinar a solução e a fronteira é atualizada com a solução retornada. Este procedimento é executato até que o estado de coleta de todos consumidores seja alterado.

Como a seleção das coletas a serem atendidas ou não atendidas é feita de forma aleatória, o método é executado várias vezes ($maxCont$) a fim de encontrar soluções mais diversificadas e aproximadas da fronteira Pareto ótima.

5.5.1 Atualização

A etapa de atualização utilizada neste trabalho garante que apenas soluções não dominadas permaneçam na fronteira Pareto aproximada $Front$. Se s não é dominada por nenhuma solução em $Front$, então ela é inserida no conjunto e as soluções por ela dominadas são removidas. Neste procedimento não se considera questões de diversidade e, também, não se controla o tamanho da fronteira, sendo que, enquanto houver soluções não dominadas sendo geradas nos procedimentos anteriores, elas serão incluídas na fronteira. O único objetivo da etapa de atualização é manter um conjunto de soluções não dominadas. Os outros aspectos são

responsabilidade dos demais procedimentos do algoritmo. A fase de atualização é apresentada no algoritmo 5.10.

Algoritmo 5.10 Etapa de atualização

Entrada: $Front$; {conjunto que se tentará inserir a solução}
Entrada: s^*; {solução para inserir}
1: $inseriu \leftarrow$ **verdadeiro**;
2: **para todo** $s \in Front$ **faça**
3: **se** $s \prec s^*$ **então**
4: $inseriu \leftarrow$ **falso**;
5: **parar**; {Interrompe o bloco de repetição}
6: **fim se**
7: **se** $s^* \prec s$ **então**
8: $Front \leftarrow Front \backslash \{s\}$
9: **fim se**
10: **fim para**
11: **se** $inseriu =$ **verdadeiro então**
12: $Front \leftarrow Front \cup \{s^*\}$
13: **fim se**
14: **retornar** $inseriu$;

5.6 Busca Geral em Vizinhança Variável Multiobjetivo (MOGVNS)

Ao se tratar de problemas de roteamento de veículos mono-objetivo com serviço de coleta e entrega, heurísticas baseadas na busca local iterativa [Subramanian et al., 2008, 2010] ou na busca geral em vizinhança variável apresentam bons resultados, sem a necessidade de ajuste de muitos parâmetros. Além disso, adaptações desta metaheurística, quando utilizadas para resolução de outros problemas de otimização combinatória multiobjetivo, também apresentam bons resultados [Geiger, 2006]. Assim, o presente trabalho implementa uma heurística baseada na metaheurística GVNS, apresentada na seção 5.4, para resolução de problemas multiobjetivo utilizando o conceito de Dominância Pareto, possibilitando, assim, a resolução de problemas de otimização combinatória multiobjetivo, em especial o problema de roteamento de veículos multiobjetivo com coleta opcional. À adaptação proposta é dado o nome de busca geral em vizinhança variável multiobjetivo (MOGVNS, do inglês *Multiobjective General Variable Neighborhood Search*) [Assis et al., 2013a].

O algoritmo proposto possui 5 etapas: (i) geração de uma solução inicial; (ii) seleção de uma solução da fronteira Pareto aproximada para ser explorada; (iii) busca local, que retorna uma solução melhorada; (iv) perturbação, que modifica uma solução e; (v) atualização, que insere uma ou mais soluções na fronteira Pareto aproximada.

Seu funcionamento é ilustrado no algoritmo 5.11. O MOGVNS inicia seu funcionamento com a geração de soluções iniciais (linha 1). As soluções geradas são inseridas no conjunto $Front$ que mantém as soluções não dominadas. Em seguida, são executados $maxIter$ iterações, onde uma solução em $Front$ é selecionada e o bloco de repetição principal do algoritmo (linhas 6–15) é executado. Neste bloco, as etapas de perturbação (linha 7) e busca local (linha 8), semelhantes ao GVNS básico, são aplicadas à solução escolhida. No algoritmo implementado a fase de busca local retorna um conjunto de soluções não dominadas S, devido às funções de vizinhanças utilizadas. O critério de aceitação é substituído por uma função de atualização, onde tenta-se inserir a nova solução gerada no conjunto $Front$ (linha 9). A variável *cont* representa o número de iterações que uma mesma solução é explorada sem a geração de uma nova solução não dominada. A sua atualização é feita da seguinte forma: se a solução gerada após a perturbação e busca local for inserida em $Front$ (indicada pela variável lógica *inserido*) então *cont* é reiniciado (linha 11), senão *cont* é incrementado de uma unidade (linha 14). O parâmetro $maxCont$ representa o número máximo de vezes que uma solução será explorada sem a geração de uma solução não dominada.

Algoritmo 5.11 *Busca Geral em Vizinhança Variável Multiobjetivo (MOGVNS)*

```
1:  Front ← gerarSoluçõesIniciais();
2:  iter ← 0;
3:  enquanto iter < maxIter faça
4:     s' ← selecionar(Front);
5:     cont ← 0;
6:     enquanto cont < maxCont faça
7:        s'' ←perturbação(s',Front);
8:        S ← buscaLocal (s'');
9:        inserido ← atualizar(Front, S);
10:       se inserido então
11:          cont ← 0;
12:          s' ← s'';
13:       senão
14:          cont ← cont + 1;
15:       fim se
16:    fim enquanto
17:    iter ← iter + 1;
18: fim enquanto
19: retornar Front;
```

Na fase de atualização do MOGVNS aplica-se o mesmo método utilizado pela heurística P_ϵ, apresentado no algoritmo 5.10. As demais fases do MOGVNS: solução inicial, seleção, perturbação e busca local, estão detalhadas a seguir.

Solução Inicial

Foram definidas duas formas para encontrar a solução inicial do algoritmo MOGVNS. A

primeira, apenas os pontos extremos da fronteira são definidos, ou seja, a solução do VRPSPD e CVRP (subseção 5.4.1). As demais soluções são encontradas a partir destas. Outro método utilizado foi a aplicação do algoritmo P_ϵ, apresentado na seção 5.5. Assim, o algoritmo retorna um conjunto de soluções não dominadas e estas soluções serão utilizadas pelo MOGVNS para explorar outras soluções no espaço dos objetivos.

Seleção

O procedimento de seleção utilizado é baseado no método distância de multidão do algoritmo NSGA-II (seção 5.7). Quanto mais distante das outras soluções uma determinada solução estiver, maior a probabilidade desta ser escolhida. Assim, é possível realizar a exploração do espaço de soluções de forma mais inteligente, onde é dado prioridade a regiões pouco exploradas, evitando a concentração da busca em determinados nichos.

Após calculada a distância de multidão, o algoritmo aplica o método da roleta para selecionar a próxima solução a ser explorada, onde a probabilidade de uma solução ser escolhida é proporcional ao valor da sua distância de multidão. Assim sendo, por aplicar o método da roleta, o cálculo da distância de multidão é um pouco diferente do realizado no NSGA-II. No MOGVNS as soluções presentes nos extremos da fronteira têm a distância de multidão igual a duas vezes a distância entre elas e a solução mais próxima, enquanto no NSGA-II, as soluções presentes nas extremidades da fronteira recebem um valor infinito.

Perturbação

Na etapa de perturbação são utilizados mecanismos que realizam modificações em uma solução, escapando de ótimos locais, permitindo a exploração de outras áreas do espaço de busca. Na seção 5.3 as perturbações utilizadas neste trabalho são apresentadas, sendo elas aplicadas em duas etapas. Primeiramente aplica-se a pertubação "Alterar Coleta" que modifica o estado de coleta de alguns consumidores. Em seguida, um dos mecanismos de perturbação (*Multiple Swap*, *Multiple Shift* e *Ejection Chain*) é escolhido aleatoriamente e aplicado à solução.

Busca Local

A etapa de busca local realiza a exploração da vizinhança de uma solução com o objetivo de encontrar outras soluções não dominadas para o problema. Neste trabalho foi utilizada uma adaptação do método da descida em vizinhança variável aleatória (Random VND), descrita na seção 5.4. O algoritmo proposto consiste em aplicar aleatoriamente as estruturas de vizinhança descritas na seção 5.2. Primeiramente, as estruturas de vizinhança inter-rota que atuam nos dois objetivos do problema são utilizadas. Sempre que uma dessas estruturas é aplicada, um conjunto de soluções é retornado. A fronteira Pareto aproximada é então atualizada dado este conjunto de soluções. Sempre que houver atualização da fronteira, aplica-se as demais estruturas de vizinhança inter-rotas que visam a redução do custo de transporte em todas as soluções que foram inseridas na fronteira. Se uma solução é aprimorada, então, aplica-se a ela, as estruturas de vizinhança intra-rota.

O algoritmo 5.12 apresenta o pseudocódigo do algoritmo adaptado do Random VND. O algoritmo recebe uma solução inicial s. Na linha 1, um conjunto S vazio é criado no qual serão armazenadas as soluções não-dominadas encontradas durante o procedimento de busca local. Na linha 2 são definidas as estruturas de vizinhança inter-rota multiobjetivo que serão aplicadas na busca (MOShift e MOSwap). Enquanto houver ainda estrutura de vizinhança para ser explorada, uma estrutura de vizinhança $\mathcal{N}_e$ é selecionada para ser aplicada à solução s (linhas 3–5). Conforme mencionado na subseção 5.2.1, as estruturas que trabalham com os dois objetivos do problema retornam um conjunto de soluções não dominadas, representado no algoritmo por S' (linha 5).

Em seguida, realiza-se a união do conjunto S e S' e as soluções dominadas são removidas, permanecendo apenas as soluções não-dominadas (linha 6). Este procedimento é realizado pela função $atualiza(S, S')$. Em seguida, as soluções pertencentes a S' que permaneceram em S passam pela fase de busca local que visa apenas a redução do custo de transporte (linha 7). Nesta etapa, o estado da demanda de coleta dos consumidores permanece fixo. Essa fase é de extrema importância uma vez que o custo de transporte é o mais difícil de ser minimizado. Assim, um número maior de buscas locais é utilizado.

Para cada solução s' pertencente a S' e inserida em S, um conjunto de estruturas de vizinhança inter-rota ($\mathcal{N}_f$) é definido . Enquanto o mínimo local de todas as estruturas de vizinhança $\mathcal{N}_f$ não for encontrado, seleciona-se uma função de vizinhança $\mathcal{N}_f$ para ser aplicada a solução s' (linhas 8–12). Como estas estruturas de vizinhança só modificam a posição do consumidor, visando a redução do custo de transporte, a função retorna apenas uma solução que será melhor que s' se possuir menor custo de transporte (linha 13). Se uma solução de melhor qualidade for encontrada, então o mesmo procedimento é feito para as estruturas de vizinhança intra-rota definidas por $\mathcal{N}_g$ (linhas 15–25).

5.7 Algoritmo Genético de Ordenação Não-Dominante II (NSGA-II)

O algoritmo genético de ordenação não-dominante II, descrito na seção 2.3, foi adaptado a fim de solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, seguindo seus princípios básicos, como mostra o algoritmo 5.13.

O primeiro passo do algoritmo é a geração da população inicial P. Partindo desta população, a população filha Q é gerada a partir da seleção, cruzamento e mutação de indivíduos da população P. Dois operadores de cruzamento são considerados neste trabalho, tendo eles igual probabilidade de serem selecionados. Cada indivíduo "filho" pode sofrer mutação, dada uma probabilidade de mutação p_{mut}. Os operadores de mutação utilizados neste trabalho são *Or-opt* e *Altera Coleta*, descritos na Subsection 5.3. Estes também têm igual probabilidade

Algoritmo 5.12 Algoritmo baseado no Random VND aplicado na fase de busca local do MOGVNS

Entrada: solução inicial s

1: $S \leftarrow \emptyset$;
{Estruturas de vizinhança inter-rota multiobjetivo (MOShift e MOSwap)}
2: Definir estruturas de vizinhança inter-rota multiobjetivo $\mathcal{N}_e$, $e = \{1, 2, \ldots, e_{max}\}$;
3: $e \leftarrow 1$;
4: **enquanto** $e < e_{max}$ **faça**
5: $S' \leftarrow \mathcal{N}_e(s)$;
6: **se** $atualiza(S, S')$ **então**
7: **para** cada solução $s' \in S'$ incluída em S **faça**
8: $s \leftarrow s'$; {Estruturas de vizinhança inter-rota}
9: Definir estruturas de vizinhança inter-rota $\mathcal{N}_f$, $f = \{1, 2, \ldots, f_{max}\}$;
10: $f \leftarrow 1$;
11: **enquanto** $f < f_{max}$ **faça**
12: $s' \leftarrow \mathcal{N}_f(s)$;
13: **se** $f(s') < f(s)$ **então**
14: $s \leftarrow s'$;
{Estruturas de vizinhança intra-rota}
15: Definir estruturas de vizinhança intra-rota $\mathcal{N}_g$, $g = \{1, 2, \ldots, g_{max}\}$;
16: $g \leftarrow 1$;
17: **enquanto** $g < g_{max}$ **faça**
18: $s' \leftarrow \mathcal{N}_g(s)$;
19: **se** $f(s') < f(s)$ **então**
20: $s \leftarrow s'$;
21: $g \leftarrow 1$;
22: Rearranjar estruturas de vizinhança intra-rota $\mathcal{N}_g$ em ordem aleatória;
23: **senão**
24: $g \leftarrow g + 1$;
25: **fim se**
26: **fim enquanto**
27: $g \leftarrow 1$;
28: Rearranjar estruturas de vizinhança inter-rota $\mathcal{N}_f$ em ordem aleatória;
29: **senão**
30: $f \leftarrow f + 1$;
31: **fim se**
32: **fim enquanto**
33: $e \leftarrow 1$;
34: Rearranjar estruturas de vizinhança inter-rota MO $\mathcal{N}_e$ em ordem aleatória;
35: **fim para**
36: **senão**
37: $e \leftarrow e + 1$;
38: **fim se**
39: **fim enquanto**
40: **retornar** S

Algoritmo 5.13 Pseudocódigo do NSGA-II

```
1: P ← Populacão Inicial();
2: Q ← Seleção, Cruzamento e Mutação (P);
3: repita
4:    F[ ] ← Ordenação por não dominância(P ∪ Q);
5:    para todo s_j ∈ F_1 faça
6:       s_j ← Busca Local(s_j);
7:    fim para
8:    P ← ∅; i ← 1;
9:    enquanto |P + F_i| ≤ N faça
10:      P ← P ∪ F_i; i ← i + 1;
11:   fim enquanto
12:   para todo s_j ∈ F_i faça
13:      d_j ← Distância de Multidão(s_j);
14:   fim para
15:   C ← Ordenação pela Distância de Multidão(d, F_i);
16:   para todo s_j ∈ C to j = N − |P| faça
17:      P ← C_j;
18:   fim para
19:   Q ← Seleção, Cruzamento and Mutação(P_{n+1});
20: até Critério de Parada
21: Front ← Soluções Não Dominadas (P ∪ Q);
22: retornar Local Search(Front);
```

de serem selecionados para serem aplicados. A partir de então, os indivíduos das populações $P \cup Q$ passarão pela método de ordenação por não-dominância, retornando um conjunto de fronteiras não dominadas. Seguindo esta ordenação, as fronteiras são incluídas na população P, preservando as melhores soluções da população anterior (elitismo). Em seguida, as soluções que se encontram na primeira fronteira não dominada são refinadas com a aplicação do algoritmo Random VND adaptado para o problema multiobjetivo (algoritmo 5.12). Cada uma destas soluções recebe um estado (refinada ou não-refinada) para que este procedimento seja aplicado apenas uma vez em cada solução e o algoritmo não se torne inviável em termos de tempo de execução.

Quando uma fronteira não puder ser totalmente incluída em P, considerando um limite máximo de N indivíduos na população, os indivíduos dessa fronteira são selecionados de acordo com o método distância de multidão. Em seguida, a técnica torneio de multidão é aplicada para seleção dos indivíduos a serem recombinados e inseridos na população Q.

A representação dos indivíduos da população, os métodos de geração da população inicial, seleção, cruzamento e mutação são descritos nas subseções a seguir.

5.7.1 Representação dos Indivíduos

Os indivíduos da população representam uma solução completa para o problema de roteamento de veículos multiobjetivo com coleta opcional, descrita na subseção 5.1.2. Cada indivíduo consiste de um conjunto de rotas e um vetor binário que armazena o estado da demanda de coleta. No vetor binário, o valor 1 na posição i indica que o consumidor i tem sua demanda de coleta atendida e o valor 0 indica o contrário.

5.7.2 Geração da População Inicial

Neste trabalho uma das formas utilizadas na geração da população inicial foi dada pela aplicação da heurística P_ϵ apresentada na seção 5.5. Caso o número de indivíduos gerados pela heurística seja menor que o tamanho pré-definido da população, então outros indivíduos são gerados aleatoriamente a fim de completá-la.

Outro procedimento também foi implementado para geração da solução inicial de forma similar a heurística mencionada [Assis et al., 2013b]. Este procedimento gera soluções mais distantes da fronteira Pareto ótima, mas com uma melhor diversidade das soluções. No procedimento de geração da população inicial, o algoritmo primeiramente gera soluções para o problema de roteamento de veículos capacitado (CVRP) e para o problema de roteamento de veículos com coleta e entrega simultâneas (VRPSPD). Ambos problemas estão localizados nas extremidades da fronteira Pareto aproximada, sendo que no primeiro problema não há atendimentos das demandas de coleta e no segundo todas estas demandas são satisfeitas. As soluções são obtidas aplicando o algoritmo baseado na busca geral em vizinhança variável (GVNS) descrito na seção 5.4.

Para uma população com p indivíduos, as demais soluções são obtidas a partir da alteração do estado das demandas de coletas das soluções obtidas para o CVRP e VRPSPD.

Iniciando o procedimento pela solução do problema de roteamento de veículo capacitado, o algoritmo calcula uma taxa na qual as demandas de coleta serão atendidas. Ao término do procedimento o último indivíduo terá todas ou quase todas as demandas de coleta satisfeitas.

O mesmo procedimento é executado para gerar as soluções iniciais partindo do VRPSPD: o algoritmo calcula a taxa no qual as demandas de coleta deixarão de ser atendidas e ao término do procedimento o último indivíduo definido terá poucas ou nenhuma demanda de coleta satisfeita.

O algoritmo 5.14 apresenta o pseudocódigo para geração da população inicial. Nas linhas 1 e 2 as soluções para o CVRP e VRPSPD são geradas utilizando o algoritmo baseado no GVNS (seção 5.4). A variável de controle *numConsColetasAtendidas* armazena, para as soluções do CVRP e VRPSPD, a quantidade de consumidores cujo estado da demanda de coleta é "*atendido*"(linhas 3 e 4). As soluções de ambos problemas são inseridas na população (linha

5). O número de soluções a serem geradas partindo de cada problema é definido na linha 6, e a taxa de incremento ou decremento da satisfação das demandas de coletas é calculada na linha 7.

Dadas as soluções de partida, o número de soluções a serem geradas e a taxa calculada, o algoritmo inicia o bloco de repetição (linhas 9 a 14) aumentando ou reduzindo a satisfação das demandas de coleta gradativamente considerando a taxa calculada na linha 7. O método *insereColeta* é, então, aplicado à solução corrente s_{CVRP} gerando uma solução na qual o número de consumidores que terão suas demandas de coleta atendidas irá variar entre $]numConsColetasAtendidas_{CVRP}$, $numConsColetasAtendidas_{CVRP} + taxa]$. Para isso, é gerado um número aleatório neste intervalo e, dado este valor, consumidores com demandas de coletas não satisfeitas são selecionados aleatoriamente para o atendimento destas demandas. Conforme mencionado nos algoritmos anteriores, este procedimento pode resultar na inviabilidade da solução devido a extrapolação da capacidade do veículo. Assim sendo, este consumidor poderá ser realocado na primeira posição viável encontrada. Caso nenhuma posição viável seja encontrada, uma nova rota é criada para a inclusão deste consumidor.

O método *removeColeta* é aplicado à solução s_{VRPSPD} (linha 11) gerando uma solução na qual o número de consumidores que não terão suas demandas de coletas atendidas irão variar entre $[numConsColetasAtendidas_{VRPSPD} - taxa, numConsColetasAtendidas_{VRPSPD}[$. De forma similar, gera-se um número aleatório neste intervalo e, então, consumidores com demandas de coletas satisfeitas são selecionados também de maneira aleatória para que as suas demandas de coleta não sejam atendidas. Este procedimento não causa inviabilidade da solução. Logo, a posição dos consumidores nas rotas permanece a mesma.

5.7.3 Fase de Recombinação

A população filha é gerada a partir da recombinação de indivíduos da população P. Primeiramente, dois indivíduos "pais" são selecionados e os indivíduos "filhos" são gerados pela aplicação de um operador de cruzamento. Dois operadores são considerados neste trabalho: Cruzamento Baseado em Rota e Cruzamento Split. Ambos operadores de cruzamento tem igual probabilidade de serem selecionados para o seu uso. Cada indivíduo "filho" pode sofrer mutação dada uma probabilidade de mutação p_{mut}. O operador de mutação utilizado neste trabalho é o *Or-opt* e *Altera Coleta* descritos na seção 5.3, também com igual probabilidade de serem selecionados para serem aplicados.

5.7.3.1 Operador de Seleção

A fase de seleção do NSGA-II ocorre em dois momentos distintos do algoritmo. O primeiro deles, quando a população inicial é gerada, o algoritmo utiliza a técnica torneio binário da seguinte forma: dados dois indivíduos a e b, se um indivíduo a domina o indivíduo b, então

Algoritmo 5.14 Geração da População Inicial

Entrada: População Pop, número de indivíduos da população p, total de consumidores n

{Gerando soluções para o CVRP e VRPSPD};
1: $s_{CVRP} \leftarrow$ gerar solução para o CVRP utilizando GVNS;
2: $s_{VRPSPD} \leftarrow$ gerar solução para o VRPSPD utilizando GVNS;
3: $numConsColetasAtendidas_{CVRP} \leftarrow 0$;
4: $numConsColetasAtendidas_{VRPSPD} \leftarrow n$;
{Inserindo soluções na população}
5: inserir(Pop, $[s_{CVRP}, s_{VRPSPD}]$);
{Calculando número de soluções geradas a partir do CVRP e VRPSPD}
6: $numSolucoes \leftarrow (p-2)/2$;
{Calculando a taxa de incremento ou decremento da satisfação das demandas de coleta}
7: $taxa \leftarrow n/numSolucoes$;
{Gerando as demais soluções da população}
8: $solucoesGeradas \leftarrow 0$;
9: **enquanto** $solucoesGeradas < numSolucoes$ **faça**
10: $s_{CVRP} \leftarrow insereColeta(taxa, s_{CVRP}, numConsColetasAtendidas_{CVRP})$;
11: $s_{VRPSPD} \leftarrow removeColeta(taxa, s_{VRPSPD}, numConsColetasAtendidas_{VRPSPD})$;
12: inserir(Pop, $[s_{CVRP}, s_{VRPSPD}]$);
13: $solucoesGeradas \leftarrow solucoesGeradas + 1$;
14: **fim enquanto**
15: **retornar** Pop;

o indivíduo a é copiado para a população Q; caso contrário, o indivíduo b é inserido na população Q. Se a não domina b e b não domina a, ambos indivíduos são copiados.

A partir de então, aplica-se o método de seleção por torneio de multidão, no qual uma solução s_i, pertencente a fronteira i após a ordenação de dominância, é considerada melhor que a solução s_j, pertencente a fronteira j, se [Deb, 2001]:

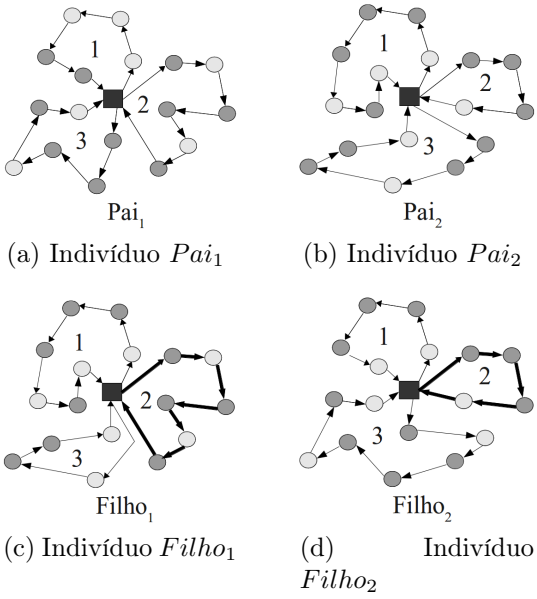
1. s_i possui um nível de não dominância melhor que s_j, ou seja, se $i < j$.
2. $i = j$ e a distância de multidão de s_i é maior que distância de multidão de s_j.

5.7.3.2 Operadores de Cruzamento

Neste trabalho foram adotados dois métodos de cruzamento: Cruzamento Baseado em Rota e Cruzamento *Split*. O Cruzamento *Split* faz uso do cruzamento sequencial comumente utilizado nos problemas do caixeiro viajante. Os métodos de cruzamento são descritos a seguir.

Cruzamento Baseado em Rota O cruzamento baseado em rota (RBX, do inglês *Route Based Crossover*), proposto por Potvin e Bengio [1996], seleciona aleatoriamente algumas rotas de um indivíduo Pai_1 para serem incluídas no indivíduo $Filho_1$. Em seguida, os demais

consumidores não contidos na solução do $Filho_1$ são inseridos de acordo com a solução do indivíduo Pai_2. De forma similar, algumas rotas do indivíduo Pai_2 são selecionadas aleatoriamente para serem incluídas no indivíduo $Filho_2$ e este indivíduo é completado de acordo com as rotas do indivíduo Pai_1.



(a) Indivíduo Pai_1 (b) Indivíduo Pai_2

(c) Indivíduo $Filho_1$ (d) Indivíduo $Filho_2$

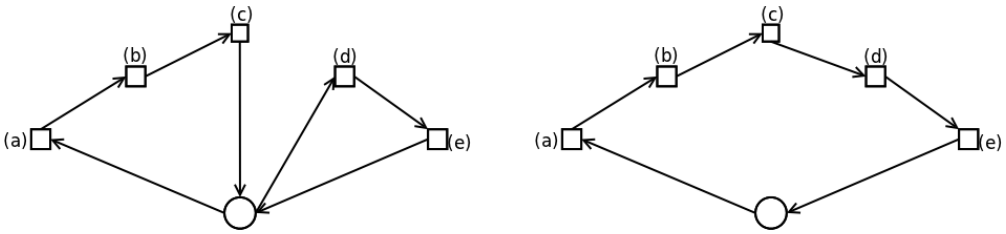
Figura 5.8: Cruzamento Baseado em Rota (RBX)

Este método é utilizado por recombinar tanto a posição dos consumidores nas rotas quanto o estado de coleta dos consumidores. Um exemplo do funcionamento do método de cruzamento RBX é apresentado na figura 5.8. O quadrado negro indica a posição do depósito e as circunferências na cor clara representam os consumidores tais que suas demandas de coletas são atendidas e as circunferências na cor escura representam os consumidores com coletas não satisfeitas. As figuras (a) e (b) são os dois indivíduos (Pai_1 e Pai_2) selecionados para o cruzamento. O indivíduo $Filho_1$ é gerado a partir da recombinação dos indivíduos Pai_1 e Pai_2, no qual a rota 2 é totalmente mantida do Pai_1 e as demais são inseridas do Pai_2. Da mesma forma, o indivíduo $Filho_2$ mantém a rota 2 do Pai_2 e o restante é inserido conforme Pai_1. Quando um consumidor é copiado de uma solução Pai para uma solução $Filho$, o estado de coleta desse consumidor é mantido do indivíduo Pai.

Cruzamento *Split* O método de cruzamento *Split* (SplitX), proposto por Prins [2004], consiste da transformação dos indivíduos "pais" em soluções para o problema do caixeiro viajante (TSP). A estas soluções, aplica-se o método de cruzamento sequencial (OX), detalhado a seguir, e então a solução para o TSP é novamente transformada em uma solução para o problema de roteamento de veículos através de um procedimento denominado *Split*.

A transformação de uma solução do problema de roteamento de veículos em uma solução do problema do caixeiro viajante é feita a partir da remoção das arestas de incidência ao

depósito, conforme figura 5.9. Apenas duas delas são mantidas a fim de incluir o depósito na solução.



(a) Solução VRP (b) Solução TSP

Figura 5.9: Transformando uma solução para o problema de roteamento de veículos (VRP) em uma solução para o problema do caixeiro viajante (TSP)

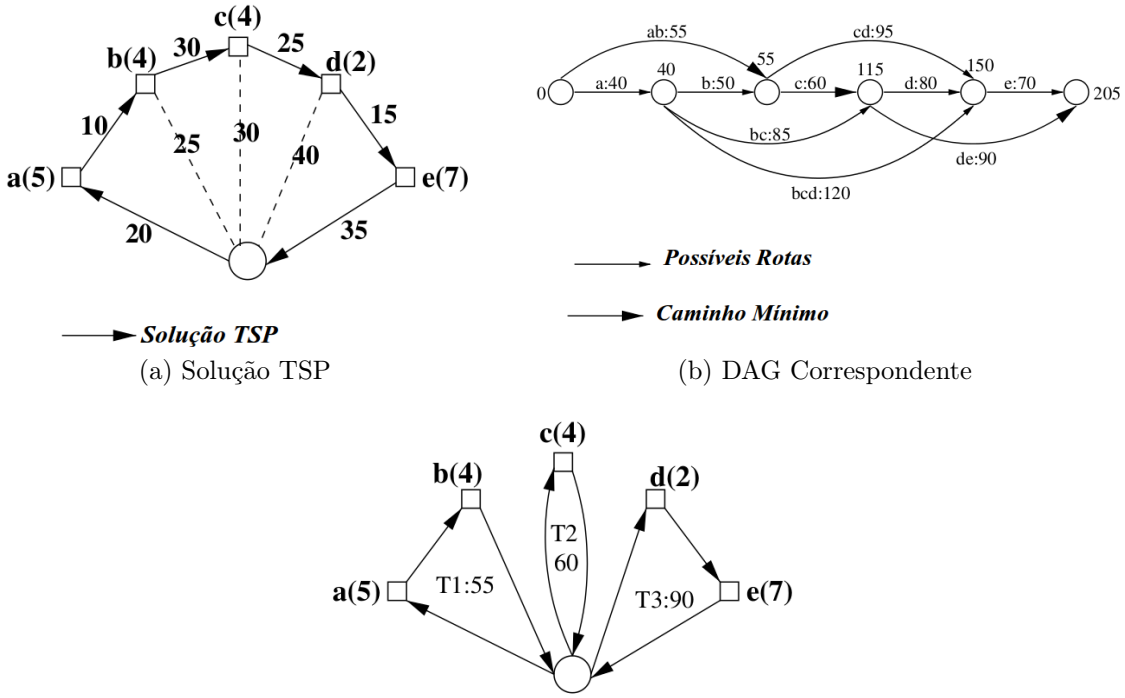
Assim, dada uma solução para o problema do caixeiro viajante, aplica-se o operador de cruzamento sequencial, muito comum para esse tipo de problema. Em seguida, o algoritmo transforma a solução obtida novamente em uma solução para o problema de roteamento de veículos através da inclusão de novas arestas de incidência ao depósito, particionando, assim, a solução do problema do caixeiro viajante.

O processo de transformação de um problema do caixeiro viajante em um problema de roteamento de veículo inicialmente constrói um grafo direcionado acíclico (DAG) D=(V,A) com V=$\{v_0, v_1, ..., v_n\}$, sendo n o número de consumidores e o vértice v_0 representando o depósito. As arestas contidas na DAG representam diferentes rotas viáveis, definidas como: para todo vértice v_i e v_j pertencente V, tal que $i < j$, o arco (v_i, v_j) pertence ao conjunto de arestas A se a capacidade do veículo nesta rota não é extrapolada.

Na figura 5.10 é mostrado como gerar um grafo direcionado acíclico (figura 5.10b) a partir de uma solução do problema do caixeiro viajante (figura 5.10c). Na solução do TSP, os números indicados entre parênteses representam a demanda de entrega de cada consumidor e os valores nas arestas representam o custo de percorrê-la. Na DAG, cada aresta indica uma rota possível de ser gerada a partir da solução do TSP. Por exemplo, a aresta $a : 40$ indica que a rota $0 - a - 0$ tem custo 40, a aresta $bcd : 120$ indica que a rota $0 - b - c - d - 0$ tem custo 120. A aresta $abc : 90$, por exemplo, não aparece na DAG, pois a carga do veículo extrapola sua capacidade máxima (Q=10).

Assim sendo, dado o grafo direcionado acíclico, busca-se, então, o caminho mais curto neste grafo. Na figura 5.10b, os valores indicados acima do vértice mostram o valor do caminho mais curto de alcançar este vértice a partir do vértice v_0. O valor 205 acima do último vértice indica que o caminho mais curto para alcançá-lo, partindo do vértice v_0, tem custo 205, sendo que este caminho passa pelas arestas: $ab : 55, c : 60, de : 90$.

Portanto, dado o caminho mais curto, é possível definir as partições que serão aplicadas



(a) Solução TSP

(b) DAG Correspondente

(c) Solução VRP

Figura 5.10: Transformando uma solução para o problema do caixeiro viajante (TSP) em uma solução para o problema de roteamento de veículos (VRP) [Prins, 2004]

à solução do problema do caixeiro viajante. Serão criadas as seguintes rotas: $0-a-b-0$, $0-c-0$ e $0-d-e-0$, como mostrado na figura 5.10a.

Cruzamento Sequencial Conforme mencionado, o operador de cruzamento *Split* utiliza o cruzamento sequencial (OX, do inglês *Order Crossover*) para recombinar indivíduos que representam soluções para o problema do caixeiro viajante.

O cruzamento sequencial copia uma subsequência da solução Pai_1, obtida a partir de dois pontos de corte, para o $Filho_1$. Os demais pontos são inseridos na ordem em que aparecem na solução Pai_2.

Um exemplo deste operador é apresentado na figura 5.11. O primeiro filho ($Filho_1$) recebe os valores contidos na subsequência $6-5-4$ gerada pelos dois pontos de corte no Pai_1, na mesma posição em que eles aparecem. O $Filho_1$ terá, então, a seguinte configuração $x-x-x-6-5-4-x-x-x$. Em seguida, retira-se do Pai_2 estes valores (6,5 e 4), gerando o trajeto $2-3-7-8-1-9$, iniciando a sequência a partir do segundo ponto de corte. Estes valores são inseridos nesta ordem no $Filho_1$, também iniciando a inserção a partir do segundo ponto de corte. Logo, ao término do cruzamento, o $Filho_1$ será $8-1-9-6-5-4-2-3-7$, conforme mostra a figura.

Como este operador foi utilizado para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional, ao copiar um consumidor para uma solução "filho" ele mantém o seu estado de coleta. Isso permite que o método consiga explorar ambos os objetivos do problema tratado.

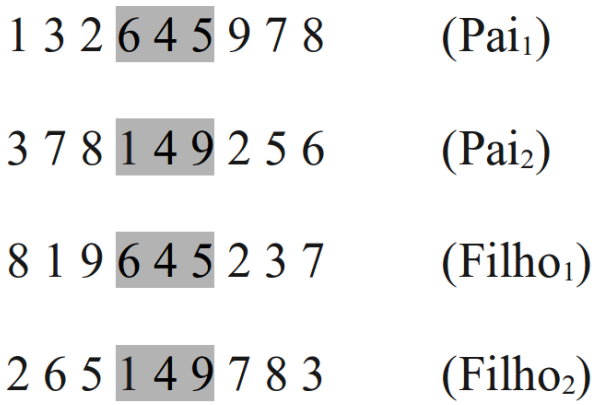


Figura 5.11: Cruzamento Sequencial (OX)

5.8 Busca Local Multiobjetivo Baseada em Indicador (IBMOLS)

Devido a complexidade computacional de diversos problemas de otimização combinatória com múltiplos objetivos, a resolução destes problemas recai em encontrar uma boa aproximação da fronteira de Pareto ótima. Em algumas situações, definir o que é uma boa aproximação da fronteira não é uma tarefa muito simples e irá depender do tomador de decisão e do cenário a ser otimizado [Basseur et al., 2011].

Os algoritmos para resolução de problemas multiobjetivo fazem uso de mecanismos para minimizar a distância da fronteira de Pareto ótima e maximizar a diversidade das soluções contidas na fronteira. Zitzler e Künzli [2004] propuseram o *indicator-based evolutionary algorithm* (IBEA) a fim de formalizar preferências em termos da relação de dominância, ou seja, os autores assumiram que a preferência do tomador de decisão é dada em termos de um indicador binário $\mathcal{I}$ de qualidade que irá associar um número real a cada fronteira de Pareto aproximada. Este indicador mede a qualidade de cada indivíduo da fronteira. Muitos indicadores têm sido propostos com a intenção de capturar diferentes preferências de otimização. Dessa forma o objetivo do problema é encontrar uma fronteira de Pareto aproximada que minimize o valor de $\mathcal{I}$.

Em geral, os algoritmos baseados em população são mais facilmente adaptáveis ao contexto multiobjetivo. Assim sendo, este trabalho explora as principais funcionalidades do algoritmo de busca local multiobjetivo baseada em indicador (IBMOLS, *Indicator-Based Multiobjective*

Local Search), proposto por Basseur et al. [2011], que trabalha com uma população de indivíduos. Além disso, este método apresenta uma fase de busca local, que também é desejável para problemas da otimização combinatória, como o VRPSP abordado neste trabalho.

Segundo Basseur et al. [2011], o IBMOLS foi definido principalmente para problemas de otimização combinatória discreto, como é o caso do problema abordado. Ele utiliza os princípios básicos da busca local e a aptidão de cada solução é obtido a partir de um indicador binário. Durante o processo de seleção, o principal objetivo é remover soluções de pior qualidade, dado o indicador aplicado, em relação aos demais indivíduos da população.

No algoritmo 5.15 são apresentados os passos básicos do IBMOLS, proposto por Basseur et al. [2011].

Algoritmo 5.15 Algoritmo IBMOLS proposto por Basseur et al. [2011]

Entrada: N (Tamanho da População), I(Indicador Binário)

1: $PO \leftarrow \emptyset$;
2: **repita**
3: $P \leftarrow$ População Inicial(N);
4: $A \leftarrow$ Soluções não dominadas (P);
5: $Aptidao[\,] \leftarrow$ Calcular Aptidão(P,I)
6: $s_w \leftarrow$ Menor($Aptidao$);
7: **para todo** solução $s \in P$ **faça**
8: **repita**
9: $s' \leftarrow$ vizinho não explorado de s;
10: $P \leftarrow P \cup s'$;
11: $Aptidao[s'] \leftarrow$ Calcular Aptidão(s', I);
12: $Aptidao[\,] \leftarrow$ Atualiza Aptidão(P,I);
13: $s_w \leftarrow$ Menor($Aptidao$);
14: $P \leftarrow P/\ s_w$;
15: $Aptidao[\,] \leftarrow$ Atualiza Aptidão(P,I);
16: **até** todas as vizinhanças tenham sido exploradas ou $s_w = s'$
17: $PO \leftarrow$ Soluções não dominadas ($PO \cup P$);
18: **fim para**
19: **até** Critério de parada
20: **retornar** PO;

Assis et al. [2013a] mostraram que o MOGVNS (ou como os autores o denominam: Multi-objective Iterated Local Search), apresentou resultados superiores ao IBMOLS, proposto por Basseur et al. [2011]. Assim sendo, neste trabalho, algumas modificações no algoritmo são apresentadas, inserindo algumas das principais características do GVNS (seção 5.6) dentre elas a fase de Perturbação (seção 5.3) e a fase de Busca Local (seção 5.2). Para o problema abordado, a necessidade de uma fase de busca local que explore todas as vizinhanças de uma determinada solução, ou seja, que seja do tipo "melhor aprimorante" ajuda na geração de soluções não dominadas mais próximas da fronteira de Pareto ótima. Mantendo

estas características em conjunto com um indicador binário é possível encontrar soluções mais próximas da fronteira de Pareto ótima, como também gerar soluções mais diversificadas.

Algoritmo 5.16 Algoritmo IBMOLS modificado

Entrada: N (Tamanho da População), I(Indicador Binário)

1: $PO \leftarrow \emptyset$;
2: **repita**
3: $P \leftarrow$ População Inicial(N);
4: $Aptidao[\,] \leftarrow$Calcular Aptidão(P,I)
5: $s_w \leftarrow$ Menor($Aptidao$);
6: **para todo** solução $s \in P$ **faça**
7: continue $\leftarrow$ TRUE;
8: **enquanto** continue **faça**
9: $s' \leftarrow$ Perturbação(s);
10: $s'' \leftarrow$ Busca Local(s');
11: **se** $s = s''$ **então**
12: $s'' \leftarrow s'$; continue $\leftarrow$ FALSE;
13: **fim se**
14: **se** $Aptidao(s'') > Aptidao(s_w)$ **então**
15: $P \leftarrow P \cup \{s''\} - \{s_w\}$;
16: Calcular Aptidão(s'', s_w, P, I);
17: $s_w \leftarrow$ Menor($Aptidao$);
18: **senão**
19: continue $\leftarrow$ FALSE;
20: **fim se**
21: **fim enquanto**
22: **fim para**
23: $PO \leftarrow SoluesNoDominadas(PO \cup P)$;
24: **até** Critério de Parada
25: **retornar** PO;

Os passos básicos do algoritmo baseado no IBMOLS, proposto neste trabalho, são mostrados no algoritmo 5.16. Inicialmente o algoritmo gera uma população inicial P. Dada uma população de tamanho N, parte dos indivíduos são obtidos pela aplicação da heurística P_ϵ (algoritmo 5.9). Os indivíduos restantes da população são gerados aleatoriamente (linha 3). Dada a população P, o valor da aptidão de cada indivíduo é obtido aplicando um indicador binário I (linha 4). O cálculo do valor da aptidão é apresentado na subseção 5.8.1. Assim, para cada solução s pertencente a P, aplica-se uma perturbação e um mecanismo de busca local (linhas 9 – 10). Os mecanismos de perturbação e busca local aplicados são similares aos métodos utilizados no algoritmo MOGVNS, apresentados na seção 5.6. Em seguida, se a solução encontrada s'' for igual a solução s, considera-se, então, apenas a perturbação feita no indivíduo s (linhas 11 – 13). Neste caso, o indivíduo que passou pela perturbação é considerado com o intuito de preservar a diversidade da população. Assim ela poderá ser novamente selecionada e novamente perturbada podendo gerar outros pontos da fronteira. O algoritmo

verifica, então, se o indivíduo gerado é melhor que o pior indivíduo, então s'' é inserido e o pior é removido da população. Dessa forma, a aptidão de todos os indivíduos serão atualizados dada a modificação efetuada (linhas 14 – 17). A variável "*continue*" garante que o processo de modificação de um indivíduo s permanece enquanto a busca local estiver gerando novos indivíduos. Caso contrário, passa-se a explorar outro indivíduo.

5.8.1 Indicador Binário (I_ϵ)

O valor da aptidão dos indivíduos da população é obtido a partir do indicador de qualidade binário I_ϵ. Este indicador apresenta bons resultados para diferentes problemas de otimização combinatória [Basseur et al., 2011]. Sejam x_1 e x_2 pertencente a fronteira de Pareto. O indicador $I_\epsilon(x_1, x_2)$ mensura a distância entre os dois pontos no espaço dos objetivos $Z = \Re^n$, sendo n o número de objetivos (Eq. 5.3). Para evitar valores negativos, os valores de todas as funções objetivos de todas as soluções foram normalizados. Sejam m_i e M_i o menor e maior valor da função objetivo i dos indivíduos contidos na população P, respectivamente, cada função objetivo i de cada indivíduo x de P é normalizado conforme Equação 5.4 [Basseur et al., 2011].

$$I_\epsilon = max_{i\in\{1,\ldots,n\}}(f_i(x_1) - f_i(x_2)) \tag{5.3}$$

$$F_i(x) = \frac{f_i(x) - m_i}{M_i - m_i} \tag{5.4}$$

Dado um indicador binário I_ϵ e um fator de escala k, a Equação 5.5 calcula a qualidade de uma solução (x) em relação às demais $(P \setminus \{x\}, x)$ [Zitzler e Künzli, 2004].

$$I(P \setminus \{x\}, x) = \sum_{z\in P\setminus\{x\}} -e^{-I(z,x)/k} \tag{5.5}$$

Assim sendo, no algoritmo 5.16, o cálculo da aptidão de todos os indivíduos da população é obtido por: $I(P\setminus\{x\}, x)$, para todo $x \in P$ (linha 4). A atualização da aptidão após inserção de indivíduo e remoção do pior indivíduo (linha 16) é dada pelos procedimentos apresentados no algoritmo 5.17.

Algoritmo 5.17 Calcula Aptidão - IBMOLS

Entrada: s'' (inserted solution), s_w (removed solution), P (population) and I (binary indicator)

1: $Aptidao(s'') \leftarrow I(P \setminus \{s''\}, s'')$;
2: $Aptidao(z) \leftarrow Aptidao(z) + I(s'', z), \quad \forall z \in P$;
3: $Aptidao(z) \leftarrow Aptidao(z) - I(s_w, z), \quad \forall z \in P$;

5.9 Conclusão

Neste capítulo apresentou-se diversos algoritmos adaptados para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional. Para a geração destes algoritmos, foi apresentado um arcabouço de procedimentos contendo uma nova estrutura de dados, algoritmos mais eficazes para verificar a viabilidade de uma solução e diferentes mecanismos de busca local e perturbação. Além disso, foi implementado um algoritmo para solucionar problemas de roteamento mono-objetivo e uma heurística para definir uma solução de partida para o problema de roteamento multiobjetivo abordado nesta tese.

Para a geração de algoritmos eficientes e mais robustos é fundamental a utilização de estruturas de dados de boa qualidade. Neste capítulo foram apresentadas as principais estruturas de dados para armazenamento de um grafo: matriz de adjacência e lista de adjacência. Após verificar suas vantagens e desvantagens, foi proposta uma nova estrutura mesclando as principais características de ambas. A nova forma de armazenamento dos dados possibilita averiguar o custo de uma aresta com complexidade computacional $O(1)$, bem como buscar um vértice mais próximo ou mais distante a outro com a mesma complexidade. Como o problema é estático, uma vez preenchida a estrutura, esta é mantida durante todos os procedimentos dos algoritmos implementados. Assim, não há custo de manutenção da estrutura proposta para o problema tratado nesta tese.

Em termos de redução da complexidade computacional dos algoritmos empregados, outra relevante contribuição encontra-se na análise de viabilidade de uma solução. Os movimentos de inserção de um consumidor em uma determinada rota podem acarretar em inviabiliade por extrapolação da capacidade do veículo. Estes movimentos são executados inúmeras vezes e, assim, qualquer contribuição neste sentido resultará em algoritmos mais rápidos e, consequentemente, mais eficientes. Neste capítulo, um novo procedimento de verificação de inviabilidade é proposto que reduz consideravelmente a complexidade computacional em relação ao método tradicional.

Para refinar as soluções encontradas, em todos os algoritmos implementados foram aplicadas estratégias de busca local. Os procedimentos apresentados atuam em um dos objetivos do problema, sendo o outro objetivo fixado em um determinado valor. Assim, diversos movimentos foram implementados e avaliados, sendo a sua maioria visando a minimização do custo do transporte que é o objetivo mais difícil de ser alcançado. Além de mecanismos de busca local, diferentes procedimentos de perturbação foram desenvolvidos para geração de uma fronteira mais diversificada, bem como para escapar de ótimos locais.

O primeiro passo para resolução do problema de roteamento de veículos multiobjetivo com coleta opcional consistiu em solucionar dois problemas mono-objetivos: problema de roteamento de veículo capacitado e problema de roteamento de veículos com coleta e entrega simultâneas. Estes problemas representam as extreminades da fronteira e, sendo elas bem

definidas, o custo computacional de encontrar os demais pontos fica reduzido, pois sabe-se que as demais soluções encontram-se no intervalo entre estas duas. Neste trabalho, o algoritmo baseado no GVNS foi aplicado para solucionar ambos problemas mono-objetivo.

As duas soluções encontradas foram utilizadas na heurística P_ϵ, também proposta neste capítulo. É a partir destas duas soluções que a heurística inicia uma fase construtiva para gerar soluções no interior da fronteira. As soluções encontradas são refinadas pelo mesmo algoritmo utilizado para solucionar os problemas mono-objetivo. Esta heurística é aplicada para definir soluções de partida para os algoritmos implementados.

Em suma, este capítulo apresentou três algoritmos para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional. Os algoritmos implementados são baseados em busca local, como o MOGVNS, baseado em população, como o NSGA-II, e mistos como o IBMOLS. Procurou-se investigar ambas abordagens a fim de buscar um algoritmo que melhor se adeque ao problema tratado e que possa resultar em soluções de melhor qualidade.

Capítulo 6

Resultados

Este capítulo apresenta os resultados dos principais algoritmos propostos neste trabalho para solucionar o problema de roteamento de veículos multiobjetivo com coleta opcional. Os experimentos realizados consistem na execução dos algoritmos: busca geral em vizinhança variável multiobjetivo (MOGVNS), algoritmo genético de ordenação não-dominante II (NSGA-II) e busca local multiobjetivo baseada em indicador (IBMOLS).

Os algoritmos foram submetidos a um conjunto de 14 problemas da literatura na qual o número de consumidores varia entre 50 a 200 consumidores e outro conjunto de 40 instâncias contendo, em cada uma delas, 50 consumidores. Os resultados foram avaliados segundo as métricas hipervolume, cardinalidade, cobertura e tempo. Para avaliar as possíveis diferenças entre o desempenho dos métodos propostos, um planejamento experimental foi utilizado possibilitando detectar diferenças significativas entre os métodos e estimar a magnitude destes. Para corroborar a qualidade das soluções, as fronteiras obtidas foram comparadas ao limite inferior dos problemas e, em duas instâncias a solução ótima foi encontrada e utilizada na comparação.

Este capítulo está organizado em 6 seções. A primeira seção remete a modelagem do problema ressaltando os objetivos do problema tratado. Os problemas teste são descritos na seção 6.2. Os parâmetros livres dos algoritmos são apresentados na seção 6.3. A seção 6.4 mostra os detalhes do planejamento experimental e a seção 6.5 apresenta e discute os resultados obtidos pelos três algoritmos avaliados. Por fim, as conclusões do capítulo são apresentadas na seção 6.6.

6.1 Objeivos Avaliados

Neste capítulo serão apresentados uma análise dos resultados obtidos pelos algoritmos implementados: MOGVNS, NSGA-II e IBMOLS. Estes algoritmos retornam um conjunto de soluções não-dominadas. Para facilitar no entendimento deste capítulo, a nomenclatura ado-

tada para representar ambos objetivos e a sua definição é novamente reportada a seguir. Outros detalhes podem ser encontrados na subseção 4.2.1.

1. Custo de transporte: somatório dos pesos das arestas utilizadas na solução. Este peso representa a distância euclidiana entre os consumidores e entre consumidores e depósito.

$$Min\ f(\mathbf{x}) = \sum_{k=1}^{k_{max}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^{k} \tag{6.1}$$

2. Coletas não atendidas: somatório da quantidade de coleta não atendida pelos veículos na solução.

$$Min\ f(\boldsymbol{\ell}) = \sum_{j=1}^{n} p_j (1 - \ell_j) \tag{6.2}$$

6.2 Instâncias

Instâncias são problemas-teste criados para um determinado problema, servindo de base para avaliação e comparação de métodos para solucioná-los. Neste trabalho, são utilizados dois conjuntos de instâncias: um deles composto por 14 instâncias, propostas por Salhi e Nagy [1999], e outro composto por 40 instâncias, propostas por Dethloff [2001]. Ambas definidas inicialmente para o problema de roteamento de veículos com coleta e entrega simultâneas (VRPSPD). Como dito na seção 4.2, o problema de roteamento de veículos multiobjetivo com coleta opcional equivale ao VRPSPD quando todas suas coletas são atendidas. Dessa forma, as instâncias podem ser utilizadas sem nenhuma alteração nos seus dados originais.

No primeiro conjunto de instâncias, o número de consumidores varia entre 50 e 200. No conjunto de instâncias de Dethloff, todos os problemas-teste contêm 50 consumidores. As instâncias armazenam informações referentes a localização dos consumidores, suas demandas de coleta e entrega e, ainda, a capacidade máxima dos veículos. A localização dos consumidores é dada pelas coordenadas (x, y), sendo que o custo entre os mesmos é dado pela distância Euclidiana. As instâncias não apresentam restrição quanto ao número de veículos utilizados, sendo considerada uma frota com um número ilimitado de veículos.

As tabelas 6.1 e 6.2 apresentam os dados das 14 instâncias de Salhi e Nagy [1999] e 40 instâncias de Dethloff [2001], respectivamente. Em ambas tabelas, a primeira coluna (Nome) indica o nome da instância, a segunda coluna (Consumidores) informa o número de consumidores. A terceira ($\sum$Entregas) e quarta ($\sum$Coletas) colunas informam a soma de todas as demandas de entrega e de coleta, respectivamente. A quinta coluna (Veículos) indica o número máximo de veículos e a última coluna (Capacidade) informa a capacidade máxima dos veículos.

Tabela 6.1: Instâncias propostas por Salhi e Nagy [1999]

Nome	**Consumidores**	**$\sum$Entregas**	**$\sum$Coletas**	**Veículos**	**Capacidade**
CMT1X	50	460.478	316.522	∞	160
CMT1Y	50	316.522	460.478	∞	160
CMT2X	75	816.894	547.106	∞	140
CMT2Y	75	547.106	816.894	∞	140
CMT3X	100	837.942	620.058	∞	200
CMT3Y	100	620.058	837.942	∞	200
CMT12X	100	942.045	867.955	∞	200
CMT12Y	100	867.955	942.045	∞	200
CMT11X	120	611.015	763.985	∞	200
CMT11Y	120	763.985	611.015	∞	200
CMT4X	150	1298.42	936.58	∞	200
CMT4Y	150	936.58	1298.42	∞	200
CMT5X	200	1871.08	1314.92	∞	200
CMT5Y	200	1314.92	1871.08	∞	200

A partir da tabela 6.1 pode ser observado um padrão no número total de coletas e de entregas para instâncias de mesmo prefixo. Por exemplo, da instância CMT1X para a instância CMT1Y as demandas de coleta e entrega são alternadas, ou seja, a demanda de entrega de um consumidor na instância CMT1X é demanda de coleta do mesmo consumidor na instância CMT1Y e vice-versa. Uma outra característica que este conjunto de instâncias apresenta está relacionada à disposição dos consumidores. Nos problemas CMT1 a CMT4, as cidades são geradas aleatoriamente no plano, ou seja, estão distribuídas igualmente no entorno do depósito. Nos problemas CMT11 e CMT12 as cidades estão dispostas em grupos e o depósito está centralizado entre estes grupos.

A tabela 6.2 apresenta 40 instâncias sendo que em 20 destas, denominadas SCA, as coordenadas dos consumidores são distribuídas uniformemente no intervalo de 0 a 100. As demais instâncias, denominadas COM, metade delas seguem esta mesma distribuição e as outras 10 instâncias são distribuídas uniformemente no intervalo de 100/3 a 200/3, assegurando que existe pelo menos uma cidade em cada 1/9 da área total [Dethloff, 2001].

6.3 Ajustes dos Parâmetros

Inicialmente os parâmetros dos três algoritmos foram ajustados individualmente de forma incremental, observando a qualidade das soluções retornadas. Após este ajuste, verificou-se o tempo de execução de todos os algoritmos no qual concluiu-se que o MOGVNS apresentou o maior tempo de execução. Em seguinda, os parâmetros dos algoritmos NSGA-II e IBMOLS

Tabela 6.2: Instâncias propostas por Dethloff [2001]

Nome	**Consumidores**	**$\sum$Entregas**	**$\sum$Coletas**	**Veículos**	**Capacidade**
CON3-0	50	2424.30	2486.16	∞	808.099
CON3-1	50	2778.77	2841.06	∞	926.257
CON3-2	50	2563.48	2495.12	∞	854.495
CON3-3	50	2548.20	2765.48	∞	849.399
CON3-4	50	2632.91	2664.88	∞	877.636
CON3-5	50	2317.62	2225.79	∞	772.54
CON3-6	50	2202.33	2090.14	∞	734.111
CON3-7	50	2487.04	2327.13	∞	829.015
CON3-8	50	2471.76	2461.52	∞	823.919
CON3-9	50	2156.47	2123.58	∞	718.823
CON8-0	50	2424.30	2486.16	∞	303.037
CON8-1	50	2778.77	2841.06	∞	347.346
CON8-2	50	2563.48	2495.12	∞	320.435
CON8-3	50	2548.20	2765.48	∞	318.524
CON8-4	50	2632.91	2664.88	∞	329.114
CON8-5	50	2317.62	2225.79	∞	289.703
CON8-6	50	2202.33	2090.14	∞	275.292
CON8-7	50	2487.04	2327.13	∞	310.881
CON8-8	50	2471.76	2461.52	∞	308.97
CON8-9	50	2156.47	2123.58	∞	269.559
SCA3-0	50	2471.06	2500.50	∞	823.685
SCA3-1	50	2317.51	2350.87	∞	772.504
SCA3-2	50	2498.01	2657.77	∞	832.671
SCA3-3	50	2578.52	2634.25	∞	859.506
SCA3-4	50	2959.02	3041.18	∞	986.34
SCA3-5	50	2439.52	2481.75	∞	813.174
SCA3-6	50	2420.02	2332.95	∞	806.675
SCA3-7	50	2500.53	2692.38	∞	833.509
SCA3-8	50	2481.03	2326.36	∞	827.01
SCA3-9	50	2261.53	2368.44	∞	753.844
SCA8-0	50	2471.06	2500.50	∞	308.882
SCA8-1	50	2317.51	2350.87	∞	289.689
SCA8-2	50	2498.01	2657.77	∞	312.252
SCA8-3	50	2578.52	2634.25	∞	322.315
SCA8-4	50	2959.02	3041.18	∞	369.877
SCA8-5	50	2439.52	2481.75	∞	304.94
SCA8-6	50	2420.02	2332.95	∞	302.503
SCA8-7	50	2500.53	2692.38	∞	312.566
SCA8-8	50	2481.03	2326.36	∞	310.129
SCA8-9	50	2261.53	2368.44	∞	282.692

foram novamente ajustados com o objetivo de proporcionar a cada método, aproximadamente, o mesmo tempo de execução.

Assim sendo, para solucionar os problemas teste, foram atribuídos os seguintes valores aos parâmetros livres dos algoritmos:

1. ILS: o algoritmo, baseado no ILS, utilizado pela heurística P_ϵ para encontrar soluções dos problemas de roteamento de veículos mono-objetivo foi definido conforme apresentado por Penna et al. [2013].
2. P_ϵ Heuristic: testes preliminares sugeriram $maxCount = 15$;
3. MOGVNS: após testes preliminares, às constantes do algoritmo foram atribuídos os seguintes valores: $maxIter = 100$ e $maxCount = 18$;
4. NSGA-II: O número de gerações do algoritmo e o tamanho da população foi definido como 200 e 100, respectivamente. A taxa de mutação (p_{mut}) é igual a 0.3 por indivíduo e a probabilidade de cruzamento é igual a 1;
5. IBMOLS: O número de gerações do algoritmo e o tamanho da população foi definido como 50 e 10, respectivamente. O fator de escala do indicador binário k foi definido como 0.1.

A solução inicial dos algoritmos MOGVNS, NSGA-II e IBMOLS foi obtida a partir da aplicação da heurística P_ϵ (seção 5.5).

6.4 Planejamento Experimental

Esta seção apresenta o planejamento experimental adotado para avaliação dos resultados obtidos pelos algoritmos MOGVNS, NSGAII e IBMOLS. Com o planejamento do experimento e a análise estatística dos dados obtidos é possível averiguar se há diferenças significativas entre o desempenho dos algoritmos e estimar o tamanho dessas diferenças.

Para cada algoritmo foram realizadas 15 replicações em cada instância de teste descrita na seção anterior. Os testes foram executados de maneira independente e foram avaliadas as seguintes medidas de qualidade: Hipervolume (S), Cardinalidade (Card), Cobertura (C) e Tempo (T), descritas na seção 2.4. Para a métrica Cobertura de Dois Conjuntos foi aplicada uma versão generalizada, denominada Cobertura de Vários Conjuntos [Batista et al., 2012]. Ao invés de quantificar o quanto um determinado algoritmo abrange outro, essa métrica calcula a quantidade que um determinado algoritmo cobre a união das fronteiras finais retornadas por todos os algoritmos, exceto a fronteira obtida por ele mesmo.

Os valores obtidos pela métrica hipervolume foram normalizados considerando o hipervolume da fronteira Pareto aproximada obtida a partir das fronteiras retornadas pelas 15 replicações de todos algoritmos. Os dados da métrica C não foram alterados, pois os valores retornados já são normalizados.

Para cada métrica foi aplicado o mesmo teste estatístico, buscando encontrar as diferenças significativas entre os algoritmos considerando cada uma delas. Como foram utilizadas diferentes instâncias de teste, o modelo estatístico que descreve os dados deste experimento pode ser escrito como:

$$y_{ijk} = \mu_i + \beta_j + \epsilon_{ijk} \begin{cases} i = 1, \ldots, a \\ j = 1, \ldots, b \\ k = 1, \ldots, r \end{cases} \tag{6.3}$$

O valor de y_{ijk} corresponde ao valor da métrica obtida pelo i-ésimo algoritmo na j-ésima instância da k-ésima replicação. O valor de μ_i representa a média geral dos valores obtidos pelo i-ésimo algoritmo dada a métrica em questão e β_j corresponde ao efeito que a j-ésima instância provoca na variável de resposta. A variável aleatória ϵ_{ijk} consiste no erro aleatório experimental, isto é, as variações provocadas por outros fatores que podem influenciar no valor de resposta e que não foram considerados no experimento. Os parâmetros a, b e r representam o número de algoritmos avaliados, o número de instâncias utilizadas e o número de replicações do experimento, respectivamente, sendo que $a = 3$, $b = 14$ *ou* 40 e $r = 15$. Por ter características distintas, os dois grupos de instâncias foram analisados separadamente, sendo um grupo contendo 14 instâncias (Salhi & Nagy) e outro 40 instâncias (Dethloff). Assim, é possível perceber a eficiência dos algoritmos em problemas de pequena magnitude, representada pelas instâncias de Dethloff, e problemas de magnitude variada, representada pelas instâncias de Salhi & Nagy.

Para melhorar a estimativa da diferença no desempenho entre os algoritmos avaliados, o experimento será realizado em blocos completos, no qual cada instância corresponde a um bloco. O termo "bloco" refere a uma unidade experimental relativamente homogênea. A "blocagem" reduz ou elimina alguns fatores inconvenientes que podem influenciar a resposta, mas não são interessantes para a análise. Neste trabalho, as diferenças entre as instâncias podem interferir nos resultados obtidos pelos algoritmos e, quando aplica-se a "blocagem" dos experimentos por instância, é possível remover o fator "instância" da análise [Montgomery, 1991].

Para cada métrica considerada, a hipótese nula indica a ausência de diferença entre os algoritmos avaliados: MOGVNS (1), NSGAII (2) e IBMOLS (3), contra sua hipótese alternativa de que existe diferença em pelo menos dois algoritmos. Caso os algoritmos apresentem o mesmo desempenho e não haja diferença significativa, a hipótese nula é aceita. Por outro lado, caso haja diferenças significativas no desempenho dos algoritmos avaliados a hipótese

nula é rejeitada. Para a análise dos experimentos foi considerado um nível de significância de 95% o que corresponde a um valor de α igual a 0.05.

Para evitar suposições de que os dados apresentam uma distribuição normal, é utilizado o teste de Friedman [Montgomery e Runger, 2003]. Este teste é uma alternativa onde não é possível afirmar ou justificar sobre a normalidade dos dados. Uma vez finalizado o teste de hipótese, caso a hipótese nula seja rejeitada, isso implica que há uma diferença significativa entre os algoritmos implementados. Assim, precisa-se estimar o magnitude dessa diferença. Neste trabalho o tamanho de efeito é obtido utilizando a função "*multicompare*" no MATLAB®. Os intervalos de confiança obtidos foram ajustados utilizando-se a correção de Dunn-Šidák [Dunn, 1961].

6.5 Resultados e Discussões

Os testes foram executados em um servidor com 4 processadores Intel® Xeon® 7550 de 2.00 GHz, 6 cores por processador, com 128 GB de memória RAM e sistema operacional GNU/Linux, distribuição Ubuntu Server 12.04.1 LTSs. Para a execução dos algoritmos MOGVNS, NSGA-II e IBMOLS um núcleo foi utilizado para executar cada instância do problema. Para definição da solução ótima e limites inferiores, todos os núcleos foram utilizados.

Os testes de hipótese foram realizados para as quatro métricas separadamente e os resultados foram avaliados separadamente para cada grupo de instância: Dethloff e Salhi & Nagy.

6.5.1 Instâncias de Dethloff

Considerando o grupo de instâncas de Dethloff, foram obtidas 1800 amostras para cada métrica, uma vez que foram avaliados 3 algoritmos em 40 instâncias de testes com 15 replicações ($3 \times 40 \times 15 = 1800$). Como o número de amostras é maior do que 25 observações então é utilizado a aproximação à função de distribuição normal. Como dito anteriormente, em todos os testes é considerado o nível de significância $\alpha = 0.05$, assim, o valor crítico correspondente é $z_{0.025} = 1.96$.

O resultado do teste de hipótese aponta para a rejeição da hipótese nula para todas as métricas avaliadas. A tabela 6.3 apresenta o p-valor obtido em cada métrica.

Tabela 6.3: P-valor obtido após teste de hipótese

Hipervolume	**Cobertura**	**Cardinalidade**	**Tempo**
$< 1.95e^{-130}$	$<8.40e^{-143}$	$<4.41e^{-155}$	0

As figuras 6.1, 6.2, 6.3 e 6.4 apresentam as médias estimadas dos algoritmos MOGVNS, NSGAII e IBMOLS para as métricas hipervolume, cobertura, cardinalidade e tempo. O MOGVNS foi inferior aos demais algoritmos apenas no tempo de execução. Neste quesito, o NSGA-II apresenta menor tempo de execução, seguido do IBMOLS e MOGVNS. Por outro lado, a análise dos dados obtidos para as métricas hipervolume, cobertura e cardinalidade apontam uma superioridade do MOGVNS em relação aos demais algoritmos. O IBMOLS foi superior ao NSGA-II nas métricas cardinalidade e hipervolume. Quanto a cobertura, não houve diferenças significativas entre este dois algoritmos.

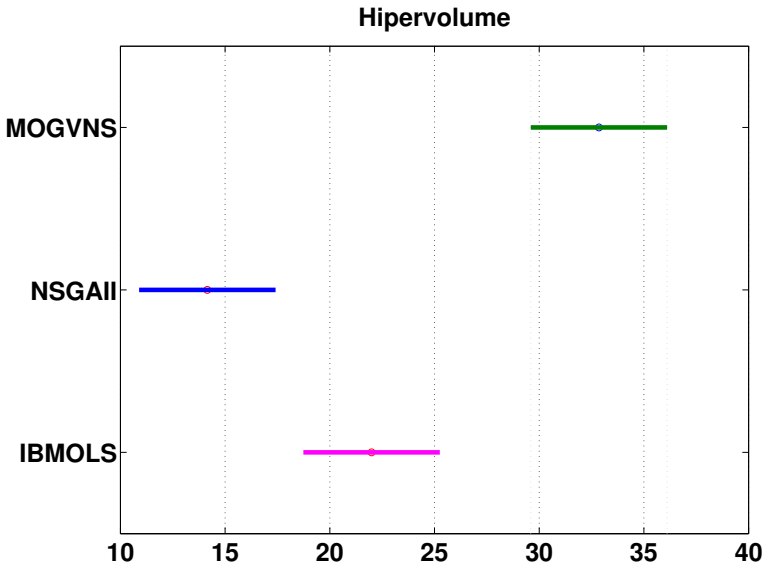


Figura 6.1: Hipervolume: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Dethloff

Os gráficos apresentados destacam a superioridade do MOGVNS, mostrando diferenças significativas entre ele e os demais algoritmos, tendo maiores valores para as médias estimadas envolvendo as métricas hipervolume, cobertura e cardinalidade. Porém, este algoritmo apresenta maior tempo de execução.

Para complementar, as tabelas 6.4 e 6.5 apresentam a média e desvio padrão das métricas: hipervolume, cobertura, cardinalidade e tempo, dos valores obtidos nas 15 replicações para cada instância e algoritmo avaliado. Na tabela também é possível verificar a superioridade do MOGVNS que apresenta média geral e desvio padrão com maiores valores em todas as métricas, o que é desejável. Quanto ao tempo de execução, o NSGAII apresenta um valor menor na média geral, confirmando um menor tempo de execução deste algoritmo.

Devido ao não conhecimento da fronteira Pareto-Ótima para todas as intâncias, fez-se uma comparação dos melhores resultados obtidos pelo MOGVNS com uma abordagem mono-objetivo. Foram utilizados apenas os dados do MOGVNS uma vez que, nas análises anteriores, verificou-se que este algoritmo apresentou melhores resultados que os demais. Assim sendo,

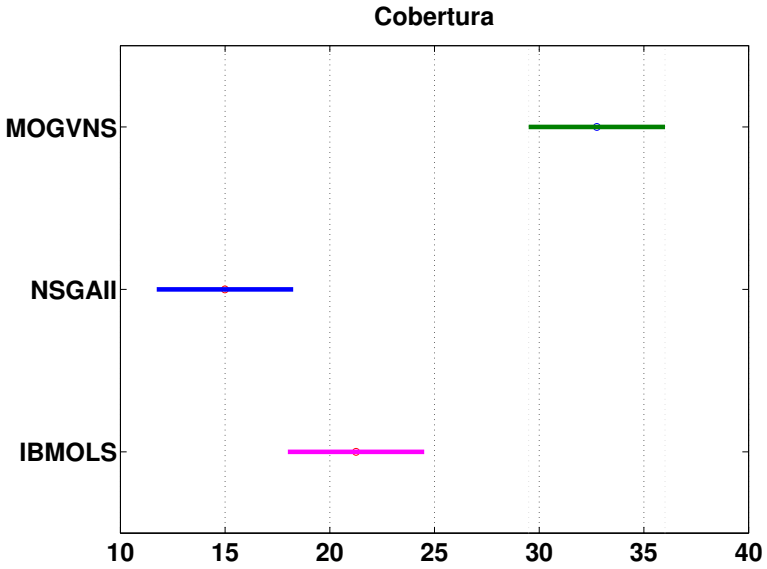


Figura 6.2: Cobertura: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Dethloff

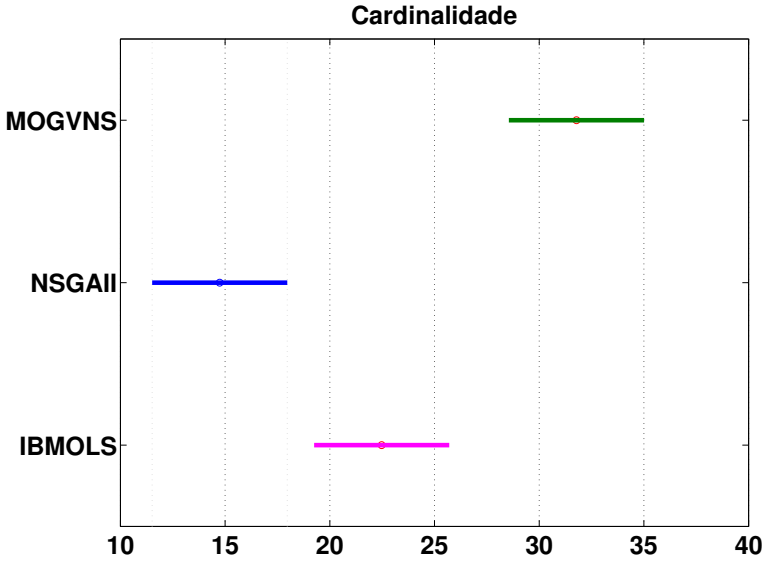


Figura 6.3: Cardinalidade: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Dethloff

para atestar a qualidade do algoritmo, as soluções obtidas foram comparadas às soluções ótimas do problema de roteamento de veículos com coleta e entrega simultânea (VRPSPD) [Subramanian et al., 2010]. As soluções consideradas na comparação são aquelas que se localizam no extremo da fronteira no qual todas as coletas são atendidas. Na tabela 6.6 esta comparação é apresentada. A última coluna indica a diferença percentual entre os resultados obtidos pelo MOGVNS e a solução ótima. O erro relativo (*gap*) é calculado da seguinte forma:

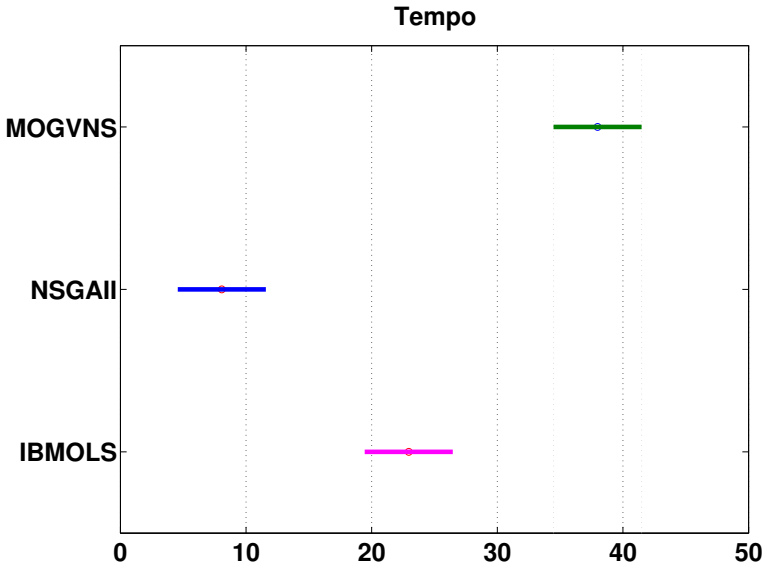


Figura 6.4: Tempo: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Dethloff

$$Erro\ relativo = \frac{\text{Solução MOGVNS} - \text{Solução Ótima}}{\text{Solução Ótima}} \times 100 \tag{6.4}$$

Como pode ser observado na tabela 6.6, o MOGVNS consegue encontrar boas soluções quando todas as demandas de coleta são consideradas, alcançando a solução ótima em 90% das instâncias, apresentando um erro relativo médio de 0.02.

A próxima subseção irá analisar os algoritmos a partir de instâncias com número de consumidores variados. Assim, será possível avaliar a qualidade dos mesmos tanto para instâncias com pequeno número de consumidores quanto para instâncias que variam o número de consumidores, sendo estas instâncias consideradas mais difíceis de resolver.

6.5.2 Instâncias de Salhi & Nagy

Considerando o grupo de instâncas de Salhi & Naghy, foram obtidas 630 amostras para cada métrica, uma vez que foram avaliados 3 algoritmos em 14 instâncias de testes com 15 replicações ($3 \times 14 \times 15 = 630$). Sendo o número de amostras maior que 25 observações então é utilizado a aproximação à função de distribuição normal. Como dito anteriormente, em todos os testes é considerado o nível de significância $\alpha = 0.05$, assim, o valor crítico correspondente é $z_{0.025} = 1.96$.

Nas métricas avaliadas o resultado do teste de hipótese aponta para a rejeição da hipótese nula para todas as métricas, exceto para o tempo de execução. Para esta métrica não houve diferença para os três algoritmos analisados. Isso confirma que os parâmetros livres dos

Tabela 6.4: Média e desvio padrão (entre parênteses) dos valores obtidos pelas métricas avaliadas.

Instâncias	Tempo ($10^3 s$)			Cardinalidade		
	MOGVNS	NSGAII	IBMOLS	MOGVNS	NSGAII	IBMOLS
CON3-0	11.14(0.42)	7.89(0.31)	9.33(0.48)	4.33(0.82)	3.67(0.82)	4.33(0.62)
CON3-1	10.42(0.41)	7.41(0.29)	8.76(0.47)	3.40(1.68)	2.47(0.52)	2.67(0.62)
CON3-2	10.64(0.45)	7.67(0.28)	8.93(0.44)	3.07(1.71)	2.13(0.99)	2.40(0.83)
CON3-3	11.00(0.43)	7.66(0.27)	9.10(0.43)	6.60(1.50)	4.93(0.70)	5.47(1.13)
CON3-4	10.68(0.41)	7.76(0.31)	8.94(0.48)	9.53(6.44)	3.07(0.26)	4.20(0.94)
CON3-5	10.89(0.44)	7.77(0.35)	9.03(0.48)	12.20(5.10)	2.13(0.35)	2.60(0.63)
CON3-6	10.70(0.42)	7.73(0.27)	8.93(0.48)	5.87(6.65)	1.20(0.41)	2.07(0.26)
CON3-7	10.21(0.39)	7.41(0.30)	8.66(0.45)	3.00(5.28)	1.07(0.26)	1.00(0.00)
CON3-8	10.85(0.42)	7.58(0.27)	8.93(0.43)	8.40(3.16)	4.80(0.86)	6.13(1.25)
CON3-9	10.99(0.42)	7.86(0.29)	9.21(0.47)	9.00(7.01)	1.07(0.26)	1.00(0.00)
CON8-0	7.97(0.37)	4.19(0.30)	5.84(0.23)	6.80(1.66)	4.67(2.06)	6.67(1.18)
CON8-1	5.91(0.31)	3.54(0.44)	4.63(0.34)	7.93(2.91)	3.93(1.16)	6.27(1.22)
CON8-2	8.72(0.25)	5.41(0.32)	7.03(0.24)	9.87(2.00)	6.13(1.77)	7.67(1.63)
CON8-3	9.16(0.29)	5.80(0.25)	7.52(0.29)	15.13(1.92)	10.40(3.40)	13.87(2.45)
CON8-4	7.49(0.37)	4.96(0.23)	6.15(0.35)	6.53(3.85)	4.27(1.03)	5.27(1.49)
CON8-5	8.08(0.39)	5.42(0.31)	6.76(0.39)	3.20(0.56)	2.00(0.93)	2.67(0.72)
CON8-6	8.92(0.31)	6.17(0.24)	7.63(0.40)	1.67(1.59)	1.07(0.26)	1.27(0.46)
CON8-7	8.44(0.28)	5.69(0.30)	7.19(0.32)	4.40(0.99)	2.47(0.74)	3.40(0.91)
CON8-8	6.51(0.46)	3.92(0.41)	5.10(0.38)	8.60(2.38)	4.13(1.30)	7.60(1.88)
CON8-9	8.53(0.28)	4.65(0.29)	6.27(0.38)	12.07(1.98)	6.67(2.38)	9.20(1.74)
SCA3-0	10.50(0.44)	7.41(0.27)	8.73(0.45)	5.67(0.90)	3.33(1.05)	3.53(0.92)
SCA3-1	10.80(0.38)	7.68(0.33)	8.94(0.46)	13.47(4.31)	2.00(0.00)	2.07(0.26)
SCA3-2	11.20(0.42)	8.06(0.28)	9.26(0.48)	15.47(0.92)	1.00(0.00)	1.00(0.00)
SCA3-3	10.79(0.41)	7.75(0.27)	9.02(0.45)	10.20(6.67)	1.60(0.51)	1.87(0.64)
SCA3-4	10.64(0.42)	7.48(0.28)	8.86(0.47)	7.27(5.06)	2.80(1.08)	3.27(0.80)
SCA3-5	10.55(0.40)	7.43(0.28)	8.77(0.46)	7.40(4.07)	2.87(0.64)	4.00(0.85)
SCA3-6	11.05(0.42)	8.04(0.26)	9.28(0.46)	15.00(3.87)	1.00(0.00)	1.00(0.00)
SCA3-7	10.62(0.41)	7.36(0.26)	8.74(0.47)	14.13(3.52)	9.73(1.39)	9.33(2.13)
SCA3-8	10.99(0.42)	7.72(0.29)	9.05(0.46)	3.60(1.30)	2.00(0.00)	2.47(0.52)
SCA3-9	10.86(0.43)	7.61(0.31)	8.97(0.44)	4.53(0.83)	3.00(0.00)	4.00(0.38)
SCA8-0	7.57(0.43)	4.54(0.39)	5.84(0.35)	7.27(1.62)	4.47(2.47)	6.60(2.06)
SCA8-1	6.37(0.55)	3.85(0.34)	5.09(0.43)	3.47(0.99)	2.73(2.02)	3.73(1.53)
SCA8-2	8.63(0.27)	5.12(0.35)	6.78(0.54)	12.40(2.35)	8.80(3.38)	9.20(1.61)
SCA8-3	6.08(0.42)	3.92(0.33)	4.84(0.32)	4.73(0.70)	3.80(0.68)	4.67(0.82)
SCA8-4	6.39(0.29)	3.83(0.40)	4.94(0.35)	6.27(0.96)	3.87(1.46)	5.80(1.08)
SCA8-5	8.25(0.38)	4.93(0.28)	6.55(0.42)	7.87(1.19)	6.53(1.64)	7.73(2.05)
SCA8-6	8.90(0.31)	5.79(0.23)	7.34(0.39)	6.00(0.76)	3.53(1.06)	5.13(0.99)
SCA8-7	9.59(0.39)	5.97(0.18)	7.66(0.38)	14.93(2.22)	14.73(4.92)	12.33(2.99)
SCA8-8	8.70(0.29)	6.01(0.30)	7.36(0.32)	5.33(5.73)	2.00(0.00)	2.00(0.00)
SCA8-9	8.14(0.35)	4.35(0.44)	6.00(0.42)	10.80(2.27)	6.93(2.12)	8.60(1.96)
Média	9.35(0.38)	6.28(0.30)	7.65(0.41)	7.94(2.74)	3.98(1.12)	4.85(1.04)

Tabela 6.5: Média e desvio padrão (entre parênteses) dos valores obtidos pelas métricas avaliadas.

Instâncias	Hipervolume			Cobertura		
	MOGVNS	NSGAII	IBMOLS	MOGVNS	NSGAII	IBMOLS
CON3-0	0.97(0.03)	0.70(0.10)	0.85(0.10)	0.81(0.17)	0.44(0.12)	0.62(0.23)
CON3-1	0.98(0.01)	0.38(0.27)	0.86(0.11)	1.00(0.00)	0.52(0.19)	0.74(0.19)
CON3-2	0.90(0.10)	0.78(0.12)	0.85(0.11)	0.67(0.23)	0.42(0.33)	0.53(0.27)
CON3-3	0.96(0.03)	0.87(0.09)	0.89(0.02)	0.87(0.12)	0.43(0.17)	0.46(0.20)
CON3-4	0.98(0.04)	0.76(0.04)	0.89(0.09)	0.98(0.05)	0.50(0.26)	0.87(0.11)
CON3-5	0.89(0.08)	0.83(0.03)	0.85(0.07)	0.95(0.10)	0.72(0.26)	0.82(0.27)
CON3-6	0.99(0.00)	0.66(0.14)	0.96(0.10)	1.00(0.00)	0.58(0.23)	0.98(0.09)
CON3-7	1.00(0.00)	0.97(0.08)	1.00(0.00)	1.00(0.00)	0.87(0.35)	1.00(0.00)
CON3-8	0.98(0.01)	0.88(0.06)	0.92(0.05)	0.91(0.08)	0.48(0.25)	0.59(0.14)
CON3-9	0.96(0.17)	0.96(0.17)	1.00(0.00)	0.93(0.26)	0.93(0.26)	1.00(0.00)
CON8-0	0.93(0.04)	0.69(0.15)	0.83(0.12)	0.77(0.18)	0.19(0.13)	0.35(0.16)
CON8-1	0.97(0.02)	0.83(0.15)	0.94(0.06)	0.79(0.22)	0.25(0.17)	0.46(0.30)
CON8-2	0.96(0.02)	0.88(0.08)	0.93(0.03)	0.73(0.17)	0.36(0.21)	0.34(0.19)
CON8-3	0.96(0.02)	0.75(0.06)	0.86(0.04)	0.70(0.22)	0.29(0.18)	0.33(0.19)
CON8-4	0.94(0.02)	0.85(0.05)	0.91(0.01)	0.95(0.10)	0.33(0.30)	0.59(0.30)
CON8-5	0.99(0.01)	0.78(0.10)	0.89(0.11)	0.89(0.13)	0.28(0.19)	0.48(0.19)
CON8-6	0.83(0.32)	0.81(0.30)	0.77(0.40)	0.78(0.39)	0.57(0.43)	0.60(0.51)
CON8-7	0.96(0.05)	0.73(0.17)	0.85(0.07)	0.89(0.18)	0.37(0.14)	0.63(0.15)
CON8-8	0.94(0.03)	0.57(0.14)	0.74(0.17)	0.74(0.18)	0.27(0.14)	0.31(0.26)
CON8-9	0.96(0.01)	0.69(0.22)	0.89(0.04)	0.75(0.17)	0.22(0.18)	0.39(0.22)
SCA3-0	0.94(0.02)	0.66(0.23)	0.90(0.05)	0.84(0.13)	0.30(0.12)	0.54(0.17)
SCA3-1	0.86(0.04)	0.84(0.02)	0.86(0.04)	0.90(0.12)	0.50(0.40)	0.93(0.14)
SCA3-2	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)
SCA3-3	0.95(0.05)	0.92(0.04)	0.94(0.04)	0.83(0.23)	0.70(0.32)	0.80(0.30)
SCA3-4	0.88(0.09)	0.51(0.16)	0.67(0.12)	0.81(0.24)	0.33(0.18)	0.52(0.20)
SCA3-5	0.90(0.04)	0.55(0.21)	0.77(0.11)	0.79(0.16)	0.37(0.19)	0.60(0.17)
SCA3-6	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)	1.00(0.00)
SCA3-7	0.96(0.02)	0.88(0.14)	0.89(0.04)	0.85(0.08)	0.45(0.20)	0.50(0.18)
SCA3-8	0.91(0.07)	0.75(0.26)	0.85(0.19)	0.83(0.21)	0.74(0.29)	0.80(0.24)
SCA3-9	0.89(0.09)	0.79(0.08)	0.83(0.08)	0.85(0.16)	0.44(0.15)	0.59(0.19)
SCA8-0	0.89(0.11)	0.71(0.18)	0.81(0.13)	0.75(0.30)	0.28(0.19)	0.42(0.25)
SCA8-1	0.96(0.06)	0.70(0.14)	0.84(0.07)	0.81(0.25)	0.21(0.19)	0.27(0.22)
SCA8-2	0.92(0.04)	0.82(0.05)	0.89(0.03)	0.67(0.18)	0.27(0.19)	0.47(0.21)
SCA8-3	0.99(0.01)	0.93(0.09)	0.95(0.07)	0.83(0.22)	0.46(0.31)	0.52(0.29)
SCA8-4	0.94(0.04)	0.68(0.10)	0.78(0.08)	0.85(0.16)	0.30(0.17)	0.39(0.22)
SCA8-5	0.97(0.02)	0.83(0.09)	0.88(0.06)	0.84(0.18)	0.33(0.23)	0.31(0.20)
SCA8-6	0.96(0.03)	0.72(0.23)	0.86(0.14)	0.85(0.16)	0.27(0.15)	0.39(0.16)
SCA8-7	0.90(0.04)	0.78(0.07)	0.79(0.05)	0.64(0.23)	0.43(0.24)	0.32(0.20)
SCA8-8	1.00(0.00)	0.98(0.03)	1.00(0.01)	1.00(0.00)	0.83(0.24)	0.97(0.13)
SCA8-9	0.88(0.04)	0.63(0.11)	0.75(0.08)	0.70(0.20)	0.23(0.20)	0.43(0.15)
Média	0.94(0.05)	0.78(0.12)	0.87(0.08)	0.84(0.15)	0.46(0.21)	0.60(0.19)

Tabela 6.6: Comparação com solução ótima do VRPSPD (Coletas 100% realizadas)

Instância	**MOGVNS**	**Ótimo***	**gap (%)**
CON3-0	616.52	616.52	0.00
CON3-1	554.47	554.47	0.00
CON3-2	519.11	518.01	0.21
CON3-3	591.19	591.19	0.00
CON3-4	588.70	588.70	0.00
CON3-5	563.70	563.70	0.00
CON3-6	499.05	499.05	0.00
CON3-7	576.48	576.48	0.00
CON3-8	523.05	523.05	0.00
CON3-9	578.25	578.25	0.00
CON8-0	857.17	857.17	0.00
CON8-1	740.85	740.85	0.00
CON8-2	712.89	712.89	0.00
CON8-3	811.07	811.07	0.00
CON8-4	772.25	772.25	0.00
CON8-5	754.88	754.88	0.00
CON8-6	678.92	678.92	0.00
CON8-7	811.96	811.96	0.00
CON8-8	767.53	767.53	0.00
CON8-9	809.00	809.00	0.00
SCA3-0	636.06	635.62	0.07
SCA3-1	697.84	697.84	0.00
SCA3-2	659.34	659.34	0.00
SCA3-3	680.04	680.04	0.00
SCA3-4	690.50	690.50	0.00
SCA3-5	659.91	659.91	0.00
SCA3-6	651.09	651.09	0.00
SCA3-7	659.17	659.17	0.00
SCA3-8	719.48	719.48	0.00
SCA3-9	681.00	681.00	0.00
SCA8-0	961.50	961.50	0.00
SCA8-1	1049.65	1049.65	0.00
SCA8-2	1045.07	1039.64	0.52
SCA8-3	983.34	983.34	0.00
SCA8-4	1065.49	1065.49	0.00
SCA8-5	1027.08	1027.08	0.00
SCA8-6	971.82	971.82	0.00
SCA8-7	1052.04	1051.28	0.07
SCA8-8	1071.18	1071.18	0.00
SCA8-9	1060.50	1060.50	0.00

*Resultados reportados por Subramanian et al. [2011].

algoritmos foram ajustados de forma que apresentem o mesmo tempo de execução para este grupo de instâncias. Para as demais métricas, o algoritmo MOGVNS apresentou diferenças significativas comparado ao NSGAII e IBMOLS. Por outro lado, o NSGAII e IBMOLS não apresentaram diferenças entre si para as métricas avaliadas. O resultado do teste de hipótese aponta para a rejeição da hipótese nula para todas as métricas avaliadas. A tabela 6.7 apresenta o p-valor obtido em cada métrica.

Tabela 6.7: P-valor obtido após teste de hipótese

Hipervolume	**Cobertura**	**Cardinalidade**	**Tempo**
$< 1.30e^{-55}$	$<2.03e^{-65}$	$<2.47e^{-66}$	0.93

As figuras 6.5, 6.6, 6.7 e 6.8 apresentam as médias estimadas dos algoritmos MOGVNS, NSGAII e IBMOLS para as métricas hipervolume, cobertura, cardinalidade e tempo.

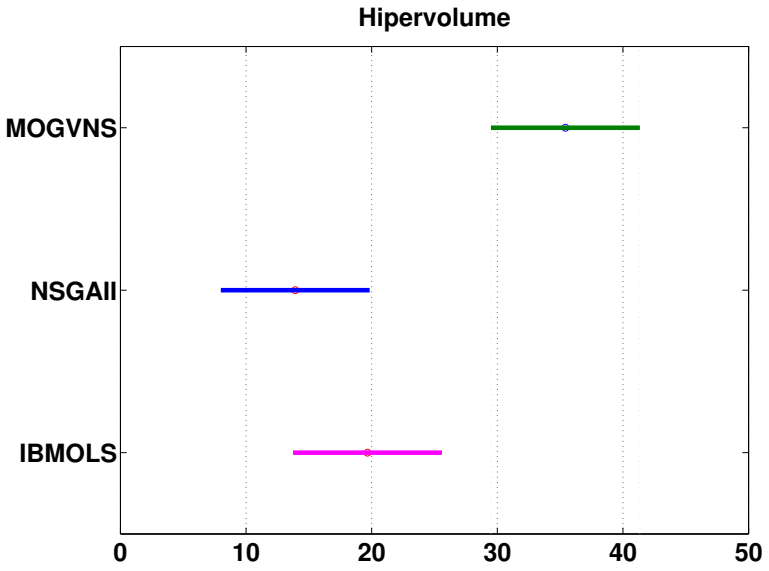


Figura 6.5: Hipervolume: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Salhi & Nagy.

Os gráficos apresentados corroboram a superioridade do MOGVNS, mostrando diferenças significativa entre ele e os demais algoritmos, com maiores valores para as médias estimadas envolvendo as métricas hipervolume, cobertura e cardinalidade. Em relação ao tempo de execução, os valores não apresentam diferença significativas.

A tabela 6.8 apresenta a média e desvio padrão das métricas: hipervolume, cobertura, cardinalidade e tempo, dos valores obtidos nas 15 replicações para cada instância e algoritmo avaliado. Na tabela também é possível verificar a superioridade do MOGVNS que apresenta média geral e desvio padrão com melhores valores em todas as métricas. Quanto ao tempo

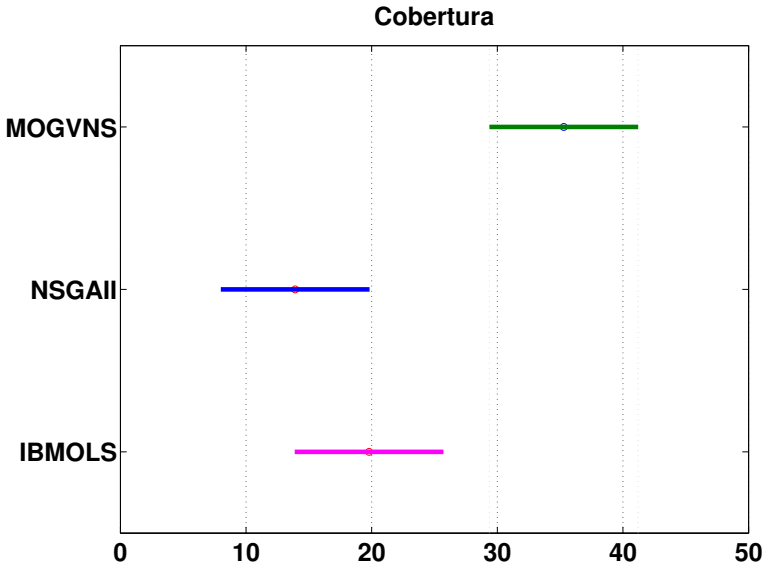


Figura 6.6: Cobertura: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Salhi & Nagy.

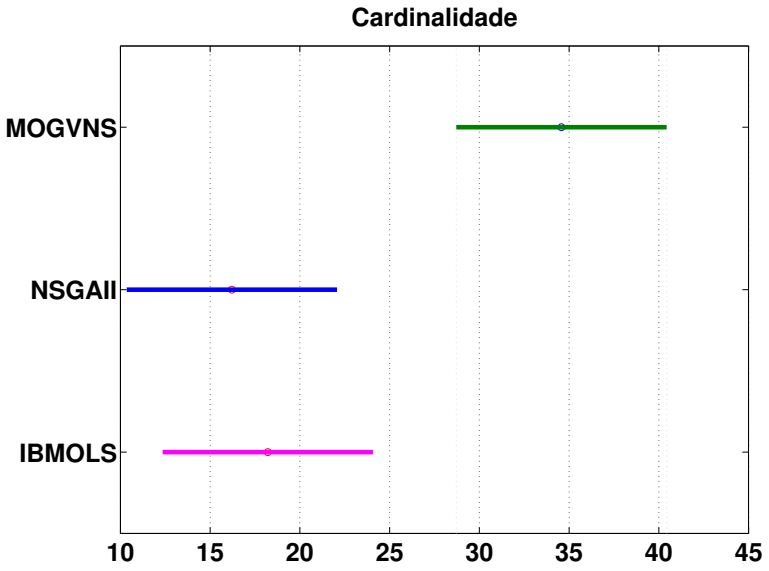


Figura 6.7: Cardinalidade: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Salhi & Nagy.

de execução, o NSGAII apresenta um valor menor na média geral, mas esta diferença não é significativa, como mostrado na análise estatística [Assis et al., 2013a,b].

Para atestar a qualidade dos algoritmos as fronteiras obtidas nas 15 replicações foram comparadas com um limite inferior ou solução ótima definida com o uso do resolvedor GUROBI [Gurobi Optimization, 2013]. O limite inferior foi obtido a partir da fixação do tempo de execução em 5 dias para cada instância de Salhi & Nagy. O tempo de execução para en-

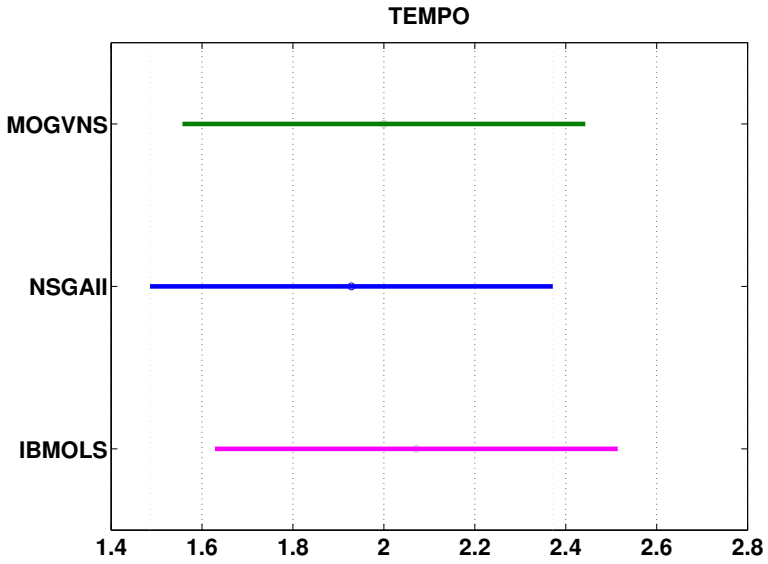


Figura 6.8: Tempo: estimativa das médias do desempenho dos algoritmos MOGVNS, NSGAII e IBMOLS aplicados às instâncias de Salhi & Nagy.

contrar a fronteira Pareto ótima das instâncias com menor número de consumidores CMT1X e CMT1Y foi de aproximadamente 2 dias para cada uma.

Para definir o limite inferior ou encontrar a fronteira Pareto ótima, o problema foi solucionado aplicando o método tradicional de resolução de problemas multiobjetivo ϵ-Restrito 2.2. Na formulação dos subproblemas, a redução do custo de transporte foi mantido na função objetivo e a minimização das demandas de coletas não atendidas foi inserida no conjunto de restrições, limitada pelo valor ϵ. Este parâmetro é inicializado com o valor 0. Este primeiro subproblema equivale ao problema de roteamento de veículos com coleta e entrega simultânea. A cada problema solucionado, o valor de ϵ é incrementando de acordo com um valor Δ pré-definido. Para encontrar o limite inferior o valor de Δ é fixado como sendo o somatório das demandas de coleta dividido por 5, uma vez que deseja-se definir até 5 soluções na fronteira. Por outro lado, para garantir a otimalidade das instâncias menores (CMT1X e CMT1Y), o valor de ϵ foi incrementando com o valor da menor demanda de coleta. Quando o valor de ϵ é igual ao somatório das demandas de coleta, então o subproblema equivale ao problema de roteamento de veículos capacitado.

Para tornar a busca pela solução ótima mais eficiente, inicialmente foi encontrada a solução ótima para o problema de roteamento de veículos capacitado, ou seja, solucionado o problema no qual ϵ é igual ao somatório das demandas de coleta. A solução ótima do CVRP indica o menor custo de transporte possível de ser obtido para qualquer valor de ϵ. Assim, o valor do custo de transporte da solução encontrada para o CVRP é utilizado para limitar a busca do problema multiobjetivo. Partindo do valor 0, a medida que o valor de ϵ é incrementado, a busca por outros pontos da fronteira é interrompida quando a solução de

Tabela 6.8: Média e desvio padrão (entre parênteses) dos valores obtidos pelas métricas avaliadas.

Instâncias	Tempo ($10^3 s$)			Cardinalidade		
(CMT)	MOGVNS	NSGAII	IBMOLS	MOGVNS	NSGAII	IBMOLS
1X	0.66(0.04)	1.74(0.39)	1.01(0.19)	3.47(0.64)	2.07(0.26)	2.80(0.56)
1Y	0.93(0.04)	2.41(1.06)	1.33(0.17)	8.93(1.28)	5.93(1.39)	8.00(1.65)
2X	1.92(0.13)	6.31(2.79)	3.23(0.73)	7.47(2.53)	3.33(1.29)	4.33(1.63)
2Y	3.84(0.19)	6.13(2.46)	6.06(0.97)	32.00(4.77)	26.47(6.95)	25.80(2.96)
3X	205.86(6.22)	38.04(7.35)	180.25(3.55)	6.20(1.15)	3.93(1.16)	5.00(1.81)
3Y	217.46(1.81)	41.10(8.97)	185.37(2.25)	19.20(1.90)	15.93(2.91)	15.67(2.66)
4X	125.69(17.89)	139.16(26.11)	959.43(26.78)	10.80(3.97)	5.53(1.30)	5.47(2.03)
4Y	143.83(20.48)	135.48(22.68)	991.37(10.45)	43.07(4.38)	34.20(8.00)	28.93(5.62)
5X	433.62(29.96)	479.66(160.27)	378.03(69.73)	14.40(4.42)	6.00(3.51)	6.67(1.45)
5Y	490.26(63.05)	437.95(114.33)	590.06(100.92)	50.40(4.36)	46.13(7.37)	30.53(6.02)
11X	79.65(3.78)	61.88(14.44)	62.05(12.15)	43.13(4.64)	31.27(5.80)	33.87(6.27)
11Y	55.56(9.10)	60.15(15.29)	41.91(6.84)	28.53(4.44)	16.27(4.67)	18.13(5.67)
12X	161.74(5.39)	26.26(6.32)	130.60(3.86)	16.60(2.67)	11.67(2.97)	11.33(2.44)
12Y	165.25(2.95)	23.89(5.12)	133.45(1.75)	30.87(3.56)	16.80(3.00)	19.47(4.42)
Média	149.02(11.50)	104.30(27.69)	261.72(17.17)	22.50(3.19)	16.11(3.61)	15.43(3.23)

Instâncias	Hipervolume			Cobertura		
(CMT)	MOGVNS	NSGAII	IBMOLS	MOGVNS	NSGAII	IBMOLS
1X	0.98(0.06)	0.86(0.18)	0.95(0.04)	0.88(0.20)	0.44(0.28)	0.66(0.26)
1Y	0.71(0.00)	0.66(0.02)	0.68(0.04)	0.67(0.16)	0.16(0.12)	0.50(0.15)
2X	0.39(0.04)	0.33(0.04)	0.33(0.03)	0.63(0.33)	0.33(0.29)	0.25(0.21)
2Y	0.91(0.03)	0.78(0.06)	0.85(0.05)	0.72(0.15)	0.24(0.14)	0.37(0.16)
3X	0.96(0.03)	0.86(0.07)	0.93(0.02)	0.62(0.21)	0.19(0.16)	0.51(0.22)
3Y	0.98(0.01)	0.86(0.06)	0.95(0.01)	0.69(0.15)	0.18(0.09)	0.45(0.10)
4X	0.85(0.10)	0.71(0.10)	0.72(0.08)	0.61(0.25)	0.18(0.13)	0.25(0.14)
4Y	0.96(0.02)	0.86(0.08)	0.91(0.03)	0.69(0.19)	0.23(0.15)	0.31(0.15)
5X	0.81(0.08)	0.64(0.11)	0.61(0.08)	0.66(0.21)	0.36(0.20)	0.21(0.21)
5Y	0.90(0.03)	0.77(0.03)	0.82(0.04)	0.63(0.19)	0.25(0.17)	0.34(0.18)
11X	0.93(0.02)	0.80(0.05)	0.85(0.04)	0.81(0.09)	0.21(0.14)	0.30(0.11)
11Y	0.78(0.09)	0.72(0.07)	0.71(0.05)	0.64(0.17)	0.27(0.18)	0.23(0.09)
12X	0.96(0.01)	0.85(0.08)	0.91(0.03)	0.74(0.16)	0.27(0.12)	0.24(0.13)
12Y	0.96(0.02)	0.87(0.05)	0.89(0.05)	0.68(0.17)	0.17(0.13)	0.26(0.10)
Média	0.86(0.04)	0.76(0.07)	0.79(0.04)	0.69(0.19)	0.25(0.17)	0.35(0.16)

um subproblema retornar o mesmo custo de transporte da solução ótima do CVRP obtida previamente. As demais solução dos subproblemas que seriam gerados posteriormente são equivalentes ou dominadas por esta, pois apresentarão o mesmo custo de transporte, mas um número igual ou maior de coletas não atendidas.

As figuras 6.9 a 6.15 mostram a fronteira geral obtidas por cada algoritmo nas 15 replica-

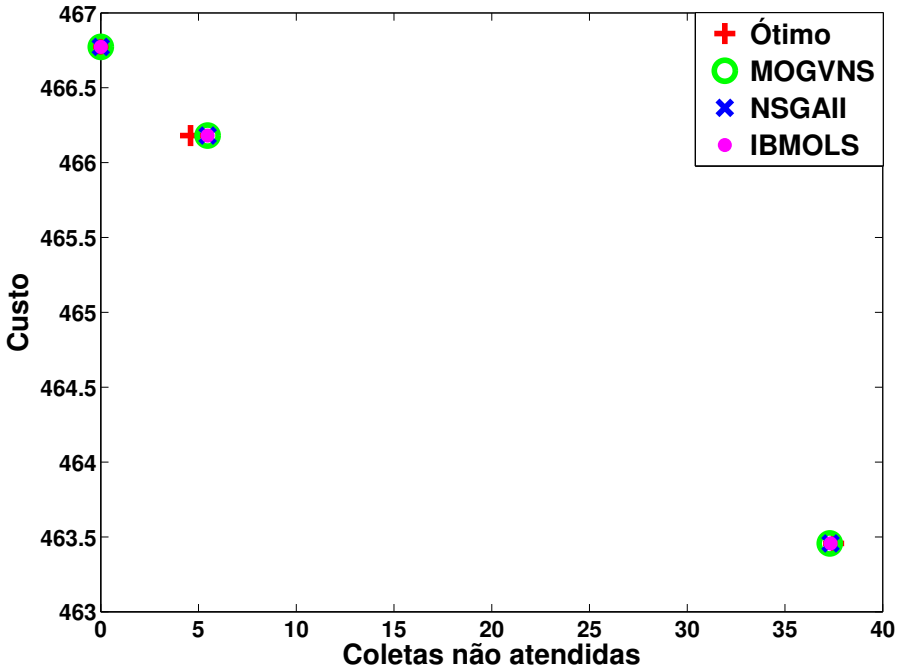
ções e a fronteira Pareto ótima ou limite inferior obtido com o resolvedor GUROBI. A solução ótima foi encontrada para duas instâncias de Salhi & Nagy, CMT1X e CMT1Y, que apresentam um menor número de consumidores (50). Observando a figura 6.9, a fronteira retornada pelos algoritmos implementados, em geral alcançam quase 90% dos pontos da fronteira Pareto ótima, sendo que, nos demais pontos, os algoritmos chegam muito próximos da solução ótima. Isto demostra que os algoritmos implementados apresentam bons resultados para instâncias de pequeno porte.

A medida que o tamanho do problema aumenta, a resolução do problema com método exatos se torna inviável. Para as demais instâncias de Salhi & Nagy, foi possível definir apenas um limite inferior. Para a maior parte das instâncias, o *gap* entre o limite inferior e superior apresentado pelo GUROBI foi bem elevado, ficando entre 10% e 20%. Ressaltando que o limite inferior foi obtido após 5 dias de execução utilizando todos os 48 núcleos do computador (4 processadores x 6 cores x 2 threads). Este dados comprovam a complexidade do problema tratado e a necessidade de estudos de métodos heurísticos.

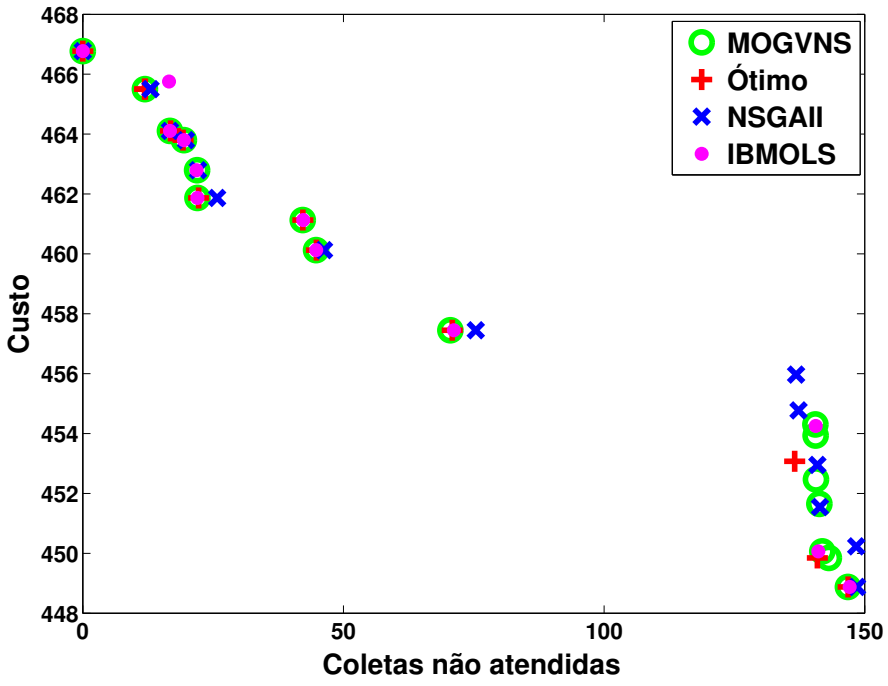
Em todos os gráficos apresentados, a fronteira geral dos algoritmos encontraram-se bem próximas uma das outras, mas pode-se observar que, para as instâncias CMT2X, CMT2Y, CMT4X, CMT5X, CMT5Y e CMT11X, a fronteira geral do MOGVNS é a que mais se aproxima do limite inferior. Além disso, os gráficos das instâncias CMT2X, CMT4X e CMT5X também comprovam a qualidade deste algoritmo considerando a cardinalidade da fronteira. Nos gráficos das demais instâncias, a fronteira geral dos três algoritmos tem comportamento muito similar.

Observando os gráficos apresentados, também é possível comprovar a importância de uma abordagem multiobjetivo para o problema tratado. A tabela 6.1 mostra que a soma das demandas de coleta das instâncias "X" é menor que a das instâncias "Y". Assim, pode-se verificar nos gráficos que, em geral, as instâncias "Y" retornam um número muito maior de pontos na fronteira. No gráfico da instância CMT1X, por exemplo, a diferença no custo entre os dois extremos da fronteira é menor que 4un, enquanto esta diferença entre os extremos da fronteira Pareto ótima da instância CMT1Y é de aproximadamente 20un. Lembrando que o custo de transporte entre os consumidores nestas duas instâncias é o mesmo, diferindo apenas nas demandas de coleta e entrega. Para o tomador de decisão, estas informações são muito relevantes. Visualizando todas as possíveis soluções que se pode adotar, este poderá decidir qual delas melhor se adequa a realidade. Considerando a instância CMT1X, por exemplo, sabendo que a diferença entre realizar todas as coletas e não realizar as coletas é mínima, ele poderá optar pela primeira opção.

Devido ao não conhecimento da fronteira Pareto-Ótima para todas as intâncias, foi feita a comparação dos melhores resultados obtidos pelo MOGVNS com abordagens mono-objetivo. Foram utilizados apenas os dados do MOGVNS uma vez que, nas análises anteriores, verificou-

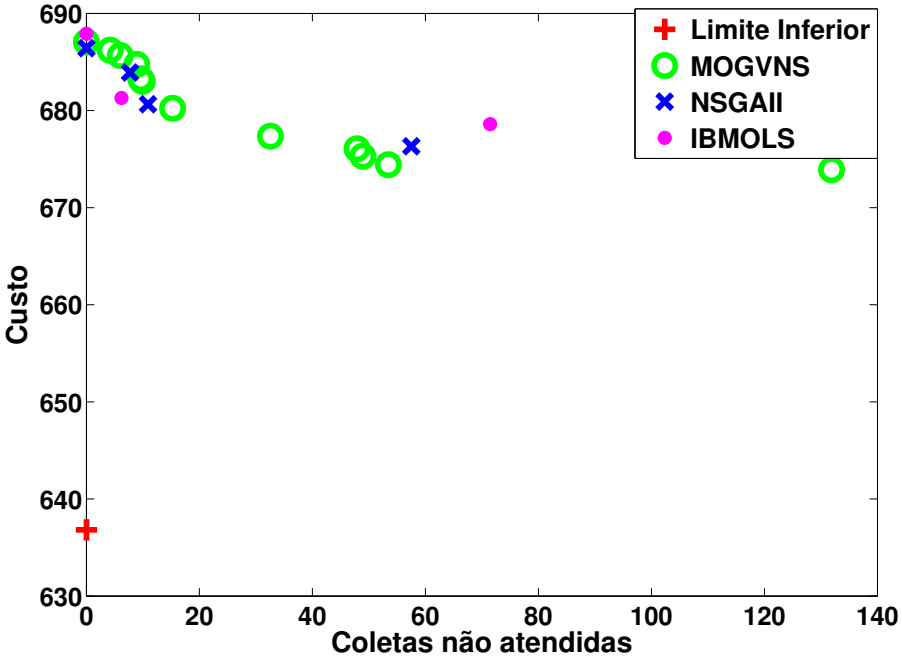


(a) CMT1X

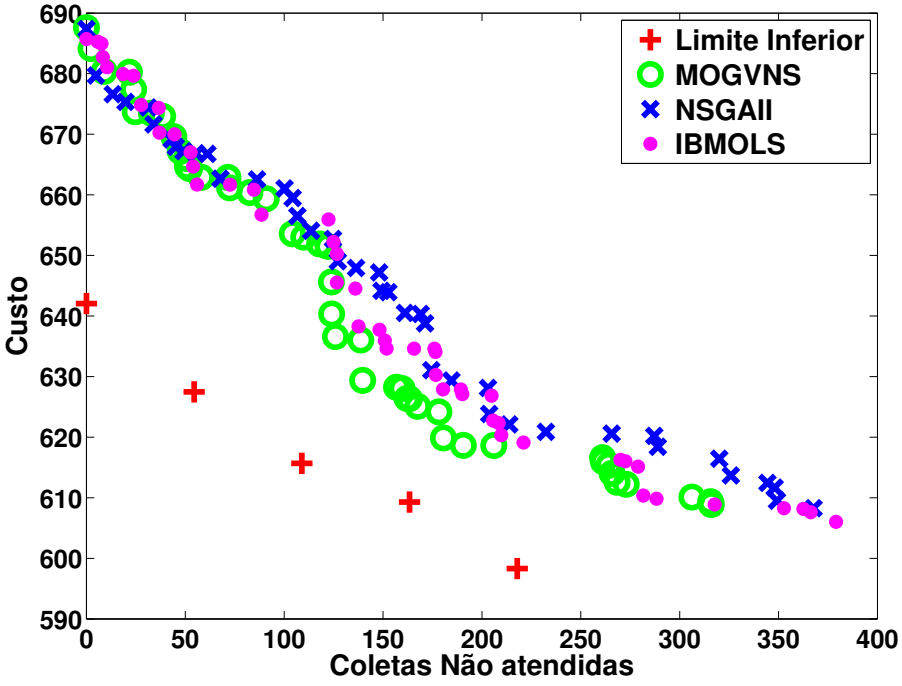


(b) CMT1Y

Figura 6.9: Instâncias CMT1X e CMT1Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

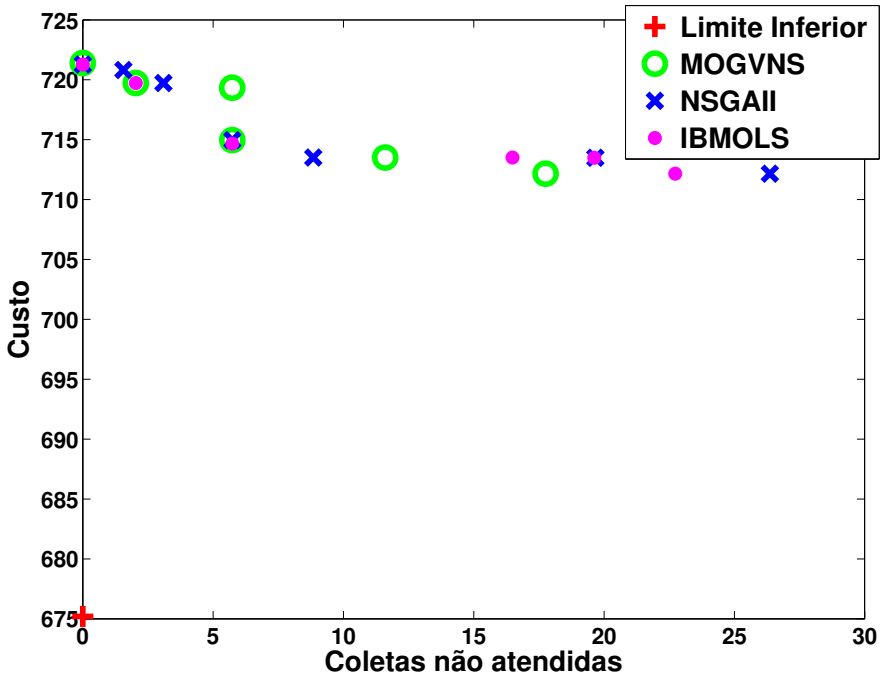


(a) CMT2X

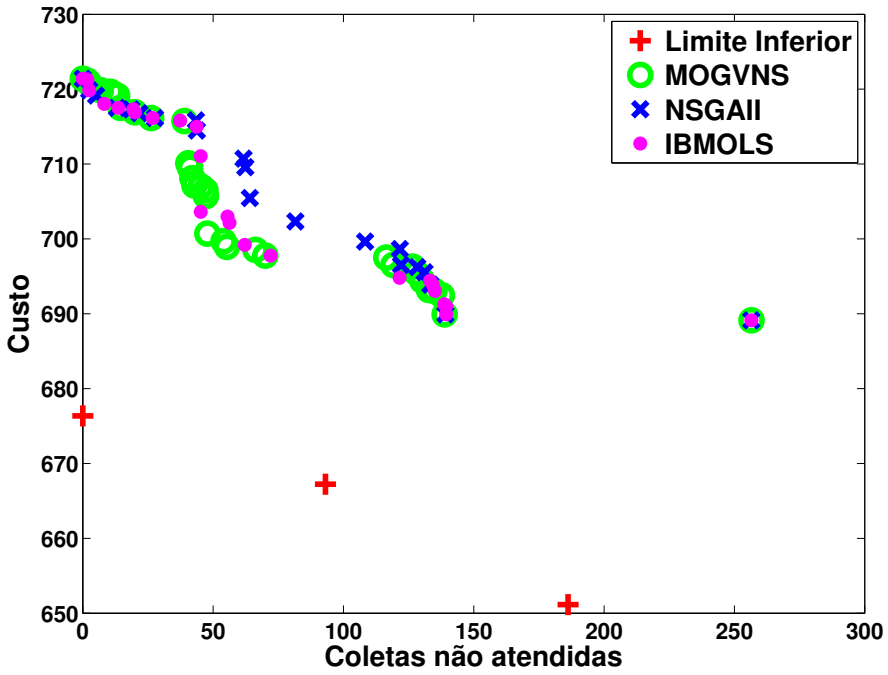


(b) CMT2Y

Figura 6.10: Instâncias CMT2X e CMT2Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

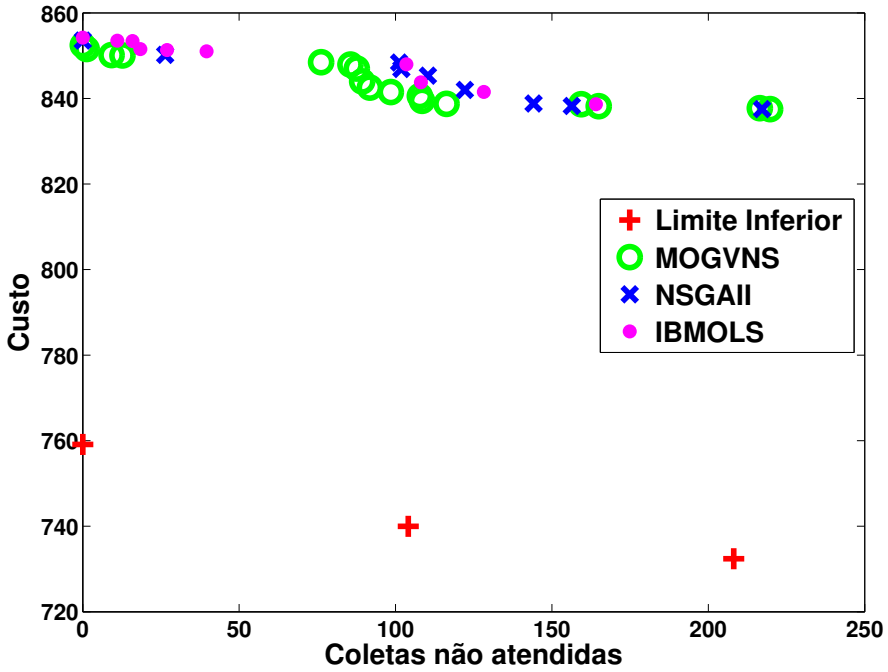


(a) CMT3X

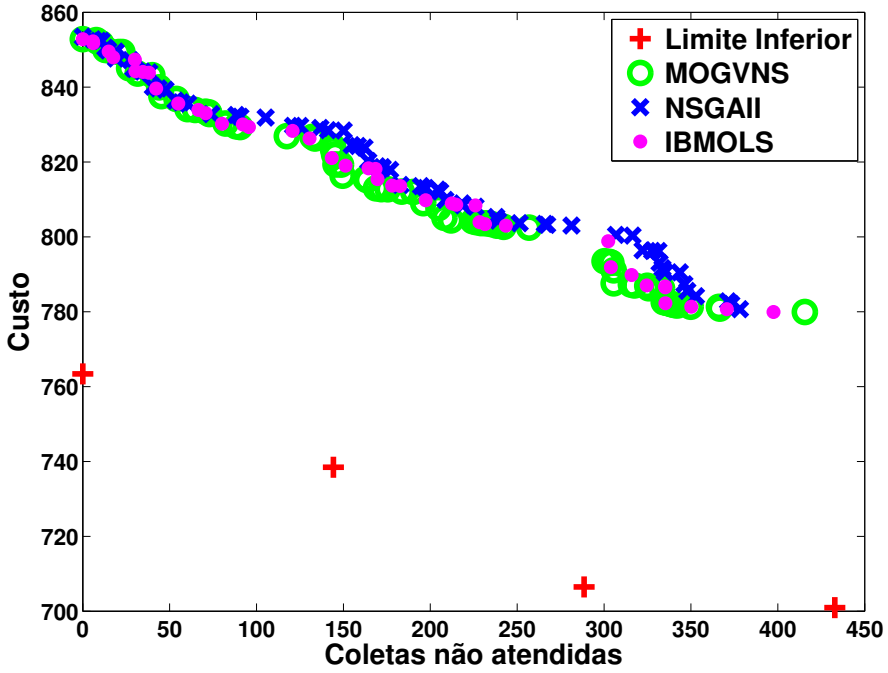


(b) CMT3Y

Figura 6.11: Instâncias CMT3X e CMT3Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

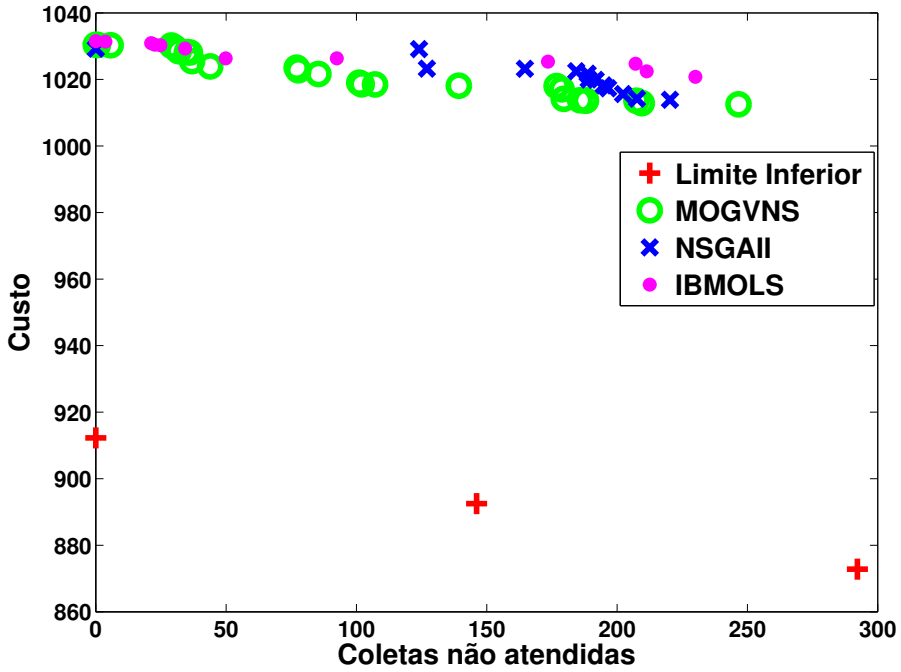


(a) CMT4X

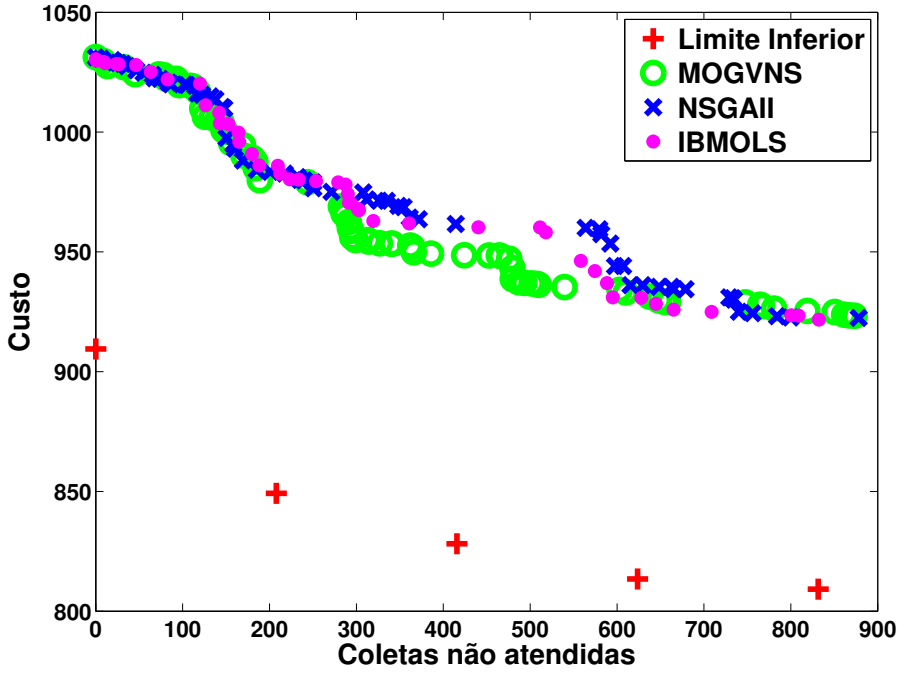


(b) CMT4Y

Figura 6.12: Instâncias CMT4X e CMT4Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

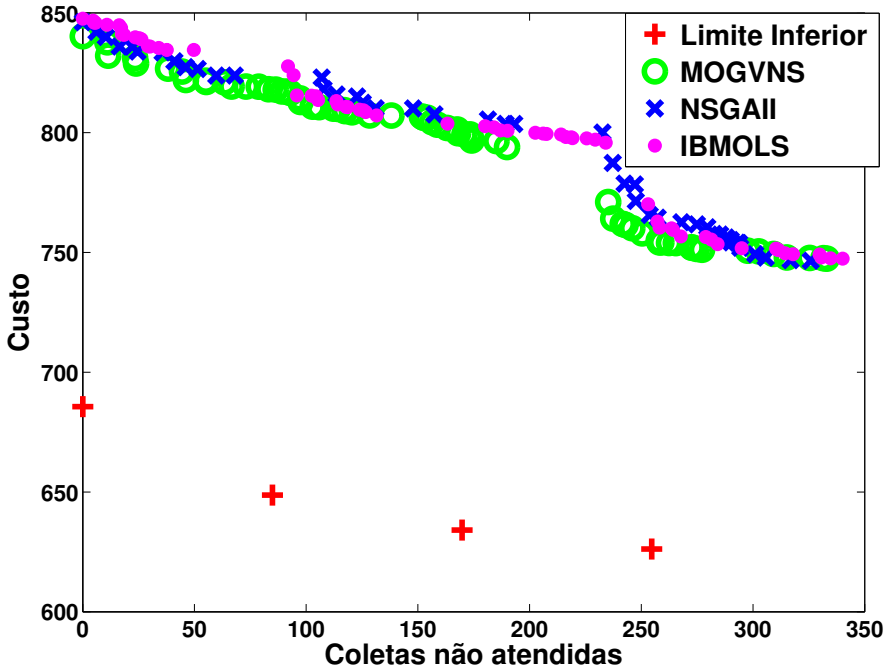


(a) CMT5X

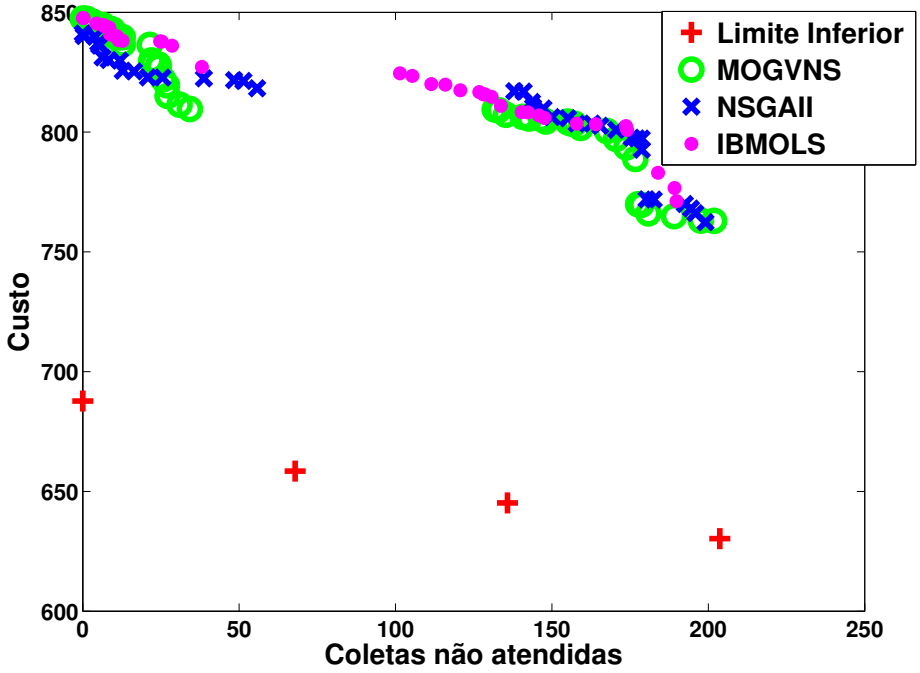


(b) CMT5Y

Figura 6.13: Instâncias CMT5X e CMT5Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

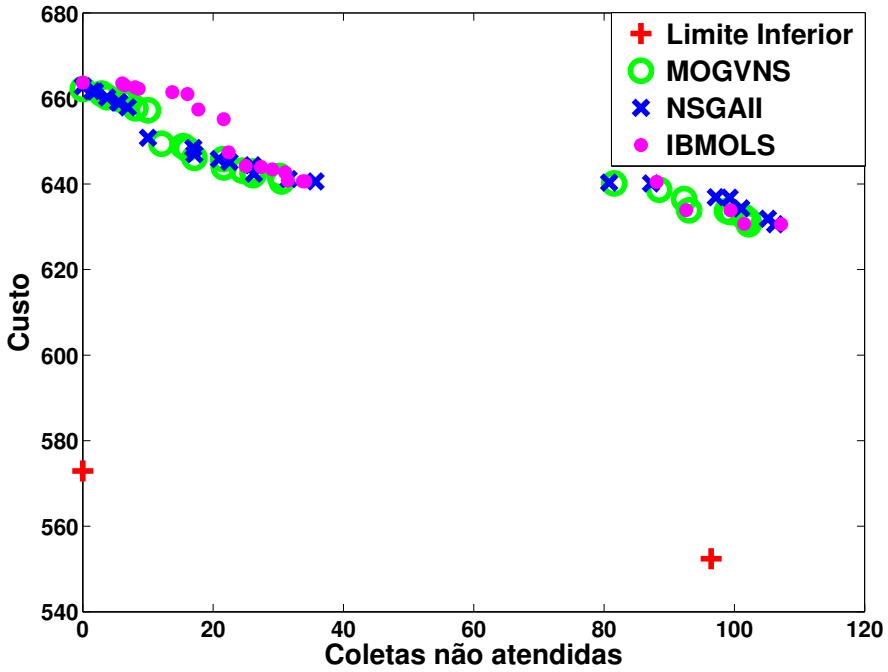


(a) CMT11X

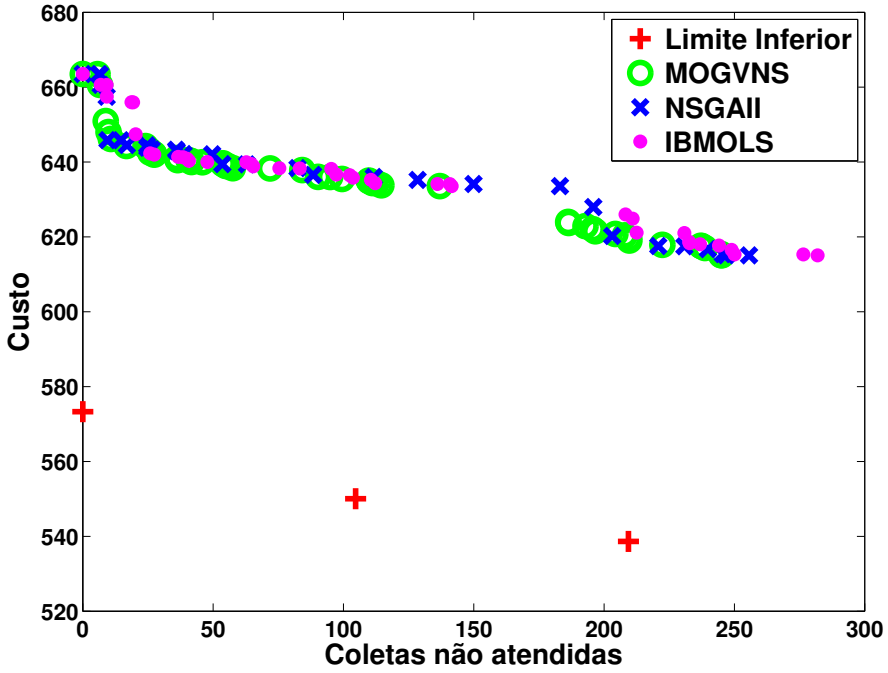


(b) CMT11Y

Figura 6.14: Instâncias CMT11X e CMT11Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.



(a) CMT12X



(b) CMT12Y

Figura 6.15: Instâncias CMT12X e CMT12Y: Limite Inferior, MOGVNS, NSGAII e IBMOLS.

Tabela 6.9: Comparação com melhores resultados para o VRPSPD (Coletas 100% realizadas)

Instância	**MOGVNS**	**Melhor conhecido***	**gap (%)**
CMT1X	466.77	466.77[a]	0.0
CMT1Y	466.77	466.77[a]	0.0
CMT2X	685.84	668.77	2.5
CMT2Y	684.89	663.25	3.3
CMT3X	721.40	721.27[a]	0.0
CMT3Y	721.40	721.27[a]	0.0
CMT4X	852.46	852.46	0.0
CMT4Y	852.83	852.35	0.0
CMT5X	1030.41	1029.25	0.1
CMT5Y	1031.34	1029.25	0.2
CMT11X	840.26	833.92	0.8
CMT11Y	847.21	830.39	2.0
CMT12X	662.22	644.70	2.7
CMT12Y	663.50	659.52	0.6

*Resultados reportados por Subramanian et al. [2010].

[a]Solução ótima [Subramanian et al., 2011].

se que este algoritmo apresentou melhores resultados que os demais. A tabela 6.9 apresenta os melhores resultados conhecidos para o problema de roteamento de veículos com coleta e entrega simultânea (VRPSPD) [Subramanian et al., 2010] e compara com as soluções obtidas pelo MOGVNS. As soluções consideradas na comparação são aquelas que se localizam no extremo da fronteira no qual todas as coletas também são atendidas. A última coluna da tabela apresenta a diferença percentual entre os resultados obtidos pelo MOGVNS e os melhores resultados conhecidos na literatura.

Como pode ser observado, o MOGVNS consegue encontrar boas soluções quando todas as demandas de coleta são consideradas, mostrando resultados bem próximos aos resultados da literatura. Em 71% das instâncias o erro relativo (*gap*) foi aproximadamente zero.

6.6 Conclusão

Este capítulo apresentou os resultados dos principais algoritmos propostos neste trabalho para o problema de roteamento de veículos multiobjetivo com coleta opcional. Os experimentos realizados consistem na execução dos algoritmos: busca geral em vizinhança variável multiobjetivo (MOGVNS), algoritmo genético de ordenação não-dominante II (NSGA-II) e buscal local multiobjetivo baseada em indicador (IBMOLS).

Os algoritmos foram submetidos a dois conjuntos de instâncias encontradas na literatura propostas para representar o problema de roteamento de veículos com coleta e entrega simultânea. O problema de roteamento de veículos multiobjetivo com coleta opcional equivale ao VRPSPD quando todas suas coletas são atendidas, assim, as instâncias puderam ser utilizadas sem nenhuma alteração nos seus dados originais. O primeiro conjunto de instâncias avaliado é composto por 40 problemas teste contendo 50 consumidores. O outro é composto por 14 problemas teste no qual o número de consumidores varia entre 50 a 200 consumidores. A análise em diferentes grupos de instâncias permitiu avaliar a qualidade dos algoritmos tanto para instâncias de pequena magnitude quanto para instâncias de tamanhos variados.

Após uma análise empírica dos resultados, este capítulo aponta ajustes de parâmetros de forma que, para instâncias com diferentes número de consumidores, os algoritmos apresentem o mesmo tempo de execução. Dessa forma, este quesito não se tornou um item que possa interferir na qualidade dos algoritmos.

Os resultados obtidos pelos testes de hipótese mostram a existência de uma diferença estatisticamente significativa no desempenho entre os algoritmos MOGVNS, NSGA-II e IBMOLS, quando aplicados na resolução do problema de roteamento de veículos multiobjetivo com coleta opcional. Os dados apresentados para ambos grupos de instâncias (Dethloff e Salhi & Nagy) demostram a superioridade do MOGVNS frente aos demais algoritmos implementados. Tanto para o grupo de instância mais homogênea, contendo 50 consumidores, quanto para o grupo de instâncias com número de consumidores variando entre 50 e 200, o algoritmo apresentou melhor desempenho nas métricas hipervolume, cobertura e cardinalidade. Apesar de apresentar tempo de execução maior para instâncias de pequena magnitude, a medida que o número de consumidores aumenta, esta diferença entre os algoritmos torna-se estatisticamente não significativa.

A fim de avaliar a qualidade das soluções obtidas pelo MOGVNS com resultados da literatura, este capítulo apresentou uma comparação deste com os melhores resultados conhecidos para o problema de roteamento de veículos com coleta e entrega simultânea. Para fazer esta comparação, apenas um dos pontos extremos da fronteira, no qual todas as demandas de coleta são satisfeitas, é utilizado nesta comparação. Os resultados apresentados pelo MOGVNS se aproximam dos melhores da literatura com uma diferença média de 0,8%. Para o conjunto de instância de 50 consumidores, a solução ótima foi encontrada em 90% das 40 instâncias

analisadas. Dessa forma, o MOGVNS é um método promissor para resolução de problemas de roteamento de veículo com múltiplos objetivos.

A fronteira geral obtida pelos algoritmos nas 15 replicações foram comparadas com um limite inferior ou solução ótima. O método tradicional de resolução de problemas multiobjetivo ϵ-Restrito foi utilizado para definir as fronteiras que representam o limite inferior dos problemas-teste e a fronteira Pareto ótima. Em cada subproblema a função objetivo consiste na minimização do custo de transporte e uma nova restrição foi inserida que limita a quantidade de coletas não atendidas por um parâmetro ϵ.

A solução ótima foi encontrada para as duas menores instâncias propostas por Salhi & Nagy, no qual todos os pontos da fronteira foram definidos. Comparando-a com as fronteiras retornadas pelos algoritmos propostos, estes apresentaram bons resultados, alcançando quase a totalidade dos pontos da fronteira Pareto ótima.

Devido a complexidade do problema e a dificuldade de se obter solução ótima, este trabalho apresenta um limite inferior para problemas com maior número de consumidores. Apesar do tempo de execução fixado em 5 dias, os limites inferiores retornados pelo resolvedor GUROBI indicam um *gap* elevado, corroborando necessidade de se aplicar métodos heurísticos para solucionar MOVRPOP em um tempo viável. Ao comparar o limite inferior com a fronteira geral obtida pelos algoritmos, as soluções do MOGVNS são as que mais se aproximaram do limite inferior e foi o algoritmo que apresentou melhor cardinalidade.

Os resultados apresentados também comprovam a importância de uma abordagem multiobjetivo para o problema tratado. Para algumas instâncias nas quais a soma das demandas de coleta é menor que a soma das demandas de entrega, a diferença no custo de transporte entre os extremos da fronteira é muito pequena. Em outras palavras, a diferença entre efetuar todas as coletas e não efetuá-las é quase insignificante para alguns casos. De posse destas informações, o tomador de decisões terá maior clareza da situação e poderá adotar aquela solução que melhor se adequa a realidade.

Em geral, os resultados obtidos mostraram que existem diferenças significativa entre os algoritmos propostos, destacando o algoritmo MOGVNS. Devido a sua fácil adaptação para outros problemas, este se mostra promissor para resolução de diferentes problemas de otimização combinatória multiobjetivo. Os limites inferiores apresentados comprovaram que o problema tratado apresenta uma complexidade computacional elevada, sendo necessário a aplicação de métodos heurísticos para solucionar o problema em um menor tempo de execução. Por fim, os gráficos das fronteiras dos algoritmos proporcionam uma visão geral das diferentes soluções que podem ser adotadas. De posse dessas informações, o tomador de decisão terá maior segurança quando optar uma das soluções da fronteira. Isso corrobora a hipótese levantada nesta tese da necessidade de uma maior exploração da abordagem multiobjetivo para problemas de roteamento de veículos com serviço de coleta e entrega.

Capítulo 7

Considerações Finais

Este capítulo apresenta as conclusões do trabalho da tese. Inicialmente, ressalta-se a necessidade de uma exploração de abordagem multiobjetivo para problemas de roteamento de veículos com serviços de coleta e entrega. Posteriormente, discorre-se também acerca dos algoritmos implementados para solucionar o problema e os resultados obtidos. Por fim, são apresentadas sugestões de trabalhos futuros.

7.1 Conclusão

Diversos problemas reais de roteamento de veículos com serviços de coleta e entrega possuem uma natureza multiobjetivo, mas, apesar disso, esta abordagem é ainda pouco explorada na literatura. Dessa forma, este trabalho inicia com uma investigação dos diversos problemas de roteamento de veículos com serviços de coleta e entrega e também dos problemas de roteamento de veículos multiobjetivo. Este estudo mostrou que na interseção destas duas classes de problemas existem poucos trabalhos relacionados.

A fim de preencher esta lacuna, este trabalho apresentou uma formulação matemática e a definição de um novo problema de roteamento de veículos multiobjetivo denominado: problema de roteamento de veículos multiobjetivo com coleta opcional. Na formulação proposta, o problema é composto por duas funções objetivo referentes ao custo total de transporte e ao atendimento das demandas de coleta. A abordagem multiobjetivo facilita na tomada de decisões em situações onde a entrega de produtos é obrigatória, mas a coleta é opcional.

Devido a complexidade computacional do problema tratado, este trabalho apresentou um conjunto de soluções heurísticas para abordá-lo, uma vez que os algoritmos exatos não seriam capazes de solucionar problemas de grande magnitude, que são, geralmente, os problemas encontrados no cotidiano das grandes empresas. Assim sendo, este trabalho explorou as especificidades das metaheurísticas baseadas em busca local e em população a fim de gerar algoritmos com características distintas a fim de avaliar quais deles melhor se adapta ao

problema abordado.

Além das metaheurísticas, métodos tradicionais de resolução de problemas de otimização multiobjetivo foram estudados com enfoque ao método ϵ-Restrito. Neste trabalho apenas algumas propriedades deste método foram utilizadas, gerando uma heurística para o problema abordado.

Dessa forma, este trabalho apresentou três novos algoritmos baseados no: MOGVNS, NSGA-II e IBMOLS que exploram as especificidades das metaheurísticas de busca local, de população. Além disso, propôs uma heurística baseada no método ϵ-Restrito utilizada para gerar soluções de partida dos algoritmos mencionados.

Além dos algoritmos implementados, este trabalho apresentou contribuições em todo arcabouço envolvido no processo de resolução do problema. Dentre elas: uma nova estrutura de dados para armazenar um grafo, redução da complexidade computacional do algoritmo que verifica a viabilidade de uma solução devido a restrição da capacidade máxima do veículo e buscas locais específicas para o problema tratado.

Para avaliação da qualidade dos algoritmos implementados, o MOGVNS, NSGA-II e IBMOLS foram submetidos a dois conjuntos de instâncias. Uma dela compostas por problemas com um número fixo de consumidores e outra com o número de consumidores variados. A análise estatística dos resultados indicou diferenças significativa entre os métodos implementados. Os resultados apontaram uma superioridade do MOGVNS frente aos demais algoritmos, considerando as métricas hipervolume, cardinalidade e cobertura. Além da métricas aplicadas, a superioridade do MOGVNS também foi constatada ao comparar as soluções obtidas pelos algoritmos com um limite inferior ou com fronteira Pareto ótima das instâncias com menor número de consumidores.

7.2 Trabalhos Futuros

Uma vez que o MOGVNS se mostrou mais promissor para solucionar o problema tratado, trabalhos futuros poderão aplicá-lo a outros problemas de otimização combinatória multiobjetivo. Além disso, outras métricas poderiam ser aplicadas, como a métrica de diversidade (Δ). Assim, outras características desejáveis de uma solução multiobjetivo poderão ser avaliadas.

Neste trabalho foram empregadas várias estruturas de vizinhança para redução do custo de transporte. Não foi feita nenhuma análise em relação a eficiência destas estruturas na resolução do problema abordado. Algoritmos mais rápidos e de melhor qualidade poderiam ser obtidos com uma análise mais detalhada da eficiência destas estruturas e com o uso de métodos de controle dimensional do problema que buscam reduzir o número de arestas a

serem consideradas durante a execução dos procedimentos de busca local [Carrano, 2007].

Outra questão a ser observada é a relação de dominância aplicada. A relação de dominância Pareto, utilizada neste trabalho, não oferece muita flexibilidade. Por essa razão é importante explorar outras relações como Cone-Dominância, proposta por Batista et al. [2011].

Nos algoritmos implementados, a exploração das soluções é feita de maneira assíncrona, ou seja, a cada iteração um ponto da fronteira é selecionado para que seja feita a busca por outras solução não-dominadas na sua vizinhança. Se esta exploração fosse realizada em paralelo reduziria o tempo computacional, gerando um número maior de soluções não-dominadas, melhor distribuídas e mais próximas às soluções Pareto-ótimas. A necessidade de redução do tempo computacional é uma importante questão levantada neste trabalho.

Todos os procedimentos citados até então referem-se a melhorias nos algoritmos que já foram implementados ou ao planejamento e análise dos resultados. São mudanças que tornam os algoritmos mais eficientes ou a análise dos resultados mais refinada. Contudo, outros problemas podem ser explorados, entre eles o problema de roteamento de veículos multiobjetivo com coleta opcional parcial. Ele se assemelha ao problema abordado ao longo deste trabalho, mas permite que parte da demanda de coleta seja atendida, não ficando restrito apenas ao atendimento total de uma demanda ou o não atendimento desta demanda. Em outras palavras, o estado de uma demanda de coleta poderá variar entre: atendida, não atendida ou parcialmente atendida. Esta variação consegue refletir melhor as necessidades reais das empresas que enfrentam este tipo de problema.

Referências Bibliográficas

Assis, L. (2007). Algoritmos para o problema de roteamento de veículos com coleta e entrega simultâneas. Master's thesis, Instituto de Ciências Exatas, Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais - Brasil.

Assis, L.; Maravilha, A.; Vivas, A.; Campelo, F. e Ramírez, J. (2013a). Multiobjective vehicle routing problem with fixed delivery and optional collections. *Optimization Letters*, 7(7):1419–1431.

Assis, L. P.; Maravilha, A. L.; Campelo, F.; Vivas, A. e Ramírez, J. A. (2013b). Problema de roteamento de veículos multiobjetivo com coleta seletiva. In Lopes, H. S.; de Abreu Rodrigues, L. C. e Steiner, M. T. A., editores, *Meta-Heurísticas em Pesquisa Operacional*, chapter 12, pp. 181–202. Omnipax, Curitiba, PR.

Associação Brasileira de Logística (2008). Associação brasileira de logística. `http:\\www.abralog.org.br`.

Associação Brasileira do Alumínio (2009). Reciclagem: Números da reciclagem. `http:\\www.abal.org.br`.

Associação Nacional de Petróleo (2013). Boletim anual de preços de 2013. `http:\\www.anp.gov.br`.

Barán, B. e Schaerer, M. (2003). A Multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows. In *Proceedings of the 21st IASTED International Conference on Applied Informatics*.

Basseur, M.; Liefooghe, A.; Le, K. e Burke, E. (2011). The efficiency of indicator-based local search for multi-objective combinatorial optimization problems. *Journal of Heuristics*, 18(2):263–296.

Batista, L.; Campelo, F.; Guimarães e F.G., Ramírez, J. (2012). The cone ϵ-dominance: An approach for evolutionary multiobjective optimization evolutionary computation. *Evolutionary Computation. [Submitted]*.

Batista, L. S.; Campelo, F.; Guimarães, F. G. e Ramírez, J. A. (2011). Pareto cone ϵ-dominance: Improving convergence and diversity in multiobjective evolutionary algorithms. In Takahashi, R. H.; Deb, K.; Wanner, E. F. e Greco, S., editores, *Evolutionary Multi-Criterion Optimization*, volume 6576 of *Lecture Notes in Computer Science*, pp. 76–90. Springer Berlin Heidelberg.

Beham, A. (2007). Parallel tabu search and the multiobjective capacitated vehicle routing problem with soft time windows. In Moreno Díaz, R.; Pichler, F. e Quesada Arencibia, A., editores, *Computer Aided Systems Theory – EUROCAST 2007*, volume 4739 of *Lecture Notes in Computer Science*, pp. 829–836. Springer Berlin Heidelberg.

Bianchessi, N. e Righini, G. (2007). Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers and Operations Research*, 34(2):578–594.

Boffey, B.; García, F.; Laporte, G.; Mesa, J. e Pelegrín, B. (1995). Multiobjective routing problems. *Top*, 3(2):167–220.

Borgulya, I. (2008). An algorithm for the capacitated vehicle routing problem with route balancing. *Central European Journal of Operations Research*, 16(4):331–343.

Bowerman, R.; Hall, B. e Calamai, P. (1995). A multi-objective optimization approach to urban school bus routing: Formulation and solution method. *Transportation Research Part A: Policy and Practice*, 29(2):107 – 123.

BRASIL (1990). Código de defesa do consumidor, lei 8078. Diário Oficial da União.

BRASIL (2002). Decreto n^o 4074, de 4 de janeiro de 2002. Diário Oficial da União.

Bruck, B. P.; dos Santos, A. G. e Arroyo, J. E. C. (2012). Hybrid metaheuristic for the single vehicle routing problem with deliveries and selective pickups. In *IEEE Congress on Evolutionary Computation*.

Carrano, E. G. (2007). *Algoritmos Evolucionários Eficientes para Otimização de Redes*. PhD thesis, Universidade Federal de Minas Gerais, Brasil.

Centro de Estudos em Logística - CEL/COPPEAD (2007). Panorama logístico: Gestão do transporte rodoviário de cargas nas empresas. `http:\\www.coppead.ufrj.br`.

Chankong, V. e Haimes, Y. Y. (1983). *Multiobjective Decision Making Theory and Methodology*. Elsevier Science, New York.

Chen, J.-F. (2006). Approaches for the vehicle routing problem with simultaneous deliveries and pickups. *Journal of the Chinese Institute of Industrial Engineers*, 23(2):141–150.

Chiang, C.-P. (2008). An efficiency frontier approach for the vehicle routing problem in a fluctuant demand environment. In *Proceedings of the 2008 Fourth International Conference on Natural Computation*.

Chitty, D. M. e Hernandez, M. L. (2004). A hybrid ant colony optimisation technique for dynamic vehicle routing. In *Genetic and Evolutionary Computation Conference*.

Coello, C. A. C.; Veldhuizen, D. A. V. e Lamont, G. B. (2011). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Press.

Confederação Nacional de Transportes (2013a). Economia em foco: Elevação do preço do diesel em 2013. `http:\\www.cnt.org.br`.

Confederação Nacional de Transportes (2013b). Pesquisa cnt de rodovias 2012. `http:\\www.cnt.org.br`.

Corberán, A.; Fernández, E.; Laguna, M. e Martí, R. (2002). Heuristic solutions to the problem of routing school buses with multiple objectives. *Journal of The Operational Research Society*, 53(4):427–435.

Cormen, T. H.; Stein, C.; Rivest, R. L. e Leiserson, C. E. (2001). *Introduction to Algorithms*. McGraw-Hill Higher Education.

Corne, D. W.; Jerram, N. R.; Knowles, J. D. e Oates, M. J. (2001). PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. In Spector, L.; Goodman, E. D.; Wu, A.; Langdon, W.; Voigt, H.-M.; Gen, M.; Sen, S.; Dorigo, M.; Pezeshk, S.; Garzon, M. H. e Burke, E., editores, *Genetic and Evolutionary Computation Conference*, pp. 283–290, San Francisco, California. Morgan Kaufmann Publishers.

Corne, D. W.; Knowles, J. D. e Oates, M. J. (2000). The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization. In Schoenauer, M.; Deb, K.; Rudolph, G.; Yao, X.; Lutton, E.; Merelo, J. J. e Schwefel, H.-P., editores, *Proceedings of the VI Conference Parallel Problem Solving from Nature*, pp. 839–848, Paris, France. Springer. Lecture Notes in Computer Science No. 1917.

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

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithm*. John Wiley & Sons.

Deb, K.; Pratap, A.; Agarwal, S. e Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182—197.

Dell'Amico, M.; Righini, G. e Salani, M. (2006). A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2):235–247.

Dethloff, J. (2001). Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up. *OR-Spektrum*, 23(1):79–96.

Dias, A. H. F. e Vasconcelos, J. A. (2002). Multiobjective genetic algorithms applied to solve optimization problems. *IEEE Transactions on Magnetics*, 38(2):1133–1136.

Doerner, K.; Focke, A. e Gutjahr, W. J. (2007). Multicriteria tour planning for mobile healthcare facilities in a developing country. *European Journal of Operational Research*, 179(3):1078–1096.

Doerner, K. e Hartl, R. (2008). Health care logistics, emergency preparedness, and disaster relief: New challenges for routing problems with a focus on the austrian situation. In Golden, B.; Raghavan, S. e Wasil, E., editores, *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*, pp. 527–550. Springer US.

Dunn, O. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64.

Edgeworth, F. Y. (1881). *Mathematical Physics*. P. Keagan, Londres, Inglaterra.

El-Sherbeny, N. (2001). *Resolution of a vehicle routing problem with multi-objective simulated annealing method*. PhD thesis, Faculté Polytechnique de Mons, Mons, Belgique.

El-Sherbeny, N.; Teghem, J. e Tuyttens, D. (2002). A multi-objective vehicle routing problem solved by simulated annealing. In *Conférence International de Mathématiques Appliquées et Sciences de l'Ingénieur*.

Empresa Brasileira de Pesquisa Agropecuária (2009). Empresa brasileira de pesquisa agropecuária. `http:\\www.embrapa.br`.

Euchi, J. e Mraihi, R. (2012). The urban bus routing problem in the tunisian case by the hybrid artificial ant colony algorithm. *Swarm and Evolutionary Computation*, 2(0):15–24.

Fonseca, C. M. e Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*.

Ford, H. (1927). *Hoje e Amanhã*. Companhia Editora Nacional, São Paulo. Tradução Monteiro Lobato.

Freitas, L. M. B. e Montané, F. T. (2008). Metaheurísticas vns-vnd e grasp-vnd para problemas de roteamento de veículos com coleta entrega simultâneas. *Simpósio de Pesquisa Operacional e Logística da Marinha*.

Gajpal, Y. e Abad, P. (2009). An ant colony system (acs) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, 36(12):3215–3223.

Garcia-Najera, A. (2009). Preserving population diversity for the multi-objective vehicle routing problem with time windows. In *Genetic and Evolutionary Computation Conference*.

Garcia-Najera, A. (2010). *Multi-Objective Evolutionary Algorithms for Vehicle Routing Problems*. PhD thesis, University of Birmingham.

Garcia-Najera, A. (2012). The vehicle routing problem with backhauls: a multi-objective evolutionary approach. In *Proceedings of the 12th European conference on Evolutionary Computation in Combinatorial Optimization*.

Garcia-Najera, A. e Bullinaria, J. A. (2009a). Bi-objective optimization for the vehicle routing problem with time windows: Using route similarity to enhance performance. In Ehrgott, M.; Fonseca, C. M.; Gandibleux, X.; Hao, J.-K. e Sevaux, M., editores, *5th International Conference on Evolutionary Multi-Criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, pp. 275–289. Springer.

Garcia-Najera, A. e Bullinaria, J. A. (2009b). Comparison of similarity measures for the multi-objective vehicle routing problem with time windows. In Ehrgott, M.; Fonseca, C. M.; Gandibleux, X.; Hao, J.-K. e Sevaux, M., editores, *Genetic and Evolutionary Computation Conference*, volume 5467 of *Lecture Notes in Computer Science*, pp. 579–586. Springer.

Garcia-Najera, A. e Bullinaria, J. A. (2010). Optimizing delivery time in multi-objective vehicle routing problems with time windows. In Schaefer, R.; Cotta, C.; Kolodziej, J. e Rudolph, G., editores, *11th International Conference on Parallel Problem Solving from Nature*, volume 6239 of *Lecture Notes in Computer Science - Part II*, pp. 51–60. Springer.

Garcia-Najera, A. e Bullinaria, J. A. (2011). An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, 38(1):287–300.

Geiger, M. (2003). A computational study of genetic crossover operators for multi-objective vehicle routing problem with soft time windows. In Habenicht, W.; Scheubrein, B. e Scheubrein, R., editores, *Multi-Criteria- und Fuzzy-Systeme in Theorie und Praxis*, pp. 191–207. Deutscher Universitätsverlag.

Geiger, M. J. (2001). Genetic algorithms for multiple objective vehicle routing. In *Metaheuristic International Conference*.

Geiger, M. J. (2006). Foundations of the pareto iterated local search metaheuristic. In *Proceedings of the 18th International Conference on Multiple Criteria Decision Making*.

Geiger, M. J. (2009). Improvements for multi-objective flow shop scheduling by pareto iterated local search. *Proceedings of the 8th Metaheuristics International Conference*.

Gelogullari, C. A. (2004). An exact algorithm for the vehicle routing problem with backhauls. *Computers and Operations Research*, 33(4):595–619.

Ghaziri, H. e Osman, I. H. (2003). A neural network algorithm for the traveling salesman problem with backhauls. *Computers and Industrial Engineering*, 44(2):267–281.

Giannikos, I. (1998). A multiobjective programming model for locating treatment sites and routing hazardous wastes. *European Journal of Operational Research*, 104(2):333 – 342.

Glover, F. W. e Kochenberger, G. A. (2003). *Handbook of Metaheuristics (International Series in Operations Research & Management Science)*. Springer.

Goetschalckx, M. e Jacobs-Blecha, C. (1993). The vehicle routing problem with backhauls: Properties and solution algorithms. Technical Report MHRC-TR-88-l3, Georgia Institute of Technology.

Gökçe, E. I. (2004). A revised ant colony system approach to vehicle routing problems. Master's thesis, School of Engineering and Natural Sciences, Sabanci Univerity.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, Boston.

Gribkovskaia, I.; Halskau, Ø.; Laporte, G. e Vlcek, M. (2007). General solutions to the single vehicle routing problem with pickups and deliveries. *European Journal of Operational Research*, 180(2):568 – 584.

Gribkovskaia, I.; Laporte, G. e Shyshou, A. (2008). The single vehicle routing problem with deliveries and selective pickups. *Computer & Operations Research*, 35(9):2908–2924.

Gurobi Optimization, I. (2013). Gurobi optimizer reference manual.

Gutiérrez, J. P. C.; Landa-Silva, D. e Moreno-Pérez, J. A. (2008). Exploring feasible and infeasible regions in the vehicle routing problem with time windows using a multi-objective particle swarm optimization approach. In *International Workshop on Nature Inspired Cooperative Strategies for Optimization*.

Gutiérrez-Jarpa, G.; Desaulniers, G.; Laporte, G. e Marianov, V. (2010). A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows. *European Journal of Operational Research*, 206(2):341 – 349.

Gutierrez-Jarpa, G.; Marianov, V. e Obreque, C. (2009). A single vehicle routing problem with fixed delivery and optional collections. *IIE Transactions*, 41(12):1067–1079.

Hong, S.-C. e Park, Y.-B. (1999). A heuristic for bi-objective vehicle routing with time window constraints. *International Journal of Production Economics*, 62(3):249 – 258.

Horn, J.; Nafpliotis, N. e Goldberg, D. E. (1994). A niched pareto genetic algorithm for multiobjective optimization. In *In Proceedings of the First IEEE Conference on Evolutionary Computation.*

Jaw, J.; Odoni, A.; Psaraftis, H. N. e Wilson, N. H. M. (1986). A heuristic algorithm for the multi-vehicle advance-request dial-a-ride problem with time window. *Transportation Research Part B*, 20(3):243–257.

Ji, P. e Chen, K. (2007). The vehicle routing problem: The case of the hong kong postal service. *Transportation Planning and Technology*, 30(2):167 – 182.

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2002). Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature.*

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2005). Enhancements of NSGA II and its application to the vehicle routing problem with route balancing. *Lecture Notes in Computer Science*, 3871:131–142.

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2007). Target aiming pareto search and its application to the vehicle routing problem with route balancing. *Journal of Heuristics*, 13(5):455–469.

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2008a). Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293–309.

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2008b). *The Vehicle Routing Problem.* Springer.

Jozefowiez, N.; Semet, F. e Talbi, E.-G. (2009). An evolutionary algorithm for the vehicle routing problem with route balancing. *European Journal of Operational Research*, 195(3):761–769.

Lacomme, P.; Prins, C. e Sevaux, M. (2006). A genetic algorithm for a bi-objective capacitated arc routing problem. *Computers & Operations Research*, 33(12):3473 – 3493. Part Special Issue: Recent Algorithmic Advances for Arc Routing Problems.

Lee, T.-R. e Ueng, J.-H. (1998). A study of vehicle routing problem with load balancing. *International Journal of Physical Distribution and Logistics Management*, 29:646–648.

Liao, X.-L. e Ting, C.-K. (2010). An evolutionary approach for the selective pickup and delivery problem. In *IEEE Congress on Evolutionary Computation.*

Lust, T. (2010). *New metaheuristics for solving MOCO problems: application to the knapsack problem, the traveling salesman problem and IMRT optimization.* PhD thesis, Faculté Polytechnique de Mons, Belgium.

M. Spada, M. B. e Liebling, T. M. (2005). Decision-aiding methodology for the school bus routing and scheduling problem. *Transportation Science*, 38(4):477–490.

Maravilha, A. L.; Morais, V. W. C.; Assis, L. e Vivas, A. (2010). Comparação entre duas abordagens bio-inspiradas aplicadas ao problema de roteamento de veículos com coleta e entrega simultâneas. *XLII Simpósio Brasileiro de Pesquisa Operacional*.

Menchik, C. (2010). Desafios do transporte na era da logística enxuta. In *Interlog*.

Meng, Q.; Lee, D.-H. e Cheu, R. L. (2005). Multiobjective vehicle routing and scheduling problem with time window constraints in hazardous material transportation. *Journal of Transportation Engineering*, 131(9):699–707.

Min, H. (1989). The multiple vehicle routing problem with simultaneous delivery and pickup points. *Transportation Research-A*, 23A(5):377–386.

Montané, F. A. T. e Galvão, R. D. (2006). A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers and Operations Research*, 33(3):595–619.

Montgomery, D. C. (1991). *Design and analysis of experiments*. John Wiley.

Montgomery, D. C. e Runger, G. C. (2003). *Applied statistics and probability for engineers*. John Wiley.

Morais, V. W. C.; Assis, L.; Caires, L. F. V. e Vivas, A. (2009). Algoritmo híbrido GRASP/VND/ILS aplicado ao problema de roteamento de veículos com coleta e entrega simultâneas. *Simpósio de Pesquisa Operacional e Logística da Marinha*.

Mosheiov, G. (1994). The traveling salesman problem with pick-up and delivery. *European Journal of Operational Research*, 79(2):299–310.

Mosheiov, G. (1998). Vehicle routing with pick-up and delivery: Tour-partition heuristics. *Computer and Industrial Engineering*, 34(3):669–684.

Mourgaya, M. e Vanderbeck, F. (2007). Column generation based heuristic for tactical planning in multi-period vehicle routing. *European Journal of Operational Research*, 183(3):1028 – 1041.

Murata, T. e Itai, R. (2005). Multi-objective vehicle routing problems using two-fold EMO algorithm to enhance solution similiraty on non-dominated set. In A. Coello Coello, C.; H. Aguirre, A. e Zitzler, E., editores, *Third International Conference Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science. Springer.

Murata, T. e Itai, R. (2007). Local search in two-fold emo algorithm to enhance solution similarity for multi-objective vehicle routing problems. In Obayashi, S.; Deb, K.; Poloni, C.; Hiroyasu, T. e Murata, T., editores, *Evolutionary Multi-Criterion Optimization*, volume 4403 of *Lecture Notes in Computer Science*, pp. 201–215. Springer Berlin Heidelberg.

Murata, T. e Itai, R. (2008). *Enhancing Solution Similarity in Multi-Objective Vehicle Routing Problems with Different Demand Periods*, chapter 7. InTech.

Ombuki, B.; Ross, B. J. e Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24(1):17–30.

Or, I. (1976). *Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking*. PhD thesis, Northwestern University.

Pacheco, J. e Marti, R. (2006). Tabu search for a multi-objective routing problem. *Journal of the Operational Research Society*, 57(1):29–37.

Papadimitriou, C. H. e Steiglitz, K. (1982). *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Pareto, V. (1896). *Cours d'économie politique*. Rouge, Lausanne,Suíça.

Park, J. e Kim, B.-I. (2010). The school bus routing problem: A review. *European Journal of Operational Research*, 202(2):311 – 319.

Parragh, S.; Doerner, K. e Hartl, R. (2008). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1,2):81–117.

Pasia, J. M.; Doerner, K. F.; Hartl, R. F. e Reimann, M. (2007). A population-based local search for solving a bi-objective vehicle routing problem. In *Proceedings of the 7th European conference on Evolutionary computation in combinatorial optimization*.

Penna, P.; Subramanian, A. e Ochi, L. (2013). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, 19(2):201–232.

Potvin, J.-Y. e Bengio, S. (1996). The vehicle routing problem with time windows part ii: Genetic search. *INFORMS Journal on Computing*, 8(2):165–172.

Potvin, J.-Y.; Duhamel, C. e Guertin, F. (1996). A genetic algorithm for vehicle routing with backhauling. *Applied Intelligence*, 6(4):345–355.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

Rahoual, M.; Kitoun, B.; Mabed, M.; Bachelet, V. e Benameur, F. (2001). Multicriteria genetic algorithms for the vehicle routing problem with time windows. In *Metaheuristic International Conference*.

Reimann, M.; Doerner, K. e Hartl, R. F. (2002). Insertion based ants for vehicle routing problems with backhauls and time windows. In *Proceedings of the Third International Workshop on Ant Algorithms*.

Ribeiro, R. e Lourenco, H. R. (2001). A Multi-Objective Model for a Multi-Period Distribution Management Problem. In *Proceedings of the 4th Metaheuristics International Conference*.

Ropke, S. e Pisinger, D. (2006). A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operation Research*, 171(3):750–775.

Salhi, S. e Nagy, G. (1999). A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of Operational Research Society*, 50(10):345–355.

Schaffer, J. D. (1984). *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University.

Sessomboon, W.; Watanabe, K.; Irohara, T. e Yoshimoto, K. (1998). A study on multi-objective vehicle routing problem considering customer satisfaction with due-time (the creation of Pareto optimal solutions by hybrid genetic algorithm). *Transaction of the Japan Society of Mechanical Engineering*.

Sousa, J. C.; Biswas, H. A.; Brito, R. e Silveira, A. (2011). A multi objective approach to solve capacitated vehicle routing problems with time windows using mixed integer linear programming. *International Journal of Advanced Science and Technology*, 28:pp. 1–8.

Srinivas, N. e Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2:221–248.

Steuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation, and Application*. Krieger Pub Co.

Subramanian, A.; Drummond, L.; Bentes, C.; Ochi, L. e Farias, R. (2010). A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899–1911.

Subramanian, A.; Ochi, L. S. e dos Anjos Formiga Cabral, L. (2008). An efficient heuristic for the vehicle routing problem with simultaneous pickup and delivery. Technical report, Instituto de Computação, Universidade Federal Fluminense.

Subramanian, A.; Uchoa, E.; Pessoa, A. A. e Ochi, L. S. (2011). Branch-and-cut with lazy separation for the vehicle routing problem with simultaneous pickup and delivery. *Operations Research Letters*, 39(5):338 – 341.

Süral, H. e Bookbinder, J. H. (2003). The single-vehicle routing problem with unrestricted backhauls. *Networks*, 41(3):127–136.

Takahashi, R. H. C. (2004). Notas de aula: Otimização escalar e vetorial. Departamento de Matemática, Universidade Federal de Minas Gerais.

Tan, K.; Cheong, C. Y. e Goh, C. K. (2007). Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839.

Tan, K.; Chew, Y. e Lee, L. (2003). A multiobjective evolutionary algorithm for solving vehicle routing problem with time window. In *IEEE International Conference on Systems, Man, and Cybernetics*.

Tan, K.; Chew, Y. e Lee, L. (2006a). A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855 – 885.

Tan, K.; Chew, Y. e Lee, L. H. (2006b). A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Computational Optimization and Applications*, 34(1):115–151.

Toth, P. e Vigo, D. (1996). *Meta-heuristics: Theory and applications*, chapter Fast local search algorithms for the handicapped persons transportation problem. Kluwer.

Toth, P. e Vigo, D. (2002). *The Vehicle Routing Problem*. Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics.

Wassan, N. A.; Wassan, A. H. e Nagy, G. (2007). A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. *Journal of Combinatorial Optimization*, 15(4):368–386.

Watanabe, S. e Sakakibara, K. (2007). A multiobjectivization approach for vehicle routing problems. In *Proceedings of the 4th international conference on Evolutionary multi-criterion optimization*.

Zhang, T.; xin Tian, W.; jie Zhang, Y. e xin Liu, S. (2008). Improved ant colony system for vrpspd with maximum distance constraint. *Systems Engineering - Theory & Practice*, 28(1):132–140.

Zitzler, E. (1999). *Evolutionary Algorithm for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology Zurich.

Zitzler, E. e Künzli, S. (2004). Indicator-based selection in multiobjective search. In *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*.

Zitzler, E.; Laumanns, M. e Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In Giannakoglou, K.; Tsahalis, D.; Periaux, J.; Papailou, P. e

Fogarty, T., editores, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pp. 95–100, Athens, Greece.

Ziviani, N. (2004). *Projeto de algoritmos: com implementações em Pascal e C*. THOMSON PIONEIRA.

Zografos, K. G. e Androutsopoulos, K. N. (2004). A heuristic algorithm for solving hazardous materials distribution problems. *European Journal of Operational Research*, 152(2):507 – 519.

Apêndices

Publicações

- Artigo completo publicado em periódico
  - ASSIS, L. P., MARAVILHA, A. L., CAMPELO, F., VIVAS, A., RAMÍREZ, J. A. **Multiobjective vehicle routing problem with fixed delivery and optional collections**. Optimization Letters. 2012.
- Capítulo de livro
  - ASSIS, L. P., MARAVILHA, A. L., CAMPELO, F., VIVAS, A., RAMÍREZ, J. A. **Problema de Roteamento de Veículos Multiobjetivo com Coleta Seletiva**. Meta-Heurísticas em Pesquisa Operacional; Heitor Silvério Lopes and Luiz Carlos de Abreu Rodrigues and Maria Teresinha Arns Steiner. Omnipax. 2013.
- Artigo completo publicado em anais de congresso
  - MARAVILHA, A., L., ASSIS, L. P., VIVAS, A., RAMÍREZ, J. A. **Abordagem multiobjetivo para problema de roteamento de veículos com serviço de coleta e entrega**. X Congresso Brasileiro de Inteligência Computacional (CBIC'2011). Fortaleza, 2011.

Optim Lett (2013) 7:1419–1431
DOI 10.1007/s11590-012-0551-z

ORIGINAL PAPER

Multiobjective vehicle routing problem with fixed delivery and optional collections

Luciana P. Assis · André L. Maravilha · Alessandro Vivas · Felipe Campelo · Jaime A. Ramírez

Received: 1 October 2011 / Accepted: 1 September 2012 / Published online: 18 September 2012

Abstract We present an adaption on the formulation for the vehicle routing problem with fixed delivery and optional collections, in which the simultaneous minimization of route costs and of collection demands not fulfilled is considered. We also propose a multiobjective version of the iterated local search (MOILS). The performance of the MOILS is compared with the ϵ-constrained (P_ϵ) ILS, the NSGA-II and the indicator-based multi-objective local search methods in the solution of 14 problem instances containing between 50 and 199 customers plus the depot. The results indicate that the MOILS outperformed the other approaches, obtaining significantly better average values for coverage, hypervolume and cardinality.

Keywords Vehicle routing problem · Selective pickups · Iterated local search · Multiobjective optimization

L. P. Assis · A. Vivas
Departamento de Computação, Universidade Federal dos Vales do Jequitinhonha e Mucuri,
Rod. MGT 367, Km 583, 5000-Alto da Jacuba, 39100-000 Diamantina, MG, Brazil
e-mail: lpassis@ufvjm.edu.br

A. Vivas
e-mail: alessandrovivas@ufvjm.edu.br

A. L. Maravilha · F. Campelo · J. A. Ramírez ( )
Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais,
Av. Antônio Carlos 6627, 31270-901 Belo Horizonte, MG, Brazil
e-mail: jramirez@ufmg.br

A. L. Maravilha
e-mail: andrelms@ufmg.br

F. Campelo
e-mail: fcampelo@ufmg.br

1 Introduction

The vehicle routing problem (VRP) is one of the most important and widely studied within the context of combinatorial optimization. It includes many relevant issues of transportation logistic, e.g., the pickup and delivery of materials, a problem of particular interest for industries concerned with the efficient management of returned goods for recycling, re-manufacturing or reuse.

In some variants of the VRP the delivery of products is mandatory, while the collection can be postponed. Traditionally, the VRP has been defined as mono-objetive and solved using exact methods. In [34], a single-VRP with unrestricted backhauls, in which a profit value is associated for each satisfied collect demand, is solved with an exact branch-and-bound algorithm. Later works [14,15] presented a branch-and-cut algorithm to solve the same problem, and a branch-and-price algorithm to solve five variants of the VRP with delivery and selective pickups with time windows. Recently, mono-objective versions of the VRP have been solved using heuristics, most of them based on local search such as simulated annealing [25] and tabu search [23,13]. These general procedures explore the space of the solution to find good solutions with a reasonable computational time [12].

Several practical VRPs are multiobjective by nature, due to the inherent necessity of considering aspects other than the minimization of the total cost [28], such as customer satisfaction, labor regulations, or other aspects of the problem [21]. There are however few works that treat the VRP as multiobjective, in particular those problems involving pickup and delivery services [20].

In terms of multiobjective approaches, metaheuristics such as genetic algorithms [8,30], tabu search [9,16,26], memetic algorithms [10] and simulated annealing [5,31,36] have been widely used to generate approximately efficient solutions for multiobjective problems. In general, algorithms based on local search perform well in multiobjective combinatorial problems [3,18,35] and could be applied to VRPs.

We discuss in this work an adaption on the VRP in order to treat its multiobjective nature with fixed deliveries and optional collections. More specifically, the objectives are defined as the minimization of route costs and of collection demands not fulfilled. We also propose a multiobjective version of the iterated local search (ILS) heuristic. The MOILS approach is applied to a set of instances containing between 50 and 199 customers plus the depot, and is compared to other approaches available in the literature [3,4,6,19].

2 Problem description

The multiobjective VRP with fixed delivery and optional collections consists in defining a set of routes that minimize the transportation cost and the number of collections not carried out, attending all delivery demands. Each customer must be visited by one vehicle, and partial execution of demands is not allowed.

2.1 Mathematical formulation

The mathematical formulation is a generalization of the model proposed in [23] for the VRP with simultaneous pickup and delivery (VRPSPD), obtained by transforming a constraint into an objective.

Let $G = (V, A)$ be a complete graph, with $V = \{0, 1, 2, \ldots, n\}$ the set of vertices and $A = \{(i, j) : i, j \in V, i \neq j\}$ the set of edges. The vertex 0 represents the depot and the others represent the customers. Each edge (i, j) has an associated value $c_{ij} \geq 0$ that represents the cost for reaching vertex j from vertex i. Each customer i has a demand d_i for delivery and a demand p_i for collection. There are $\overline{k}$ homogeneous vehicles available, each with capacity Q. The parameter y_{ij} is the sum of the collected load between the depot and the node i (included) driven to node j. The parameter z_{ij} is the sum of the load delivered to customers after node i (excluded) driven to node j. We use the edge variable x_{ij}^k, which is equal to one if edge (i, j) is traveled by vehicle k and zero otherwise, and the choice variable ℓ_j, which is equal to one if pickup demand of customer j is satisfied, and zero otherwise.

The objectives of this problem can be expressed as:

$$\text{minimize} \begin{cases} \sum_{k=1}^{\overline{k}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k \\ \sum_{j=1}^{n} p_j (1 - \ell_j) \end{cases} \tag{1}$$

$$\text{Subject to: } \sum_{i=0}^{n} \sum_{k=1}^{\overline{k}} x_{ij}^k = 1, \qquad j = 1, \ldots, n \tag{2}$$

$$\sum_{i=0}^{n} x_{ij}^k - \sum_{i=0}^{n} x_{ji}^k = 0, \qquad j = 0, \ldots, n \quad \text{and} \quad k = 0, \ldots, \overline{k} \tag{3}$$

$$\sum_{j=1}^{n} x_{0j}^k \leq 1, \qquad k = 1, \ldots, \overline{k} \tag{4}$$

$$\sum_{i=0}^{n} z_{ij} - \sum_{i=0}^{n} z_{ji} = d_j, \qquad \forall j \neq 0 \tag{5}$$

$$\sum_{i=0}^{n} y_{ij} - \sum_{i=0}^{n} y_{ji} = p_j \ell_j, \qquad \forall j \neq 0 \tag{6}$$

$$y_{ij} + z_{ij} \leq Q \sum_{k=1}^{\overline{k}} x_{ij}^k, \qquad i, j = 0, \ldots, n \tag{7}$$

$$x_{ij}, \ell_j \in \{0, 1\}, \qquad i, j = 0, \ldots, n \tag{8}$$

$$y_{ij}, z_{ij} \geq 0, \qquad i, j = 0, \ldots, n \tag{9}$$

The two objective functions (1) are the total cost of the routes and the collections not carried out, and have to be minimized. The constraints model the following

requirements: each point of demand must be visited by only one vehicle (2); the same vehicle arrives and departs from each client (3); a maximum of $\overline{k}$ vehicles can be used (4); all delivery demands must be met (5); when variable ℓ_j is equal to one, the pickup demand of customer j must be satisfied (6); all demands should be transported on the arcs included in the solution (7). In addition, (8) represents the integrality restriction and (9) a non-negativity constraint for collection and delivery demands.

2.2 Multiobjective iterated local search

The iterated local search (ILS) algorithm is based in the idea that the local search procedure can be improved from the generation of new solutions for the starting point by perturbing visited local optimal solutions [22].

Some heuristic approaches found in literature to solve the VRP with simultaneous pickup and delivery indicate that the use of ILS is a good alternative [32]. Some of its advantages are: the existence of few adjusting parameters when compared to the evolutionary methods, simplicity, robustness, effectiveness and the ease of implementation [22].

In this section, we present an adaptation of the ILS for the solution of combinatorial optimization multiobjective problems. This approach, called the multiobjective iterated local search (MOILS), is presented in Algorithm 1.

Algorithm 1: Multiobjective Iterated Local Search (MOILS)

```
   Input: maxIter, maxCount
 1 Front ← generateInitialSolutions();
 2 iter ← 0;
 3 while iter < maxIter do
 4     s' ← select(Front);
 5     count ← 0;
 6     while count < maxCount do
 7         s'' ← perturb(s');
 8         s'' ← localSearch(s'');
 9         inserted ← update(Front, s'');
10         if inserted then
11             count ← 0;
12             s' ← s'';
13         else
14             count ← count + 1;
15     iter ← iter + 1;
   Output: Front
```

First, the MOILS constructs two initial solutions (line 1) by efficient hybrid algorithm based on ILS and random variable neighborhood descent (RVND), proposed in [27]. These solutions represent the extremes of the front, one being a point of maximum transportation cost and the other a point of minimum transportation cost, where

no collections are performed. When all collection demands should be attended the problem is reduced to the mono-objective VRPSPD, and when no collection demand is attended it is reduced to the mono-objective capacitated VRP (CVRP).

The algorithm executes $maxIter$ iterations (lines 3–15), where one non-dominated solution, in the front set, is selected (line 4) to be exploited (lines 6–14). The selection procedure is based on the crowding-distance [6], which allows less explored regions to have higher selection priority. In this work, the crowding-distance is calculated in a different manner, with the value of the extreme solutions equal to twice the distance between this point and the nearest solution. We have proposed this modification in order to guarantee the exploration of the two initial solutions (equivalent to the CVRP and VRPSPD solutions) in early iterations and to give advantage to more intermediate solutions after the iterative process starts generating more non-dominated points.

Given a selected solution, the algorithm executes $maxCount$ iterations to explore it. In each iteration, this solution is perturbed and local search is applied, according to the ILS. Here, perturbation is done in two phases. In the first, the collection status (attempt or not attempt) of some customers randomly selected is changed. When the pickup demand ceases to be fulfilled, the respective customer can remain in the same position of the route. Otherwise, the load stored by the vehicle that will satisfy this demand must be verified. If its capacity is exceeded, the customer is inserted in the first viable position found. When no viable position is found, this customer is included in a new route. In the second phase, the algorithm applies one of the following perturbation mechanisms randomly chosen: Multiple Swap that reallocate different customers of the different routes randomly selected; Multiple Shift that exchanges two customers of distinct routes; and Ejection Chain that relocates a customer from route R_1 to route R_2, a customer from R_2 to R_3, and so on.

The local search phase explores the neighborhood of a solution with the objective of finding other non-dominated solutions for the problem. We have used the random variable neighborhood search (RVND) refinement method [27]. We implemented 12 types of movements to define the neighborhood, six inter-route and six intra-route. The inter-route movements are: (1) *Shift(1,0)*, (2) *Shift(2,0)*, that relocate a customer or two adjacent customers, (3) *Swap(1,1)*, (4) *Swap(2,1)*, (5) *Swap(2,2)* that interchange two customers, or two adjacent customers with one customers or with others two adjacent customers, and (6) *Crossover* that removes two arcs (i, j) and (i', j') and inserts two new arcs (i, j') and (i', j). The intra-route movements are: (1) *Or-opt*, (2) *Or-opt 2*, (3) *Or-opt 3* that relocate one customer, or two or three adjacent customers, (4) *2-opt* that removes two non-consecutive arcs (i, j) and (i', j') and replaces by the arcs (i, i') and (j, j'), (5) *Exchange* two customers, and (6) *Reverse* the direction of the route, aimed at reducing the load of the vehicle. Figure 1 shows examples of these operations, in which the highlighted arcs and customers indicate the changes in the original solution.

The best improvement strategy was used in all neighborhood structures. The computational complexity of each one of these moves is $O(n^2)$, except to reverse that is $O(n)$. After RVND executions, the algorithm verifies if it is possible to attempt the pickup demands of some customers. The collection of one customer is carried out only if this demand does not exceed the capacity of the vehicle and if it can remain in the same position of the route.

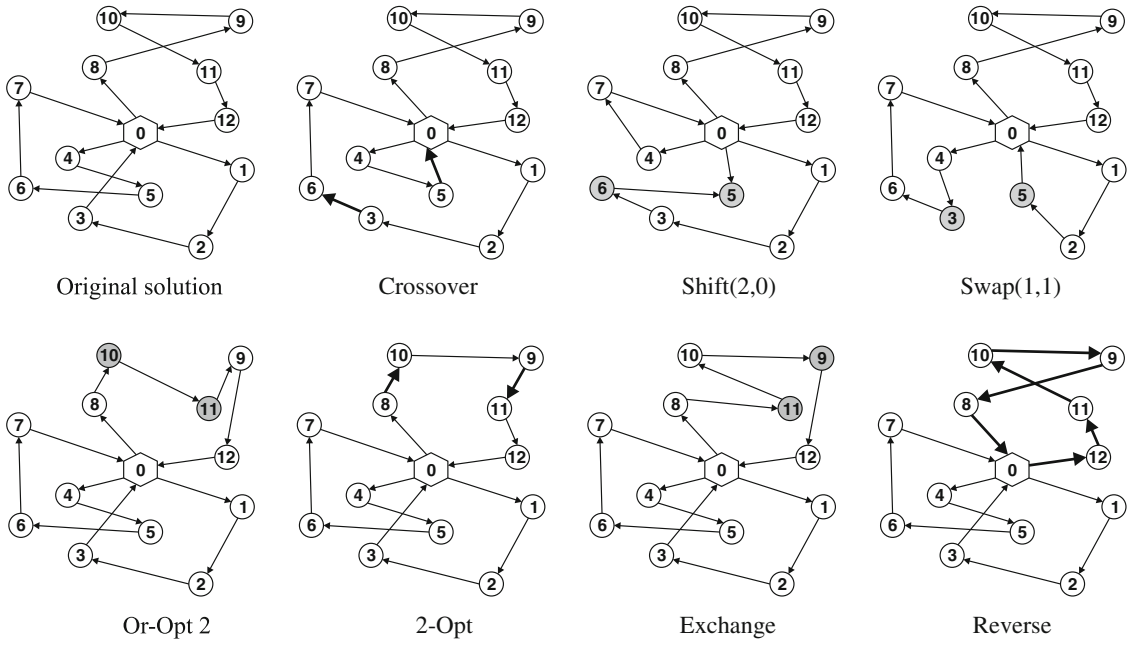


Fig. 1 Neighborhood structures

The solution generated in the perturbation and local search procedures is then included in the front (line 9). The updating phase verifies if a specific solution s is non-dominated by any solution, then it is inserted in the set and the solutions dominated by it are eliminated. The variable *count* represents the number of iterations that one solution is explored without generating a new non-dominated solution. If one solution is inserted in the front then *count* is restarted (line 11), otherwise it is increased by one unit (line 14).

3 Experimental results

3.1 Test problems and performance metrics

To evaluate the performance of the MOILS, an experiment was set up[1] considering 14 problem instances ranging from 50 to 199 customers plus the depot [29]. The travel cost is calculated as the Euclidean distance between the points. Problems 1–5 have randomly placed cities in the plane, while problems 11 and 12 have cities appearing in clusters, in which the depot is not centered.

These instances were solved using the following approaches:

1. the MOILS, as described in Sect. 2.2. After preliminary testing, the constants were set as $maxIter = 100$ and $maxCount = 15$.
2. ϵ-constrained approach (P_ϵ) [4] using the ILS as the base algorithm, in order to compare the performance of our proposed MOILS against that of an adapted

[1] The algorithm implementations, problem instances, and statistical routines used in the analysis of this experiment are available online at the address http://www.cpdee.ufmg.br/~fcampelo/files/OPTL2012a./.

single-objective iterated local search. Preliminary testing suggested the use of 50 as the number of iterations for this method [22];
3. non-dominated Sorting Genetic Algorithm II (NSGA-II) [6], which is a standard evolutionary multiobjective algorithm. The implementation used crossover and mutation operators already adapted for VRPs [19], with a population size of 500 evolving over 250 iterations;
4. indicator-Based Multi-Objective Local Search (IBMOLS) [3], a local search-based method for multiobjective optimization included to provide a better comparison baseline. The implementation used the epsilon binary indicator, employed the same local search operators as the MOILS and the RM population generation method with the mutation based on perturbation mechanism of the MOILS. Preliminary testing suggested using a scale factor of 0.1, population size of 10 and 50 generation.

To evaluate the relative performance of these approaches, three performance metrics were considered: the hypervolume (S), standardized to the interval (0, 1) independently for each problem; the Coverage of Two Sets [37]; and the cardinality (Card) of the final solution set returned. Both the S and CS-metrics return values representing percentages, while the Cardinality metric provides absolute values.

The free parameters of the algorithms used were adjusted in order to provide each method with approximately the same run time. No significant differences were detected in the runtime of the algorithms tested ($p > 0.05$).

3.2 Statistical design

We have employed statistical tests designed to detect significant differences and to estimate their magnitude for each quality metric. The data used was composed of the metrics calculated for the final Pareto-fronts obtained on eight independent runs of each algorithm on each problem.

For each metric, the experiment was designed as a randomized complete block design (RCBD) with the algorithms as levels of the experimental factor, and the problems as a blocking factor [24]. By treating the problems as blocks, it was possible to model and remove the effects of different instances on the performance of the algorithm, and obtain an overall performance difference across all test instances used. The null hypotheses of absence of differences among the algorithms evaluated over all problems were considered against two-sided alternatives. To avoid the assumptions of the F test, the more versatile Gore test [11], a more robust alternative for the ANOVA, was employed.

After testing for significance, least squares estimators of the block (instance) effects were obtained [24] and subtracted from the samples, thus allowing a problem-independent estimation of the effect size for each algorithm. The estimations of effect size were calculated by means of the Hodges–Lehmann (HL) estimator of the median of differences between two independent samples [17]. A more detailed description of the statistical methods is available in an earlier work [1].

Table 1 Mean values and standard deviations (in the parentheses) for the performance metrics used

Instances	Hypervolume				Cardinality			
(CMT)	MOILS	P_ϵ	NSGA-II	IBMOLS	MOILS	P_ϵ	NSGA-II	IBMOLS
1X	0.91 (0.16)	0.60 (0.17)	0.86 (0.16)	0.97 (0.02)	3.12 (0.6)	6.25 (3.6)	2.12 (0.3)	2.75 (0.5)
1Y	0.95 (0.02)	0.90 (0.05)	0.87 (0.07)	0.91 (0.03)	11.12 (1.1)	8.62 (1.5)	7.5 (1.2)	7.87 (1.9)
2X	0.88 (0.07)	0.77 (0.09)	0.83 (0.10)	0.83 (0.08)	8.12 (2.8)	9.50 (2.1)	4.00 (1.6)	4.50 (2.1)
2Y	0.93 (0.01)	0.82 (0.05)	0.84 (0.06)	0.89 (0.02)	33.62 (6.5)	27.25 (4.7)	31.12 (5.3)	25.12 (4.3)
3X	0.94 (0.05)	0.74 (0.20)	0.91 (0.05)	0.86 (0.07)	6.37 (0.9)	8.87 (3.5)	3.37 (1.1)	4.37 (1.3)
3Y	0.96 (0.00)	0.76 (0.05)	0.86 (0.04)	0.88 (0.06)	16.63 (2.6)	13.63 (2.9)	15.5 (5.1)	15.38 (4.0)
4X	0.90 (0.04)	0.83 (0.08)	0.78 (0.06)	0.73 (0.07)	15.25 (2.6)	12.63 (5.0)	5.0 (2.5)	6.50 (1.6)
4Y	0.95 (0.02)	0.88 (0.06)	0.83 (0.09)	0.92 (0.04)	42.13 (3.3)	32.63 (5.5)	35.5 (8.0)	30.7 (11.9)
5X	0.91 (0.04)	0.82 (0.05)	0.85 (0.06)	0.87 (0.05)	17.25 (4.2)	16.25 (4.3)	8.25 (5.2)	6.75 (1.8)
5Y	0.94 (0.01)	0.82 (0.04)	0.83 (0.03)	0.91 (0.02)	51.63 (5.9)	52.0 (10.6)	59.88 (14.1)	32.13 (4.8)
11X	0.94 (0.03)	0.87 (0.02)	0.87 (0.03)	0.88 (0.02)	44.75 (5.5)	42.13 (5.3)	40.75 (7.1)	21.25 (3.3)
11Y	0.91 (0.08)	0.65 (0.08)	0.70 (0.03)	0.76 (0.09)	27.63 (5.6)	20.25 (3.8)	20.5 (2.8)	14.75 (3.3)
12X	0.96 (0.02)	0.75 (0.04)	0.87 (0.08)	0.88 (0.04)	14.63 (3.7)	9.25 (2.2)	14.38 (3.4)	11.75 (4.5)
12Y	0.98 (0.01)	0.85 (0.06)	0.91 (0.03)	0.90 (0.05)	32.75 (3.7)	14.50 (3.6)	20.00 (5.4)	13.25 (2.9)
Mean	0.93 (0.04)	0.79 (0.07)	0.84 (0.06)	0.87 (0.05)	23.21 (3.5)	19.55 (4.19)	19.13 (4.51)	14.08 (3.44)

Table 1 continued

Instances	Coverage			
(CMT)	MOILS	P_ϵ	NSGA-II	IBMOLS
1X	0.80 (0.29)	0.53 (0.09)	0.65 (0.15)	0.76 (0.16)
1Y	0.53 (0.24)	0.63 (0.20)	0.36 (0.12)	0.60 (0.14)
2X	0.63 (0.18)	0.54 (0.13)	0.65 (0.30)	0.64 (0.24)
2Y	0.67 (0.06)	0.42 (0.09)	0.55 (0.15)	0.67 (0.12)
3X	0.59 (0.37)	0.60 (0.14)	0.72 (0.25)	0.61 (0.24)
3Y	0.74 (0.08)	0.34 (0.05)	0.56 (0.13)	0.61 (0.14)
4X	0.67 (0.17)	0.59 (0.10)	0.48 (0.09)	0.40 (0.14)
4Y	0.68 (0.12)	0.52 (0.16)	0.40 (0.12)	0.54 (0.22)
5X	0.64 (0.19)	0.59 (0.15)	0.56 (0.18)	0.58 (0.15)
5Y	0.73 (0.09)	0.39 (0.06)	0.55 (0.08)	0.70 (0.09)
11X	0.72 (0.08)	0.56 (0.06)	0.39 (0.10)	0.45 (0.08)
11Y	0.71 (0.14)	0.55 (0.06)	0.41 (0.10)	0.41 (0.10)
12X	0.73 (0.11)	0.42 (0.08)	0.58 (0.07)	0.51 (0.05)
12Y	0.80 (0.10)	0.31 (0.02)	0.49 (0.09)	0.49 (0.09)
Mean	0.69 (0.16)	0.50 (0.10)	0.53 (0.14)	0.57 (0.14)

Table 2 Estimated difference in average performance between the row and column algorithms for the performance metrics

	Hypervolume			Coverage			Cardinality		
	Pϵ	NSGA-II	IBMOLS	Pϵ	NSGA-II	IBMOLS	Pϵ	NSGA-II	IBMOLS
MOILS	0.11	0.08	0.06	0.70	0.32	0.13	2.56	4.32	6.06
Pϵ	–	−0.03	−0.05	–	−0.28	−0.54	–	n.s	3.50
NSGA-II	–	–	n.s	–	–	−0.18	–	–	1.73

Only results significant at a 95 % confidence level (adjusted for multiple hypothesis testing using Dunn-Šidák correction [7]) are shown. Positive values indicate higher average value for the algorithm in the row. OBS:
n.s. not statistically significant

3.3 Results and discussion

The results obtained for the experimental comparison are summarized in Table 1, which reports the mean and standard deviation values of the three metrics considered in each problem; and Table 2, where the results of the statistical analysis are summarized and the magnitude of the statistically significant differences are presented.

For the hypervolume, all algorithms presented a relatively solid performance in most problems, as shown in Table 1, with the P_ϵ returning slightly smaller values than the other three methods. The statistical analysis reported in Table 2 confirms this observation, showing that all algorithms presented small but statistically significant differences, with the MOILS showing the best results (11 % superior to the P_ϵ, 8 % better than the NSGA-II and 6 % superior to the IBMOLS). This result indicates that the proposed approach was generally able to obtain better fronts, either by returning a well-spread set of solutions, or points that were closer to the real Pareto-optimal front.

The average differences obtained for the cardinality metric were also relatively modest. Table 1 shows that, for each particular problem, all algorithms returned approximately the same mean number of solutions, with no large trends being easily discernible. Table 2 shows that, when integrated over all problems, small but statistically significant differences were observed, with the MOILS returning on average 2.56 more solutions than the P_ϵ, 4.32 more than the NSGA-II, and 6.06 more than the IBMOLS.

The differences in performance observed for the coverage metric were considerably large. By examining the results in Table 2, it can be easily seen that the MOILS was able to outperform the P_ϵ by an average of 0.7, the NSGA-II by 0.32 and the IBMOLS by 0.13. This is a strong indicator of superiority, since it means that, on average, the fronts found by the MOILS were able to dominate, on average, about 70 % of the ones yielded by the P_ϵ method, 32 % of those obtained by the NSGA-II and 13 % of the fronts returned by the IBMOLS. To simplify the reporting of the relative coverage values, a generalized version of the Coverage of Two Sets, called Coverage of Many Sets [2], was employed to generate the data shown in Table 1. Instead of quantifying how much a given algorithm covers another, this generalized metric instead measures how much a given algorithm covers the *union* of the final fronts returned by all algorithms except itself.

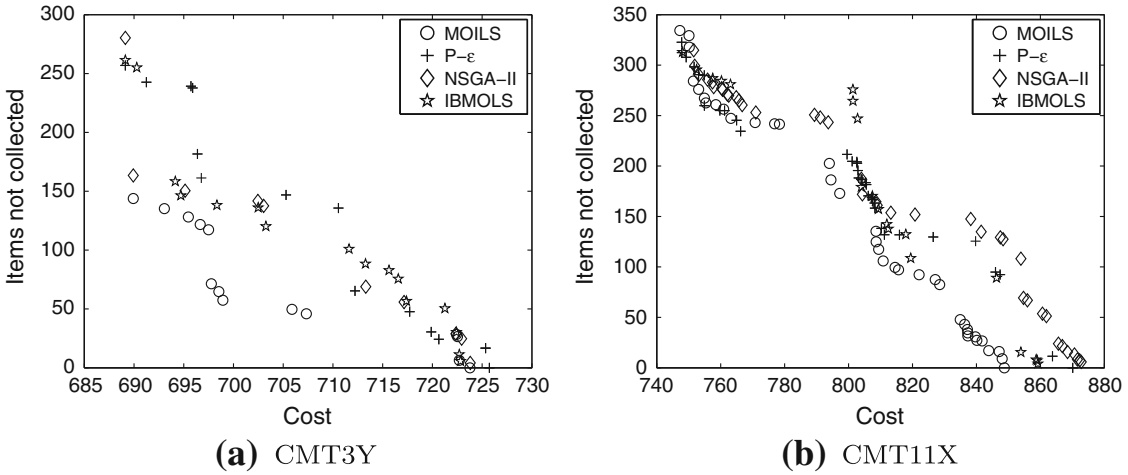


Fig. 2 Typical fronts obtained by the algorithms on selected problems

Table 3 Comparison with best known single-objective results for the VRPSPD

	Instances (CMT)						
	1X	1Y	2X	2Y	3X	3Y	4X
MOILS	466.77	466.77	688.05	693.09	721.40	723.28	857.44
Best known	466.77[a]	466.77[a]	668.77[b]	663.25[b]	721.27[a]	721.27[a]	852.46[b]
Error* (%)	0.00	0.00	2.88	4.50	0.02	0.28	0.58
	Instances (CMT)						
	4Y	5X	5Y	11X	11Y	12X	12Y
MOILS	855.74	1037.25	1032.68	848.72	850.67	665.85	664.00
Best known	852.35[b]	1029.25[b]	1029.25[b]	833.92[b]	830.39[b]	644.70[b]	659.52[b]
Error* (%)	0.40	0.78	0.33	1.77	2.44	3.28	0.68

* Percent difference between the results obtained by MOILS and the best known
[a] Optimal solution obtained by Branch-and-cut [33]
[b] Results of the parallel ILS-RVND algorithm reported in the literature [32]

Figure 2 shows two representative examples of fronts obtained by the algorithms. Since the true Pareto-optimal front is not known for the problems considered, it is not possible to objectively evaluate the absolute quality of the fronts obtained. However, the extreme point of the fronts obtained for the case in which all pickups are performed is equivalent to the corresponding VRPSPD instance. Table 3 shows a comparison of the extreme point of the non-dominated fronts obtained by the MOILS with the best known results for this problem [32], showing that, at least for the extreme points, the MOILS approach is able to find solutions that are generally very close to the best available in the literature.

4 Conclusions

We proposed an adaptation of the mathematical formulation for the multiobjective VRP with fixed delivery and optional collections, which takes into account the total

cost of the solution and the number of pickups performed. A multiobjective version of the iterated local search algorithm (MOILS) was proposed and compared on a test set of 14 benchmark problems against the P_ϵ, NSGA-II and IBMOLS approaches. The results obtained show that the MOILS was able to significantly outperform the three methods, obtaining superior values for coverage, hypervolume and cardinality across the set of test problems used.

Acknowledgments This work was supported by the following agencies: National Council for Research and Development (CNPq), grants 306910/2006-3 and 472446/2010-0; the Coordination for the Improvement of Higher Education Personnel (CAPES); and the Research Foundation of the State of Minas Gerais (FAPEMIG, Brazil), grants Pronex: TEC 01075/09 and Pronem: CEX APQ-04611-10.

References

1. Batista, L.S., Campelo, F., Guimarães, F.G., Ramírez, J.A.: Pareto cone ϵ-dominance: improving convergence and diversity in multiobjective evolutionary algorithms. Lect. Notes Comput Sci. (Springer) **6576**, 76–90 (2011)
2. Batista, L.S., Campelo, F., Guimarães, F.G., Ramírez, J.A.: The cone ϵ-dominance: an approach for evolutionary multiobjective optimization. Evol. Comput. (submitted)
3. Basseur, M., Liefooghe, A., Le, K., Burke, E.K.: The efficiency of indicator-based local search for multi-objective combinatorial optimization problems. J. Heuristics **18**(2), 263–296 (2011)
4. Chankong, V., Haimes, Y.Y.: Multiobjective decision making theory and methodology. Elsevier Science, New York (1983)
5. Czyzak, P., Jaszkiewicz, A.: Pareto simulated annealing: a metaheuristic technique for multiple-objective combinatorial optimization. J. Multi-Crit. Decis. Anal. **3**, 83–104 (1998)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
7. Dunn, O.J.: Multiple Comparisons Among Means. J. Am. Stat. Assoc. **56**, 52–64 (1961)
8. Fonseca, C.M., Fleming, P.J.: An overview of evolutionary algorithms in multiobjective optimization. Evol. Comput. **3**, 1–16 (1995)
9. Gandibleux, X., Mezdaoui, N., Fréville, A.: A tabu search procedure to solve multiobjective combinatorial optimization problems. In: Caballero, R., Steuer, R. (eds.) Proceedings Volume of MOPGP 96. Springer, Berlin (1996)
10. Goh, C.K., Ong, Y.S., Tan, K.C.: Multiobjective memetic algorithms. Springer, Berlin (2009)
11. Gore, A.: Some nonparametric tests and selection procedures for main effects in two-way layouts. Ann. Inst. Stat. Math. **27**(1), 487–500 (1973)
12. Gendreau, M., Potvin, J.Y., Bräysy, O., Hasle, G., Løkketangen, A.: Metaheuristics for the vehicle routing problem and its extensions: a categorized bibliography. In: Golden, B., Raghavan, S., Wasil, E. (eds.) The Vehicle Routing Problem—Latest Advances and New Challenges. Springer, New York (2008)
13. Gribkovskaia, I., Laporte, G., Shyshou, A.: The single vehicle routing problem with deliveries and selective pickups. Comput. Oper. Res. **35**, 2908–2924 (2008)
14. Gutiérrez-Jarpa, G., Marianov, V., Obreque, C.: A single vehicle routing problem with fixed delivery and optional collections. IIE Trans. **41**, 1067–1079 (2009)
15. Gutiérrez-Jarpa, G., Desaulniers, G., Laporte, G., Marianov, V.: A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows. Eur. J. Oper. Res. **206**, 341–349 (2010)
16. Hansen, M.P.: Tabu search for multiobjective optimization: MOTS. MCDM Conference (1997)
17. Hodges, J., Lehmann, E.: Estimation of location based on ranks. Ann. Math. Stat. **34**, 598–611 (1963)
18. Jaszkiewicz, A.: Genetic local search for multi-objective combinatorial optimization. Eur. J. Oper. Res. **137**(1), 50–71 (2002)
19. Jozefowiez, N., Semet, F., Talbi, E.G.: An evolutionary algorithm for the vehicle routing problem with route balancing. Eur. J. Oper. Res. **195**(3), 761–769 (2007)

20. Jozefowiez, N., Semet, F., Talbi, E.G.: Multi-objective vehicle routing problems. Eur. J. Oper. Res. **189**(2), 293–309 (2008)
21. Lee, T.-R., Ueng, J.-H.: A study of vehicle routing problem with load balancing. Int. J. Phys. Distrib. Logist. Manag. **29**, 646–648 (1998)
22. Lourenço H.R., Martin O.C., Stützle T.: Iterated local search. In: Glover F., Kochenberger G.A. (eds.) Handbook of Metaheuristics, pp. 321–353. Kluwer Academic Publishers, Boston (2003)
23. Montané, F.A.T., Galvão, R.D.: A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Comput. Oper. Res. **33**, 595–619 (2006)
24. Montgomery, D.: Design and analysis of experiments, 7th edn. Wiley, New York (2008)
25. Osman, I.H.: Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Ann. Oper. Res. **41**, 421–451 (1993)
26. Pacheco, J., Martí, R.: Tabu search for a multiobjective routing problem. J. Oper. Res. Soc. **57**(1), 29–37 (2006)
27. Penna, P., Subramanian, A., Ochi, L.: An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. J. Heuristics (2011). doi:10.1007/s10732-011-9186-y
28. Ribeiro, R., Lourenco, H.R.: A multi-objective model for a multi-period distribution management problem. In: Metaheuristic International Conference (MIC), 97–101 (2001)
29. Salhi, S., Nagy, G.: A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. J. Oper. Res. Soc. **50**, 1034–1042 (1999)
30. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: International Conference on Genetic Algorithm and Their Applications (1985)
31. Serafini, P.: Simulated annealing for multi-objective optimization problems. In: Tzeng, G.H., Wang, H.F., Wen, V.P., Yu, P.L. (eds.) Multiple Criteria Decision Making. Expand and Enrich the Domains of Thinking and Application, pp. 283–292. Springer, Berlin (1994)
32. Subramanian, A., Drummond, L., Bentes, C., Ochi, L., Farias, R.: A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. Comput. Oper. Res. **37**(11), 1899–1911 (2010)
33. Subramanian, A., Uchoa, E., Pessoa, A.A., Ochi, L.S.: Branch-and-cut with lazy separation for the vehicle routing problem with simultaneous pickup and delivery. Oper. Res. Lett. **39**(5), 338–341 (2011)
34. Süral, H., Bookbinder, J.H.: The single-vehicle routing problem with unrestricted backhauls. Networks **41**(3), 127–136 (2003)
35. Ulungu, E.L., Teghem, J.: Multi-objective combinatorial optimization problems: a survey. J. Multi-Crit. Decis. Anal. **3**, 83–104 (1994)
36. Ulungu, E.L., Teghem, J., Fortemps, Ph: Tuyttens: MOSA method: a tool for solving multiobjective combinatorial optimization problems. J. Multi-Crit. Decis. Anal. **8**, 221–236 (1999)
37. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**, 117–132 (2003)

Capítulo 12

Problema de Roteamento de Veículos Multiobjetivo com Coleta Seletiva

Luciana Pereira de Assis*, André Luiz Maravilha Silva,
Felipe Campelo Franca Pinto, Alessandro Vivas Andrade e Jaime Arturo Ramírez

Resumo: Este capítulo apresenta uma abordagem multiobjetivo para o problema de roteamento de veículos com coleta seletiva, cujos objetivos são a minimização dos custos das rotas e das demandas de coletas não atendidas. Propõe-se uma estrutura de dados que melhor se ajusta ao problema, estruturas de vizinhanças que exploram ambos objetivos do problema e um algoritmo que verifica a viabilidade de uma solução com menor custo computacional. Para solucionar o problema, são discutidas três meta-heurísticas: o ILS multiobjetivo (MOILS), NSGA-II e o método ϵ-Restrito, as quais são aplicadas à 14 instâncias contendo entre 50 e 199 consumidores. Os resultados indicam que o MOILS é superior aos outros algoritmos, obtendo valores médios para cobertura, hipervolume e cardinalidade significativamente melhores.

Palavras-chave: Problema de roteamento de veículos, Meta-heurística, Otimização multiobjetivo.

Abstract: *This chapter presents a multiobjective approach for the vehicle routing problem with optional collections, whose objectives are the minimization of the route costs and of the not fulfilled collection demands. It proposes a data structure that best suits the problem, neighborhood structures that exploit both goals of the problem and an algorithm that checks the feasibility of a solution with lower computational cost. To solve the problem, three metaheuristics are discussed: the multiobjective iterated local search (MOILS), NSGA-II and the ϵ-Constrained method, which are applied to fourteen instances containing between 50 and 199 customers. The results indicate that the MOILS outperforms the other approaches, obtaining significantly better average values for coverage, hypervolume and cardinality over the set of used test problems.*

Keywords: *Vehicle Routing problem, Metaheuristics, Multiobjective optimization.*

1. Introdução

O problema de roteamento de veículos (VRP, *Vehicle Routing Problem*) é um dos mais importantes e mais amplamente estudados no contexto da otimização combinatória, por englobar grande parte dos problemas reais de logística de transporte. Em geral, este problema consiste em definir rotas otimizadas que atendam às demandas de um conjunto de consumidores considerando uma frota finita de veículos capacitados.

Uma vez que existe um custo de transporte associado entre cada par de consumidores, e entre cada consumidor e o depósito, o objetivo do problema é definir rotas de menor custo total. Este problema é caracterizado por uma natureza combinatória, ou seja, o domínio da função a ser otimizada é dada pela permutação ideal de um conjunto de arestas que conectam os pares de consumidores e conectam os consumidores ao depósito. Com o aumento no número de consumidores, torna-se extremamente difícil encontrar soluções ótimas a partir de métodos exatos.

Assim sendo, diversas meta-heurísticas têm sido propostas para solucionar problemas para os quais métodos exatos não são capazes de fornecer soluções em um tempo computacional viável. Dentre estas, destacam-se os métodos baseados em buscas locais, como a Busca Local Iterativa (ILS, *Iterated Local Search*) (Lourenço et al., 2003), e os métodos baseados em população como os Algoritmos Genéticos (Holland, 1992).

Em alguns problemas de roteamento encontrados na literatura as demandas dos consumidores podem ser tanto de entrega quanto de coleta de mercadorias. O estudo deste fluxo de material está inserido no contexto da logística reversa, que planeja toda a atividade de entrega dos produtos e, também, o caminho inverso, ou

*Autor para contato: lpassis@ufvjm.edu.br

seja, o retorno dos produtos à central de distribuição. Algumas indústrias estão cada vez mais preocupadas com o gerenciamento do retorno do material consumido para o depósito a fim de reciclá-los, remanufaturá-los ou agregar novos valores aos mesmos.

O problema de roteamento de veículos com coletas e entregas simultânea (VRPSPD, *Vehicle Routing Problem with Simultaneous Pickup and Delivery*) é um problema básico de logística reversa. O objetivo deste problema é minimizar os custos de transporte, atendendo a todas as demandas de coleta e entrega simultaneamente (Subramanian et al., 2011).

O problema de roteamento de veículos com serviço de coleta e entrega, proposto por Sural & Bookbinder (2003), ilustra situações onde o atendimento das demandas de coleta não é obrigatório. Este problema, denominado *Single-Vehicle Routing Problem with Unrestricted Backhauls*, associa um lucro a cada demanda de coleta, caso esta seja atendida. O objetivo consiste em definir um percurso que atenda a todas as demandas de entrega e, também, realize as coletas que forem consideradas lucrativas, minimizando uma função objetivo denominada custo líquido (custo total da rota subtraído do lucro total obtido através das coletas realizadas). Um algoritmo exato (*Branch-and-Bound*) é aplicado para solucionar instâncias do problema com poucos consumidores. Gutiérrez-Jarpa et al. (2010) apresentam uma abordagem exata (*Branch-and-Price*) para solucionar problemas de roteamento de veículos com entrega e coleta seletiva e janela de tempo. Outro problema similar é apresentado por Gribkovskaia et al. (2007), denominado *Single Vehicle Routing Problem With Delivery and Selective Pickups*. Neste problema, os consumidores podem ser visitados duas vezes; uma vez para realização das entregas e outra para coletas. O objetivo do problema também é a minimização do custo líquido.

Apesar da natureza multiobjetivo destes problemas, os trabalhos citados apresentam uma modelagem mono-objetivo e fazem uso de um lucro associado às demandas para diferenciá-las. Contudo, o estudo de problemas de roteamento de veículos com múltiplos objetivos vem apresentando maior destaque nos últimos anos (Jozefowiez et al., 2008), mostrando que em algumas situações reais é necessário visualizar diversos cenários, dados os diversos objetivos do problema, para se tomar uma decisão. Assim sendo, ao modelar problemas de maneira multiobjetivo, estes se tornam mais interessantes para aplicações práticas.

Este capítulo aborda o problema de roteamento de veículos multiobjetivo com coleta seletiva (MOVRPSP, *Multiobjective Vehicle Routing Problem with Selective Pickup*) cujos objetivos são: minimização dos custos de transporte e minimização das demandas de coletas não atendidas. Esta formulação pode ser vista como uma generalização do VRPSPD, uma vez que a restrição de atendimento de todas as demandas de coleta se transforma em um objetivo do problema.

Para avaliar diferentes estratégias de resolução do MOVRPSP, este trabalho apresenta três algoritmos: Busca Local Iterativa Multiobjetivo (MOILS, *Multiobjective Iterated Local Search*) (Geiger, 2006), Algoritmo Genético de Ordenação Não-Dominante II (NSGA-II, *Non-Dominated Sorting Genetic Algorithm-II*) (Deb et al., 2002) e heurística P_ϵ-ILS, baseada no método ϵ-Restrito (Chankong & Haimes, 1983). Estes algoritmos exploram as características dos algoritmos baseados em busca local, em população e nos métodos tradicionais de resolução de problemas multiobjetivo, respectivamente.

Este capítulo está organizado como se segue. A Seção 2 apresenta a definição do problema e sua modelagem matemática multiobjetivo. Em seguida, na Seção 3, as estruturas de dados e os algoritmos aplicados na resolução do problema são apresentados. Os resultados computacionais são mostrados na Seção 4 e, por fim, na Seção 5 são feitas as conclusões deste trabalho.

2. Definição do Problema

O problema de roteamento de veículos multiobjetivo com coleta seletiva (MOVRPSP) consiste em definir um conjunto de rotas que atenda a todas as demandas de entrega, minimizando os custos de transporte e o número de coletas não realizadas. Cada consumidor deve ser visitado uma única vez e o atendimento parcial das demandas não é permitido.

Os dois objetivos a serem otimizados são conflitantes, ou seja, melhorias em um provocam degradações no outro, não existindo uma solução que seja ótima para ambos os objetivos simultaneamente. Assim, não é uma tarefa trivial identificar se uma determinada solução é de melhor qualidade em relação a outra. Isto leva ao conceito de dominância de Pareto (Definição 1), sendo a solução do problema um conjunto de soluções não dominadas. Neste trabalho existe uma diferença entre um conjunto de soluções não dominadas e um conjunto Pareto-ótimo. Um conjunto de soluções não dominadas é definido no contexto de uma amostra do espaço de busca, enquanto que o conjunto ótimo de Pareto é definido em relação a todo o espaço de busca (Definições 2 e 3).

Definição 1 (Dominância de Pareto) *Dado um vetor objetivo* $Z(x) = (f_1(x), \ldots, f_m(x))$*, com* m *funções objetivo para minimização. Uma solução* x' *domina uma outra solução* x'' *se, e somente se, para todos os*

objetivos $f_i(x') \leq f_i(x'')$ *com* $i = 1, \ldots, m$ *e, existe pelo menos um objetivo* $f_i(x') < f_i(x'')$. *Esta relação de dominância é representada pela notação* $x' \prec x''$ *(Coello Coello et al., 2007).*

Definição 2 (Solução Pareto-Ótima) x^* *é uma solução Pareto-Ótima se não existe qualquer outra solução* x *tal que* $x \prec x^*$, *ou seja,* x^* *não é dominada por nenhuma outra solução (Coello Coello et al., 2007).*

Definição 3 (Conjunto Pareto-Ótimo) *O conjunto* $\mathcal{X}^*$ *é um conjunto Pareto-Ótimo se é composto por todas as soluções Pareto-Ótimas. O conjunto-imagem* $\mathcal{Y}^*$ *associado ao conjunto Pareto-Ótimo é denominado fronteira Pareto-Ótima (Coello Coello et al., 2007).*

2.1 Modelagem matemática

A modelagem matemática apresentada nesta seção é uma adaptação da modelagem proposta por Montané & Galvão (2006) para o VRPSPD.

Seja $V = \{1 \ldots n\}$ o conjunto de consumidores e V_0 o conjunto de consumidores mais o depósito ($V_0 = V \cup \{0\}$). Um custo $c_{ij} \geq 0$ é associado a cada arco que conecta os pares de consumidores e os consumidores ao depósito. As demandas de coleta e entrega do consumidor j são representadas por p_j e d_j, respectivamente. O número de veículos disponíveis é dado por $\overline{k}$, sendo Q a capacidade destes. O parâmetro y_{ij} é o somatório das cargas coletadas entre o depósito e o nó i (inclusive) dirigida ao nó $j \in V_0$. Já z_{ij} é o somatório das demandas de entrega dos consumidores visitados após o nó i (exclusive) e dirigida ao nó $j \in V_0$. A variável de decisão binária x_{ij}^k assume valor 1 se a aresta (i, j) é transposta pelo veículo k e 0, caso contrário. A variável de decisão binária ℓ_j recebe valor unitário se a demanda de coleta do consumidor j é satisfeita e 0, caso contrário.

O modelo matemático pode ser expresso por:

$$\text{Minimize} \begin{cases} \sum_{k=1}^{\overline{k}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k \\ \sum_{j=1}^{n} p_j (1 - \ell_j) \end{cases} \tag{1}$$

Sujeito a:

$$\sum_{i=0}^{n} \sum_{k=1}^{\overline{k}} x_{ij}^k = 1, \qquad j = 1, \ldots, n \tag{2}$$

$$\sum_{i=0}^{n} x_{ij}^k - \sum_{i=0}^{n} x_{ji}^k = 0, \qquad j = 0, \ldots, n \text{ e } k = 0, \ldots \overline{k} \tag{3}$$

$$\sum_{j=1}^{n} x_{0j}^k \leq 1, \qquad k = 1, \ldots, \overline{k} \tag{4}$$

$$\sum_{i=0}^{n} z_{ij} - \sum_{i=0}^{n} z_{ji} = d_j, \qquad \forall j \neq 0 \tag{5}$$

$$\sum_{i=0}^{n} y_{ij} - \sum_{i=0}^{n} y_{ji} = p_j \ell_j, \qquad \forall j \neq 0 \tag{6}$$

$$y_{ij} + z_{ij} \leq Q \sum_{k=1}^{\overline{k}} x_{ij}^k, \qquad i, j = 0, \ldots, n \tag{7}$$

$$x_{ij}, \ell_j \in \{0, 1\}, \qquad i, j = 0, \ldots, n \tag{8}$$

$$y_{ij}, z_{ij} \geq 0, \qquad i, j = 0, \ldots, n \tag{9}$$

As duas funções objetivo, dadas na Equação (1), representam o custo total das rotas e as coletas não realizadas, ambas devem ser minimizadas.

Estas funções objetivo devem satisfazer algumas restrições. Cada ponto de demanda deve ser visitado por um único veículo (Equação (2)). A Equação (3) representa a restrição de conservação do fluxo. A Equação (4) indica que $\overline{k}$ veículos, no máximo, podem ser utilizados. A Equação (5) obriga a satisfação de todas as demandas de entrega. A Equação (6) garante o atendimento da demanda de coleta do consumidor j quando a variável de decisão ℓ_j assumir valor unitário. A Equação (7) define que as demandas devem ser transportadas

nos arcos incluídos na solução e ainda impõem um limite para a carga total transportada pelo veículo. A Equação (8) representa a restrição de integralidade e, finalmente, a Equação (9) representa as restrições de não-negatividade para demandas de coleta e entrega.

3. Estratégias de Solução

Nesta seção serão apresentadas as ferramentas e métodos utilizados para resolução de problemas de roteamento de veículos multiobjetivo com coleta seletiva.

3.1 Estrutura de dados

A maneira como os dados de entrada de um problema são armazenados interferem diretamente na eficiência dos algoritmos implementados. Os problemas de roteamento de veículos podem ser representados através de um grafo completo, onde os consumidores e o depósito compõem o conjunto de vértices e as arestas representam as conexões entre consumidores e entre consumidores e depósito. Existem diversas formas de representar um grafo e este capítulo apresenta uma nova estrutura a fim de unir as vantagens de duas estruturas bem tradicionais: matriz de adjacência e lista de adjacência.

Para a nova estrutura, foi implementada uma matriz de adjacência, onde cada elemento (i, j) da matriz é uma célula de uma lista duplamente encadeada composta por: vértice adjacente a i, o custo da aresta (i, j) e dois ponteiros. Partindo da diagonal principal da matriz, os ponteiros criam uma lista duplamente encadeada, sendo que os ponteiros da direita e esquerda ordenam crescente e decrescente, respectivamente, os vértices adjacentes a i pelo custo da aresta (i, j). A Figura 1 ilustra como os dados são mantidos na estrutura de dados proposta. Nesta Figura foi representado apenas um dos ponteiros para facilitar a visualização. A matriz, à direita, apresenta os vértices e, entre parênteses, os custos das arestas.

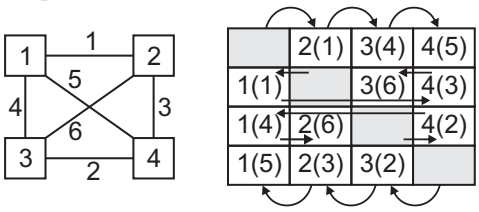


Figura 1. Nova estrutura de dados para armazenar um grafo.

3.2 Representação de uma solução

A representação de uma solução para problemas de roteamento de veículos pode ser feita através de um conjunto de vetores, cada um indicando uma rota e armazenando os consumidores na ordem em que devem ser visitados.

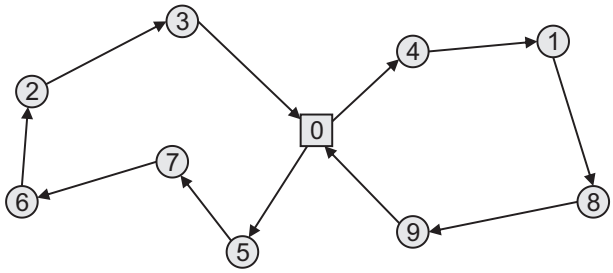
Porém, nos problemas de roteamento multiobjetivo com coleta seletiva, além da ordem em que os consumidores são visitados, deve-se armazenar quais demandas de coleta serão atendidas. Para isso, utiliza-se um vetor binário informando, para cada consumidor, o estado de sua demanda de coleta, atendida ou não atendida. A Figura 2 apresenta uma solução para o problema de roteamento de veículos multiobjetivo com coleta seletiva. Nessa ilustração, os dois primeiros vetores mostram as rotas que compõem a solução, indicando a ordem em que os consumidores devem ser visitados, sempre começando e terminado no depósito, representado pelo número 0. O último vetor informa o estado da demanda de coleta de cada consumidor, sendo representada pelo valor 1, demanda de coleta atendida, ou 0, demanda de coleta não atendida.

3.3 Análise da viabilidade das rotas

Durante a construção, ou na alteração de uma solução de um problema de roteamento de veículos com serviços de coleta e entrega de mercadorias, é necessário verificar se o atendimento de uma demanda não extrapola a capacidade máxima do veículo.

Uma implementação simples para este procedimento pode ser feita calculando-se a carga do veículo após o atendimento de cada demanda dos consumidores da rota modificada, verificando, assim, se a capacidade do veículo não será extrapolada com a inserção das demandas de entrega e coleta do novo consumidor. Apesar de sua fácil implementação, o desempenho do algoritmo para geração de rotas válidas pode ser prejudicado, já que este processo tem complexidade $O(l)$, sendo l o comprimento da rota.

Como a avaliação da viabilidade é realizada inúmeras vezes, qualquer aperfeiçoamento realizado melhora o desempenho dos algoritmos que o utilizam. Assim, é proposto o armazenamento de algumas informações



Rota 1: [0-4-1-8-9-0] - **Rota 2:** [0-5-7-6-2-3-0]
Estado Coleta: [1-1-0-1-0-0-1-1-1]

Figura 2. Representação de uma solução para o problema de roteamento de veículos multiobjetivo com coleta seletiva.

adicionais sobre a carga do veículo ao longo da rota, tornando mais eficiente esta análise de viabilidade. O Algoritmo 1 apresenta o pseudo-código do procedimento proposto para análise de viabilidade das rotas.

Algoritmo 1: Análise de viabilidade.

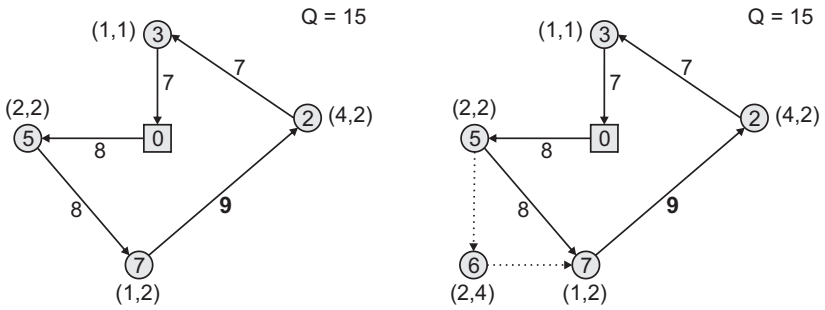
```
   // Verifica a carga do veículo ao partir e retornar ao depósito
1  se (SomaEntregas + E(x)) > Q então
2      retorna falso;
3  se (SomaColetas + C(x)) > Q então
4      retorna falso;
   // Verifica se o local onde o consumidor será inserido está antes do ponto de maior
      carga
5  se p ≤ maxPosicao então
6      se (infoCarga[maxPosicao] + C(x)) > Q então
7          retorna falso;
8      se E(x) > C(x) então
9          para i = 0 até p − 1 faça
10             se (infoCarga[i] + E(x)) > Q então
11                 retorna falso;
12 senão
       // Consumidor será inserido depois do ponto de maior carga
13     se (infoCarga[maxPosicao] + E(x)) > Q então
14         retorna falso;
15     se E(x) < C(x) então
16         para i = p até l − 1 faça
17             se (infoCarga[i] + C(x)) > Q então
18                 retorna falso;
19 retorna verdadeiro;
```

Sejam $E(x)$ e $C(x)$ as demandas de entrega e coleta do consumidor x, respectivamente. O Algoritmo 1 analisa, inicialmente, se a carga do veículo não extrapola a capacidade máxima (Q), quando o veículo parte do depósito e quando ele retorna ao depósito (linhas 2–7). Em seguida, verifica se a posição onde o veículo apresenta maior carga está antes ou depois da posição p que se deseja inserir o consumidor x (linha 9).

Caso o consumidor seja inserido antes da posição na qual o veículo atinge o ponto de maior carga ($maxPosicao$), então a carga máxima do veículo será acrescida apenas da demanda de coleta do consumidor. O vetor $infoCarga$ armazena a carga presente no veículo em cada posição da rota (linhas 6–7). A Figura reffig:viabilidade11.a exemplifica uma rota na qual o depósito está representado por um quadrado, os consumidores representados por circunferências, as demandas de entrega e coleta de cada consumidor representadas entre parênteses, respectivamente. O valor nas arestas indica a carga presente no veículo

($infoCarga$), sendo a carga máxima do veículo destacada em negrito. Na Figura 3.b, pretende-se inserir um vértice (6) antes do ponto no qual o veículo atinge sua carga máxima (9). Na situação ilustrada, a carga máxima do veículo será acrescida apenas da demanda de coleta do consumidor 6, $C(6) = 4$.

Se a carga máxima do veículo acrescido da demanda de coleta do consumidor inserido for maior que a capacidade do veículo, então o consumidor x não pode ser inserido na posição p (linhas 6–7). Caso contrário, o algoritmo verifica se a demanda de entrega $E(x)$ é menor que a demanda de coleta $C(x)$ (linhas 8–11). Desta maneira, garante-se que a carga máxima do veículo permanecerá na mesma aresta e, assim, a inserção no consumidor não inviabiliza a solução. Senão, torna-se necessário verificar a carga do veículo no trajeto entre o depósito e o vértice inserido, pois a carga máxima do veículo pode agora estar neste intervalo.



(a) Solução Inicial. (b) Inserção de um novo vértice (6).

Figura 3. Análise de viabilidade.

Quando o ponto de inserção p estiver após o ponto de maior carga, o procedimento é análogo (linhas 12–18). No fim, se o veículo não extrapola sua capacidade máxima, então é retornado *verdadeiro*, indicando que a inserção do consumidor na posição p é viável (linha 19).

Neste procedimento, quando $p \leq maxPosicao$ e $E(x) \leq C(x)$ ou $p > maxPosicao$ e $C(x) \leq E(x)$, tem-se o melhor caso, com complexidade $O(1)$, pois ele não executa nenhum bloco de repetição. Apenas no pior caso o Algoritmo 1 apresenta complexidade $O(l)$.

O pior caso ocorre em duas situações: (i) quando se trata da inserção de um consumidor x imediatamente anterior à posição de maior carga, a posição de maior carga ocorre no final da rota e a demanda de entrega do consumidor x é maior que a demanda de coleta x; (ii) quando ocorre a inserção de um consumidor x logo após a posição de maior carga e a posição de maior carga ocorre no início da rota, e a demanda de coleta do consumidor x é maior que a demanda de coleta de x. Em ambos casos, as linhas 9–11 ou 16–18 serão executadas aproximadamente l vezes.

3.4 Busca local iterativa multiobjetivo (MOILS)

A meta-heurística busca local iterativa (ILS, *Iterated Local Search*) consiste em aprimorar o procedimento de busca local gerando novas soluções de partidas através de mecanismos de perturbações aplicados à solução ótima local (Lourenço et al., 2003).

Ao se tratar de problemas de roteamento de veículos mono-objetivo com serviços de coleta e entrega, heurísticas baseadas no ILS apresentam bons resultados, sem a necessidade de ajuste de muitos parâmetros (Subramanian et al., 2010). Além disso, adaptações desta meta-heurística, quando utilizadas para resolução de outros problemas de otimização combinatória multiobjetivo, também apresentam bons resultados (Geiger, 2006). Assim, este capítulo irá apresentar uma heurística baseada na meta-heurística ILS para resolução de problemas multiobjetivo utilizando o conceito de Dominância de Pareto, possibilitando, assim, a resolução de problemas de otimização combinatória multiobjetivo, em especial o MOVRPSP. À adaptação proposta é dado o nome de busca local iterativa multiobjetivo (MOILS, *Multiobjective Iterated Local Search*).

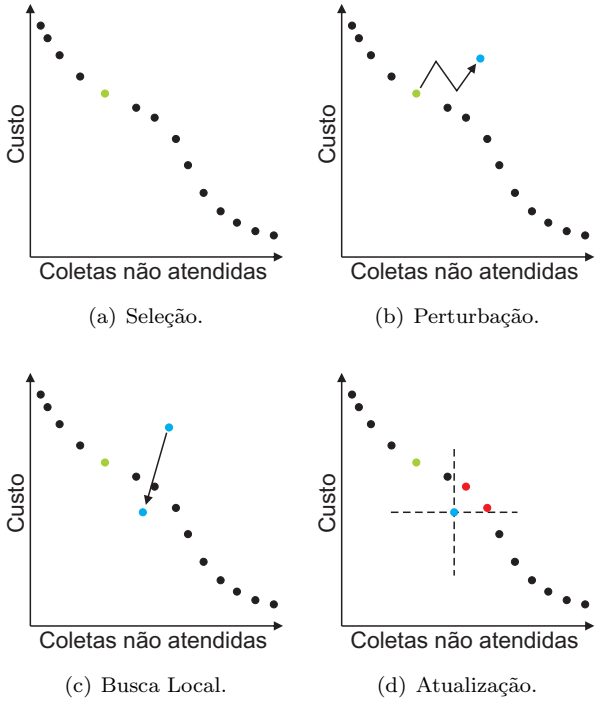
O algoritmo proposto possui 5 etapas: (i) geração de uma solução inicial; (ii) seleção de uma solução da fronteira de Pareto aproximada para ser explorada; (iii) busca local, que retorna uma solução melhorada; (iv) perturbação, que modifica uma solução; e (v) atualização da fronteira de Pareto aproximada dada uma solução para o problema.

O funcionamento desta heurística é ilustrado no Algoritmo 2. O MOILS inicia seu funcionamento com a geração de soluções iniciais (linha 1). As soluções geradas são inseridas no conjunto $Front$ que mantém um conjunto de soluções não dominadas. Em seguida, são executados $maxIter$ iterações, onde uma solução em $Front$ é selecionada e o bloco de repetição principal do algoritmo (linhas 6–15) é executado. Neste bloco, as etapas de perturbação (linha 7) e busca local (linha 8), semelhantes ao ILS, são aplicadas à solução escolhida.

Dada a solução gerada, o conjunto *Front* é atualizado (linha 9). Se a solução gerada após a perturbação e busca local for inserida em *Front* (indicada pela variável lógica *inserido*), então *cont* é reiniciado (linha 11); senão *cont* é incrementado de uma unidade (linha 14). O parâmetro *maxCount* representa o número máximo de vezes que uma solução será explorada sem a inserção de uma solução não dominada no conjunto *Front*.

Algoritmo 2: Algoritmo MOILS.

```
   Entrada: maxIter
   Entrada: maxCount
1  Front ← gerarSoluçõesIniciais();
2  iter ← 0;
3  enquanto iter < maxIter faça
4      s' ← selecionar(Front);
5      cont ← 0;
6      enquanto cont < maxCount faça
7          s'' ← perturbação(s');
8          s'' ← buscaLocal(s'');
9          inserido ← atualizar(Front, s'');
10         se inserido então
11             cont ← 0;
12             s' ← s'';
13         senão
14             cont ← cont + 1;
15     iter ← iter + 1;
16 retorna Front;
```



(a) Seleção. (b) Perturbação.

(c) Busca Local. (d) Atualização.

Figura 4. Etapas do algoritmo MOILS.

A Figura 4 ilustra as etapas do algoritmo MOILS. Na Figura 4.a, uma solução contida na fronteira de Pareto aproximada é selecionada. Nas Figuras 4.b e 4.c, esta solução passa pelos procedimentos de perturbação e busca local, respectivamente. Por fim, a fronteira é atualizada com a nova solução gerada, Figura 4.d.

Os procedimentos de geração da solução inicial, seleção, busca local, perturbação e atualização são detalhados nas subseções seguintes.

3.4.1 Solução inicial

A solução inicial para o MOVRPSP consiste em encontrar uma solução para o problema de roteamento de veículos com coleta e entrega simultânea (VRPSPD) e para o problema de roteamento de veículo capacitado (CVRP, *Capacitated Vehicle Routing Problem*), sendo estas soluções extremos da fronteira de Pareto aproximada. A solução para ambos problemas é obtida aplicando o algoritmo baseado no ILS, proposto por Penna et al. (2012). Assim sendo, o ILS é executado duas vezes, sendo uma para solucionar o VRPSPD, no qual todas as demandas de coleta do problema são consideradas e outra para solucionar o CVRP, desconsiderando as demandas de coleta dos consumidores.

As soluções encontradas representam os pontos extremos da fronteira de Pareto aproximada, conforme Figura 5. Uma vez encontrados os extremos e sendo estes de boa qualidade, os demais passos do algoritmo consistem em encontrar as soluções contidas no intervalo entre os dois pontos extremos já definidos.

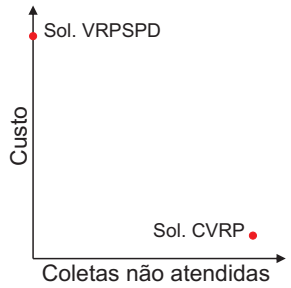


Figura 5. Pontos extremos da fronteira de Pareto aproximada.

3.4.2 Seleção

O procedimento de seleção utilizado é baseado na *crowding-distance* do algoritmo NSGA-II, proposto por Deb et al. (2002). Quanto mais distante das outras soluções uma determinada solução estiver, maior a probabilidade desta ser escolhida. Assim, é possível realizar a exploração do espaço de soluções de forma mais inteligente, onde é dado prioridade a regiões pouco exploradas, evitando a concentração da busca em determinados nichos.

Neste trabalho, o cálculo da *crowding-distance* é um pouco diferente do realizado no NSGA-II. No MOILS as soluções presentes nos extremos da fronteira têm a *crowding-distance* igual a duas vezes a distância entre elas e a solução mais próxima, enquanto no NSGA-II, as soluções presentes nas extremidades da fronteira recebem um valor infinito. Esta alteração é necessária já que após a geração das soluções iniciais, a fronteira é composta apenas pelas duas soluções que se encontram nas extremidades da fronteira (solução do CVRP e solução do VRPSPD). Assim, para reduzir a possibilidade de serem demasiadamente selecionadas durante todo o procedimento, o valor da *crowding-distance* dessas soluções precisa ser reduzido.

3.4.3 Busca Local

A etapa de busca local realiza a exploração da vizinhança de uma solução com o objetivo de encontrar outras soluções não dominadas para o problema. No algoritmo apresentado, foi utilizado o método busca em descida em vizinhança variável aleatória (RVND, *Randomized Variable Neighborhood Descent*), proposta por Penna et al. (2012), que explora uma solução aplicando aleatoriamente diferentes estruturas de vizinhanças.

Neste trabalho, foram utilizadas um conjunto de estruturas de vizinhanças que podem ser divididas em dois grupo: inter-rotas e intra-rota. As estruturas de vizinhança inter-rotas são aquelas que envolvem duas ou mais rotas no movimento. Já as intra-rotas envolvem apenas uma única rota. A busca realizada em todas as estruturas de vizinhança foi baseada na estratégia melhor aprimorante que avalia todas as soluções vizinhas antes do movimento ser aplicado. As Subseções 3.4.3.1 e 3.4.3.2 descrevem estas estruturas.

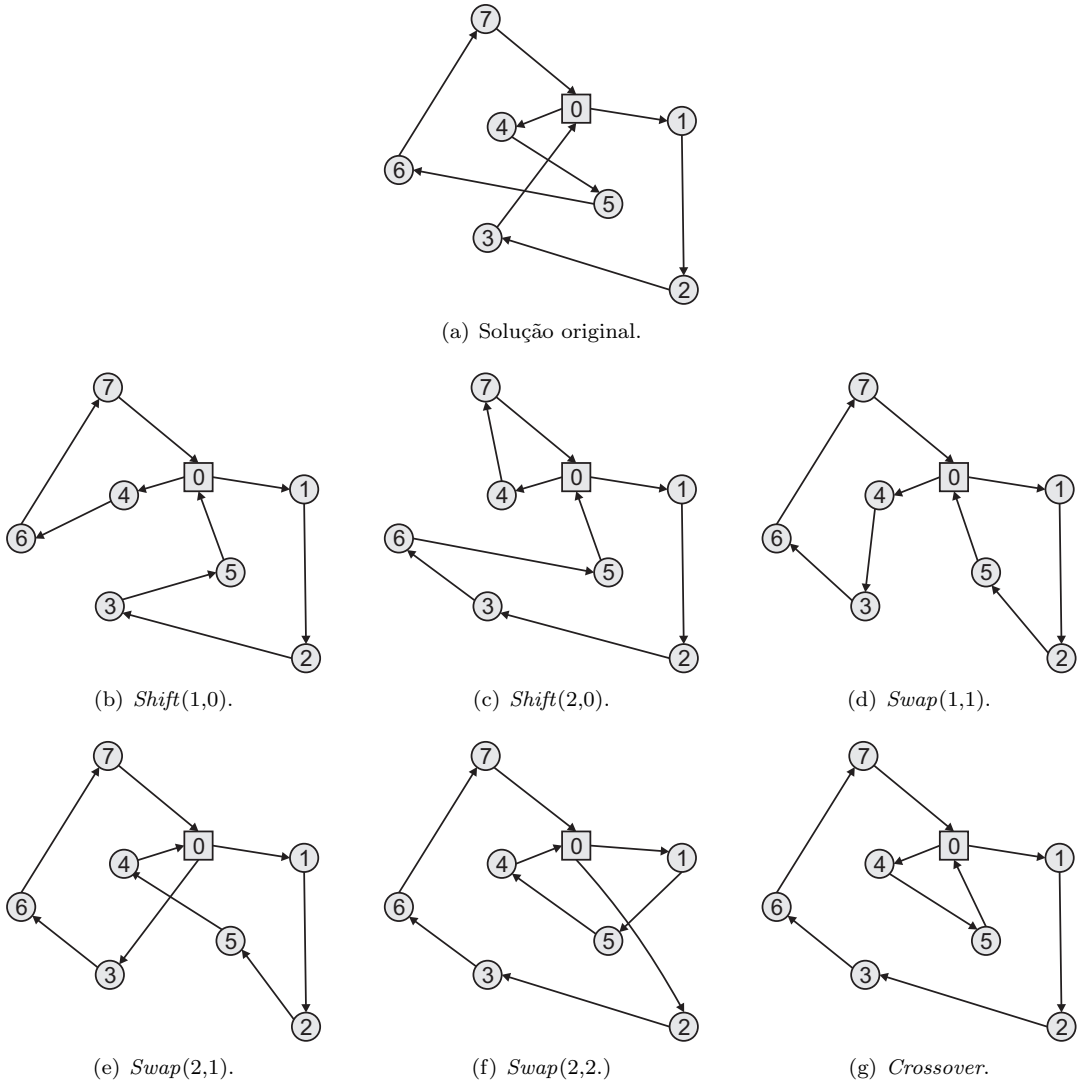
3.4.3.1 Estruturas de vizinhança inter-rotas

Foram implementadas dois tipos de estruturas de vizinhança inter-rotas: que exploram soluções levando em consideração apenas o objetivo custo de transporte e outras que consideram, além do custo de transporte, o objetivo de minimização das demandas de coleta não atendidas.

As seguintes buscas locais visam a redução do custo de transporte e podem ser aplicadas a diversos problemas de roteamento mono-objetivo.

- ***Shift*(1,0):** um consumidor é transferido de uma rota para outra rota. Na Figura 6.b, o consumidor 5 é removido de sua rota e inserido em outra rota.

- **Shift(2,0):** dois consumidores adjacentes são transferidos de uma rota para outra rota, continuando adjacentes, mas não necessariamente na mesma ordem. Na Figura 6.c, os consumidores 5 e 6 são removidos e inseridos em outra rota.
- ***Swap*(1,1):** permutação entre dois consumidores de duas rotas distintas. Na Figura 6.d, os consumidores 3 e 5 são permutados.
- ***Swap*(2,1):** permutação de dois consumidores adjacentes de uma rota com um consumidor de outra rota. Os dois consumidores adjacentes removidos devem permanecer adjacentes quando inseridos na outra rota, porém não necessariamente na mesma ordem. Na Figura 6.e, o consumidor 3 é trocado com os consumidores 4 e 5.
- ***Swap*(2,2):** permutação de dois consumidores adjacentes de uma rota com outros dois consumidores, também adjacentes, de outra rota. Assim como na Swap(2,1), os consumidores adjacentes devem permanecer adjacentes mas não necessariamente na mesma ordem. Na Figura 6.f, os consumidores 4 e 5 são trocados com os consumidores 2 e 3.
- ***Crossover*:** uma aresta (i, j) de uma rota r_1 e outra aresta (i', j') de uma rota r_2 são removidas. Em seguida as arestas (i, j') e (i', j) são inseridas. Na Figura 6.g, as arestas $(3, 0)$ e $(5, 6)$ são removidas e as arestas $(3, 6)$ e $(5, 0)$ são inseridas.
- ***TryEmptyRoute*:** remove os consumidores de uma determinada rota e tenta reinseri-los nas demais rotas pertencentes à solução a fim e reduzir o número de veículos utilizados na solução.



(a) Solução original.

(b) *Shift*(1,0).

(c) *Shift*(2,0).

(d) *Swap*(1,1).

(e) *Swap*(2,1).

(f) *Swap*(2,2.)

(g) *Crossover*.

Figura 6. Estruturas de vizinhança inter-rotas.

Devido à natureza multiobjetivo do MOVRPSP, proposto neste trabalho, faz-se necessária a utilização de estruturas de vizinhança que atuem nos outros objetivos e não apenas no custo de transporte. Este capítulo apresenta duas estruturas inter-rotas que atuam tanto na redução do custo de transporte quanto na minimização das demandas de coletas não atendidas. O objetivo é melhorar a diversidade das soluções e aproximá-las da fronteira de Pareto ótima.

Estas estruturas de vizinhanças são adaptações da *Shift*(1,0) e *Swap*(1,1), descritas anteriormente. O procedimento é similar, modificando apenas o estado da demanda de coleta (satisfeita ou não satisfeita) dos consumidores envolvidos no movimento. Assim, ao invés de retornar uma única solução, retornarão um conjunto de soluções não dominadas.

- ***MOShift*(1,0):** um consumidor é transferido de uma rota para outra rota. A transferência é feita considerando o estado de coleta atual do consumidor e, em seguida, a busca local é aplicada novamente alterando o estado de coleta do consumidor envolvido no movimento. Logo, para cada consumidor pertencente à solução, duas soluções vizinhas são encontradas.

- ***MOSwap*(1,1):** permutação entre dois consumidores de duas rotas distintas. A permutação é feita considerando os estados atuais das demandas de coleta dos consumidores e alternando estes estados. Logo, para cada consumidor pertencente à solução, quatro novas soluções vizinhas são geradas: uma considerando os estados atuais das demandas de coleta de ambos consumidores, a segunda alterando o estado da demanda de coleta de um dos consumidores, a terceira alterando o estado do outro consumidor e, por fim, alterando os estados das demandas de coletas dos dois consumidores envolvidos no movimento.

A Figura 7 exemplifica os movimentos *MOShift*(1,0) e *MOSwap*(1,1). Nesta figura, o valor nas arestas representa a carga do veículo naquele ponto e os valores entre parênteses indicam a demanda de entrega e coleta de cada consumidor, respectivamente. As Figuras 7.b e 7.c mostram as duas possíveis soluções retornadas pela estrutura de vizinhança *MOShift*(1,0). Na Figura 7.b, o consumidor 3 é realocado em outra rota desconsiderando a sua demanda de coleta e, na Figura 7.c, o consumidor 3 é realocado considerando o atendimento desta demanda. As Figuras 7.d, 7.e, 7.f e 7.g representam as quatro soluções geradas pela estrutura de vizinhança *MOSwap*(1,1). Na Figura 7.d, os consumidores 3 e 5 são permutados desconsiderando a demanda de coleta de ambos. Na Figura 7.e, estes mesmos consumidores são permutados desconsiderando a demanda de coleta do consumidor 3 e, na Figura 7.f, desconsiderando a demanda de coleta do consumidor 5. Na Figura 7.g, estes consumidores são permutados considerando o atendimento da demandas de coleta de ambos. Nas duas estruturas de vizinhança, dadas as soluções geradas, apenas as soluções não dominadas são retornadas.

Para evitar que movimentos inter-rota sejam executados desnecessariamente, uma estrutura de dados auxiliar, proposta por Penna et al. (2012), foi utilizada indicando se uma determinada busca local poderá ser aplicada à uma rota específica. Se, em uma iteração anterior, a estrutura de vizinhança i foi aplicada à rota j e não houve nenhuma melhoria da solução, então a estrutura de vizinhança i fica desabilitada para a rota j até que esta rota sofra alguma modificação.

3.4.3.2 Estruturas de vizinhança intra-rota.

Um dos objetivos das estruturas de vizinhança intra-rota é reduzir os custos de transporte das rotas. Assim, quando as estruturas de vizinhança inter-rotas não puderem gerar soluções na fronteira de Pareto aproximada, aplica-se, então, as estruturas de vizinhança intra-rota a fim de reduzir ainda mais os custos de transporte. As estruturas de vizinhança intra-rota implementadas são descritas a seguir.

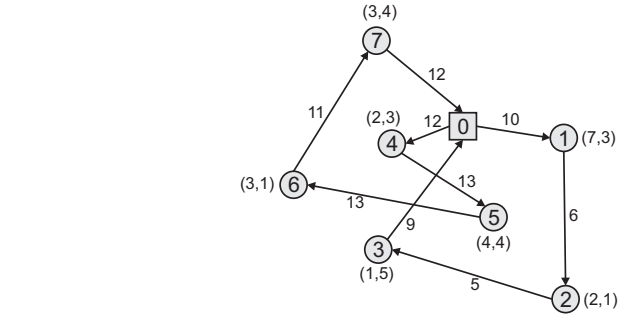
- ***Or-Opt*:** um consumidor é inserido em outra posição da mesma rota. Na Figura 8.b, o consumidor 5 é alterado de posição.

- ***Or-Opt* 2:** dois consumidores adjacentes são removidos e inseridos em outra posição da mesma rota. Na Figura 8.c, os consumidores 4 e 5 são removidos e inseridos em outra posição.

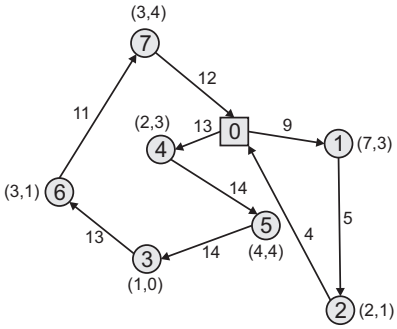
- ***Or-Opt* 3:** três consumidores adjacentes são removidos e inseridos em outra posição da mesma rota. Na Figura 8.d, os consumidores adjacentes 2, 3 e 4 são removidos e inseridos em outra posição.

- **2-*Opt*:** duas arestas (i, j) e (i', j') não consecutivas da mesma rota são removidas. Em seguida as arestas (i, i') e (j, j') são inseridas. Na Figura 8.e, as arestas $(3, 4)$ e $(5, 6)$ são removidas e as arestas $(3, 5)$ e $(4, 6)$ são inseridas.

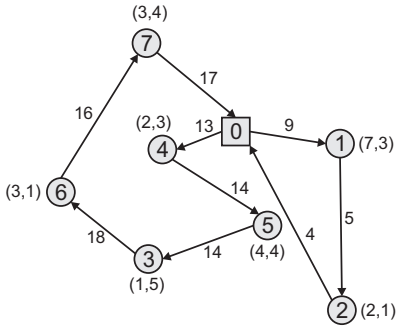
- ***Exchange*:** realiza a permutação entre dois consumidores de uma mesma rota. Na Figura 8.f, os consumidores 3 e 6 são permutados.



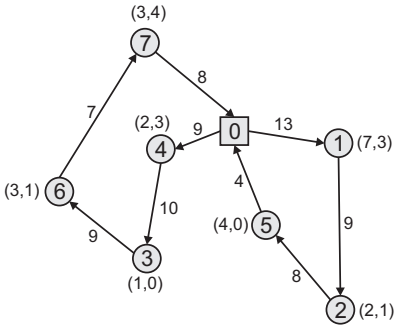
(a) Solução original.



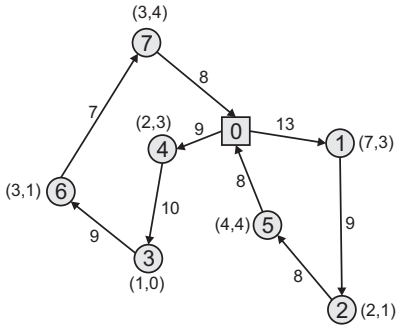
(b) *MOShift*(1,0) Solução 1.



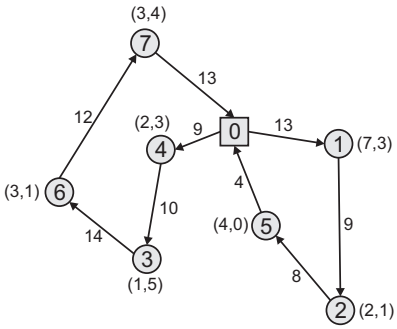
(c) *MOShift*(1,0) Solução 2.



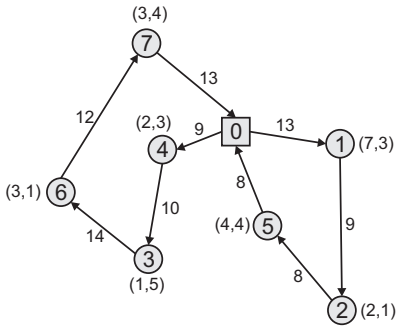
(d) *MOSwap*(1,1) Solução 1.



(e) *MOSwap*(1,1) Solução 2.



(f) *MOSwap*(1,1) Solução 3.

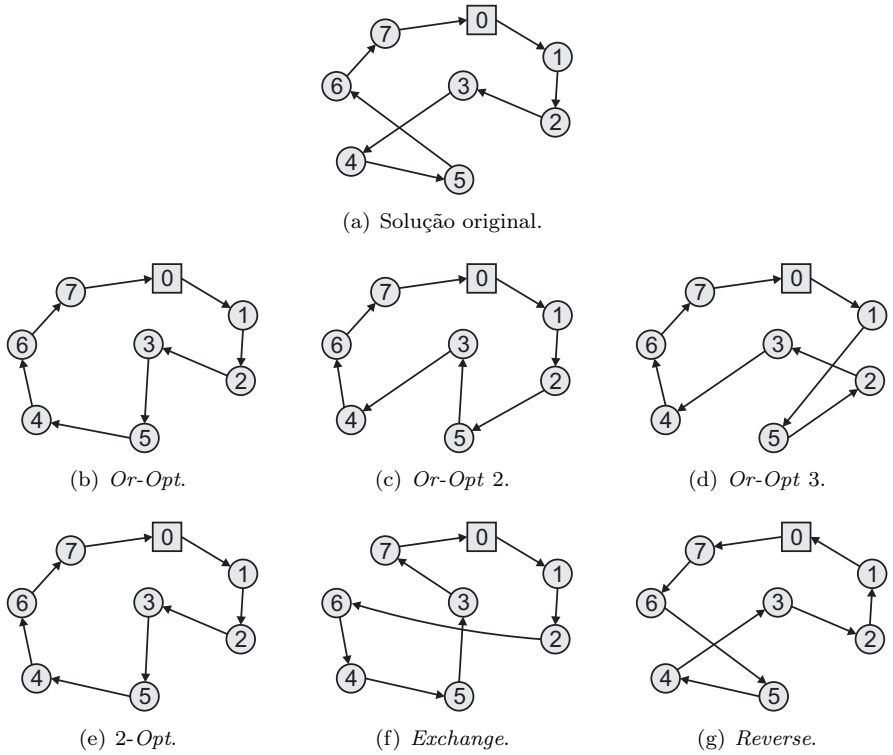


(g) *MOSwap*(1,1) Solução 4.

Figura 7. Estruturas de vizinhança inter-rotas.

- ***Reverse*:** inverte a direção da rota visando reduzir a carga máxima no veículo. Na Figura 8.g, todas as arestas têm a direção alterada, invertendo o sentido da rota.

Para reduzir o número de demandas de coleta não atendidas, foi criada uma estrutura de vizinhança denominada *Adiciona Coleta*. Esta estrutura verifica se a demanda de coleta de cada consumidor pode ser satisfeita sem extrapolar a capacidade do veículo, dada a demanda de coleta deste consumidor. Desta forma, o consumidor não precisará ser realocado em outra posição ou rota.



(a) Solução original.

(b) *Or-Opt*. (c) *Or-Opt* 2. (d) *Or-Opt* 3.

(e) 2-*Opt*. (f) *Exchange*. (g) *Reverse*.

Figura 8. Estruturas de vizinhança intra-rotas.

3.4.4 Perturbação

Na etapa de perturbação do algoritmo são utilizados mecanismos que realizam modificações em uma solução, escapando de ótimos locais, permitindo a exploração de outras áreas do espaço de busca. Neste trabalho, foram criadas perturbações que modificam as soluções em relação a ambos os objetivos do problema de roteamento de veículos multiobjetivo com coleta seletiva.

Assim sendo, o método de perturbação do algoritmo MOILS consiste em aplicar, primeiramente, o procedimento *Altera Coleta* que visa a modificação da solução em relação ao atendimento da demanda de coleta. Neste método, se o consumidor tiver a sua coleta realizada, então esta demanda não será mais atendida, e vice-versa. Quando a demanda de coleta de um consumidor deixar de ser atendida, o consumidor permanece na mesma posição. Caso contrário, ao passar a atender uma demanda de coleta, a capacidade do veículo pode ser extrapolada, inviabilizando a solução. Nestes casos, procura-se uma posição viável para inserir o consumidor. Caso não seja encontrada nenhuma posição viável, uma nova rota é criada para que este consumidor possa ser inserido.

Após a realização do procedimento *Altera Coleta*, a etapa de perturbação do algoritmo MOILS seleciona aleatoriamente e executa, ainda, um dos mecanismos de perturbação, descritos a seguir, que modificam a posição dos consumidores na solução.

- ***Multiple Swap*:** realiza trocas entre consumidores de rotas distintas escolhidos aleatoriamente;
- ***Multiple Shift*:** realoca um consumidor escolhido aleatoriamente de uma rota para outra rota;
- ***Ejection Chain*:** realoca um consumidor de uma rota R_1 em uma rota R_2, outro da rota R_2 em uma rota R_3, e assim sucessivamente até que um consumidor da última rota seja inserido na primeira rota.
- ***Or-Opt*:** mover 1 a 3 consumidores consecutivos de uma rota para outra.

Em todos estes quatro mecanismos de perturbação listados, a inserção dos consumidores é feita na primeira posição viável encontrada.

3.4.5 Atualização

A etapa de atualização verifica se uma determinada solução s não é dominada pelas soluções presentes na fronteira de Pareto aproximada, no Algoritmo 2 representada pelo conjunto $Front$. Se s não é dominada por nenhuma solução em $Front$, então ela é inserida no conjunto e as soluções por ela dominadas são removidas.

3.5 Heuristica P$_\epsilon$-ILS

Um método bastante comum para solucionar problemas com múltiplos objetivos baseia-se na transformação deste problema em diversos problemas com um único objetivo, onde as soluções ótimas destes problemas representam as soluções da fronteira de Pareto. Estes métodos definem um conjunto de problemas parametrizados de diferentes formas, sendo que os valores nos quais se parametrizam estes problemas determinarão quais soluções do conjunto Pareto serão alcançadas. Desta forma, a tarefa de escolher esta sequência de valores de parâmetros é extremamente difícil. Em muitos casos é impossível definir valores tais que toda a fronteira de Pareto seja encontrada.

O método ϵ-Restrito, proposto por Chankong & Haimes (1983), é baseado na escalarização no qual um dos objetivos é minimizado, enquanto os demais objetivos são limitados por meio de restrições. As soluções pertencentes a fronteira de Pareto são obtidas através de uma variação sistemática do limite destas restrições.

Definição 4 *Seja m o número de objetivos para determinado problema multiobjetivo, se $x^* \in \mathcal{F}_x$ é eficiente, então existem $i \in \{1, 2, ..., m\}$ e números reais ϵ_j, $j = 1, ..., m$ $(j \neq i)$ tais que x^* resolve (Chankong & Haimes, 1983):*
$x^* = arg\ min_{x \in \mathcal{F}_x}\ f_i(x)$
Sujeito a: $\{\ f_j(x) \leq \epsilon_j,\ j = 1, ..., m\ (j \neq i)$

Segundo a definição 4, o conjunto de soluções eficientes é constituído por cada solução x^* obtida para cada ϵ_j. Assim, a principal vantagem do método ϵ-Restrito é que o mesmo possibilita encontrar a fronteira de Pareto independente da estrutura do problema multiobjetivo, podendo ele ser não-convexo e discreto (Chankong & Haimes, 1983).

Baseada nos princípios básicos do método ϵ-Restrito, a heurística P$_\epsilon$ consiste em definir, a cada iteração, um conjunto de consumidores no qual suas demandas de coleta serão atendidas. Desta forma, este sub-problema gerado é solucionado utilizando um algoritmo baseado no ILS.

Algoritmo 3: Heurística P$_\epsilon$-ILS.

```
   Entrada: maxCount
1  s_CVRP ← gerar solução para o CVRP utilizando ILS;
2  s_VRPSPD ← gerar solução para o VRPSPD utilizando ILS;
3  atualizar(Front, s_CVRP);
4  atualizar(Front, s_VRPSPD);
5  cont ← 0;
6  enquanto cont < maxCount faça
7      s'_CVRP ← Adiciona Coleta(s_CVRP);
8      Atualizar(Front, s'_CVRP);
9      enquanto ∃ consumidor_j ∈ s'_CVRP | ℓ_j = 0 faça
10         Definir j aleatoriamente;
11         ℓ_j ← 1;
12         s'_CVRP ← Insere uma coleta(consumidor_j, s'_CVRP);
13         s'_CVRP ← ILS(s'_CVRP);
14         atualizar(Front, s'_CVRP);
15     enquanto ∃ consumidor_j ∈ s'_VRPSPD | ℓ_j = 1 faça
16         Definir j aleatoriamente;
17         ℓ_j ← 0;
18         s'_VRPSPD ← ILS(s'_VRPSPD);
19         atualizar(Front, s'_VRPSPD);
20     cont ← cont + 1;
21 retorna Front;
```

O passo inicial define os pontos de partida do algoritmo (linhas 1 e 2). Estes pontos consistem nas soluções do CVRP e VRPSPD, nos quais nenhuma demanda de coleta é atendida e todas as demandas de coletas são atendidas, respectivamente. A busca local iterativa, apresentada em Penna et al. (2012), foi utilizada para resolver ambos problemas. As duas soluções são, então, incluídas na fronteira de Pareto aproximada.

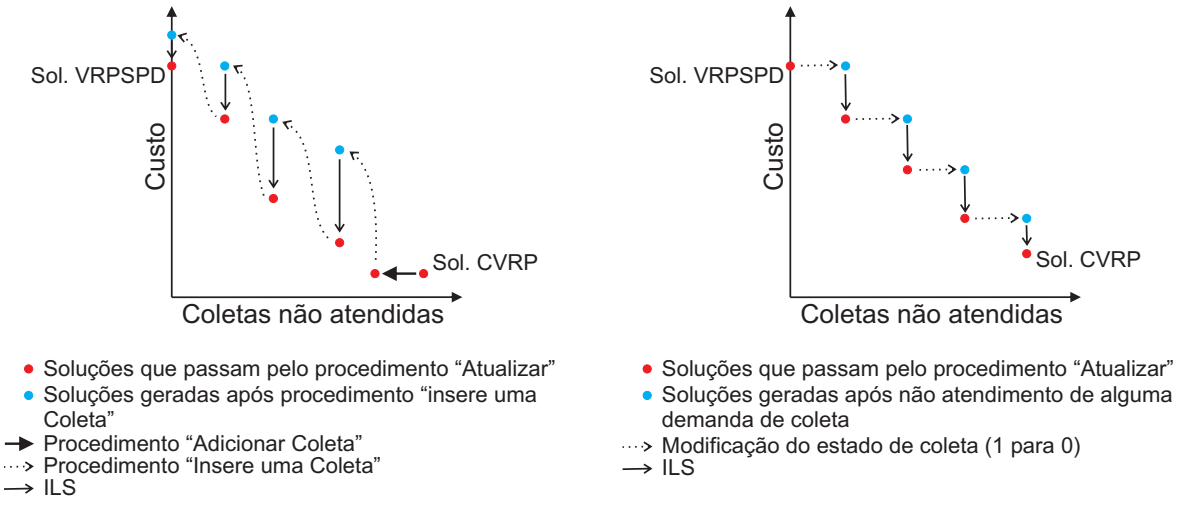
O algoritmo irá executar *maxCount* iterações (linhas 4–14), no qual a solução do CVRP será modificada gradativamente até que todas as demandas de coleta sejam satisfeitas. Por outro lado, a solução do VRPSPD será também modificada gradativamente até que nenhuma demanda de coleta seja atendida.

Iniciando pela solução s'_{CVRP}, na linha 7, para cada rota aplica-se a busca local intra-rota *Adiciona Coleta*, definida na Subseção 3.4.3.2. Dessa forma, sem alterar a posição dos consumidores na rota, busca-se atender o máximo de demandas de coleta possível. Dada esta solução, a fronteira *Front* é atualizada aplicando o mesmo procedimento do algoritmo MOILS.

Em seguida, enquanto houver demandas de coleta não atendidas na solução s'_{CVRP}, as demandas de coleta não atendidas serão selecionadas aleatoriamente, uma a uma, para serem satisfeitas. Este procedimento resulta na realocação do consumidor, uma vez que o atendimento de uma demanda de coleta pode acarretar na extrapolação da capacidade do veículo. Assim, este consumidor é realocado na melhor posição encontrada dentre todas as rotas, considerando o custo de transporte. Quando nenhuma posição viável é encontrada, uma nova rota é criada para a inclusão deste consumidor. Para refinar a solução obtida, aplica-se o método ILS (Penna et al., 2012) à esta solução. Finalmente, atualiza-se o conjunto de soluções não dominadas *Front*. A Figura 9.a ilustra os passos da heurística P_ϵ-ILS partindo da solução do CVRP.

O procedimento descrito para a solução s'_{CVRP} é repetido para a solução s'_{VRPSPD} com algumas modificações. Cada demanda de coleta satisfeita é selecionada aleatoriamente, uma a uma, para deixarem de ser atendidas. Este procedimento não requer a realocação do consumidor, uma vez que o mesmo não acarreta em nenhuma inviabilidade da solução. Da mesma forma, para refinar a solução, o método ILS é utilizado e o conjunto *Front* é atualizado. A Figura 9.b mostra as soluções geradas pela heurística P_ϵ-ILS, partindo de uma solução do VRPSPD.

Como a seleção das coletas a serem atendidas ou não atendidas é feita de forma aleatória, o método é executado várias vezes ($maxCount$) a fim de encontrar soluções mais diversificadas e aproximadas da fronteira de Pareto ótima.



(a) Partindo de Sol. CVRP (b) Partindo de Sol. VRPSPD

Figura 9. Etapas da heurística P_ϵ-ILS.

3.6 Algoritmo genético de ordenação não-dominante II (NSGA-II)

O algoritmo genético de ordenação não-dominante II (NSGA-II, *Non-Dominated Sorting Genetic Algorithm-II*) foi adaptado a fim de solucionar o MOVRPSP. Os passos do algoritmo seguem os princípios básicos do NSGA-II, proposto por Deb et al. (2002).

Para o MOVRPSP, os indivíduos da população representam uma solução completa do problema tratado, ou seja, um conjunto de rotas e um vetor binário que armazena o estado da demanda de coleta dos consumidores (atendida ou não atendida). O método de geração da população inicial P, assim como nos algoritmos MOILS e P_ϵ-ILS, inicia buscando soluções para o VRPSPD e CVRP para então inseri-las em P. Para uma população com p indivíduos, as demais soluções são obtidas a partir da alteração do estado das demandas de coletas das soluções obtidas para o CVRP e VRPSPD, sendo que metade dos indivíduos da população serão gerados a partir da solução do CVRP e a outra metade a partir do VRPSPD. Iniciando o procedimento pela solução do CVRP, o algoritmo calcula uma taxa ($2n/p$, sendo n é o número de consumidores) no qual as demandas de coleta serão atendidas. Ao término do procedimento, o último indivíduo terá todas ou quase todas as demandas de coleta satisfeitas. Procedimento similar é executado para gerar as soluções iniciais partindo do VRPSPD, sendo que, ao término do procedimento, o último indivíduo definido terá poucas ou nenhuma demanda de coleta satisfeita.

A população filha Q é gerada a partir da recombinação de indivíduos da população P. Dois operadores de cruzamento são considerados neste trabalho, tendo eles igual probabilidade de serem selecionados. Cada indivíduo *filho* pode sofrer mutação, dada uma probabilidade de mutação p_{mut}. Os operadores de mutação utilizados neste trabalho são *Or-opt* e *Altera Coleta*, descritos na Seção 3.4.4. Estes também têm igual probabilidade de serem selecionados para serem aplicados.

A fase de seleção do NSGA-II ocorre em dois momentos distintos do algoritmo. O primeiro deles, após a população inicial ser gerada, o algoritmo utiliza a técnica torneio binário da seguinte forma: dados dois indivíduos a e b, se um indivíduo a domina o indivíduo b, então o indivíduo a é selecionado para o processo de recombinação, caso contrário, o indivíduo b é selecionado. Se a não domina b e b não domina a, ambos indivíduos são selecionados. A partir de então, o método de seleção realiza uma ordenação por não-dominância dos indivíduos pertencentes a $P \cup Q$, retornando um conjunto de fronteiras não dominadas. Seguindo esta ordenação, as fronteiras são incluídas na nova população P, preservando as melhores soluções da população anterior (elitismo). Quando um fronteira não puder ser totalmente incluída em P, os indivíduos dessa fronteira são selecionados de acordo com o método *Crowding-Distance*. Em seguida, a técnica torneio de multidão é aplicada para seleção dos indivíduos a serem recombinados e inseridos na população Q.

No torneio de multidão uma solução s_i, pertencente a fronteira i após a ordenação de dominância, é considerada melhor que a solução s_j, pertencente a fronteira j, se (Deb et al., 2002):

1. s_i possui um nível de não dominância melhor que s_j, ou seja, se $i < j$.
2. $i = j$ e a distância de multidão de s_i é maior que distância de multidão de s_j.

O Algoritmo 4 e a Figura 10 ilustram o funcionamento do NSGA-II implementado.

Algoritmo 4: Pseudocódigo do NSGA-II.

```
1  g ← 0; // contador de gerações
2  P_g ← Gerar População Inicial(); // População Pai
3  Q_g ← Realizar Seleção Torneio, Cruzamento e Mutação(P_g); // População Filha
4  repita
5  |  R_g ← P_g ∪ Q_g;
6  |  F[ ] ← Ordenação por não-dominância(R_g);
7  |  P_{g+1} ← ∅;
8  |  i ← 1;
9  |  enquanto |P_{g+1} + F_i| ≤ N faça
10 |  |  P_{g+1} ← P_{g+1} ∪ F_i;
11 |  |  i ← i + 1;
12 |  para todo solução s_j ∈ F_i faça
13 |  |  d_j ← crowding-distance(s_j);
14 |  s ← Ordenar F_i de forma ascendente de acordo com d_j;
15 |  para j = 1 até j = N − |P_{g+1}| faça
16 |  |  P_{g+1} ← s_j;
17 |  Q_{g+1} ← Realizar Seleção Torneio de Multidão, Cruzamento e Mutação(P_{g+1});
18 |  g ← g + 1;
19 até Critério de Parada;
20 R_g ← P_g ∪ Q_g;
21 Fronteira ← Obter soluções não dominadas(R_g);
22 retorna Fronteira;
```

3.6.1 Operadores de cruzamento

Neste trabalho foram adotados dois métodos de cruzamento: cruzamento baseado em rota e *split*, descritos a seguir.

3.6.1.1 Cruzamento baseado em rota

O cruzamento baseado em rota (RBX, *Route Based Crossover*) seleciona aleatoriamente algumas rotas de um indivíduo pai_1 para serem incluídas no indivíduo $filho_1$. Em seguida, os demais consumidores não contidos na solução do $filho_1$ são inseridos de acordo com a solução do indivíduo pai_2. De forma similar, algumas rotas do indivíduo pai_2 são selecionadas aleatoriamente para serem incluídas no indivíduo $filho_2$ e este indivíduo é completado de acordo com as rotas do indivíduo pai_1.

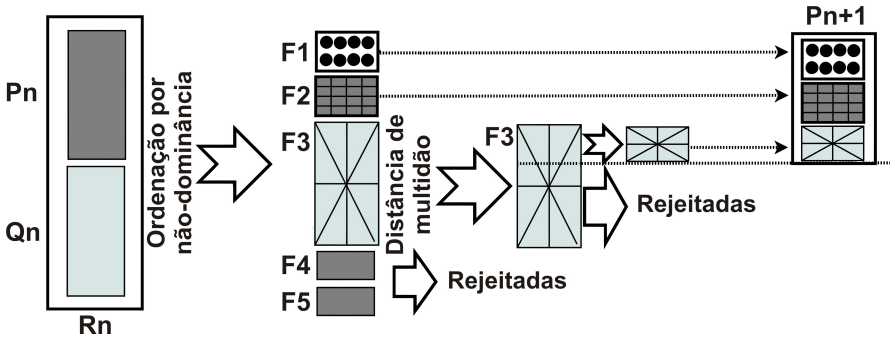
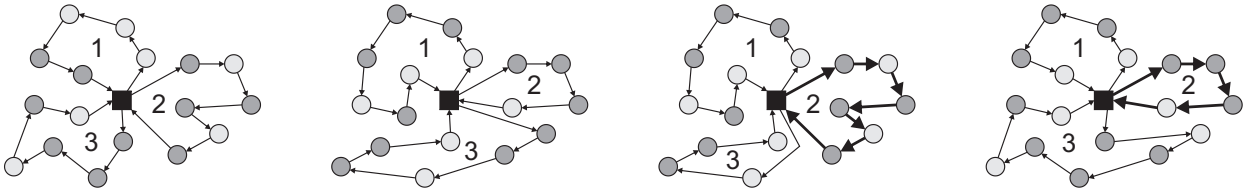


Figura 10. Algoritmo NSGA-II.



(a) Pai_1 (b) Pai_2 (c) $Filho_1$ (d) $Filho_2$

Figura 11. Cruzamento baseado em rota (RBX).

Este método é utilizado por recombinar tanto a posição dos consumidores nas rotas quanto o estado de coleta dos consumidores. A Figura 11 exemplifica o funcionamento do método de cruzamento RBX. O quadrado negro indica a posição do depósito e as circunferências na cor clara representam os consumidores, tais que suas demandas de coletas são atendidas, e as circunferências na cor escura representam os consumidores com coletas não satisfeitas. As Figuras 11.a e 11.b são os dois indivíduos (Pai_1 e Pai_2) selecionados para o cruzamento. O indivíduo $Filho_1$ é gerado a partir da recombinação dos indivíduos Pai_1 e Pai_2, no qual a rota 2 é totalmente mantida do Pai_1 e as demais são inseridas do Pai_2. Da mesma forma, o indivíduo $Filho_2$ mantém a rota 2 do Pai_2 e o restante é inserido conforme Pai_1. Quando um consumidor é copiado de uma solução Pai para uma solução $Filho$, o estado de coleta desse consumidor é mantido do indivíduo Pai.

3.6.1.2 Cruzamento *split*

O método de cruzamento *split*, proposto por Prins (2004), consiste da transformação dos indivíduos *pais* em soluções para o problema do caixeiro viajante (TSP, *Traveling Salesman Problem*). A estas soluções, aplica-se o método de cruzamento sequencial (OX) e, então, a solução para o TSP é novamente transformada em uma solução para o VRP através de um procedimento denominado *split*.

O processo de transformação de um problema do caixeiro viajante em um problema de roteamento de veículo, inicialmente, constrói um grafo direcionado acíclico (DAG), D=(V,A) com V={0,1,...,n}, sendo n o número de consumidores e o vértice 0 representando o depósito. As arestas contidas na DAG representam diferentes rotas viáveis, definidas como: para todo vértice i e j pertencente V, tal que $i < j$, o arco (i, j) pertence ao conjunto de arestas A se a capacidade do veículo nesta rota não é extrapolada.

A Figura 12 mostra como gerar um grafo direcionado acíclico (Figura 12.b) a partir de uma solução do problema do caixeiro viajante (Figura 12.a). Na solução do TSP (Figura 12.a), os números indicados entre parênteses representam a demanda de entrega de cada consumidor e os valores nas arestas representam o custo de percorrê-la. Na DAG, cada aresta indica uma rota possível de ser gerada a partir da solução do TSP. Por exemplo, na aresta $a : 40$ indica que a rota $0 - a - 0$ tem custo 40, a aresta $bcd : 120$ indica que a rota $0 - b - c - d - 0$ tem custo 120. A aresta $abc : 90$, por exemplo, não aparece na DAG, pois a carga do veículo extrapola sua capacidade máxima (Q=10).

Assim sendo, dado o grafo direcionado acíclico, busca-se, então, o caminho mais curto neste grafo. Na Figura 12.b, os valores indicados acima do vértice mostram o valor do caminho mais curto de alcançar este vértice a partir do vértice 0. O valor 205 acima do último vértice indica que o caminho mais curto para alcançá-lo, partindo do vértice 0, tem custo 205, sendo que este caminho passa pelas arestas: $ab : 55, c : 60, de : 90$.

Portanto, dado o caminho mais curto, é possível definir as partições que serão aplicadas à solução do problema do caixeiro viajante. Serão criadas as seguintes rotas: $0 - a - b - 0$, $0 - c - 0$ e $0 - d - e - 0$, como mostrado na Figura 12.c.

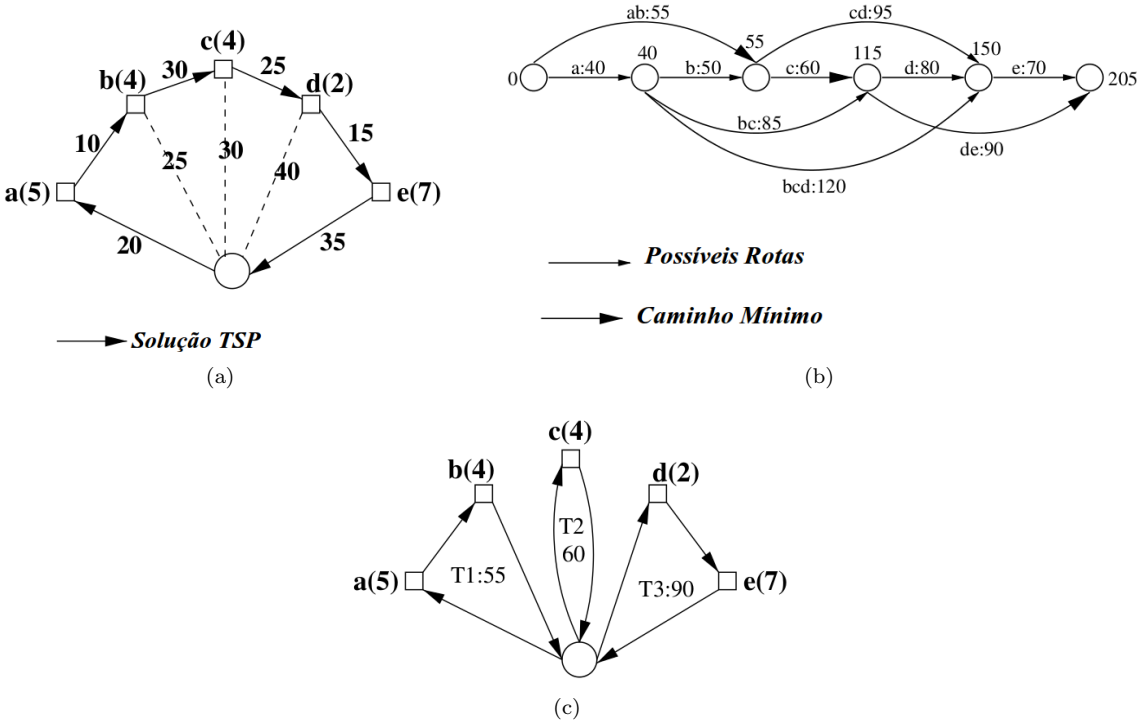


Figura 12. Transformando uma solução do TSP em uma solução para o VRP, de acordo com o cruzamento *split* (Prins, 2004).

4. Resultados Experimentais

Para avaliar o desempenho dos algoritmos MOILS, NSGA-II e P_ϵ-ILS, os testes foram realizados a partir da execução de cada algoritmo em um conjunto de 14 instâncias contendo entre 50 e 199 consumidores, mais o depósito (Salhi & Nagy, 1999). O custo de transporte é calculado como a distância Euclidiana entre os pontos. Nos problemas CMT1 a CMT5, os consumidores estão aleatoriamente distribuídas no plano, enquanto nos problemas CMT11 e CMT12, estes aparecem em grupos, no qual o depósito não está centralizado.

Os parâmetros livres dos algoritmos comparados foram ajustados com o objetivo de proporcionar a cada método, aproximadamente, o mesmo tempo de execução. Uma análise estatística similar à empregada nesta seção não apontou diferenças significativas no tempo de execução dos algoritmos.

Para solucionar os problemas teste, foram atribuídos os seguintes valores aos parâmetros livres dos algoritmos:

1. MOILS: após testes preliminares, às constantes do algoritmo foram atribuídos os seguintes valores: $maxIter = 100$ e $maxCount = 15$;
2. P_ϵ: testes preliminares sugeriram $maxCount = 50$;
3. NSGA-II: O número de gerações do algoritmo e o tamanho da população foi definido como 250 e 500, respectivamente. A taxa de mutação (p_{mut}) é igual a $0,3$;
4. ILS: o algoritmo, baseado no ILS, utilizado em todos os algoritmos para encontrar soluções dos problemas de roteamento de veículos mono-objetivo foi definido conforme apresentado por Penna et al. (2012).

Para avaliar o desempenho relativo para os três algoritmos, três métricas foram consideradas: Cardinalidade (Card), Hipervolume (S), Cobertura de dois conjuntos (CS) que são definidos a seguir.

- Cardinalidade (Card): número de soluções contidas na fronteira de Pareto.
- Hipervolume (S): é dado pela soma dos hipercubos formados pelas soluções não dominadas obtidas pelos algoritmos. Esta métrica calcula o volume da região coberta entre os pontos das soluções na fronteira encontrada e um determinado ponto de referência (W). Para problemas de minimização, maiores hipervolumes indicam melhores soluções.

- Cobertura (C): dadas duas fronteiras de Pareto aproximadas A e B, C(A,B) calcula o número de soluções em B que são fracamente dominadas por outra fronteira A, dada a cardinalidade de B. Se C(A,B) é igual a 1, então todas as soluções em B são dominadas por soluções da fronteira A. Por outro lado, se o valor retornado é igual a 0, isso aponta para uma situação oposta na qual nenhuma das soluções de B são dominadas por A. Para aplicação destas métricas, é necessário avaliar tanto C(A,B) quanto C(B,A), pois estes valores não são complementares. A Equação 10 apresenta como se obtém o valor da cobertura dadas duas fronteira Pareto aproximadas A e B (Zitzler, 1999).

$$C(A,B) = \frac{|\{y \in B : \exists x \in A, x \preceq y\}|}{|B|} \tag{10}$$

Neste trabalho, a métrica hipervolume é calculada considerando como ponto de referência (W) o maior valor de cada objetivo do problema, obtido após a execução de todas as replicações de todos os algoritmos. O valor do Hipervolume foi normalizado no intervalo $(0, 1)$, independentemente, para cada problema. Para ambas as métricas, Hipervolume e Cobertura, os valores obtidos representam a diferença entre porcentagem, enquanto a métrica Cardinalidade fornece valores absolutos.

4.1 Planejamento estatístico

Foram empregados testes estatísticos a fim de detectar diferenças significativas entre os algoritmos e estimar sua magnitude, os quais foram realizados, independentemente, para cada métrica avaliada. Esta análise foi aplicada em um conjunto de dados composto pela avaliação das métricas dadas às soluções obtidas em 8 replicações de cada algoritmo em cada problema teste.

Para cada métrica, o experimento foi projetado como um planejamento completo em blocos aleatorizados (RCBD, *Randomized Complete Bloc Design*), sendo os algoritmos os níveis do fator experimental e os problemas como um fator de blocagem (Montgomery, 2008). Ao considerar os problemas como blocos, é possível modelar e remover os efeitos das diferentes instâncias nos resultados, e obter as diferenças reais no desempenho dos algoritmos, dados os múltiplos problemas teste utilizados. A hipótese nula de ausência de diferença entre os algoritmos foi testada contra a hipótese alternativa bilateral. Para evitar a necessidade de se atender às premissas de normalidade do teste F, foi utilizado o teste de Gore (Parsad, 2002), uma alternativa não-paramétrica robusta para análise de variância em blocos.

Após os testes de significância, os estimadores dos efeitos dos blocos (instâncias) foram obtidos (Montgomery, 2008) e subtraídos das amostras, permitindo, assim, uma estimativa problema-independente dos tamanhos dos efeitos de cada algoritmos. As magnitudes das diferenças de desempenho foram obtidas através do estimador de Hodges-Lehmann (HL) (Hodges & Lehmann, 1963), que fornece uma medida da mediana das diferenças entre duas amostras independentes. Uma descrição mais detalhada dos métodos estatísticos estão disponíveis em Batista et al. (2011).

4.2 Resultados e discussões

Os resultados obtidos nos testes aplicados mostram diferenças estatisticamente significativas entre os algoritmos avaliados. A média e o desvio padrão dos valores obtidos estão sumarizados na Tabela 1. A Tabela 2 sintetiza os resultados das análises estatísticas, mostrando a grandeza das diferenças estatisticamente significativas.

Para a métrica Hipervolume, todos os algoritmos apresentaram diferenças relativamente pequenas, como mostrado na Tabela 1, com o algoritmo P_ϵ apresentando um desempenho um pouco pior que os demais métodos. A análise estatística, apresentada na Tabela 2, confirma esta observação, mostrando que todos os algoritmos apresentaram diferenças relativamente pequenas, mas estatisticamente significantes, com o MOILS mostrando os melhores resultados (11% superior ao P_ϵ-ILS e 8% melhor que o NSGA-II). Estes resultados indicam que o MOILS, em geral, é capaz de obter melhores fronteiras por retornar um conjunto de soluções melhor distribuídas ou pontos mais próximos da fronteira Pareto-ótima.

As diferenças médias obtidas pela métrica Cardinalidade também foram relativamente modestas. A Tabela 1 mostra que, para cada problema, todos os algoritmos retornaram aproximadamente o mesmo número médio de soluções. A Tabela 2 mostra que, quando considerado todos os problemas, diferenças pequenas mas estatisticamente significantes foram observadas. Os resultados obtidos a partir do algoritmo MOILS apresentam um número maior de soluções presentes na fronteira de Pareto aproximada. Em média, o MOILS retorna 2.56 soluções a mais que o P_ϵ-ILS e 4.32 a mais que o NSGA-II.

Diferenças mais significativas nos resultados dos algoritmos podem ser observadas para a métrica de Cobertura. O MOILS supera a heurística P_ϵ-ILS por uma média de 0.7 e o NSGA-II por 0.32. Este é um indicador muito evidente de superioridade, uma vez ele indica que as fronteiras geradas pelos MOILS

Tabela 1. Média e desvio padrão (entre parênteses) dos valores obtidos pelas métricas avaliadas.

Instâncias	**Hipervolume**			**Cardinalidade**		
(CMT)	**MOILS**	**P$_\epsilon$**	**NSGA-II**	**MOILS**	**P$_\epsilon$**	**NSGA-II**
1X	0,91(0,16)	0,60(0,17)	0,86(0,16)	3,12(0,6)	6,25(3,6)	2,12(0,3)
1Y	0,95(0,02)	0,90(0,05)	0,87(0,07)	11,12(1,1)	8,62(1,5)	7,5(1,2)
2X	0,88(0,07)	0,77(0,09)	0,83(0,10)	8,12(2,8)	9,50(2,1)	4,00(1,6)
2Y	0,93(0,01)	0,82(0,05)	0,84(0,06)	33,62(6,5)	27,25(4,7)	31,12(5,3)
3X	0,94(0,05)	0,74(0,20)	0,91(0,05)	6,37(0,9)	8,87(3,5)	3,37(1,1)
3Y	0,96(0,00)	0,76(0,05)	0,86(0,04)	16,63(2,6)	13,63(2,9)	15,5(5,1)
4X	0,90(0,04)	0,83(0,08)	0,78(0,06)	15,25(2,6)	12,63(5,0)	5,0(2,5)
4Y	0,95(0,02)	0,88(0,06)	0,83(0,09)	42,13(3,3)	32,63(5,5)	35,5(8,0)
5X	0,91(0,04)	0,82(0,05)	0,85(0,06)	17,25(4,2)	16,25(4,3)	8,25(5,2)
5Y	0,94(0,01)	0,82(0,04)	0,83(0,03)	51,63(5,9)	52,0(10,6)	59,88(14,1)
11X	0,94(0,03)	0,87(0,02)	0,87(0,03)	44,75(5,5)	42,13(5,3)	40,75(7,1)
11Y	0,91(0,08)	0,65(0,08)	0,70(0,03)	27,63(5,6)	20,25(3,8)	20,5(2,8)
12X	0,96(0,02)	0,75(0,04)	0,87(0,08)	14,63(3,7)	9,25(2,2)	14,38(3,4)
12Y	0,98(0,01)	0,85(0,06)	0,91(0,03)	32,75(3,7)	14,50(3,6)	20,00(5,4)
Média	0,93(0,04)	0,79(0,07)	0,84(0,06)	23,21(3,5)	19,55(4,19)	19,13(4,51)

Instâncias	**Cobertura**		
(CMT)	**MOILS**	**P$_\epsilon$**	**NSGA-II**
1X	0,80(0,29)	0,53(0,09)	0,65(0,15)
1Y	0,53(0,24)	0,63(0,20)	0,36(0,12)
2X	0,63(0,18)	0,54(0,13)	0,65(0,30)
2Y	0,67(0,06)	0,42(0,09)	0,55(0,15)
3X	0,59(0,37)	0,60(0,14)	0,72(0,25)
3Y	0,74(0,08)	0,34(0,05)	0,56(0,13)
4X	0,67(0,17)	0,59(0,10)	0,48(0,09)
4Y	0,68(0,12)	0,52(0,16)	0,40(0,12)
5X	0,64(0,19)	0,59(0,15)	0,56(0,18)
5Y	0,73(0,09)	0,39(0,06)	0,55(0,08)
11X	0,72(0,08)	0,56(0,06)	0,39(0,10)
11Y	0,71(0,14)	0,55(0,06)	0,41(0,10
12X	0,73(0,11)	0,42(0,08)	0,58(0,07)
12Y	0,80(0,10)	0,31(0,02)	0,49(0,09)
Média	0,69(0,16)	0,50(0,10)	0,53(0,14)

Tabela 2. Diferenças estimadas no desempenho médio entre os algoritmos (linha × coluna) para as métricas consideradas. Somente os efeitos estatisticamente significativos a um nível de confiança de 95% (ajustado para o teste de múltiplas hipóteses usando Correção Dunn-Šidák (Dunn, 1961)) são apresentados. Os valores positivos indicam um valor médio maior para o algoritmo da linha. Informações redundantes foram removidas da tabela a fim de melhorar a legibilidade.

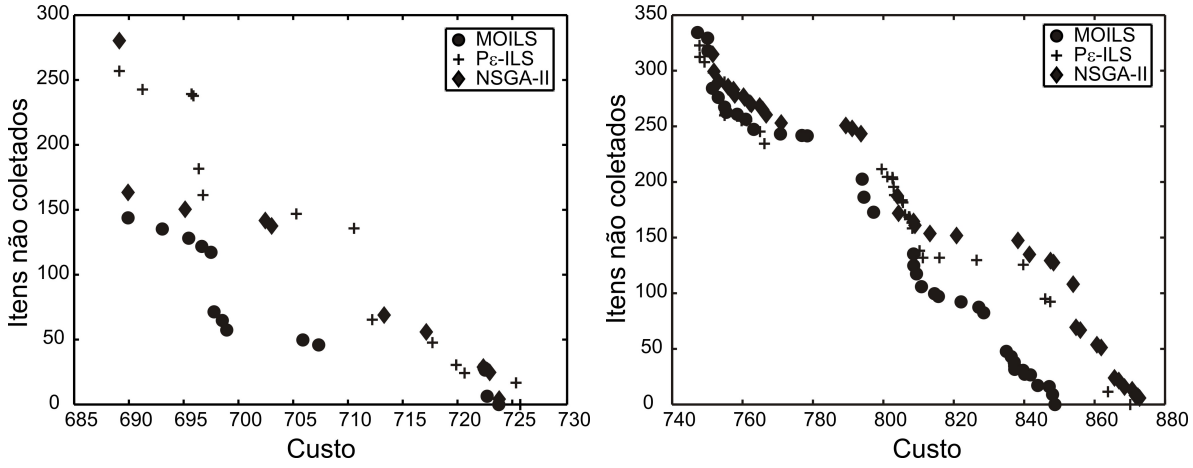
	Hipervolume		**Cobertura**		**Cardinalidade**	
	P$_\epsilon$-ILS	**NSGA-II**	**P$_\epsilon$-ILS**	**NSGA-II**	**P$_\epsilon$-ILS**	**NSGA-II**
MOILS	0,11	0,08	0,70	0,32	2,56	4,32
P$_\epsilon$-ILS	-	-0,03	-	-0,28	-	n.s

*"n.s." significa "estatisticamente não significativo".

foram capazes de dominar, em média, cerca de 70% das fronteiras correspondentes obtidas pelo P$_\epsilon$-ILS, e 32% no caso do NSGA-II. Para simplificar a apresentação dos valores relativos de cobertura, uma versão generalizada da métrica Cobertura de Dois Conjuntos, denominada Cobertura de Vários Conjuntos (Batista et al., 2012), foi utilizado para gerar os dados apresentados na Tabela 1. Ao invés de quantificar o quanto um

determinado algoritmo abrange outra, essa métrica generalizada calcula a quantidade que um determinado algoritmo cobre a união das fronteiras finais retornados por todos os algoritmos, exceto em a fronteira obtida por ele mesmo.

A Figura 13 mostra dois exemplos de fronteiras obtidas pelos três algoritmos. Uma vez que a fronteira Pareto-ótima é desconhecida para o problema considerado, não é possível avaliar a quantidade absoluta da fronteira retornada pelos algoritmos propostos neste capítulo. Contudo, um ponto extremo da fronteira é equivalente à instância correspondente do VRPSPD. A Tabela 3 apresenta uma comparação deste ponto extremo com os melhores resultados conhecidos para o VRPSPD (Subramanian et al., 2010). Pode-se perceber que, neste ponto, o MOILS encontra soluções muito próximas da melhor solução conhecida da literatura.



(a) CMT3Y. (b) CMT11X.

Figura 13. Exemplo das fronteiras obtidas pelos algoritmos implementados.

Tabela 3. Comparação do MOILS com os melhores resultados conhecidos para o VRPSPD mono-objetivo (100% das coletas atendidas).

	Instâncias (CMT)						
	1X	1Y	2X	2Y	3X	3Y	12X
MOILS	466,77	466,77	688,05	693,09	721,40	723,28	665,85
Melhor	466,77[2]	466,77[2]	668,77[1]	663,25[1]	721,27[2]	721,27[2]	644,70[1]
Erro*(%)	0,00	0,00	2,88	4,50	0,02	0,28	3,28

	Instâncias (CMT)						
	12Y	11X	11Y	4X	4Y	5X	5Y
MOILS	664,00	848,72	850,67	857,44	855,74	1037,25	1032,68
Melhor	659,52[1]	833,92[1]	830,39[1]	852,46[1]	852,35[1]	1029,25[1]	1029,25[1]
Erro**(%)	0,68	1,77	2,44	0,58	0,40	0,78	0,33

[1] Resultados reportados em Subramanian et al. (2010).
[2] Solução ótima (Subramanian et al., 2011).
** Diferença percentual entre MOILS e os melhores conhecidos.

5. Conclusões

Diversos problemas reais de roteamento de veículos com serviços de coleta e entrega possuem uma natureza multiobjetivo, mas, apesar disso, esta abordagem é ainda pouco explorada na literatura. Este capítulo apresentou uma formulação matemática adaptada ao problema de roteamento de veículos multiobjetivo denominado problema de roteamento de veículos multiobjetivo com coleta seletiva.

Devido à complexidade computacional do problema tratado, foi apresentado um conjunto de métodos heurísticos para solucioná-lo. Assim, explorou-se as especificidades das meta-heurísticas baseadas em

busca local, em população e nos métodos tradicionais de resolução de problemas multiobjetivo, resultando, respectivamente, nos algoritmos: MOILS, NSGA-II e P_ϵ-ILS

Após análises estatísticas, os testes de hipótese mostram diferenças significativas no desempenho dos algoritmos, apontando uma superioridade do MOILS em relação aos outros algoritmos. O bom desempenho do algoritmo baseado no MOILS pôde ser também observado na comparação com os resultados da literatura. Neste trabalho, um dos pontos extremos da fronteira, no qual todas as demandas de coleta são atendidas, é comparado com os melhores resultados conhecidos do problema de roteamento de veículos com coleta e entrega simultânea. Os resultados mostraram que as soluções obtidas pelo MOILS se aproximam das melhores da literatura com um erro relativo médio de 0,9%, sendo que, em muitas instâncias, este erro é quase nulo, confirmando o bom desempenho deste algoritmo.

6. Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro (edital 004/2010/PIBIT-CNPq-UFVJM, projeto PQ:306910/2006-3 e 472446/2010-0) e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão de bolsa de estudos por meio do Programa de Fomento à Pós-Graduação (PROF).

Referências

Batista, L.S.; Campelo, F.; Guimarães, & F. G. Ramírez, J.A., Pareto cone ϵ-dominance: Improving convergence and diversity in multiobjective evolutionary algorithms. In: *Proceedings of the 6th international conference on Evolutionary Multi-criterion Optimization*. Heidelberg, Germany: Springer-Verlag, v. 6576 de *Lecture Notes in Computer Science*, p. 76–90, 2011.

Batista, L.S.; Campelo, F.; Guimarães, & F. G. Ramírez, J.A., The cone ϵ-dominance: An approach for evolutionary multiobjective optimization evolutionary computation. *Evolutionary Computation*, 2012 *[Submitted]*.

Chankong, V. & Haimes, Y.Y., *Multiobjective Decision Making: Theory and Methodology*. New York, USA: Elsevier Science, 1983.

Coello Coello, C.A.; Lamont, G.B. & van Veldhuizen, D.A., *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2a edição. New York, USA: Springer Science, 2007.

Deb, K.; Pratap, A.; Agarwal, S. & Meyarivan, T., A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182—197, 2002.

Dunn, O., Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

Geiger, M.J., Foundations of the pareto iterated local search metaheuristic. In: *Proceedings of the 18th International Conference on Multiple Criteria Decision Making*. Chania, Greece, p. 19–23, 2006.

Gribkovskaia, I.; Halskau, O.; Laporte, G. & Vlcek, M., General solutions to the single vehicle routing problem with pickups and deliveries. *European Journal of Operational Research*, 180(2):568 – 584, 2007.

Gutiérrez-Jarpa, G.; Desaulniers, G.; Laporte, G. & Marianov, V., A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows. *European Journal of Operational Research*, 206(2):341 – 349, 2010.

Hodges, J.L. & Lehmann, E.L., Estimation of location based on ranks. *Annals of Mathematical Statistics*, 34:598–611, 1963.

Holland, J.H., *Adaptation in Natural and Artificial Systems*. Cambridge, USA: MIT Press, 1992.

Jozefowiez, N.; Semet, F. & Talbi, E.G., From single-objective to multi-objective vehicle routing problems: Motivations, case studies, and methods. In: Golden, B.L.; Raghavan, S. & Wasil, E.A. (Eds.), *The Vehicle Routing Problem: Latest Advances and New Challenges*. Berlin, Germany: Springer, v. 43 de *Operations Research/Computer Science Interfaces Series*, p. 445–471, 2008.

Lourenço, H.R.; Martin, O.C. & Stützle, T., Iterated local search. *AIROnews*, VIII(3):11–14, 2003.

Montané, F.A.T. & Galvão, R.D., A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers and Operations Research*, 33(3):595–619, 2006.

Montgomery, D.C., *Design and Analysis of Experiments*. 7a edição. New York, USA: J. Wiley & Sons, 2008.

Parsad, R., Non-parametric methods in analysis of experimental data. In: R. Parsad R. Srivastava, V.K.G. (Ed.), *Design and Analysis of Agricultural Experiments*. New Delhi, India: Indian Agricultural Statistics Research Institute, Mod. V, p. 693–694, 2002.

Penna, P.H.V.; Subramanian, A. & Ochi, L.S., An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, :1–322012Http://dx.doi.org/10.1007/s10732-011-9186-y.

Prins, C., A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, 2004.

Salhi, S. & Nagy, G., A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of Operational Research Society*, 50(10):1034–1042, 1999.

Subramanian, A.; Drummond, L.M.A.; Bentes, C.; Ochi, L.S. & Farias, R., A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899 – 1911, 2010.

Subramanian, A.; Uchoa, E.; Pessoa, A.A. & Ochi, L.S., Branch-and-cut with lazy separation for the vehicle routing problem with simultaneous pickup and delivery. *Operations Research Letters*, 39(5):338–341, 2011.

Sural, H. & Bookbinder, J.H., The single-vehicle routing problem with unrestricted backhauls. *Networks*, 41(3):127–136, 2003.

Zitzler, E., Evolutionary Algorithm for Multiobjective Optimization: Methods and Applications. PhD Thesis, Swiss Federal Institute of Technology Zurich, Computer Engineering and Networks Laboratory, Zurich, Switzerland, 1999.

Notas Biográficas

Luciana Pereira de Assis é graduada em Ciência da Computação (Centro Universitário de Belo Horizonte, 2004) e mestre em Ciência da Computação (UFMG, 2007). Atualmente é professora Assistente do departamento de Computação da UFVJM e doutoranda junto ao Programa de Pós-Graduação em Engenharia Elétrica da UFMG.

André Luiz Maravilha Silva é graduado em Sistemas de Informação (UFVJM, 2011) e atualmente é mestrando junto ao Programa de Pós-Graduação em Engenharia Elétrica da UFMG.

Felipe Campelo Franca Pinto é graduado em Engenharia Elétrica (UFMG, 2003), mestre e doutor em Engenharia (Hokkaido University, Japão, 2006 e 2009, respectivamente). Atualmente é professor Adjunto do departamento de Engenharia Elétrica da UFMG, atuando também como co-investigador junto ao Laboratory of Hybrid Systems, Hokkaido University.

Alessandro Vivas Andrade é graduado, mestre e doutor em Engenharia Elétrica (UFMG, 1999, 2002 e 2008, respectivamente). Atualmente é professor Adjunto do Departamento de Computação da UFVJM.

Jaime Arturo Ramírez é graduado e mestre em Engenharia Elétrica (UFMG, 1986 e 1990, respectivamente) e doutor em Engenharia Elétrica (Imperial College London, UK, 1994). Atualmente é professor Titular do Departamento de Engenharia Elétrica da UFMG.

X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

ABORDAGEM MULTIOBJETIVO PARA UM PROBLEMA DE ROTEAMENTO DE VEÍCULOS COM SERVIÇOS DE COLETA E ENTREGA

André L. Maravilha, Luciana Assis, Alessandro Vivas
Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM)
{msilva.andreluiz,lupassis,alessandro.vivas}@gmail.com

Jaime A. Ramírez
Universidade Federal de Minas Gerais (UFMG)
jramirez@ufmg.br

Resumo – Este artigo apresenta uma abordagem multiobjetivo para um problema de roteamento de veículos com entregas obrigatórias e coletas opcionais. Este problema tem como objetivo definir rotas com custo mínimo e, ao mesmo tempo, atender o maior número de demandas de coleta. O algoritmo proposto para solucionar o problema é uma adaptação do *Iterated Local Search* que retorna um conjunto de soluções não dominadas. Para avaliar a eficiência deste algoritmo, foram utilizadas instâncias do Problema de Roteamento de Veículos com Coletas e Entregas Simultâneas mono-objetivo. Os resultados mostraram que o algoritmo proposto apresenta um bom desempenho quanto ao número de soluções retornadas, extensão e distribuição das soluções Pareto. A qualidade dos resultados foi analisada também considerando os objetivos individualmente. Neste caso, os resultados também foram satisfatórios e se aproximam dos resultados da literatura.

Palavras-chave – Logística & Transportes, Metaheurísticas, Otimização Combinatória.

Abstract – This paper presents an approach to a multiobjective vehicle routing problem with deliveries and selective pickups. This issue aims to define routes with minimum cost and, at the same time, attend as many demands of collection. The proposed algorithm to solve the problem is an adaptation of Iterated Local Search algorithm which returns not only a solution but a set of nondominated solutions. To evaluate the efficiency of this algorithm, we used instances of the Vehicle Routing Problem with Simultaneous Pickup and Delivery single objective. The results showed that the proposed algorithm presents a good performs on the number of solutions returned, extent and uniform distribution of Pareto solutions. Another form used in this study to evaluate the quality was to analyze the objectives individually. In this case, the results were also satisfactory and closer to the literature.

Keywords – Logistics & Transport, Metaheuristics, Combinatorial Optimization.

1 INTRODUÇÃO

O estudo de roteamento de veículos tem um papel importante em indústrias que trabalham com um fluxo intenso de materiais. Em regiões com grande área territorial, o custo com o transporte pode impactar diretamente no custo final do produto. Segundo a Associação Brasileira de Logística [1], os custos de transporte correspondem a 60% dos custos logísticos das empresas. Os custos logísticos no Brasil atingiram, em 2008, um valor equivalente a 11,6% do Produto Interno Bruto (PIB), ou seja, R$ 349,0 bilhões. O item de maior representatividade foi o transporte, com 6,9% do PIB (R$207,0 bilhões).

Em diversos casos, além de realizar a entrega de produtos, é necessário realizar a coleta dos resíduos gerados após o consumo destes produtos para reutilização ou armazenagem em local adequado. O estudo deste fluxo de materiais está inserido no contexto da logística reversa que planeja toda a atividade de entrega dos produtos, e também o caminho inverso, o retorno dos produtos à central de distribuição.

Na otimização combinatória, um problema básico de logística reversa é o Problema de Roteamento de Veículos com Coletas e Entregas Simultâneas (VRPSPD, do inglês *Vehicle Routing Problem with Simultaneous Pickup and Delivery*). O objetivo deste problema, proposto por Min [2], é minimizar o custo do transporte, atendendo às demandas de coleta e entrega simultaneamente. Cada cliente deve ser visitado uma única vez e por um único veículo, onde se tem disponível uma frota de veículos homogênea e limitada. É fácil perceber que, se nenhum consumidor necessita do serviço de coleta, o VRPSPD pode ser reduzido ao Problema de Roteamento de Veículos (VRP, do inglês *Vehicle Routing Problem*), mostrando que é um problema NP-Difícil.

Desde que foi proposto, vários são os estudos disponíveis na literatura para o VRPSPD. Dell'Amico [3] apresenta uma modelagem matemática para o problema e faz uso da técnica *Branch-and-Price* para solucioná-la. Para a fase de avaliação (*pricing*) são apresentadas duas técnicas, uma utilizando programação dinâmica e outra com relaxação do espaço de estados.

Diversos autores apresentam algoritmos baseados nas metaheurísticas *Tabu Search* [4–7], *Greedy Random Adaptative Search Procedure* (GRASP) [8], *Iterated Local Search* (ILS) [9, 10] e *Variable Neighborhood Descent* (VND) [8–10], pois estas apresentam bons resultados na solução do VRPSPD. Nestes trabalhos, além do uso destas metaheurísticas, são apresentadas, também, diversas técnicas geração de soluções de partida dos algoritmos.

Subramanian et al [10] apresenta uma abordagem paralela para solucionar o VRPSPD, na qual é utilizado a metaheurística *Iterated Local Search* (ILS) em conjunto com um procedimento *Variable Neighborhood Descent* com ordenação aleatória (RVND). Até então, este detém os melhores resultados para grande parte das instâncias (problemas teste) do VRPSPD.

Abordagens bio-inspiradas vem se mostrando como uma boa opção na resolução de problemas de otimização combinatória. Nos trabalhos [11, 12] são propostos algoritmos baseados em colônia de formigas para solução do VRPSPD. Estas abordagens apresentam resultados de boa qualidade, principalmente por permitirem a exploração de diversas soluções, entretanto, tem um maior custo computacional. Maravilha et al. [13] também apresenta um algoritmo baseado em colônia de formigas, além de um algoritmo baseando no sistema imunológico. É feita a comparação entre as duas abordagens, sendo que o algoritmo imunológico apresenta melhores resultados.

Apesar das inúmeras aplicações práticas do VRPSPD, existem situações onde a realização da coleta não é obrigatória, permitindo a escolha de quais coletas realizar. Um exemplo desta situação pode ser observado nas indústrias de agrotóxicos. No Brasil, os usuários tem um prazo de um ano para devolver as embalagens vazias para os estabelecimentos comerciais. As empresas produtoras e comercializadoras de agrotóxicos tem um prazo de um ano para coleta e destinação das embalagens. Com isso, não é necessário atender a todas demandas imediatamente, já que tem-se o prazo de um ano para efetuar a coleta, podendo selecionar quais delas realizar.

Devido a esta característica, pode-se reduzir ainda mais o custo da rota, pois um veículo ao não realizar uma determinada coleta, possuirá mais espaço livre para a carga de produtos de entrega, permitindo assim, atender mais clientes. Apesar da realização das demandas de coleta não ser obrigatória, ela é desejada, pois assim evita uma futura visita ao consumidor para recolher o produto que ainda não foi coletado. Portanto, deve-se maximizar o número de coletas realizadas.

Um problema de otimização combinatória com estas características é o Problema de Roteamento de Veículos com Entregas e Coletas Seletivas (SVRPDSP, do inglês *Single Vehicle Routing Problem with Deliveries and Selective Pickups*) [14]. Nele, cada consumidor possui demandas de entrega e coleta, e para cada demanda de coleta existe um lucro associado pela sua realização. O objetivo é definir um percurso que atenda a todas as demandas de entrega e, também, realize as coletas que forem consideradas lucrativas, minimizando uma função objetivo denominada custo líquido (custo total da rota subtraído do lucro total obtido através das coletas realizadas). Gribkovskaia et al. [14] apresentam a formulação matemática do SVRPDSP e propõem um algoritmo baseado na metaheurística *Tabu Search*, além de algumas heurísticas construtivas e de melhoria.

Sural e Bookbinder [15] tratam de um outro problema semelhante ao SVRPDSP, denominado Problema de Roteamento de Veículos com *Backhauls* Irrestrito (SVRPUB, do inglês *Single Vehicle Routing Problem with Unrestricted Backhauls*). A diferença entre eles é que no SVRPUB nem todos os consumidores apresentam demandas de coleta. Para resolvê-lo, os autores propõem algumas técnicas de melhoria para modelagem matemática e, em seguida, aplicam o software CPLEX para resolver de maneira exata o modelo proposto, permitindo encontrar a solução ótima, para problemas de tamanho médio, em tempo razoável.

Outro trabalho que também aborda o roteamento de veículos com entregas obrigatórias e coletas opcionais é [16]. Neste os autores tratam do Problema de Coleta Seletiva e Entrega (SPDP, *Selective Pickup and Delivery Problem*, onde é considerado um conjunto de pontos para aluguel de bicicletas e estas bicicletas devem ser distribuídas de acordo com a demanda de cada ponto. Neste problema a distribuição dos produtos é feita entre os pontos de consumo, não existindo um depósito, mas apenas um ponto de partida. Para solução do problema, os autores utilizam um algoritmo genético.

Os trabalhos encontrados na literatura, que abordam problemas de roteamento de veículos com coletas não obrigatórias, fazem uso de formulações mono-objetivo, onde a função objetivo é dada, geralmente, pelo custo total das rotas menos o lucro obtido pelas coletas realizadas. Mas, as características desta classe de problema possibilitam a formulação de um modelo multiobjetivo, onde podem ser consideradas duas funções objetivo: (i) a minimização do custo total da rota e (ii) a maximização do número total de coletas realizadas.

O estudo de problemas de roteamento de veículos com múltiplos objetivos vem apresentando maior destaque nos últimos anos, como pode ser observado pelo trabalho de [17] em que faz uma revisão da literatura para problemas de roteamento de veículosentrega multiobjetivo. A maior parte dos trabalhos apresentados neste estudo foram publicados a partir do ano 2000.

Em algumas situações reais é necessário visualizar diversos cenários, dados os diversos objetivos do problema, para se tomar uma decisão. Assim sendo, ao modelar problemas de maneira multiobjetivo, estes se tornam mais interessantes para aplicações práticas [18]. Apesar de serem mais atraentes para aplicações reais, o grau de complexidade é maior pois envolvem objetivos conflitantes.

O presente trabalho aborda o Problema de Roteamento de Veículos com Entrega e Coleta Seletiva de forma multiobjetivo, apresentando uma adaptação da metaheurística ILS para geração de soluções do mesmo. As demais seções estão organizadas como segue: Seção 2 são apresentadas a definição do problema e sua modelagem matemática multiobjetivo. Em seguida (Seção 3), é proposto o algoritmo ILS multiobjetivo. Na Seção 4 são apresentados os resultados computacionais obtidos pelo algoritmo, e por fim, na Seção 5 são feitas as conclusões deste trabalho.

2 DEFINIÇÃO DO PROBLEMA

O Problema de Roteamento de Veículos com Entrega e Coleta Seletiva Multiobjetivo (MOVRPDSP, do inglês *Multiobjective Vehicle Routing Problem with Delivery and Selective Pickup*) pode ser definido como: dado um grafo completo $G = (V, A)$, onde $V = \{0, 1, 2, \ldots, n\}$ é o conjunto de vértices e $A = \{(i, j) : i, j \in V, i \neq j\}$ o conjunto de arestas. O vértice 0 representa o depósito e os demais vértices representam os consumidores. Cada aresta (i, j) tem um valor $c_{ij} \geq 0$ associado que representa

o custo de se alcançar o vértice j a partir do vértice i. Cada consumidor i tem uma demanda d_i de entrega e uma demanda p_i de coleta. Tem-se disponível uma frota de $\overline{k}$ veículos homogêneos de capacidade Q para realizar as demandas.

Aqui, todas as demandas de entrega devem ser realizadas, porém, as demandas de coleta são opcionais, mas desejadas. Então, o objetivo é construir rotas que minimizem o custo total de se realizar o trajeto, atendendo todas as demandas de entrega, e ao mesmo tempo, minimizar o número de coletas não realizadas.

Os veículos não podem ter sua capacidade extrapolada em nenhum momento do trajeto. Outra restrição é que um consumidor deve ser visitado uma única vez e por um único veículo, ou seja, se este consumidor tiver a demanda de coleta atendida, esta deverá ser realizada simultaneamente com a demanda de entrega. Não é permitido a realização parcial das demandas.

Os dois objetivos a serem otimizados são conflitantes, ou seja, a melhora em um provoca a piora do outro, não existindo uma solução que seja ótima para ambos os objetivos simultaneamente. Assim, não é uma tarefa trivial identificar se uma determinada solução é de melhor qualidade em relação a outra. Isso leva ao conceito de dominância de Pareto (Definição 1), sendo a solução do problema um conjunto de soluções não dominadas (Definições 2, 3).

Definição 1 (Dominância de Pareto) *Dado um vetor objetivo $Z(x) = (f_1(x), \ldots, f_m(x))$, com m funções objetivo para minimização. Uma solução x' domina uma outra solução x'' se, e somente se, para todos os objetivos $f_i(x') \leq f_i(x'')$ com $i = 1, \ldots, m$ e, existe pelo menos um objetivo $f_i(x') < f_i(x'')$. Esta relação de dominância é representada pela notação $x' \prec x''$.*

Definição 2 (Solução Pareto-Ótima) *x^* é uma solução Pareto-Ótima se não existe qualquer outra solução x tal que $x \prec x^*$, ou seja, x^* não é dominada por nenhuma outra solução.*

Definição 3 (Conjunto Pareto-Ótimo) *O conjunto $\mathcal{X}^*$ é um conjunto Pareto-Ótimo se é composto por todas as soluções Pareto-Ótimas. O conjunto-imagem $\mathcal{Y}^*$ associado ao conjunto Pareto-Ótimo é denominado fronteira Pareto-Ótima.*

Encontrar o conjunto Pareto-Ótimo pode ser computacionalmente intratável [18]. Desta forma, a abordagem a problemas de otimização combinatória multiobjetivo é caracterizada pelo desenvolvimento de técnicas para tentar encontrar um conjunto de soluções viáveis que representam pontos próximos à fronteira Pareto-Ótima.

2.1 MODELAGEM MATEMÁTICA

A modelagem matemática apresentada nesta seção é uma adaptação da modelagem proposta em [4] para o Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas.

Sendo V o conjunto de consumidores, num total de n consumidores e V_0 o conjunto de consumidores mais o depósito ($V_0 = V \cup \{0\}$). Para alcançar um consumidor j a partir do consumidor i, existe um custo $c_{ij} \geq 0$. As demandas de coleta e entrega do consumidor j são representadas por p_j e d_j, respectivamente, com $j = 1, 2, 3, \ldots, n$. O número de veículos disponíveis é dado por $\overline{k}$, sendo Q a capacidade destes. O parâmetro y_{ij} é o somatório das cargas coletadas entre o depósito e o nó i (inclusive) dirigida ao nó $j \in V_0$. Já z_{ij} é o somatório das demandas de entrega dos consumidores visitados após o nó i (exclusive) e dirigida ao nó $j \in V_0$.

Considerando ainda as seguintes variáveis:

$$x_{ij}^k = \begin{cases} 1 & \text{, se arco } (i,j) \text{ faz parte da rota trafegada pelo veículo } k, \\ 0 & \text{, caso contrário.} \end{cases}$$

$$pc_j = \begin{cases} 1 & \text{, se a demanda de coleta do consumidor } j \text{ é satisfeita,} \\ 0 & \text{, caso contrário.} \end{cases}$$

O modelo matemático pode ser expresso por:

$$\min \sum_{k=1}^{\overline{k}} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k \tag{1}$$

$$\min \sum_{j=1}^{n} p_j - \sum_{j=1}^{n} p_j pc_j \tag{2}$$

sujeito a

$$\sum_{i=0}^{n} \sum_{k=1}^{\overline{k}} x_{ij}^k = 1, \qquad j = 1, \ldots, n \tag{3}$$

$$\sum_{i=0}^{n} x_{ij}^k - \sum_{i=0}^{n} x_{ji}^k = 0, \qquad j = 0, \ldots, n \text{ e } k = 0, \ldots, \overline{k} \tag{4}$$

$$\sum_{j=1}^{n} x_{0j}^k \leq 1, \qquad k = 1, \ldots, \overline{k} \tag{5}$$

$$\sum_{i=0}^{n} z_{ij} - \sum_{i=0}^{n} z_{ji} = d_j, \qquad \forall j \neq 0 \tag{6}$$

$$y_{ij} + z_{ij} \leq Q \sum_{k=1}^{\overline{k}} x_{ij}^{k}, \qquad i,j = 0,\ldots,n \tag{7}$$

$$x_{ij} \in \{0,1\}, \qquad i,j = 0,\ldots,n \tag{8}$$

$$y_{ij} \geq 0, \qquad i,j = 0,\ldots,n \tag{9}$$

$$z_{ij} \geq 0, \qquad i,j = 0,\ldots,n \tag{10}$$

A função objetivo, dada na Equação 1, visa a minimização do custo total da rota, já a Equação 2, apresenta a segunda função objetivo, que visa a minimização das coletas não realizadas.

Estas funções objetivo devem satisfazer algumas restrições. Cada ponto de demanda deve ser visitado por um único veículo (Equação 3). A Equação 4 representa a restrição de conservação do fluxo. A Equação 5 indica que $\overline{k}$ veículos, no máximo, podem ser utilizados. A Equação 6 obriga a satisfação de todas as demandas de entrega. A Equação 7 define que as demandas devem ser transportadas nos arcos incluídos na solução e ainda impõem um limite para a carga total transportada pelo veículo. A Equação 8 representa a restrição de integralidade e as Equações 9 e 10 representam as restrições de não-negatividade para demandas de coleta e entrega, respectivamente.

3 O ALGORITMO ILS MULTIOBJETIVO

Diversas técnicas heurísticas podem ser encontradas na literatura para solução de problemas multiobjetivo, dentre elas, os algoritmos evolucionários, propostos em [19–22], vem se destacando.

Ao se tratar de problemas de roteamento de veículos mono-objetivo com serviço de coleta e entrega, heurísticas baseadas no ILS [9, 10] apresentam bons resultados e não necessita de ajuste de muitos parâmetros. Além disso, adaptações do ILS foram utilizadas para resolução de outros problemas de otimização combinatória multiobjetivo, apresentando bons resultados [23]. Assim, o presente trabalho propõe uma adaptação desta metaheurística, possibilitando a resolução de problemas de otimização combinatória multiobjetivo, em especial o MOVRPDSP.

O algoritmo ILS é baseado na ideia de que um procedimento de busca local pode ser melhorado a partir da geração de novas soluções de partida, sendo estas novas soluções obtidas por meio de perturbações em uma solução ótima local [24]. Ele possui 4 etapas: (i) geração de uma solução inicial; (ii) busca local, que retorna uma solução melhorada; (iii) perturbação, que modifica uma solução e; (iv) critério de aceitação, que decide qual solução será utilizada na próxima iteração do algoritmo.

À adaptação proposta é dado o nome de *Multiobjective Iterated Local Search* (MOILS). Seu funcionamento é ilustrado no Algoritmo 1. O MOILS inicia seu funcionamento com a geração de soluções iniciais (linha 1). Neste trabalho, esta etapa foi realizada com a geração de duas soluções através do ILS. A primeira atendendo todas as demandas de coleta e a segunda, não atendendo a nenhuma demanda de coleta. As soluções geradas são inseridas no conjunto $Front$ que armazenará as soluções não dominadas.

Em seguida, são executados $maxIter$ iterações, onde uma solução em $Front$ é selecionada e o bloco de repetição principal do algoritmo (linhas 6–15) é executado. Neste bloco, as etapas de perturbação (linha 7) e busca local (linha 8), semelhantes ao ILS, são aplicadas à solução escolhida. O critério de aceitação é substituído por uma função de atualização, onde tenta-se inserir a nova solução gerada no conjunto $Front$ (linha 9).

A variável $cont$ representa o número de iterações realizadas sem a geração de uma solução não dominada. A sua atualização é feita da seguinte forma: se a solução gerada após a perturbação e busca local for inserida em $Front$ (indicada pela variável lógica $inserido$) então $cont$ é reiniciado (linha 11), senão $cont$ é incrementado de uma unidade (linha 13).

O parâmetro $maxCont$ representa o número máximo de vezes que uma solução será explorada sem a geração de uma solução não dominada.

As subseções a seguir descrevem o funcionamento das etapas de seleção, perturbação, busca local e atualização do MOILS, empregadas neste trabalho.

3.1 SELEÇÃO

O procedimento de seleção utilizado é baseado na *crowding-distance* do algoritmo NSGA-II (*Nondominated Sorting Genetic Algorithm II*) [19]. No algoritmo proposto, é atribuído, a cada solução, uma probabilidade proporcional à sua *crowding-distance*. Em seguida, é feito o sorteio de uma solução baseado nesta probabilidade.

Quanto mais distante das outras soluções, uma determinada solução estiver, maior a probabilidade desta ser escolhida. Assim é possível realizar a exploração do espaço de soluções de forma mais inteligente, onde é dado prioridade a regiões pouco exploradas, evitando a concentração da busca em determinados nichos.

O cálculo da *crowding-distance* utilizada neste trabalho é um pouco diferente do realizado no NSGA-II. As soluções presentes nos extremos da fronteira tem a *crowding-distance* igual a duas vezes a distância entre elas e a solução mais próxima. Enquanto no NSGA-II, as soluções presentes nas extremidades da fronteira recebem um valor infinito.

Algoritmo 1 Pseudo-código do algoritmo MOILS

```
1: Front ← gerarSoluçõesIniciais();
2: iter ← 0;
3: enquanto iter < maxIter fazer
4:    s' ← selecionar(Front);
5:    cont ← 0;
6:    enquanto cont < maxCont fazer
7:       s'' ← perturbação(s');
8:       s'' ← buscaLocal(s'');
9:       inserido ← atualizar(Front, s'');
10:      se inserido então
11:         cont ← 0;
12:      se não
13:         cont ← cont + 1;
14:      fim se
15:   fim enquanto
16:   iter ← iter + 1;
17: fim enquanto
18: retornar Front;
```

3.2 PERTURBAÇÃO

Na etapa de perturbação são utilizados mecanismos que realizam modificações em uma solução, escapando de ótimos locais, permitindo a exploração de outras áreas do espaço de busca. As modificações realizadas não podem ser muito pequenas, pois não permitiriam a exploração de novas soluções, nem muito grandes, pois assim, as buscas locais seriam realizadas a partir de pontos completamente aleatórios.

Neste trabalho, a perturbação foi realizada em duas etapas. Primeiro, é feita a alteração do status de realização da coleta de alguns consumidores selecionados aleatoriamente. Se o consumidor tiver a sua coleta realizada, então ela não será mais realizada, e vice-versa. Em seguida, um do seguintes mecanismos de perturbação é escolhido aleatoriamente e aplicado à solução: Multiple Swap, Multiple Shift, Ejection Chain.

3.3 BUSCA LOCAL

A etapa de busca local realiza a exploração da vizinhança de uma solução, a fim de encontrar uma outra solução ainda melhor. Neste trabalho foi utilizado o método de refinamento VND com as estruturas de vizinhança Shift(1,0), Swap(1,1), Crossover, Or-opt, 2-opt e Exchange.

Após a execução do VND, tenta-se realizar a demanda de coleta dos consumidores que não estão sendo atendidos. A demanda de coleta de um consumidor será atendida se mantiver a solução válida. Para todas as estruturas de vizinhança foi utilizado a estratégia melhor-aprimorante.

3.4 ATUALIZAÇÃO

A etapa de atualização verifica se uma determinada solução s não é dominada pelas soluções presentes no conjunto $Front$. Se s não é dominada por nenhuma solução em $Front$, então ela é inserida no conjunto e as soluções por ela dominadas são removidas.

4 TESTES COMPUTACIONAIS

O algoritmo foi implementado em linguagem C++, com o ambiente de desenvolvimento Eclipse. Os testes foram executados em uma máquina com 2 processadores Intel(R) Xeon(R) E5506 de 2.13 GHz, com 16 GB de memória RAM e sistema operacional GNU/Linux, distribuição Ubuntu Server 10.04 de 64 bits.

Para testar a eficiência do algoritmo proposto, foi utilizado um conjunto de 14 instâncias [25], comumente encontradas na literatura para o VRPSPD, e com número de consumidores variando entre 50 e 199. Antes de executar os testes com estas instâncias, foram realizados experimentos preliminares para que os parâmetros pudessem ser ajustados. Através destes experimentos, o número de iterações do algoritmo ($maxIter$) foi definido como 100 e o número de iterações permitidas sem a ocorrência de soluções não dominadas ($maxCont$) foi definido como 18.

4.1 MÉTRICAS DE DESEMPENHO

A análise da qualidade dos resultados para problemas multiobjetivo é mais complexa que para problemas com um único objetivo, pois é avaliado um conjunto de soluções ao invés de uma única solução. Portanto, é necessário aplicar algumas métricas de desempenho para analisar algumas características do conjunto de soluções.

As principais características avaliadas em otimização multiobjetivo são: (i) a proximidade da fronteira de Pareto-Ótima; (ii) a cardinalidade, que indica o número de soluções obtidas; (iii) a distribuição, isto é, se as soluções estão distribuídas de maneira uniforme ao longo da fronteira; e (iv) a extensão, ou seja, o intervalo de valores que é coberto pela fronteira para cada objetivo.

Como não se tem conhecimento da Fronteira Pareto-Ótima para o problema aqui abordado, então a primeira característica apresentada não será considerada. Para avaliação das características *iii* e *iv* são utilizadas métricas apresentadas em [26].

4.2 RESULTADOS OBTIDOS

Na Tabela 1 são apresentados o melhor, a média e o pior resultados para avaliação das características citadas anteriormente. A segunda coluna (Cons.) da tabela informa o número de consumidores envolvidos. Os resultados presentes nesta tabela referem-se a 30 execuções do algoritmo.

Tabela 1: Medidas de eficiência

Instância	Cons.	Cardinalidade			Extensão			Distribuição		
		Melhor	Média	Pior	Melhor	Média	Pior	Melhor	Média	Pior
CMT1X	50	13	6,47	1	104,77	35,99	0,00	10,18	4,76	1,00
CMT1Y	50	29	20,03	12	283,49	188,77	146,70	25,07	16,10	7,83
CMT2X	75	24	12,33	4	284,97	94,16	24,73	22,52	10,78	2,67
CMT2Y	75	48	35,43	27	401,77	314,42	210,97	45,96	33,58	24,54
CMT3X	100	30	19,13	12	379,18	125,98	37,69	28,14	16,84	9,82
CMT3Y	100	50	38,97	27	356,16	255,61	167,04	47,25	36,64	24,08
CMT12X	100	37	25,80	18	293,74	168,59	134,99	34,11	23,89	15,41
CMT12Y	100	55	33,43	23	339,87	207,59	130,47	53,44	31,45	21,18
CMT11X	120	71	55,07	35	454,13	339,07	214,33	69,09	52,76	32,82
CMT11Y	120	69	46,50	25	261,96	219,39	172,89	67,47	44,40	23,17
CMT4X	150	35	26,10	16	742,86	261,28	93,01	33,29	23,91	11,78
CMT4Y	150	68	52,43	39	696,62	500,17	367,03	66,45	50,51	36,58
CMT5X	199	43	29,53	18	653,73	293,05	150,16	40,44	27,60	16,47
CMT5Y	199	73	56,40	42	956,75	740,07	552,39	71,11	54,58	39,80

Para a avaliação de cardinalidade do conjunto de soluções, as instâncias com poucos consumidores apresentaram resultados razoáveis, exceto a instância CMT1X, onde o cardinalidade foi muito baixa. Entretanto, para instâncias envolvendo um maior número de consumidores, os resultados obtidos pelo MOILS se mostraram bons, apresentando uma quantidade significativa de soluções não dominadas nos conjuntos retornados.

Em relação à extensão da fronteira retornada pelo algoritmo, assim como na avaliação da cardinalidade, quanto mais complexa a instância, ou seja, quanto maior o número de consumidores envolvidos, maior é o valor retornado. Isso ocorre pois, quanto mais consumidores existirem maior o número de cenários, ou soluções, que podem ser obtidos.

A instância CMT1X, no pior resultado para a extensão, apresentou o valor 0.00. Este resultado é referente à solução que apresentou cardinalidade igual a 1 (pior caso na avaliação da cardinalidade para a instância). A extensão desta fronteira apresenta este valor pois ela é formada por apenas um único ponto no espaço de soluções.

Ao verificar a qualidade da distribuição das soluções ao longo da fronteira, percebe-se que seus resultados estão próximos ao da cardinalidade. Então, fica evidente que o MOILS consegue gerar soluções distribuídas de maneira uniforme pela fronteira, representando bem as configurações possíveis entre custo e coletas.

Esse resultado indica, também, que o MOILS consegue evitar a concentração em determinados nichos do espaço de soluções. Isto deve-se principalmente à técnica de seleção utilizada, baseada na *crowding-distance*.

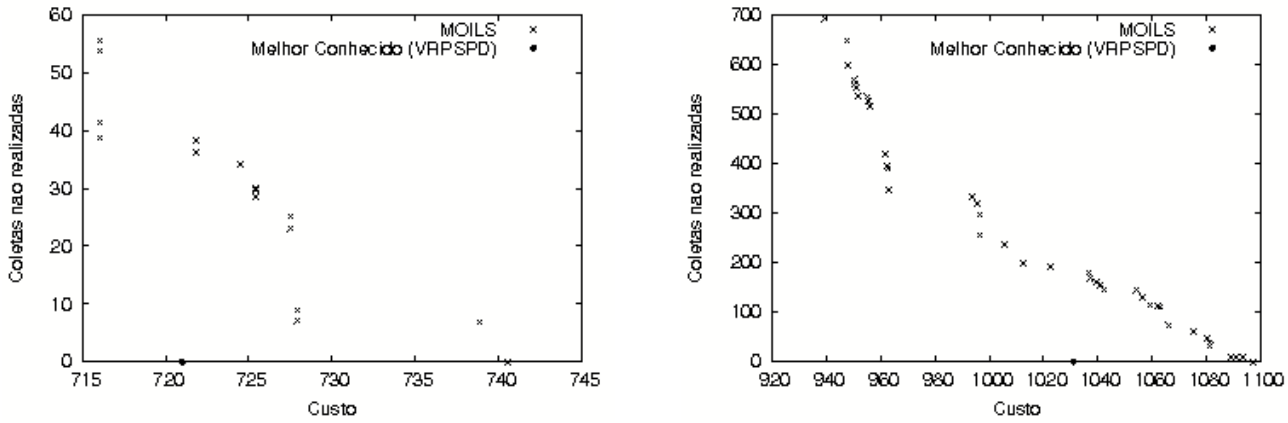
A Figura 1 apresenta as fronteiras encontradas para as instâncias CMT3X e CMT5Y, respectivamente, em uma execução do algoritmo. Apesar da instância CMT3X (Figura 1(a)) apresentar um bom resultado para a qualidade da distribuição, existe um espaço considerável entre as soluções na fronteira. Isso ocorre devido à sua cardinalidade pequena em relação à extensão. Para a instância CMT5Y (Figura 1(b)), que obteve bons resultados para cardinalidade e distribuição, o gráfico apresenta uma fronteira visualmente melhor.

O tempo para execução do algoritmo também é avaliado. A Tabela 2 apresenta o tempo médio, em minutos, para sua execução.

Tabela 2: Média de tempo para execução em minutos

Instância	CMT1X	CMT1Y	CMT2X	CMT2Y	CMT3X	CMT3Y	CMT12X
Tempo (min.)	2,84	6,54	4,93	12,56	26,32	50,08	29,54
Instância	**CMT12Y**	**CMT11X**	**CMT11Y**	**CMT4X**	**CMT4Y**	**CMT5X**	**CMT5Y**
Tempo (min.)	36,06	131,97	86,36	66,31	150,52	122,56	299,47

Devido ao não conhecimento da fronteira Pareto-Ótima, é feita a comparação dos melhores resultados obtidos pelo MOILS com abordagens mono-objetivo. A Tabela 3 apresenta os melhores resultados conhecidos [10] quando todas as coletas são realizadas, e compara com os obtidos pelo MOILS. Ela apresenta ainda, o melhor resultado encontrado considerando apenas a minimização do custo (f_1) e as coletas não realizadas (f_2) associada, e também, o custo associado quando todas as coletas são realizadas.



(a) CMT3X (b) CMT5Y

Figura 1: Gráficos das fronteiras para as instâncias CMT3X e CMT5Y

Tabela 3: Melhores resultados obtidos para ambos objetivos

Instância	**Considerando f_1**		**Considerando f_2 (Coletas 100% realizadas)**		
	Melhor f_1	**f_2 associado**	**f_1 associado**	**Melhor conhecido***	**gap** (%)**
CMT1Y	448,88	149,50	474,27	466,77	1,61
CMT2X	677,29	112,11	692,26	668,77	3,51
CMT2Y	610,03	295,93	716,58	663,25	8,04
CMT3X	716,05	55,44	726,28	721,27	0,69
CMT3Y	689,13	256,73	739,77	721,27	2,56
CMT12X	630,68	106,43	683,85	644,70	6,07
CMT12Y	615,40	248,04	680,72	659,52	3,21
CMT11X	751,73	347,42	867,78	833,92	4,06
CMT11Y	797,80	183,52	878,19	830,39	5,76
CMT4X	838,26	264,75	882,18	852,46	3,49
CMT4Y	783,42	367,95	886,91	852,35	4,05
CMT5X	1015,01	282,09	1087,97	1030,55	5,57
CMT5Y	939,17	691,26	1097,53	1030,55	6,50

* Resultados apresentados em [10].
** Diferença percentual entre os resultados obtidos pelo MOILS e os melhores conhecidos.

Os valores obtidos pelo MOILS, quando atendendo a todas as demandas de coleta, são próximos aos melhores resultados conhecidos na literatura para x das y instâncias utilizadas nos testes. Estes valores próximos permite ao tomador de decisões, em um momento que for desejado atender a todas as coletas, obter soluções de qualidade no que diz respeito ao seu custo.

5 CONCLUSÕES

Neste trabalho é proposto uma adaptação do algoritmo ILS para resolução de problemas multiobjetivo, e sua aplicação no Problema de Roteamento de Veículos com Entrega e Coleta Seletiva Multiobjetivo. Este algoritmo mostrou-se como uma técnica eficiente na geração de soluções não dominadas para o problema, principalmente para instâncias que envolvem um número maior de consumidores.

Os resultados encontrados são fortemente dependentes dos mecanismos utilizados nas etapas de perturbação e busca local, sendo que a utilização de outros mecanismos podem melhorar ainda mais os resultados obtidos. Para instâncias pequenas, onde a qualidade foi inferior, comparado às outras instâncias, pode-se executar o algoritmo por um número maior de iterações, visando a obtenção de uma fronteira mais consistente.

Pelo fato de apresentar uma estrutura genérica, assim como o ILS, a alteração das etapas deste algoritmo não interferem no seu comportamento geral, podendo ser utilizado em outros problemas multiobjetivo.

Assim, trabalhos futuros poderão explorar melhor o conjunto de mecanismos de perturbação e busca local como, também, aplicar o algoritmo proposto a outros problemas de otimização combinatória multiobjetivo.

AGRADECIMENTOS

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro (edital 004/2010/PIBIT-CNPq-UFVJM e projeto PQ:306910/2006-3), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão de bolsa de estudo através do Programa de Fomento à Pós-Graduação (PROF) e à Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) que possibilitaram a realização deste trabalho.

REFERÊNCIAS

[1] Associação Brasileira de Logística. "Associação Brasileira de Logística". www.aslog.org.br, 2008. Acessado em Abril de 2011.

[2] H. Min. "The multiple vehicle routing problem with simultaneous delivery and pickup points". *Transportation Research Part A: General*, vol. 23A, no. 5, pp. 377–386, 1989.

[3] M. Dell'Amico, G. Righini and M. Salani. "A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection". *Transportation Science*, vol. 40, pp. 235–247, 2005.

[4] F. A. T. Montané and R. D. Galvão. "A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and deliveriy service". *Computers & Operations Research*, vol. 33, pp. 595–619, 2006.

[5] J.-F. Chen. "Approaches for the vehicle routing problem with simultaneous deliveries and pickups". *Journal of the Chinese Institute of Industrial Engineers*, vol. 23, no. 2, pp. 141–150, 2006.

[6] N. A. Wassan, A. H. Wassan and G. Nagy. "A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries". *Journal of Combinatorial Optimization*, vol. 15, no. 4, pp. 368–386, 2007.

[7] N. Bianchessi and G. Righini. "Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery". *Computers & Operations Research*, vol. 34, pp. 578–594, 2007.

[8] L. M. B. Freitas and F. A. T. Montané. "Metaheurísticas VNS-VND e GRASP-VND para problemas de roteamento de veículos com coleta entrega simultâneas". *Simpósio de Pesquisa Operacional e Logística da Marinha*, 2008.

[9] A. Subramanian, L. S. Ochi and L. dos Anjos Formiga Cabral. "An efficient heuristic for the vehicle routing problem with simultaneous pickup and delivery". Technical report, Instituto de Computação, Universidade Federal Fluminense, 2008.

[10] A. Subramanian, L. Drummond, C. Bentes, L. Ochi and R. Farias. "A parallel heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery". *Computers & Operations Research*, vol. 37, no. 11, pp. 1899 – 1911, 2010. Metaheuristics for Logistics and Vehicle Routing.

[11] E. I. Gökçe. "A revised ant colony system approach to vehicle routing problems". Master's thesis, School of Engineering and Natural Sciences, Sabanci Univerity, 2004.

[12] Y. Gajpal and P. Abad. "An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup". *Computers & Operations Research*, 2009.

[13] A. L. Maravilha, V. W. C. Morais, L. Assis and A. Vivas. "Comparação entre duas abordagens bio-inspiradas aplicadas ao problema de roteamento de veículos com coleta e entrega simultâneas". *XLII Simpósio Brasileiro de Pesquisa Operacional*, 2010.

[14] I. Gribkovskaia, G. Laporte and A. Shyshou. "The single vehicle routing problem with deliveries and selective pickups". *Computers & Operations Research*, vol. 35, pp. 2908–2924, 2008.

[15] H. Süral and J. H. Bookbinder. "The single-vehicle routing problem with unrestricted backhauls". *Networks*, vol. 41, no. 3, pp. 127–136, 2003.

[16] X.-L. Liao and C.-K. Ting. "An evolutionary approach for the selective pickup and delivery problem". In *Proc. IEEE Congress Evolutionary Computation (CEC)*, pp. 1–8, 2010.

[17] N. Jozefowiez, F. Semet and E.-G. Talbi. "Multi-objective vehicle routing problems". *European Journal of Operational Research*, vol. 189, pp. 293–309, 2008.

[18] E. Angel, E. Bampis and L. Gourvès. *Approximation in Multiobjective Problems*, chapter 28, pp. 28.1–28.15. Chapman & Hall/CRC computer & information science, 2007.

[19] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan. "A fast and elitist multiobjective genetic algorithm: NSGA-II". *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182—197, 2002.

[20] A. Corberán, E. Fernández, M. Laguna and R. Martí. "Heuristic solutions to the problem of routing school buses with multiple objectives". *Journal of the Operational Research Society*, vol. 53, no. 4, pp. 427–435, 2002.

[21] A. H. F. Dias and J. A. Vasconcelos. "Multiobjective genetic algorithms applied to solve optimization problems". *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 1133–1136, 2002.

[22] N. Jozefowiez, F. Semet and E.-G. Talbi. "An evolutionary algorithm for the vehicle routing problem with route balancing". *European Journal of Operational Research*, vol. 195, pp. 761–769, 2009.

[23] M. J. Geiger. "Foundations of the Pareto Iterated Local Search Metaheuristic". In *MCDM*, pp. 19–23, Grécia, Junho 2006.

[24] H. R. Lourenço, O. C. Martin and T. Stützle. *Iterated Local Search*, chapter 11, pp. 321–353. Kluwer Academic Publishers, 2003.

[25] S. Salhi and G. Nagy. "A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling". *Journal of the Operational Research Society*, vol. 50, pp. 1034–1042, 1999.

[26] E. Zitzler, K. Deb and L. Thiele. "Comparison of multiobjective evolutionary algorithms: empirical results". *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.