FRANCISCO HENRIQUE DE FREITAS VIANA

ALGORITMO PARA O PROBLEMA DE ROTEAMENTO DINÂMICO DE VEÍCULOS COM JANELAS DE TEMPO E TEMPOS DE VIAGEM VARIÁVEIS

Belo Horizonte
05 de março de 2007

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

ALGORITMO PARA O PROBLEMA DE ROTEAMENTO DINÂMICO DE VEÍCULOS COM JANELAS DE TEMPO E TEMPOS DE VIAGEM VARIÁVEIS

Proposta de dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

FRANCISCO HENRIQUE DE FREITAS VIANA

Belo Horizonte
05 de março de 2007

UNIVERSIDADE FEDERAL DE MINAS GERAIS

FOLHA DE APROVAÇÃO

Algoritmo para o Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis

FRANCISCO HENRIQUE DE FREITAS VIANA

Proposta de dissertação defendida e aprovada pela banca examinadora constituída por:

D. Sc. Geraldo Robson Mateus – Orientador
Universidade Federal de Minas Gerais

D. Sc. Guilherme Bastos Alvarenga
Universidade Federal de Lavras (DCC/UFLA)

D. Sc. Sérgio Ricardo de Souza
Centro Federal de Educação Tecnológica (DCC/CEFET-MG)

D. Sc. Sebastian Alberto Urrutia
Universidade Federal de Minas Gerais (DCC/UFMG)

Belo Horizonte, 05 de março de 2007

Resumo

É notório que o custo final das mercadorias no comércio varejista decorre, em grande parte, dos gastos com o transporte de bens. Neste contexto, surge o problema de roteamento de veículos que visa a otimizar as rotas de uma frota que tem a incumbência de prestar serviços de coleta ou de entrega em pontos de demanda. Diante da necessidade do atendimento de requisições efetuadas no decorrer do período de operação da frota, surge o problema de roteamento dinâmico de veículos. Este trabalho traz uma abordagem capaz de fornecer soluções de boa qualidade para o problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis. Foi proposto um arcabouço, baseado em uma heurística de geração de colunas que utiliza um algoritmo evolucionário dinâmico com o intuito de gerar um conjunto de soluções para o referido problema. Associado a este algoritmo, foi utilizada uma formulação matemática do problema de partição de conjuntos que visa a obter a melhor combinação de rotas que atendem às restrições do problema, tendo como base as rotas pertencentes às soluções encontradas pelo algoritmo evolucionário. Foram apresentadas três políticas de roteamento dinâmico para determinar o momento no qual as novas requisições efetuadas são repassadas ao algoritmo evolucionário dinâmico. A política *Online* repassa as requisições logo que são geradas. A política *Por Demanda* aramazena as novas requisições em uma fila e as repassa ao algoritmo dinâmico somente quando a fila atinge o seu limite de capacidade. Já a política *Periódica*, por sua vez, espera um período de tempo fixo entre os repasses de novas requisições ao algoritmo evolucionário dinâmico. O arcabouço foi testado através de uma adaptação das instâncias clássicas de problemas de roteamento de veículos propostas por Solomon (1987). Foram realizados testes das três políticas de roteamento dinâmico apresentadas e os resultados foram analisados através de estatísticas.

Abstract

It is well-know that the final cost of commercial products is mostly due to expenses with their transport. In this context, appears the vehicle routing problem that aims to optimize the routes of a fleet which is responsible of providing pick-up and delivery services. In face of the necessity in attending requests that come during the fleet's operation time, appears the dynamic vehicle routing problem. This dissertation brings an approach to obtain good quality solutions to the time-dependent dynamic time-window vehicle routing problem. A framework was proposed based on column generation heuristic that uses a evolutionary algorithm aiming to generate a solution set to the problem. Moreover, the evolutionary algorithm generates a set of columns, or a set of routes that attends the problem's constraints, for a set partitioning problem formulation. The optimal solution for the set partitioning problem is a good solution for the vehicle routing problem. Three dynamic policies were presented to determinate the moment at which the new requests are sent to the dynamic evolutionary algorithm. The *Online* policy sends the request as soon as they are generated. The *Size Demand* policy stores the new requests in a buffer and send them forward to the dynamic evolutionary algorithm only when the buffer is full. While the *Periodic* policy waits a fixed period of time before forwarding a new requests to the dynamic evolutionary algorithm. The framework was tested with the classic instances proposed by Solomon (1987). It was reported tests using the three policies presented and the results were analyzed by statistics.

Aos meus pais Chaguinha e Viana,
à minha tia Teresinha,
por me incentivarem sempre em todos os momentos da minha vida.
E à Juliana, pois, sem o seu apoio e o seu incentivo, tudo teria sido muito mais difícil.

Agradecimentos

Em primeiro lugar, agradeço a Deus, pelo dom da vida. Agradeço também pela sua luz nos momentos nos quais tive que tomar decisões importantes na minha vida.

Agradeço aos meus pais (Viana e Chaguinha), pelo carinho, pelo amor, por tudo que sempre fazem por mim, desde que eu nasci. Agradeço muito por estarem sempre presentes na minha vida, por terem acompanhado cada passo meu e por sempre estarem na retaguarda.

Agradeço à minha tia Tê, pelo seu carinho, pelo seu amor, pela sua dedicação e pelas suas orações diárias.

Agradeço à Vó Lourdes, ao Tio Alvim e à Mazé.

Agradeço a todos os meus professores do curso de Ciência da Computação da Universidade Estadual do Ceará-UECE e, em especial, ao Prof. Marcos José Negreiros Gomes que, mais do que um professor, tornou-se um amigo. Sempre que dele precisamos, ele está pronto a nos atender, apesar de todos os seus compromissos e ocupações.

Agradeço ao meu orientador, Prof. Geraldo Robson Mateus, pelas instruções, pelas conversas, pelo apoio, pela compreensão e pela dedicação ao orientar o meu trabalho.

Agradeço aos membros da banca por terem aceitado prontamente o convite para participarem da comissão examinadora desta dissertação.

Agradeço aos amigos do LaPO - Laboratório de Pesquisa Operacional do Departamento de Ciência da Computação - DCC/UFMG pelo convívio e pelo companheirismo. Agradeço também aos amigos da UFMG que não são do LaPO, mas são agregados, como Alla, Nakamura, etc.

Agradeço a todos os amigos que fiz na UECE por terem me incentivado a vir fazer mestrado na UFMG e por terem me ajudado a tomar esta decisão tão difícil. Agradeço a todos: Diana, Marcos, César, Rosivelt, Joel, Lia, Lisse, Henrique, Jeandro, Alex, Dayvison, Pedro, etc.

Agradeço a todos os amigos da Capela São Judas Tadeu (Comunidade Vila Pery/Maraponga) pelo carinho e pelas orações.

Agradeço de um modo especial à Juliana, companheira de sempre, simplesmente por ser a minha razão de viver, por ser a minha maior motivação para tudo que faço na minha vida, inclusive esta dissertação que, ao lado dos meus pais, a ela dedico.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Este capítulo tem por objetivo explanar acerca da **motivação** (seção 1.1) que impulsionou a realização deste trabalho, bem como expor, em linhas gerais, o **problema** que é abordado nessa dissertação (subseção 1.1.1), com ênfase no **objetivo** que se pretendeu alcançar ao planejá-la (subseção 1.1.2). Por fim, descreve-se, sucintamente, o **conteúdo** de cada capítulo que se segue (seção 1.2).

1.1 Motivação

Grande parcela do custo final de mercadorias ou de serviços deve-se ao gasto com o transporte desses bens até o consumidor final.

Tanto na etapa do transporte de matérias-primas para a fabricação dos produtos, como na distribuição destas mercadorias para centros comerciais, bem como na entrega dos referidos produtos ao consumidor final, há grandes despesas relacionadas ao transporte.

Estes gastos, muitas vezes excessivos por não se adotar um modelo de otimização que vise a minimizar tais despesas, interferem, de forma bastante significativa, nos preços das mercadorias no comércio varejista.

Esta problemática tem incentivado tanto os pesquisadores como as empresas que lidam com transporte a buscarem alternativas que possam vir a melhorar a logística de transporte, no tocante ao uso inteligente de frotas de veículos para o transporte de mercadorias, de passageiros, bem como de lixo coletado em centros urbanos, dentre outras aplicações.

Neste contexto, surge a idéia do *Problema de Roteamento de Veículos*, cujo objetivo primordial é gerar rotas para os veículos de uma frota, visando a minimizar os custos operacionais de transporte, utilizando como métricas a distância total percorrida, a quantidade máxima de veículos utilizados em cada jornada, dentre outras.

Em geral, existem restrições operacionais inerentes ao problema, que dificultam o alcance do seu objetivo. Estas restrições devem-se a fatores, como, por exemplo, a natureza dos bens que são transportados pela frota, bem como a qualidade com que o serviço deva ser prestado, seja ele de entrega ou de coleta.

O *Problema de Roteamento de Veículos - PRV*, quando lida com uma frota de apenas um

veículo, recai no caso particular do *Problema do Caixeiro Viajante - PCV*. Existem diversas variantes deste problema, tais como *Problema de Roteamento de Veículos Capacitado*, que se caracteriza por considerar que os veículos da frota possuem uma capacidade de carga limitada; *Problema de Roteamento de Veículos com Janelas de Tempo*, que considera que cada cliente deve ser servido por um veículo em um intervalo de tempo pré-estabelecido; *Problema de Roteamento de Veículos com Múltiplos Depósitos*, que trata de casos nos quais há várias garagens de onde podem partir os veículos para cumprir uma jornada e para onde devem retornar após tê-la cumprido; *Problema de Roteamento de Veículos com Coleta e Entrega Simultâneas*, que considera a possibilidade de um veículo ter que efetuar serviço de coleta e de entrega na mesma jornada; *Problema de Roteamento Dinâmico de Veículos*, que lida com os casos nos quais não se tem conhecimento de todas as requisições no início do período de operação dos veículos, de modo que requisições efetuadas, enquanto a frota já estiver realizando a sua jornada, são alocadas às rotas em curso ou às novas rotas criadas para este fim; *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*, que é uma variante do problema de roteamento dinâmico de veículos, a qual considera que o tempo de viagem entre dois nós de demanda não é constante, ou seja, pode variar de acordo com o momento no qual se trafega em um determinado trecho. Estas variações nos tempos de viagem ocorrem devido a congestionamentos, a chuvas, a acidentes na pista, a obras nas estradas e a outros fatores. Este problema considera também uma janela de tempo de atendimento associada a cada requisição efetuada.

Uma discussão mais detalhada a respeito de algumas das principais variações do *Problema de Roteamento de Veículos*, bem como uma revisão bibliográfica mostrando como este problema vem sendo tratado, é apresentada no capítulo 3.

1.1.1 Problema Abordado

Nesta dissertação, o problema tratado é uma combinação de algumas das variações de *PRV* aqui citadas. Trata-se do *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*.

Neste problema, parte-se das seguintes premissas:

- há uma frota de K veículos em um depósito central antes do início do período de operação;
- há um e somente um depósito central de onde partem os veículos e para onde eles retornam após a jornada;
- ao depósito central é associada uma janela de tempo, que corresponde ao período de operação dos veículos;
- há um conjunto de requisições a serem atendidas no início do período de operação dos veículos;

- novas requisições podem ser efetuadas a qualquer momento do período de operação dos veículos, sendo atendidas no mesmo período, caso sejam efetuadas até um tempo limite estabelecido *a priori*. Caso contrário, as requisições são armazenadas para serem atendidas no período seguinte;
- cada requisição é caracterizada por uma demanda (carga), que corresponde a d unidades de capacidade (peso, volume) que a referida carga irá ocupar no veículo;
- cada requisição é efetuada por um cliente, que possui uma localização na qual o serviço de coleta deve ser efetuado por um dos veículos da frota. Esta localização foi tratada como sendo um par de coordenadas cartesianas *(x,y)*;
- cada requisição deve ser atendida por um e somente um veículo;
- a frota é homogênea, ou seja, todos os veículos possuem o mesmo limite de capacidade Q;
- associada a cada requisição i, existe uma janela de tempo de atendimento $[e_i, l_i]$. A janela de tempo impõe que uma requisição deva ser atendida preferencialmente no intervalo de tempo por ela especificado. Considera-se que as janelas de tempo são flexíveis, ou seja, soluções que as infringem também são consideradas viáveis, todavia, tais soluções são penalizadas proporcionalmente ao tempo de atraso e ao tempo de espera;
- a cada requisição é associada uma previsão de tempo de serviço de coleta. Isto consiste em uma estimativa de quanto tempo o veículo, que irá atender a esta requisição, ficará parado no ponto de demanda especificado pelas coordenadas do cliente que efetuou a referida requisição;
- o tempo de viagem necessário para um veículo realizar um percurso entre dois quaisquer pontos de demanda varia de acordo com o momento da sua partida do ponto de origem ao ponto de destino.

1.1.2 Objetivo do problema

O objetivo do problema é encontrar um conjunto de rotas que satisfaça a todas as restrições do problema, minimizando a distância total percorrida, o número de veículos usados, o tempo total de atraso dos veículos quanto à chegada aos nós de demanda, bem como minimizando o tempo total de espera do veículo nos nós de demanda quando o momento da chegada ao nó antecede o início da respectiva janela de tempo. Trata-se de um problema multi-critério, pois são consideradas ,simultaneamente, as quatro métricas acima citadas.

1.2 Estrutura desta dissertação

Esta dissertação está organizada como se segue. No capítulo 2 são introduzidos alguns conceitos básicos necessários para uma boa compreensão dos demais capítulos que compõem este

trabalho, pois trata de situar o problema abordado dentro da classe de problemas *NP-Difíceis* e de apresentar, em linhas gerais, as técnicas de resolução utilizadas neste trabalho. Este capítulo também apresenta alguns conceitos relacionados às técnicas que foram aplicadas ao problema abordado.

No capítulo 3 são apresentadas as principais variantes do *Problema de Roteamento de Veículos - PRV*, bem como comentários acerca de alguns trabalhos da literatura relacionados a cada uma das derivações de *PRV* listadas.

O capítulo 4 explica detalhadamente a forma como o problema foi abordado nesta dissertação. É apresentado um gerador de requisições, as políticas de roteamento dinâmico de veículos analisadas (Online, Por Demanda e Periódica), o algoritmo evolucionário dinâmico e como ele é utilizado como parte de uma heurística de geração de colunas para se obter soluções de boa qualidade para o problema abordado.

O capítulo 5, por sua vez, destina-se a explicar, pormenorizadamente, o algoritmo evolucionário dinâmico proposto nesta dissertação para o *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*. Este algoritmo foi destacado neste capítulo por ser considerado a etapa mais crucial do procedimento de solução apresentado no capítulo 4.

O capítulo 6 apresenta os cenários de teste aos quais o procedimento de solução descrito no capítulo 4 foi submetido, os resultados obtidos, bem como uma análise estatística desses resultados.

Por fim, o capítulo 7 apresenta a conclusão a que se chegou mediante a análise de tais resultados e as principais idéias de trabalhos futuros, bem como melhorias que podem ser feitas no *arcabouço* apresentado nesta dissertação.

Capítulo 2

Conceitos Básicos

Este capítulo trata de explicar alguns conceitos e técnicas, cujo conhecimento é necessário para um melhor entendimento do trabalho desenvolvido nesta dissertação. Na seção 2.1, define-se a forma mais simples do *Problema de Roteamento de Veículos - PRV*. Na seção 2.1.1, discute-se a complexidade computacional do PRV. Na seção 2.1.2, são listadas algumas aplicações do mundo real, as quais fazem uso do PRV e de suas variações. Finalmente, a seção 2.2 traz uma introdução aos conceitos básicos das técnicas de resolução aplicadas ao PRV que são utilizadas nesta dissertação.

2.1 Problema de Roteamento de Veículos

Dada uma frota de veículos com K veículos, um conjunto de pontos V a serem visitados por esta frota e um depósito central no qual estão estacionados os veículos antes do início da jornada de trabalho, pode-se afirmar que o *Problema de Roteamento de Veículos* consiste em definir k rotas de veículos, onde uma rota é definida como sendo um ciclo que inicia no depósito central, percorre um subconjunto dos pontos V e termina, evidentemente, no depósito central. A escolha das k rotas se dá de acordo com o critério de otimização. As métricas mais comuns que se deseja minimizar são: distância total percorrida e número de veículos, ou de uma maneira geral, minimizar os custos.

Este conjunto de k rotas deve ser definido considerando-se a condição de que cada um dos pontos $v \in V$ deve pertencer a uma, e somente a uma, das k rotas.

Esta seria a definição mais básica do problema, pois não considera que os veículos têm capacidade limitada, nem que existe demanda de coleta ou de entrega associada a cada ponto.

Este problema foi proposto primeiramente em (Dantzig e Ramser, 1959), onde é apresentada uma formulação matemática para o referido problema.

Muitas variações deste problema, entretanto, já foram definidas na literatura, das quais, algumas, são apresentadas formalmente no capítulo 3.

2.1.1 Complexidade Computacional

O *Problema de Roteamento de Veículos - PRV*, bem como suas variações, são casos mais genéricos do *Problema do Caixeiro Viajante - PCV*. Como o PCV pertence à classe de problemas *NP-Difícil*, por conseguinte, o PRV também é um problema *NP-Difícil* (Thompson e Psaraftis (1993)). Este fato justifica a aplicação de métodos heurísticos a este problema.

2.1.2 Aplicabilidade

O Problema de Roteamento de Veículos tem sido utilizado em diferentes aplicações, na maioria das vezes voltadas para a área de transportes, tais como:

- Distribuição de bebidas em pontos de venda;
- Transporte escolar;
- Distribuição e coleta de dinheiro em caixas eletrônicos;
- Serviços de entrega em geral;
- Coleta de lixo em centros urbanos.

Este trabalho foi projetado para ser aplicado em um projeto de otimização de cadeia de suprimentos, denominado *OTIMAL*, desenvolvido no Laboratório de Pesquisa Operacional - LaPO, vinculado ao Departamento de Ciência da Computação - DCC/UFMG.

2.1.2.1 O Projeto OTIMAL

O Projeto *OTIMAL (Otimização Integrada em Aplicações Logísticas - Produção e Transporte - Projeto CT-INFO/CNPq - DCC/UFMG)* é um projeto do CT-INFO/CNPq que visa a otimizar uma cadeia de suprimentos de uma indústria. O objetivo é otimizar desde a etapa de planejamento da produção, passando pela otimização do sequenciamento de máquinas, pela otimização da distribuição dos produtos nos centros distribuidores, pela otimização do empacotamento dos pedidos de entrega e pelo roteamento de veículos.

Este projeto está em vias de desenvolvimento e já existe um protótipo integrando os módulos de planejamento da produção, sequenciamento e roteamento de veículos.

O trabalho apresentado nesta dissertação enquadra-se no módulo de roteamento de veículos do projeto OTIMAL.

2.2 Técnicas de Resolução

Diversas técnicas têm sido aplicadas às diversas variações do PRV. Desde formulações matemáticas, algoritmos enumerativos, algoritmos aproximativos, bem como métodos heurísticos têm sido utilizados na literatura para se obter soluções satisfatórias para estes problemas.

2.2.1 Algoritmos Heurísticos

O foco deste trabalho é utilizar algoritmos heurísticos para o problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis. Mais especificamente, dentre as diversas opções de algoritmos heurísticos, o objetivo é aplicar algoritmos evolucionários. Por isso, vale destacar os conceitos básicos de algoritmos genéticos, pois, no decorrer do texto, tais conceitos serão imprescindíveis para uma boa compreensão.

2.2.1.1 Algoritmos Genéticos

Algoritmos Genéticos - AG são métodos adaptativos que podem ser usados para resolver problemas de otimização. Eles são baseados nos processos genéticos dos organismos biológicos. Através de várias gerações, populações de indivíduos evoluem de acordo com os princípios da seleção natural, de modo que sobrevivem os indivíduos mais adaptados, princípio definido por Charles Darwin em *The Origin of Species*.

De maneira análoga ao processo de evolução das espécies, algoritmos genéticos são aptos a evoluírem soluções (indivíduos) de problemas combinatoriais. Os princípios básicos dos AGs foram definidos rigorosamente pela primeira vez por Holland (1975) e aprimorados por muitos autores como: Davis (1987), Davis (1991), Grefenstette (1986), Grefenstette (1990), Goldberg (1989) e Michalewicz (1992).

Algoritmos Genéticos fazem uma anologia direta com a genética natural. Tais algoritmos trabalham com uma *população* de *indivíduos*, cada um representando uma solução (nem sempre viável) para o problema a ser resolvido. A cada indivíduo é associado um valor que corresponde à sua aptidão, ou seja, indica quão boa é a referida solução para o problema em estudo.

É dada a oportunidade dos indivíduos se reproduzirem através do operador genético de cruzamento. Os novos indivíduos gerados devem possuir características herdadas de seus ancestrais.

Uma nova população completa de possíveis soluções é então produzida através da seleção dos melhores indivíduos da geração corrente e dos novos indivíduos gerados na fase de cruzamento. Esta nova população traz consigo uma alta proporção de características herdadas dos bem adaptados membros da geração anterior. Dessa maneira, ao longo do tempo, as boas características herdadas de gerações anteriores misturam-se, a ponto de fazer com que a busca por melhores soluções atinja regiões mais promissoras do espaço de busca.

Neste trabalho, optou-se por utilizar um algoritmo evolucionário, o qual se diferencia dos algoritmos genéticos por não possuir uma fase se cruzamento entre os indivíduos da população.

2.3 Conclusão

Neste capítulo, foi definida a versão mais simples do problema de roteamento de veículos. Discutiu-se também um pouco acerca da complexidade do problema, sendo este enquadrado na classe de problemas *NP-Difíceis*. Foi mostrada, de maneira sucinta, em que atividades

econômicas o problema tem tido maior relevância e aplicabilidade. Por fim, foi feito um comentário preliminar sobre os conceitos básicos de algoritmos genéticos, imprescindíveis para o bom entendimento deste trabalho. Algumas referências foram destacadas, de modo a facilitar a busca de mais detalhes destes conceitos.

Capítulo 3

Trabalhos Relacionados

Este capítulo descreve algumas das variações do *Problema de Roteamento de Veículos*, bem como as técnicas e algoritmos apresentados na literatura que visam a buscar melhores soluções para cada variante do problema.

3.1 Problema de Roteamento de Veículos Capacitado - PRVC

Esta variante do problema pode ser definida da seguinte forma: seja um grafo completo $G = (V, E)$, onde $V = v_0, v_1, ..., v_n$ é o conjunto de vértices e $E = e_0, e_1, ..., e_m$ é o conjunto de arestas. O vértice v_0 representa o depósito, enquanto que os demais vértices representam os clientes que efetuaram requisições de coleta ou de entrega.

O custo de viagem entre os vértices i e j é denotado por c_{ij} e assume-se que os custos são simétricos, ou seja, $c_{ij} = c_{ji}\ \forall (i, j) \in E$. Se a frota de veículos tem capacidade idêntica Q, o problema é dito ser *homogêneo*; caso contrário, o problema é *heterogêneo*.

Cada cliente i tem a ele associada uma demanda q_i, onde $0 < q_i \leq Q$. Cada cliente deve ser visitado por um único veículo e nenhum veículo pode atender a um conjunto de clientes cuja demanda total exceda a sua capacidade.

O objetivo do problema é encontrar um conjunto de rotas de custo mínimo, de modo que os veículos iniciem e terminem suas jornadas no depósito.

3.1.1 Formulação Matemática

Seja x_{ij} o número de vezes que um veículo trafega entre os vértices i e j (considerando que x_{ij} e x_{ji} representam a mesma variável). Seja $V_c = V \setminus 0$ o conjunto de clientes (igual ao conjunto de vértices V, excetuando-se o depósito).

Dado um conjunto de clientes $S \subseteq V_c$, $q(S)$ representa o somatório das demandas dos clientes pertencentes ao conjunto S e $\delta(S)$ é o conjunto de arestas incidentes aos vértices pertencentes ao conjunto S.

Uma formulação para o problema é apresentada a seguir:

$$\min \qquad \Sigma_{e \in E}\ c_e\ x_e \tag{3.1}$$

$$Subject \quad to: \ \Sigma_{e\in\delta(i)} \ x_e = 2 \quad \forall i \in V \tag{3.2}$$

$$\Sigma_{e\in\delta(0)} \ x_e = 2K \tag{3.3}$$

$$\Sigma_{e\in\delta(S)} \ x_e = 2K(S) \quad \forall S \subseteq V \tag{3.4}$$

$$x_e \leq 1 \quad \forall e \in E\backslash\delta(0) \tag{3.5}$$

$$x_e \geq 0 \quad \forall e \in E \tag{3.6}$$

O bloco de restrição 3.2 especifica que cada cliente deve ser visitado por apenas um veículo.

As restrições do bloco 3.3 impõem que K veículos devem sair e retornar ao depósito. O bloco 3.4 corresponde às restrições de eliminação de subciclos, as quais especificam, que para qualquer subconjunto S de clientes, $K(S)$ veículos devem entrar e sair de S. Em outras palavras, 3.4 impõe que todos os subconjuntos do conjunto de clientes V deve ser atendido pela quantidade adequada de veículos.

O cálculo de $K(S)$ é um problema *NP-Difícil*, haja vista que implica a resolução de uma instância do *Bin-Packing Problem* (Lysgaard et al. (2003)).

A função objetivo, mostrada em 3.1, indica que o objetivo é minimizar o somatório do produto do custo associado a cada aresta $e \in E$ pela quantidade de vezes que algum veículo trafega por esta aresta. Uma aresta pode nunca ser utilizada caso não pertença a nenhuma rota ou ser utilizada uma única vez no caso em que for alocada a alguma rota. Caso o grafo seja simétrico, uma aresta e pode ainda ser utilizada duas vezes na mesma rota, no caso de se ter, na solução, uma rota com um único cliente a ser visitado.

Lysgaard et al. (2003) apresentam um novo algoritmo *branch-and-cut* para o *PRVC*. Este algoritmo usa uma variedade de planos de corte e, para cada uma das classes de desigualdades, é descrito um algoritmo de separação. Também são descritos importantes procedimentos do *branch-and-cut*, como as regras de *branching*, a estratégia de seleção de nós e o gerenciamento dos cortes. Resultados computacionais mostram que o algoritmo é competitivo. Em particular, foram resolvidas (no ótimo), pela primeira vez, três instancias de Augerat a saber: B-n50-k8, B-n66-k9 and B-n78-k10.

Moghaddama et al. (2006) abordam um *PRVC* no qual a demanda de um nó pode ser dividida em vários veículos. O objetivo é minimizar o custo relacionado à frota e à distância total viajada. O custo relacionado à frota depende do número de veículos utilizados e da capacidade total não utilizada. Moghaddama et al. (2006) apresentam um modelo de *Programação Linear Inteira Mista* para este problema. É mostrada também uma meta-heurística *Simulated Annealing (SA)* para resolvê-lo. A análise apresentada no artigo sugere que o modelo dado consegue atender à quantidade de clientes estabelecida, usando o mínimo número de veículos e a máxima capacidade. O SA pode alcançar soluções ótimas ou próximas da ótima para problemas de pequeno porte. Para instâncias de grande porte, o *gap* médio entre a solução encontrada pela meta-heurística SA e o limite inferior obtido pela formulação matemática é de 25%.

Toth e Vigo (2002) faz uma revisão literária sobre algoritmos exatos baseados em *branch-and-bound*, propostos nos últimos anos para resolver o *PRVC*. É mostrado que tais algoritmos

conseguem resolver instâncias de *PRVC* bem maiores em relação aos problemas que eram resolvidos por abordagens anteriores. Além disso, pelo menos para os casos nos quais a matriz de custos é assimétrica, mostra-se que algoritmos baseados na técnica de *branch-and-bound* ainda representam o *estado da arte* no que que se refere a soluções exatas para o *PRVC*.

Lewis e Sexton (2007) propõem uma heurística para o *PRVC* com demanda unitária. Assume-se que exista uma frota de veículos ilimitada, de modo que, se a frota for fixa, essa heurística pode produzir soluções inviáveis.

Longo et al. (2006) resolvem o *Capacitated Arc Routing Problem–CARP* usando uma transformação deste problema para o *PRVC*. É apresentado um algoritmo *branch-and-cut* para resolver o *PRVC* e os resultados foram significativamente melhores do que os obtidos por métodos exatos aplicados ao próprio *CARP*. Foram melhorados os limites inferiores de quase todas as instâncias que estão ainda em aberto na literatura e, para algumas dessas instâncias, encontrou-se a solução ótima.

Tarantilis (2005) desenvolve um método de programação de memória adaptativa para resolver o *PRVC*. Este método foi denominado *Solutions Elite PArts Search – SEPAS*. Trata-se de um método iterativo que gera uma solução inicial através de uma técnica de diversificação sistemática e armazena suas rotas em uma memória adaptativa. Posteriormente, uma heurística construtiva faz uma junção de componentes dessas rotas que estão na memória adaptativa. Por fim, um procedimento de *Busca Tabu* tenta melhorar a solução construída pela heurística e a memória adaptativa é adequadamente atualizada. SEPAS foi testado em dois *benchmarks* e obteve soluções de boa qualidade em tempo computacional razoável. Comparações com outras meta-heurísticas mostram que SEPAS é um dos métodos mais bem sucedidos aplicados ao PRVC, alcançando resultados que desviam menos do que 0.2% das melhores soluções encontradas na literatura.

3.2 Problema de Roteamento de Veículos Capacitado com Janelas de Tempo - PRVCJT

Além das características já enumeradas na seção 3.1, esta variação do *PRV* pressupõe que, associada a cada requisição, existe uma janela de tempo de atendimento, que pode ser interpretada como sendo o intervalo de tempo no qual a requisição deve ser servida preferencialmente. Soluções que não atendem a uma ou mais janelas de tempo podem ser tratadas de diferentes formas, de acordo com as peculiaridades da aplicação. Eis alguns dos tratamentos possíveis para tais soluções:

- podem ser consideradas inviáveis, quando os requisitos da aplicação impõem que todos os atendimentos devem respeitar as janelas de tempo;
- podem ser consideradas viáveis, mas penalizadas proporcionalmente ao somatório do tempo de atraso de todos os atendimentos que iniciaram posteriormente ao término da respectiva janela de tempo;

- podem ser consideradas viáveis, mas penalizadas proporcionalmente ao somatório do tempo de espera de todos os veículos que chegaram ao local de atendimento anteriormente ao início da respectiva janela de tempo;

- podem ser consideradas viáveis, mas penalizadas proporcionalmente tanto ao somatório do tempo de atraso como ao somatório do tempo de espera nos pontos de demanda;

Dondo e Cerda (2007) apresentam uma heurística de três fases para o *Problema de Roteamento de Veículos com Janelas de Tempo – PRVJT*, considerando múltiplos depósitos e frota heterogênea. Foi introduzido um modelo de Programação Linear Inteira Mista, que é capaz de encontrar a solução ótima para casos com cerca de vinte e cinco nós de demanda. Na fase I, são encontrados os *clusters* viáveis eficientes em termos de custo.

Na fase II, uma formulação matemática é utilizada para atribuir os clusters aos respectivos veículos, bem como para definir a ordem de visitação de cada rota. Na fase III, outro modelo matemático determina o tempo de chegada dos veículos em cada nó de demanda. Várias instâncias da literatura, tanto clusterizadas como aleatórias (em termos de localização dos clientes), considerando janelas de tempo, foram utilizadas para testes e foram resolvidas em tempos computacionais aceitáveis.

Alvarenga et al. (2007) propuseram uma heurística robusta para o *PRVJT* usando a distância total percorrida pela frota como métrica principal. Trata-se de uma heurística de geração de colunas aplicada ao *PRVJT* que faz uso de um *Algoritmo Genético - AG* e de uma formulação matemática para o *Problema de Partição de Conjuntos*. Os resultados obtidos são muito expressivos, estabelecendo melhores soluções para algumas instâncias da literatura.

Bent e Hentenryck (2006) apresentam uma heurística híbrida de duas fases para o Problema de Roteamento de Veículos com Coleta e Entrega e Janelas de Tempo. A primeira fase utiliza uma heurística baseada em *Simulated Anealing* para reduzir o número de rotas, enquanto a segunda fase usa *Large Neighborhood Search - LNS*, para reduzir custo de viagem.

Mautora e Naudina (2007) apresentam vários modelos arco-estados que podem ser aplicados a numerosos problemas de roteamento de veículos, dentre os quais o Problema de Roteamento de Veículos com Janelas de Tempo. Nesses modelos, as variáveis correspondem aos estados dos veículos quando eles trafegam através de um arco. Uma relaxação do problema fornece um limite inferior, que é utilizado no algoritmo *Branch-and-Bound*, aplicado na resolução exata do problema, todavia, para utilizar um número de variáveis e de restrições pseudo-polinomial, geração de linhas e de colunas foram utilizadas. Geralmente, em algoritmos *Branch-and-Bound*, é necessário que se tenha um limite inferior muito eficiente para minimizar o tamanho da árvore de soluções. Nessa abordagem, todavia, embora os limites inferiores tenham decrescido, o tempo computacional reduziu significativamente.

Kallehaugea et al. (2006) consideram um PRVJT ao qual se aplica uma relaxação da restrição que impõe que cada cliente deve ser visitado por um e somente um veículo. Dessa forma, obtém-se um subproblema de caminho mínimo restrito. É apresentado um algoritmo de plano de corte estabilizado dentro de um *framework* de programação linear, resolvendo o problema dual lagrangeano associado. Foram resolvidos dois problemas propostos em 2001,

com 400 e 1000 clientes respectivamente. Até o presente momento, são as maiores instâncias do problema para as quais foram encontradas suas soluções ótimas.

3.3 Problema de Roteamento Dinâmico de Veículos Capacitado com Janelas de Tempo - PRDVCJT

No PRDVCJT, há a possibilidade de algumas requisições surgirem no decorrer do período de operação dos veículos. Desta forma, faz-se necessário realizar um reajuste de rotas, com o intuito de incluir, nas rotas previamente estabelecidas, visitas aos clientes que fizeram as novas requisições. Estas novas demandas são aceitas a qualquer momento, entretanto, serão atendidas no atual período de operação somente as requisições que chegarem até um determinado momento - *deadline* - estabelecido *a priori*.

A literatura existente acerca do Problema de Roteamento Dinâmico de Veículos está mais voltada para o Problema do Caixeiro Viajante Dinâmico (Psaraftis, 1988), (Bertsimas e Ryzin, 1991), (Regan et al., 2002), *Dynamic Traveling Repairman Problem* (Bertsimas e Ryzin, 1991) (Bertsimas e Ryzin, 1993), (Xu, 1994), *Dynamic dial-a-ride problem e Dynamic Vehicle Allocation Problem* (Powell, 1986), (Powell, 1987), (Powell, 1988), (Regan et al., 1996), (Regan et al., 1998), (Cheung e Powell., 1996).

Haghani e Jung (2005) apresenta uma formulação para o Problema de *Roteamento de Veículos Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*. É também mostrado um algoritmo genético para resolver o referido problema. Trata-se de um problema de roteamento de veículos (entrega ou coleta) com janela de tempo, no qual se considera uma frota com múltiplos veículos com diferentes capacidades, requisições de serviços em tempo real, bem como variações em tempo real dos tempos de viagem entre os nós de demanda.

3.4 Problema de Roteamento Dinâmico de Veículos Capacitado com Janelas de Tempo e Tempos de Viagem Variáveis - PRDVCJTTVV

Nesta variante do problema, o tempo que um veículo necessita para percorrer um trecho entre dois pontos de demanda quaisquer varia de acordo com o momento da partida. Este tipo de modelagem visa a buscar uma maior aproximação do que ocorre no mundo real, devido a congestionamentos, acidentes, obras na pista, dentre outros fatores que podem interferir no tempo de viagem entre dois pontos de demanda.

Enquanto muitos pesquisadores têm se dedicado a propor melhores soluções para as vertentes mais genéricas do Problema do Caixeiro Viajante e do Problema da Roteamento de Veículos, pesquisas a respeito do Problema de Roteamento de Veículos com Tempos de Viagem Variáveis, no qual os tempos de viagem variam de acordo com o momento da partida, são mais esparsas (Haghani e Jung, 2005).

A razão da escassez de referências acerca do *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis* deve-se ao fato de que este cenário é mais difícil de modelar e de resolver (Ichoua et al., 2003).

Na literatura, os artigos que fazem referência a este problema (Malandraki, 1989), (Malandraki e Daskin, 1992), (Hill e Benton, 1992), (Malandraki e Dial, 1996), examinam tanto o *Problema de Roteamento de Veículos com Tempos de Viagem Variáveis -PRVTVV* como o *Problema do Caixeiro Viajante Dependente de Tempo – PCVDT*, que se trata de um caso especial de *PRVTVV* quando a frota é composta de somente um veículo. Estes trabalhos mostram também formulações de *Programação Linear Inteira Mista* que incluem restrições de janela de tempo, bem como restrições de capacidade dos veículos. Os tempos de viagem são computados usando *step functions*.

Heurísticas gulosas, tais como a heurística do vizinho mais próximo, são propostas para resolver tanto o *PCVDT* dinâmico como o *PCVDT* estático. No caso estático, não surgem novos pontos a serem visitados após o início do período de operação.

Quando se têm poucos nós de demanda (entre 10 e 25 nós), pode-se utilizar uma abordagem exata, aplicando-se técnicas como o algoritmo *branch-and-cut*. Malandraki e Dial (1996) propõem um algoritmo de programação dinâmica para resolver o *PCVDT*. Malandraki e Daskin (1992), consideram um *PRVDT* sem janela de tempo e é proposto um modelo baseado na velocidade da viagem. São reportados resultados computacionais de exemplos de pequeno porte (cinco nós de demanda).

Apesar da dependência de tempo ter sido, até então, pouco estudada no contexto do Problema de Roteamento de Veículos - PRV, esta abordagem tem sido amplamente aplicada a problemas semelhantes, tais como o problema do menor caminho dependente de tempo (Hill e Benton, 1992), o problema da escolha do caminho dependente de tempo e o problema do caixeiro viajante (Hadley, 1964).

Muitos pesquisadores têm adotado regras simples para integrar componentes dependentes de tempo aos seus modelos. Eles utilizam fatores multiplicadores para representar variações no tempo de viagem (Shen e Potvin, 1995). Em geral, trabalha-se com variações nas velocidades médias dos veículos a cada fatia de tempo do período de operação. Claramente, trata-se de uma aproximação não muito eficiente das condições reais de tráfego.

Neste tipo de formulação, o horizonte de interesse é discretizado em pequenos intervalos de tempo. O tempo de viagem e a função custo para cada *link* são calculados em função do tempo em que cada veículo parte do nó de origem (Ichoua et al., 2003). Esta estratégia tem sido amplamente aplicada em muitos tipos de problemas de transporte dependentes de tempo (Chabini, 1997). Entretanto, sabe-se que, no mundo real, o tempo de viagem varia continuamente ao longo do tempo. Por outro lado, destacam-se algumas limitações para a abordagem que utiliza funções de tempo de viagem contínuas, a saber:

- estas funções são ainda aproximações do que é observado no mundo real;
- geralmente são feitas suposições para que se obtenha modelos mais fáceis de se lidar (Ichoua et al., 2003).

Ichoua et al. (2003) propõem uma metodologia de solução do PRVJT, usando *Tabu Search*. Resultados experimentais mostram que a modelagem dependente do tempo implica ganhos significativos se comparados com os modelos que utilizam *step functions* e com os modelos que não consideram a dependência de tempo.

Haghani e Jung (2005) apresenta uma formulação para o *PRDVJTTVV*. É também proposto um algoritmo genético para resolver o problema. É tratado um problema de coleta ou entrega com janelas de tempo flexíveis. Considera-se, também, uma frota com diferentes capacidades, requisições de serviço em tempo real e variações também em tempo real do tempo de viagem gasto entre os nós de demanda.

Chen et al. (2006) formulam um problema de roteamento de veículos dependente do tempo com janelas de tempo através de uma série de modelos de programação linear inteira mista, que consideram tempos de viagens variando em tempo real, bem como demandas solicitadas também em tempo real. Nesta nova variante do problema, não se tem o conhecimento de todas as informações necessárias para o planejamento de rotas, no início do período de operação dos veículos.

A função que define a variação do tempo de viagem em função do momento da partida também varia em tempo real. Esta estratégia é uma forma de simular a ocorência de incidentes inesperados, como acidentes, chuvas fortes, dentre outros. São definidos vários submodelos de Programação Inteira Mista, sendo cada um desses submodelos um caso especial de um PRVJT em um determinado momento em que os tempos de viagens e/ou as demandas tiverem sido alteradas.

Esta dissertação trata desta variante do problema discutida nesta seção.

3.5 Conclusão

Nesta revisão bibliográfica, observou-se que existem na literatura soluções de boa qualidade para diversas variações do problema de roteamento de veículos, bem como para suas variantes que consideram capacidade limitada dos veículos e janelas de tempo. Verificou-se que ainda há uma lacuna no que se refere a variações do problema que lidam com o aspecto dinâmico, nas quais não se tem todas as informações para a resolução do problema no início do período de operação da frota. São escassos, também, materiais acerca de abordagens que considerem os tempos de viagem variáveis.

Consegue-se perceber, entretanto, que soluções que obtiveram bons resultados, quando aplicadas ao problema estático, podem ser adaptadas ao contexto do problema dinâmico e podem vir a preencher estas lacunas.

Capítulo 4

Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis

Este capítulo tem por objetivo descrever o *arcabouço* desenvolvido para ser aplicado ao problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis, mostrando todas as etapas que integram o procedimento de solução proposto. Na seção 4.1, apresenta-se em linhas gerais o *arcabouço*.

Em seguida, mostra-se o seu modelo de funcionamento (seção 4.2). Nas subseções seguintes, o modelo é detalhado. A subseção 4.2.1 explica como funciona a aplicação cliente, que gera as requisições em tempo real para o algoritmo dinâmico. A subseção 4.2.2 trata de explanar acerca das políticas de roteamento dinâmico propostas. Em seguida, é descrita a entidade denominada Otimizador, responsável pelo controle do funcionamento do *arcabouço* (subseção 4.2.3).

A seguir, define-se o conceito de ilhas de evolução, bem como o seu funcionamento (subseção 4.2.4). Na subseção 4.2.5 é discutido o Problema de Partição de Conjuntos e como este problema enquadra-se no *arcabouço* apresentado. Em seguida, apresenta-se a fase deste procedimento de solução que faz uso de um problema reduzido e justifica-se o porquê de se ter criado esta etapa (4.2.6). Por fim, apresenta-se uma conclusão que traz consigo algumas considerações finais acerca do *arcabouço* abordado neste capítulo.

4.1 Introdução

Este capítulo descreve um *arcabouço* desenvolvido para obter-se soluções de boa qualidade para o *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis - PRDVJTTVV*.

O *arcabouço* é parametrizável, podendo ser utilizado para resolver outras variantes de PRV que não considerem janelas de tempo, tempos de viagem variáveis, bem como o aspecto dinâmico.

Para que sejam relaxadas as restrições de janelas de tempo das requisições, pode-se configurar as janelas de tempo de todas as requisições como sendo igual à janela de tempo do depósito central, ou seja, igual ao período de operação dos veículos que inicia no tempo zero(0) e perdura até o término da janela de tempo do depósito.

Para que o caráter dinâmico do prolema seja relaxado, ou seja, para que todas as requisições sejam previamente conhecidas, basta que se configure o parâmetro α (grau de dinamismo) para 0%. Deste modo, o algoritmo dinâmico não irá receber novas requisições durante o período de operação da frota e irá tratar o problema como um *Problema de Roteamento de Veículos Estático*.

Existe um modelo de tráfego, acoplado ao *arcabouço*, responsável por regular os tempos de viagens entre quaisquer dois pontos de demanda i e j, onde os pontos i e j indicam duas localizações de clientes que efetuaram requisições. O modelo de tráfego utilizado nos testes apresentados neste trabalho é descrito no capitulo 5.

4.2 Modelo do Arcabouço

Nesta seção, descreve-se, em alto nível, o funcionamento geral deste *arcabouço*, cuja descrição mais detalhada de cada uma de suas fases é apresentada nas seções subsequentes deste capítulo.

Para executar-se uma simulação de otimização de rotas de uma frota, faz-se necessário o desenvolvimento de uma aplicação cliente, que gere as requisições em tempo real, no decorrer do período de operação dos veículos. Este período corresponde à janela de tempo do depósito central, ou seja, é o período que começa no instante em que o primeiro veículo ou o primeiro conjunto de veículos parte do depósito até o tempo limite que eles têm para retornar à garagem.

A Figura 4.1 mostra o arcabouço desenvolvido para resolver o *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*.

4.2.1 Gerador de Requisições e Consumidor de Requisições

Para efeito de teste, neste trabalho é apresentada uma aplicação que faz uso das instâncias clássicas de PRV propostas por Solomon (1987). Entretanto, o *arcabouço* pode ser facilmente utilizado por uma aplicação de uma empresa de transportes real, que receba requisições de coleta no decorrer do período de operação da frota.

O funcionamento da aplicação pode ser compreendido como um problema clássico de *Produtor X Consumidor*. O *Produtor* é representado por uma *Thread*, que gera requisições em tempo real em uma fila compartilhada por esta e por outra *Thread*, a qual representa o *consumidor*. O *Consumidor* pertence ao *arcabouço* e, executando em paralelo com o *Produtor*, verifica periodicamente se há novas requisições na fila. Em caso afirmativo, a *Thread* consumidora repassa as novas requisições para a *Política de Roteamento Dinâmico de Veículos* e esvazia a fila.

É importante destacar que o acesso à fila de novas requisições é controlado por *semáforo*, não permitindo, desta forma, que a *Thread* produtora e a consumidora acessem à fila, região

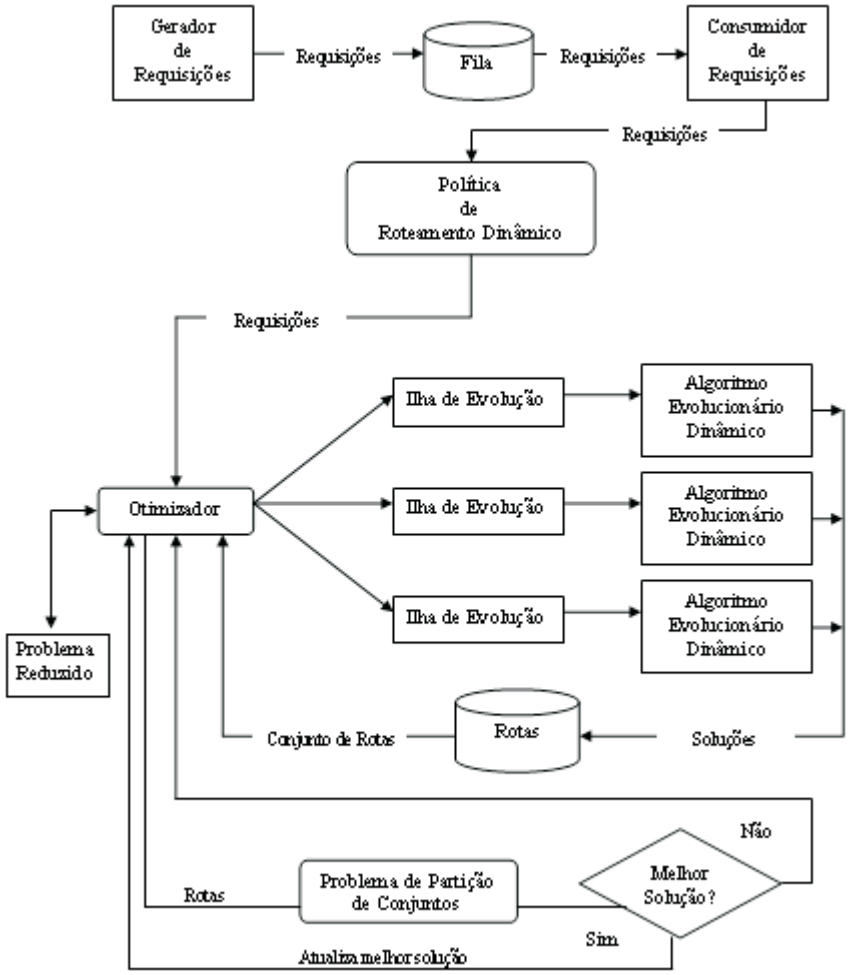

Figura 4.1: *Modelo geral do arcabouço.*

crítica, simultaneamente.

O repasse das novas requisições ao Otimizador não pode ser feito diretamente pela aplicação cliente, que gera as requisições, pois esta realimentação ocorre no momento em que há um reajuste de rotas e este momento é definido pela política de roteamento dinâmico de veículos adotada. Ao Otimizador cabe o papel de gerente ou coordenador de todo o processo de otimização. Tais políticas pertencem ao *arcabouço* e serão descritas na subseção subsequente (4.2.2).

4.2.2 Políticas de Roteamento Dinâmico de Veículos

Nesta seção, são descritas políticas que determinam o momento no qual o *Otimizador* deve receber as novas requisições que porventura tenham sido efeutadas e, a partir de então, procurar soluções para o PRV, considerando, evidentemente, as novas requisições recebidas. O *Otimizador* é a entidade responsável por controlar o funcionamento do *Algoritmo Dinâmico*.

Isto implica um reajuste no grafo do problema, pois faz-se necessária, no instante em que as novas requisições são repassadas ao algoritmo dinâmico, a inserção de novas arestas conectando as novas requisições (vértices) aos vértices já existentes no grafo. Assim, ao término deste ajuste, tem-se um grafo completo.

Caso já houvesse uma lista de clientes previamente cadastrados, quando surgissem novas requisições efetuadas por tais clientes, não seria necessário realizar a operação de reajuste do grafo a cada vez que se adiciona uma nova requisição ao problema. Todavia, neste trabalho, não se considera que já existem clientes cadastrados, daí a necessidade deste reajuste. Em um primeiro momento, não seria necessário que se criassem arestas conectando o novo vértice (cliente que efetuou uma nova requisição) a todos os demais vértices, para que a nova requisição fosse inserida no problema. Entretanto, optou-se por trabalhar com grafo completo para facilitar as operações de troca inter-rotas e intra-rotas que são efetuadas pelos algoritmos de mutação e de cruzamento.

4.2.2.1 *Online*

Nesta política, todas as requisições que são efetuadas são imediatamente repassadas ao *Otimizador*. Este, por sua vez, trata de realizar a incorporação das novas requisições ao conjunto de requisições do algoritmo dinâmico.

4.2.2.2 Por Demanda

Nesta política, as requisições são repassadas ao *Otimizador* somente quando a fila que armazena as novas requisições atinge uma quantidade de requisições definida *a priori* nas configurações do *arcabouço*.

4.2.2.3 Periódica

Esta política consiste em armazenar as novas requisições na fila durante um período de tempo pré-estabelecido. Ao término do período, as requisições armazenadas são repassadas ao *Otimizador* e tem início um novo *round*.

4.2.3 O Algoritmo Dinâmico

O Algoritmo Dinâmico inicia sua execução a partir do momento que se conhece o depósito central e um conjunto inicial de requisições a serem atendidas em um dado período de operação.

Por ser um algoritmo dinâmico, ele encontra uma solução inicial viável para o problema e continua procurando por melhores soluções, ao passo que a solução inicial já é repassada para os caminhoneiros.

Obviamente, que o primeiro destino de cada veículo da frota, definido na solução inicial, será mantido, hava vista que os veículos já deixaram o depósito viajando nesta direção.

O algoritmo parte do pressuposto de que o tempo disponível para ele encontrar uma solução melhor do que a solução inicial é o intervalo de tempo que tem início no momento

em que a frota parte do depósito até o instante em que um dos veículos tenha concluído sua primeira visita. Neste instante, o algoritmo atualiza as rotas, caso tenha encontrado uma solução de melhor qualidade do que a solução corrente, e passa a considerar que as rotas estão fixas até o segundo cliente a ser visitado. O processo repete-se até que se conclua o período de operação.

4.2.4 As ilhas de Evolução

Com o intuito de atingir uma maior região do espaço de busca de soluções e sabendo-se que a aleatoriedade influi na região que é explorada pelo algoritmo evolucionário, foi utilizada a estratégia de submeter a mesma instância do problema a n_i ilhas de evolução. Cada ilha de evolução consiste em gerar uma solução para o PRDVJTTVV através de um algoritmo evolucionário.

Cada ilha corresponde a uma *Thread*, haja vista que são processos independentes, ou seja, uma evolução não interfere na outra. Em cada uma dessas ilhas de evolução, pode-se executar uma meta-heurística. Neste trabalho, foi utilizado o algoritmo evolucionário apresentado no 5.

Ao término da execução de todas as ilhas de evolução, tem-se n_i soluções.

4.2.5 Problema de Partição de Conjuntos - PPC

Todas as rotas das n_i soluções encontradas pelas ilhas de evolução são adicionadas a um conjunto global de rotas *RouteSet*.

Utiliza-se então um modelo matemático do *Problema de Partição de Conjuntos* para selecionar o subconjunto de *RouteSet* que atende a todas as requisições com custo mínimo.

A união do algoritmo evolucionário dinâmico, que gera rotas (colunas) de boa qualidade, com o *PPC* que utiliza as rotas geradas pelo AE e encontra a melhor combinação delas, é denominada heurística de geração de colunas para o problema de roteamento dinâmico de veículos.

4.2.5.1 Formulação Matemática

Esta seção apresenta a formulação matemática do Problema de Partição de Conjuntos utilizada na heurística de geração de colunas que é aplicada ao problema abordado.

$$\min \quad \Sigma_{r \in RouteSet} \; c_r x_r \;\; \textit{Subject to:} \tag{4.1}$$

$$\Sigma_{r \in RouteSet} \; \sigma_{ir} \;\; x_r = 1 \quad \forall i \in C \tag{4.2}$$

$$x_r \in 0, 1 \tag{4.3}$$

No qual:

- C é o conjunto de clientes que efetuaram requisições;

- c_r é o custo total associado a uma rota. Este custo considera a distância total percorrida, o tempo de atraso e o tempo de espera dos veículos nos pontos de demanda;
- x_r é uma variável de decisão. Se a rota r faz parte da solução, esta variável assume o valor 1 (um). Caso contrário, o valor assumido é 0 (zero);
- σ_{ir} indica as requisições alocadas à rota r. Este parâmetro tem valor igual a 1 (um) se a requisição i está contida na rota r. Nesta fase, já se sabe quais requisições estão contidas em cada rota, pois esta informação é gerada pelo algoritmo evolucionário;
- *RouteSet* é o conjunto de rotas viáveis geradas pelo algoritmo evolucionário dinâmico.

Pode-se perceber que, se estivessem contidas em *RouteSet* todas as rotas viáveis para o problema, através da aplicação do problema de partição de conjuntos, tería-se a solução ótima para o problema.

4.2.6 O Problema Reduzido

Objetivando diversificar o conjunto de rotas de *RouteSet*, cria-se um problema reduzido a partir do problema original.

O problema reduzido (Figura 4.2) tem um conjunto de requisições a serem atendidas, gerado a partir de escolha aleatória de um subconjunto das requisições atendidas pelo problema original. De forma análoga à resolução do problema original, o problema reduzido também é submetido a n_i ilhas de evolução e a melhor combinação das rotas encontradas por todas as ilhas é obtida através da execução do *Problema de Partição de Conjuntos*.

O objetivo desta fase do arcabouço é encontrar rotas que não contenham todos as requisições que existem no problema original. Isto pode ser útil na fase do Problema de Partição de Conjuntos, visto que a tendência é que se tenha menos interseções de requisições entre as rotas geradas, contribuindo para que haja um maior número de combinações entre elas.

4.3 Conclusão

Neste capítulo, foram apresentadas todas as etapas que compõem o *arcabouço* desenvolvido. Conclui-se que o arcabouço está subdividido em fases bem definidas que podem ser facilmente substituídas, caso haja o desejo de estendê-lo. A frequência, a quantidade, bem como a distribuição das requisições geradas ao longo do tempo podem ser definidas por uma aplicação real de uma empresa que faz uso de transporte para coleta ou para entrega de bens. O *algoritmo evolucionário* pode ser substituído também por uma outra abordagem heurística, como Busca Tabu, ou até mesmo por uma abordagem exata, utlizando técnicas como *branch-and-bound*, por exemplo.

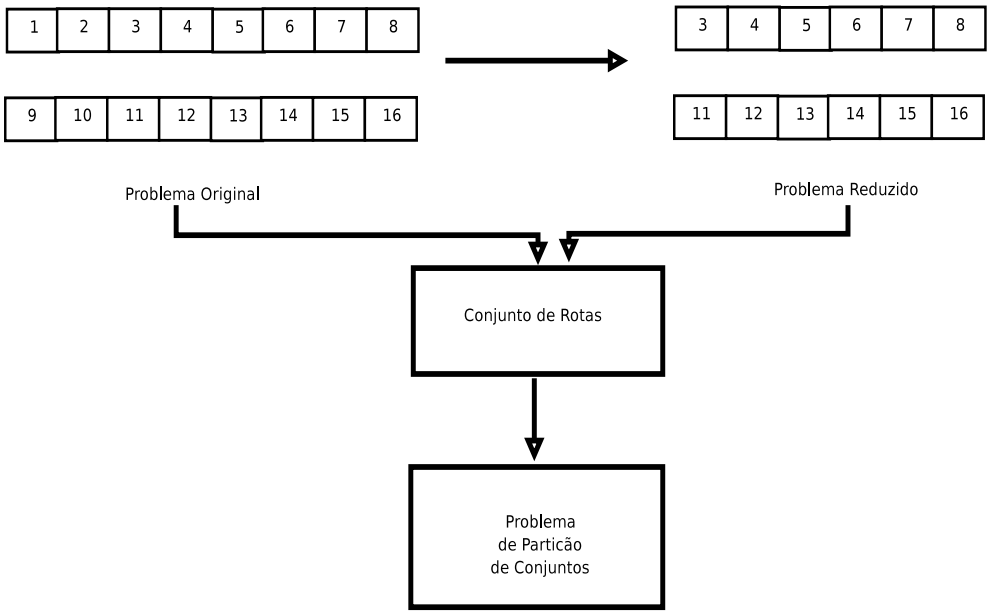

Figura 4.2: Problema Reduzido

Capítulo 5

Algoritmo Evolucionário Aplicado ao Problema Abordado

O algoritmo evolucionário dinâmico desenvolvido tem como objetivo minimizar uma função que considera métricas diversas. Para diferenciá-lo do algoritmo evolucionário tradicional, o mesmo será denominado de Algoritmo Evolucionário Dinâmico. O *arcabouço* é flexível a ponto de permitir que o usuário priorize a minimização da distância ou da quantidade de veículos. São penalizadas as soluções que não respeitam as janelas de tempo associadas às requisições.

A seção 5.1 detalha a estrutura de uma requisição. A seção 5.2 mostra um paralelo entre a representação de um indivíduo do Algoritmo Evolucionário Dinâmico com uma representação voltada para o caso estático. A seção 5.3 destaca a função de avaliação do problema. A seção 5.4 dá uma noção geral do funcionamento do algoritmo evolucionário dinâmico. A seção 5.5 explica como foi implementado o algoritmo que gera a população inicial do algoritmo evolucionário dinâmico. A seção 5.6 trata da função que mede a adaptação dos indivíduos. As demais fases do algoritmo (seleção, geração de novas soluções viáveis, elitismo e mutação) são descritas nas seções 5.7, 5.8, 5.9 e 5.10. A seção 5.11 discute a forma como foi implementada a dependência de tempo. A seção 5.12 discute a inserção de novas requisições em tempo real e a seção 5.13 apresenta uma conclusão objetiva sobre o algoritmo evolucionário dinâmico.

5.1 A Estrutura das Requisições

As requisições podem ser entendidas como pedidos de coleta que são efetuados a uma empresa, que atua direta ou indiretamente na área de transporte. Assume-se que uma requisição possui os seguintes atributos:

- um identificador id;
- um tempo no qual a requisição foi efetuada $registrationTime$;
- uma demanda de coleta q_i;

- um par ordenado de coordenadas cartesianas (x, y), que indica a localização do cliente que efetuou a requisição;
- uma janela de tempo de atendimento $[e,l_i]$;
- uma previsão de tempo de serviço.

O Problema é representado como um grafo completo bi-direcionado $G(V, E)$, no qual V e E crescem dinamicamente, à medida que novas requisições são efetuadas e incorporadas pelo algoritmo.

5.2 Representação dos Indivíduos e dos Cromossomos

Adaptando a estrutura dos algoritmos evolucionários ao problema de roteamento de veículos, pode-se mapear suas respectivas entidades da seguinte forma:

- uma solução do problema de roteamento de veículos é mapeada para um indivíduo;
- uma população pode ser compreendida como sendo um conjunto de indivíduos, ou seja, um conjunto de soluções;
- cada indivíduo é composto por um conjunto de cromossomos, onde cada cromossomo representa uma rota, que faz parte de uma solução;
- cada cromossomo contém um conjunto de genes que representam as requisições a serem atendidas em uma determinada rota.

O depósito central é omitido desta representação, haja vista que o problema em si já determina que todas as rotas iniciem e terminem no depósito. Assim, por economia de estrutura de armazenamento, o depósito não é representado em cada cromossomo.

Para adaptar esta estrutura ao contexto dinâmico do problema, são necessários alguns ajustes:

1. A partir de um certo tempo, pode-se ter uma estimativa da localização de cada veículo em trânsito, uma vez que as distâncias entre quaisquer dois pontos de demanda são conhecidas e pode-se estimar o tempo de viagem entre os referidos pontos através de uma função que retorna o tempo de viagem, utilizando, como parâmetros, o tempo de partida, o ponto de origem i, o ponto de destino j e a distância d_{ij};
2. De posse da localização estimada de cada veículo, assume-se que o novo depósito ou depósito artificial de cada rota passa a ser o próximo destino de cada veículo, ou seja, se um veículo está entre os nós i e j, considera-se que o novo depósito desta rota é o nó j. Foi dada a este cliente a nomenclatura de novo depósito por ele ser o novo ponto de partida da rota, considerando-se o tempo corrente;

3. As requisições a serem atendidas por uma rota que, na representação dos cromossomos, estão localizadas após o novo depósito, podem ser trocadas de posição dentro da mesma rota ou, até mesmo, podem ser atribuídas a outras rotas. As requisições posicionadas anteriormente ao novo depósito de cada rota, todavia, são tidas como já atendidas e, evidentemente, não podem ser mais trocadas nem de posição nem de rota. O próprio novo depósito também não pode sofrer alteração no seu posicionamento dentro da rota nem ser trocado de rota, porque parte-se do pressuposto de que já há um veículo locomovendo-se na sua direção.

A Figura 5.1 ilustra como seria a representação de um indivíduo no contexto estático, na qual são exibidas três rotas (cromossomos) que compõem uma solução (indivíduo). Na primeira rota, são visitados os clientes 3, 11, 7, 2 e 9. Pela segunda rota, são atendidos os clientes 12, 8, 13, 5 e 10. Por fim, na terceira rota os clientes servidos são: 4, 14, 6, 15 e 1. A Figura 5.2 mostra uma adaptação da representação do indivíduo do problema estático que se adequa ao contexto dinâmico do problema. As requisições que estão marcadas com a cor cinza representam as requisições que já foram atendidas pela respectiva rota (cromossomo). As requisições marcadas em preto correspondem aos *novos depósitos* ou *depósitos artificiais*.

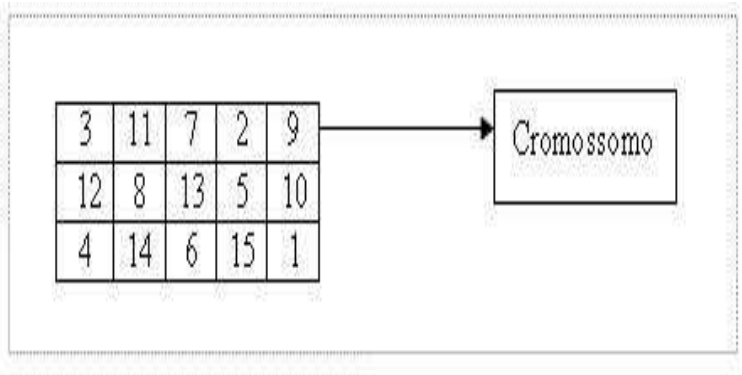

Figura 5.1: Representação de um indivíduo no contexto do problema de roteamento estático de veículos.

5.3 A função de avaliação

Dada uma solução viável, o seu custo pode ser calculado através da função:

$$FuncaoCusto = nv * cv + dt * cd + pa * ta + pe * te \tag{5.1}$$

sendo:

- $FuncaoCusto$: é o custo associado a uma solução;
- nv: é o número de veículos utilizado nesta solução;

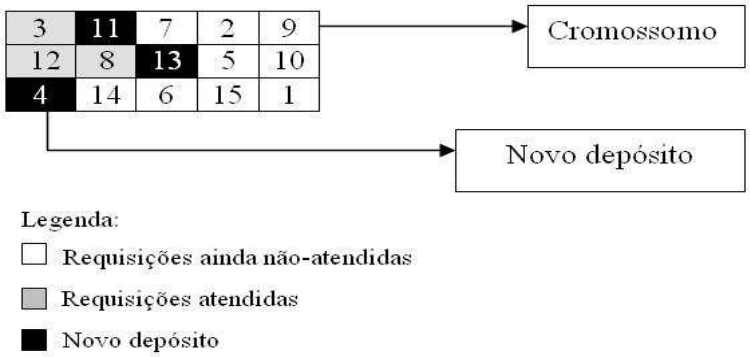

Figura 5.2: Representação de um indivíduo no contexto do problema de roteamento dinâmico de veículos.

- *cv*: é o custo fixo associado a cada veículo usado nesta solução. Este valor é um parâmetro passado ao *arcabouço*;
- *dt*: é a distância total percorrida, considerando-se todas as rotas desta solução;
- *cd*: custo associado a cada unidade de distância percorrida. Este valor é um parâmetro passado ao *arcabouço*; ;
- *pa*: penalidade por unidade de tempo de atraso. Este valor é um parâmetro passado ao *arcabouço*;
- *ta*: tempo total de atraso, considerando-se todos os atendimentos realizados por todos os veículos usados nesta solução;
- *pe*: penalidade por unidade de tempo de espera. Este valor é um parâmetro passado ao *arcabouço*;
- *te*: tempo total de espera, considerando-se todos os atendimentos realizados por todos os veículos usados nesta solução.

Esta função é utilizada pelo Algoritmo Evolucionário Dinâmico para mensurar a aptidão de uma solução. Uma solução do problema abordado pode ser compreendida como sendo um *indivíduo*, no contexto do algoritmo evolucionário.

5.4 Idéia Geral do Algoritmo

O Algoritmo Evolucionário Dinâmico segue o modelo clássico de algoritmos evolucionários como mostra o *Algoritmo* 1. Algumas adaptações foram necessárias, entretanto, para que ele se adequasse ao contexto dinâmico do problema abordado. Este algoritmo é uma *Thread*, disparada por uma ilha de evolução que, paralelamente a outras, executa durante o período de operação dos veículos constantemente em busca de melhores soluções. Quanto mais ilhas

Algoritmo 1: Algoritmo Evolucionário Dinâmico

Entrada: Depósito Central depot e o conjunto inicial de requisições requests
Saida : A melhor solução encontrada VRPSolution

```
1  inicializa o tempo;
2  population ⟵ InitialPopulation(nIndiv );
3  while CurrentTime ≤ EndOfPeriod do
4      if NewRequests ≠ ∅ then
5          UpdateVRP();
6      NewPopulation ⟵ Selection(population, IndivRate);
7      NewPopulation ⟵ NewSolutionsGenerate(NewPopulation);
8      Elitism(NewPopulation, population);
9      Mutation(NewPopulation, MutationRate);
10     atualiza o tempo;
11     UpdateBestSolution(VRPSolution);
```

de evolução executando em paralelo, há uma tendência de se explorar uma maior parcela do espaço de busca de soluções.

Foi considerado que o tempo é um valor numérico que inicia em -1 e é atualizado a cada iteração do algoritmo evolucionário, até alcançar o tempo que marca o término do período de operação dos veículos. No tempo 0 (zero), uma solução inicial já terá sido gerada e os veículos partem, nesse instante, para cunprir suas jornadas.

Na linha 0, são listados os parâmetros que devem ser passados para que o algoritmo inicie o seu processamento. A princípio, devem ser conhecidas as informações a respeito do depósito central (*depot*), tais como coordenadas cartesianas, horário a partir do qual os veículos podem partir do depósito e horário no qual todos os veículos já devem ter retornado ao local de origem. Quando o algoritmo inicia a sua execução, algumas requisições já são conhecidas e outras serão efetuadas no decorrer do período de operação. A lista destas requisições conhecidas *a priori* são também passadas como parâmetro para o algoritmo evolucionário.

Na linha 1, o tempo é inicializado com o valor 0 (zero). Na linha 2, é executado o algoritmo *Dynamic Stochastic Push-Forward Insertion Heuristic* baseado no algoritmo *Stochastic PFIH*, proposto por Alvarenga et al. (2003) e no algoritmo *PFIH*, proposto por Solomon (1987). Este algoritmo será detalhado na seção 5.5.

A linha 3 indica que a condição de parada do algoritmo é o término do período de operação dos veículos. Melhorias podem ser realizadas, entretanto, para que o algoritmo termine, também, caso todos os veículos já tenham retornado ao depósito, desde que o tempo atual seja superior ao horário limite para inserir-se novas requisições durante o período em curso.

Na linha 4, mostra-se que a cada iteração do algoritmo é feito um teste para verificar se há novas requisições disponíveis para serem incorporadas à solução corrente. Vale ressaltar que podem existir requisições já efetuadas, mas que não estão disponíveis para as ilhas de evolução. Isto ocorre quando a política de roteamento dinâmico é *Periódica* ou *Por Demanda*.

As novas requisições disponíveis, caso existam, são neste momento adicionadas ao conjunto

de requisições do problema e uma nova solução é gerada, de modo que estas novas requisições já sejam atribuídas a alguma rota, conforme procedimento descrito na seção 5.12.

Na linha 6 são selecionados os indivíduos que irão participar da fase de *crossover*. Na linha 7 são geradas novas soluções viáveis. A linha 8 consite na fase de elitismo, que garante que os melhores indivíduos irão sobreviver para a próxima geração. Na linha 9 são executados os algoritmos de mutação, que visam a explorar outras regiões do espaço de busca de soluções. Por fim, o tempo é atualizado e depois verifica-se se foi encontrada uma solução melhor do que a melhor soluçào já encontrada. Em caso afirmativo, atualiza-se a melhor solução (11).

5.5 População Inicial

O algoritmo evolucionário inicia com a geração da população inicial. É definido, como parâmetro do *arcabouço*, a quantidade de indivíduos que irão compor a população. Em geral, aplica-se uma heurística gulosa de construção, para se gerar uma solução viável para o problema em tempo polinomial.

Solomon (1987) propôs a heurística *Push-Forward Insertion Heuristic - PFIH* que utiliza uma função para determinar o custo de se atribuir uma nova requisição a uma rota qualquer. Verifica-se que este algoritmo é determinístico e que a escolha da primeira requisição atribuída a uma determinada rota impacta na escolha das demais requisições. Isto se deve à utilização desta função determinística.

Alvarenga et al. (2003) adaptaram a heurística proposta inicialmente por (Solomon, 1987) de modo que a escolha da primeira requisição de cada rota seja efetuada de maneira aleatória, através da aplicação de uma distribuição de probabilidade. Esta adaptação permitiu que fossem obtidas poulações iniciais com um maior grau de diversidade a cada execução do algoritmo evolucionário. Este algoritmo foi aplicado como gerador de população inicial do algoritmo evolucionário proposto por Alvarenga et al. (2003) para o *Problema de Roteamento de Veículos com Janelas de Tempo*. Esta heurística foi denominada *Stochastic Push-Forward Insertion Heuristic - SPFIH*.

Apresenta-se uma heurística que consiste em uma adaptação do algoritmo *SPFIH* para que ele possa ser aplicado no contexto do *Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*. Isto é necessário porque este algoritmo não é utilizado somente na geração da população inicial, mas, também, quando novas requisições precisam ser incorporadas a uma solução, cujas rotas já foram parcialmente percorridas pelos veículos.

O Algoritmo 2 tem como entrada as informações relativas ao deposito central *depot*, as requisições já conhecidas *requests*, o instante atual *time* e a solução corrente $VRPSolution$. No caso da utilização deste algoritmo para se gerar a população inicial, não se tem solução corrente e, portanto, este parâmetro não é passado ao algoritmo.

De acordo com o parâmetro *time*, pode-se facilmente obter uma previsão da localização de cada veículo em trânsito. Neste caso, quando o algoritmo é utilizado na geração da população inicial, este passo não tem efeito, pois o tempo atual é igual a 0 (zero).

Algoritmo 2: Stochastic Dynamic Push-Forward Insertion Heuristic - SDPFIH

Entrada: Depósito Central depot, o conjunto inicial de requisições requests, instante atual time, a solução atual VRPSolution e o número de indivíduos a serem gerados nIndiv
Saida : População population com nIndiv indivíduos

```
1  atualiza o novo depósito de cada rota de VRPSolution de acordo com time;
2  population ⟵ CreatePopulation;
3  for i ⟵ 1 to nIndiv do
4      Individual ⟵ CreateIndividual;
5      route ⟵ CreateRoute;
6      while existir requisição não atribuída a nenhuma rota do
7          request ⟵ RandomRequest(requests, seed );
8          Add(route, request );
9          BestCost ⟵ INIFINITY;
10         BestPosition ⟵ NULL;
11         BestRequest ⟵ NULL;
12         for i ⟵ 1 to NReq do
13             request ⟵ NReq_i;
14             EvaluateNewRequest(request, route, Cost, Position );
15             if restrição de capacidade não é violada then
16                 UpdateBestCost(Cost, Position, request, BestCost, BestPosition, BestRequest );
17             vai para a próxima iteração;
18         if BestCost = INIFINITY then
19             AddRoute(Individual, route );
20             route ⟵ CreateRoute;
21             Break;
22         AddRequest(route, request, BestRequest, BestPosition );
23         Remove(NReq,request );
24 Add(Individual, population);
```

Neste algoritmo, para cada nova rota:

- seleciona-se aleatoriamente a primeira requisição a ela atribuída;
- avalia-se, nesta rota, a inserção de cada uma das requisições não atríbuídas;
- caso seja viável, insere-se na rota a requisição que menos impactou no custo da rota, na posição que acarreta também o menor custo de inserção;

Este processo é repetido até que todas as requisições tenham sido atribuídas a alguma rota.

5.6 A função de Aptidão dos Indivíduos

Esta função determina quão bem adaptado é um indivíduo. Observa-se que a função de aptidão é inversamente proporcional à função de avaliação (seção 5.3).

Formalmente, pode-se definir a função de aptidão dos indivíduos como sendo:

$$Fitness = 1/FuncaoCusto \tag{5.2}$$

onde a *FuncaoCusto* é a função definida na seção 5.3.

5.7 Seleção

O procedimento de *seleção* seleciona i indivíduos da população para participarem da fase de geração de novas soluções viáveis.

Algoritmo 3: Seleção

Entrada: População atual population, taxa de indivíduos a serem selecionadosRate
Saida : Lista de indivíduos selecionados selected

```
1 count ⟵ Size(population ) ∗ Rate;
2 for i ⟵ 1 to count do
3 |   winner ⟵ competition(population );
4 |   AddIndividual(selected, winner );
```

Este algoritmo recebe, como entrada, a população atual *population* e uma taxa *rate*, ou seja, um percentual dos indivíduos da população que deve ser selecionado ao término deste procedimento. O parâmetro *rate* é definido nas configurações do *arcabouço*.

Na linha 1, é efetuado o cálculo de quantos indivíduos devem ser selecionados. Em seguida, através de um método de torneio *Competition*, os indivíduos são selecionados um a um.

A seleção de um indivíduo pelo método do torneio ocorre da seguinte forma:

- são selecionados, aleatoriamente, n [1] indivíduos da população;
- desses indivíduos selecionados, o vencedor do torneio é aquele que apresentar maior valor de função de aptidão (seção 5.6).

Pode-se observar, nas linhas 2 a 4, que este processo de torneio é repetido até que a quantidade de indivíduos a ser selecionada seja atingida.

5.8 Geração de novas soluções viáveis

Diversos algoritmos têm sido estudados na literatura para serem aplicados na fase de cruzamento de algoritmos genéticos voltados para o *Problema de Roteamento de Veículos*.

A idéia geral da fase de cruzamento mais convencional é utilizar material genético proveniente de dois indivíduos genitores, com o intuito de gerar-se um ou dois novos indivíduos, dependendo do algoritmo utilizado.

O mais natural seria um algoritmo definido basicamente nos seguintes passos:

[1] o valor de n é um parâmetro de configuração do *arcabouço*.

- agrupam-se os indivíduos aos pares;
- para cada par de indivíduos, gera-se um novo indivíduo que herda equitativamente rotas (cromossomos) crompletas dos dois indivíduos ancestrais;
- quando não for mais possível inserir no novo indivíduo rotas completas, procura-se inserir as requisições que ainda não tenham sido atribuídas a nenhuma rota nas rotas já existentes no novo indivíduo. Isto pode ser feito avaliando-se o custo de inserção de cada uma dessas requisições em cada posição, considerando-se todas as rotas existentes no novo indivíduo;
- por fim, quando não for mais possível utilizar nenhuma das alternativas apresentadas nos dois ítens anteriores, novas rotas são criadas para que sejam alocadas as requisições que porventura tenham sobrado;

Esta abordagem tem a vantagem de utilizar o material genético de dois indivivíduos para gerar um novo indivíduo. Entretanto, em virtude da restrição de que uma requisição deve ser atribuída a uma e somente uma rota, pode ocorrer que poucas rotas sejam herdadas por completo pelos indivíduos descendentes. Isto implica que, ao término do algoritmo, os novos indivíduos (soluções) gerados podem não ser de boa qualidade, pois apresentam muitas rotas e um custo elevado.

Buscou-se, neste trabalho, uma alternativa ao método de cruzamento que apresentasse a tendência de gerar indivíduos de boa qualidade. A desvantagem desta abordagem é que o indivíduo gerado herda material genético de apenas um ancestral, proporcionando, assim, uma diversificação menor do que a abordagem citada anteriormente. Vale ressaltar que o algoritmo aqui proposto não é um cruzamento.

Todavia, esta última estratégia tende a proporcionar melhores resultados no que diz respeito à qualidade dos indivíduos (soluções) geradas. Tal abordagem é pormenorizada no Algoritmo 4.

Este algoritmo de geração de novas soluções viáveis recebe, como entrada, uma população de indivíduos (0) e, ao seu término, uma nova geração terá sido gerada. Para cada indivíduo da população original, são selecionadas duas rotas aleatoriamente (linhas 3 e 4).

São selecionados, de maneira aleatória, pontos de corte nas duas rotas selecionadas (linhas 5 e 6). O conjunto de requisições posterior ao ponto de corte da primeira rota selecionada é trocado com o conjundo de requisições posterior ao ponto de corte da segunda rota selecionada, como pode-se verificar nas linhas 7 a 14.

5.9 Elitismo

O *elitismo* consiste em garantir a sobrevivência dos melhores indivíduos da população para gerações futuras. Isto se dá com o intuito de garantir que regiões promissoras do espaço de busca não deixem de ser exploradas em virtude da tendência à diversificação da população, característica inerente ao algoritmo.

Algoritmo 4: Geração de novas soluções viáveis

Entrada: População atual population
Saida : População após a fase de geração de novas soluções viáveis

```
1  for i ⟵ 1 to Size(population ) do
2      Individual ⟵ population_i;
3      Route[1] ⟵ RandomRoute(Individual );
4      Route[2] ⟵ RandomRoute(Individual );
5      Cut[1] ⟵ RandomPosition(Route[1] );
6      Cut[2] ⟵ RandomPosition(Route[2] );
7      for j ⟵ Cut[1] to Size(Route[1] ) do
8          Request ⟵ RemoveRequest(j);
9          Add(RequestList[1], Request );
10     for j ⟵ Cut[2] to Size(Route[2] ) do
11         Request ⟵ RemoveRequest(j);
12         Add(RequestList[2], Request );
13     AddRequest(RequestList[1], Route[2] );
14     AddRequest(RequestList[2], Route[1] );
15     if Route[1] e Route[2] são viáveis then
16         UpdateIndividual(Individual );
```

Neste trabalho, é proposto um *elitismo* em duas fases. Após a fase de geração de novas soluções viáveis do algoritmo evolucionário, a nova população é composta por menos indivíduos do que a população original, haja vista que apenas um subconjunto de indivíduos é selecionado para a fase de geração de novas soluções viáveis. A quantidade de indivíduos selecionados para a fase de geração de novas soluções viáveis e, por conseguinte, a quantidade de novos indivíduos gerados nesta fase é determinada pelo parâmetro $IndivRate$, que se trata de um percentual de indivíduos que provavelmente são selecionados para a fase de geração de novas soluções viáveis.

Neste contexto, os melhores indivíduos da população original são adicionados à nova geração, até que a nova população contenha a mesma quantidade de indivíduos da geração ancestral. Isto consiste na primeira fase do *elitismo*.

A exemplo do que ocorre na primeira fase do *elitismo*, na segunda fase ocorre que, após as mutações, a população está com um número reduzido de indivíduos, de acordo com o parâmetro *mutationRate*. Como na primeira fase, a população é completada com os indivíduos mais promissores provenientes da geração anterior.

5.10 Mutação

Foram implementados alguns algoritmos de mutação que têm o propósito tanto de procurar soluções de melhor qualidade do que as atuais, como de diversificar a população em virtude da necessidade de se ter uma melhor exploração do espaço de busca de soluções, bem como para evitar de se recair em uma solução intermediária de boa qualidade que impeça uma evolução

satisfatória da população.

O primeiro passo, que é comum a todos os algoritmos de mutação que foram utilizados, consiste em, mediante o tempo atual, estimar a localização de cada veículo e, por conseguinte, determinar qual o próximo cliente a ser visitado por cada um deles. Este cliente é denominado de novo depósito, pois, do ponto de vista dos algoritmos de mutação, este cliente é considerado como um ponto de partida. A seguir, os algoritmos de mutação adotados são descritos sucintamente.

5.10.1 Migração aleatória de Requisição

Este operador pode ser definido nos seguintes passos executados para cada indivíduo selecionado para mutação (conforme ilustra a Figura 5.3):

- seleciona-se uma rota aleatoriamente. Desta rota, seleciona-se uma requisição qualquer;
- insere-se a requisição selecionada em outra rota escolhida aleatoriamente;
- se a solução obtida for viável, a alteração é ratificada; caso contrário, a modificação é desfeita.

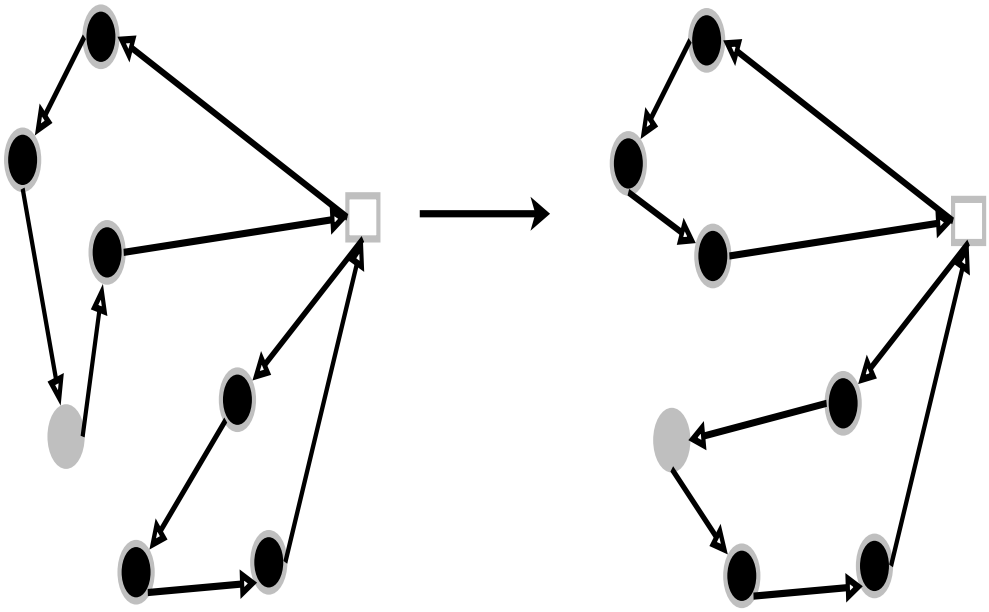

Figura 5.3: Migração Aleatória de Requisições

5.10.2 Acréscimo Mínimo de Custo

Este operador pode ser descrito nos passos a seguir, executados para cada indivíduo selecionado para mutação:

- escolhe-se uma rota *route* do indivíduo, aleatoriamente ;

- procura-se, dentre as requisições das demais rotas, qual a que acarretaria o menor acréscimo de custo, se inserida em *route* e, evidentemente, removida da sua rota de origem;
- efetua-se a migração da referida requisição para *route*.

5.10.3 Troca de Requisições Similares

Este operador evolucionário escolhe uma rota aleatoriamente e uma requisição i qualquer desta rota. Em seguida, busca, nas demais rotas, requisições que possuem janelas de tempo similares. A requisição j que possuir janela de tempo com maior grau de similaridade com a requisição i troca de rota com a requisição i.

O grau de similaridade de janela de tempo de duas requisições i e j é maior, quanto menor for a diferença entre o término de suas respectivas janelas de tempo. A troca das requisições é realizada independente do valor absoluto da diferença mínima encontrada entre duas janelas de tempo. Entretanto, no ato da troca, verifica-se a viabilidade da troca. Se a solução resultante da troca continuar sendo uma solução viável, esta troca é efetivada.

5.10.4 Troca de Requisições com Redução de Custo

Este operador verifica todas as possibilidades de troca de requisições entre duas rotas selecionadas aleatoriamente. A operação de troca é efetuada somente na hipótese de implicar redução no valor da função de avaliação.

5.10.5 Intercalação de Rotas

Este operador escolhe aleatoriamente duas rotas de cada indivíduo da população e tenta realizar uma união entre elas de maneira aleatória. Muitas vezes, algumas requisições sobram, por não ser mais viável inseri-las na rota resultante da intercalação. Neste caso, as requisições remanescentes são inseridas nas outras rotas. Caso também esta operação não seja viável para todas as requisições restantes, a alternativa passa a ser a criação de novas rotas, até que todas as requisições sejam atribuídas a alguma rota. Para a criação de novas rotas utiliza-se o algoritmo usado na geração da população inicial *SDPFIH*.

5.10.6 Divisão de Rotas

Este operador divide uma rota em duas novas rotas. O ponto de corte é gerado aleatoriamente, de modo que as requisições posicionadas anteriormente ao ponto de corte originem uma rota e as demais requisições originem outra nova rota (Figura 5.4).

5.11 Dependência de Tempo

Neste trabalho, considera-se um problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis. É importante, portanto, uma discussão mais por-

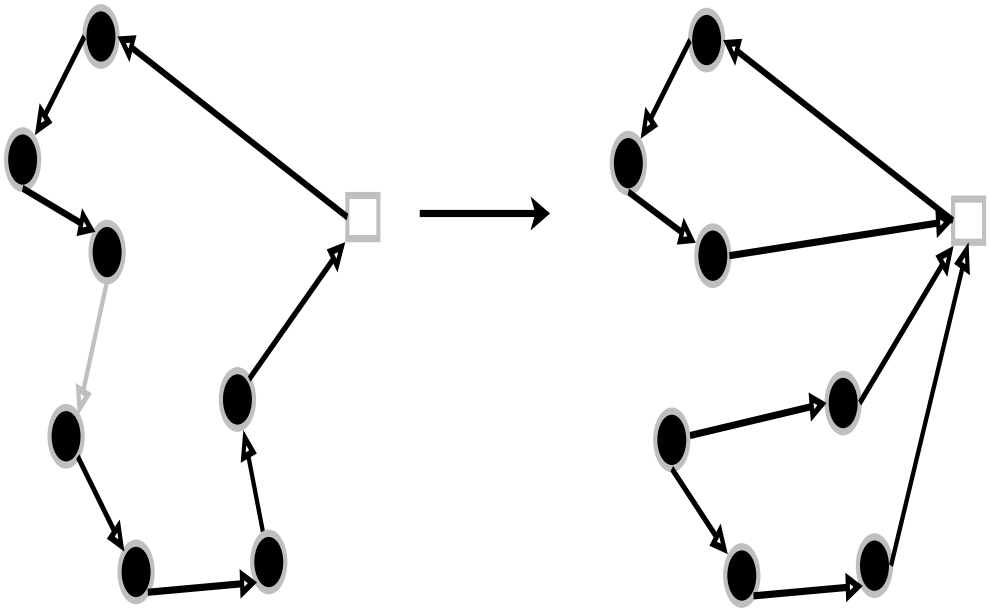

Figura 5.4: Divisão de Rotas

menorizada da forma como o *arcabouço* trata a questão da dependência de tempo, ou seja, como são determinados os tempos de viagem variáveis.

O período de operação da frota é determinado pela janela de tempo do depósito central. Assim, define-se que o período de operação é dado pela expressão:

$$T = l_0 - e_0 \tag{5.3}$$

onde e_0 é o início da janela de tempo do depósito central, ou seja, o momento a partir do qual os veículos podem partir do depósito e l_0 é o término da janela de tempo associada ao depósito, ou seja, é o horário limite para os veículos retornarem ao depósito central.

Nos parâmetros do *arcabouço*, define-se a quantidade de alterações (λ) na previsão de tempo de viagem entre dois pontos de demanda i e j. O período de operação T é dividido em λ estágios e a velocidade média dos veículos varia em cada um dos λ estágios.

Como, evidentemente, a distância entre quaisquer pontos de demanda i e j é constante e como a velocidade média (V_m) dos veículos varia em cada um dos λ estágios que compõem o período de operação λ, verifica-se que os tempos de viagem entre os pontos de demanda i e j irão variar no decorrer tempo.

Verifica-se, todavia, que esta ainda não seria a melhor solução para simular uma previsão de tempos de viagem variáveis, pois a velocidade média dos veículos estaria atrelada apenas ao momento da partidade um vértice i para um outro vértice , quando deveria também estar vinculada ao trecho que se deseja percorrer. j, sendo i e j pertencentes a V. Por este motivo, foi elaborado um modelo mais complexo, que vincula o valor da velocidade média de um veículo ao trecho (i, j) que ele precisa percorrer e ao momento no qual ele parte de i em direção a j. Este novo modelo é dado pela expressão:

$$T_{ij} = d_{ij}/(M * V_m) \tag{5.4}$$

na qual:

- T_{ij} é a estimativa de tempo de viagem entre os pontos de demanda i e j;
- M é um fator multiplicador gerado a partir do trecho (i, j). Este fator associa o tempo de viagem ao trecho que se deseja percorrer;
- V_m é a velocidade média dos veículos considerando-se apenas o estágio (fatia de tempo) do período operação no qual se deseja iniciar o percurso;
- d_{ij} é a distância referente ao trecho (i, j).

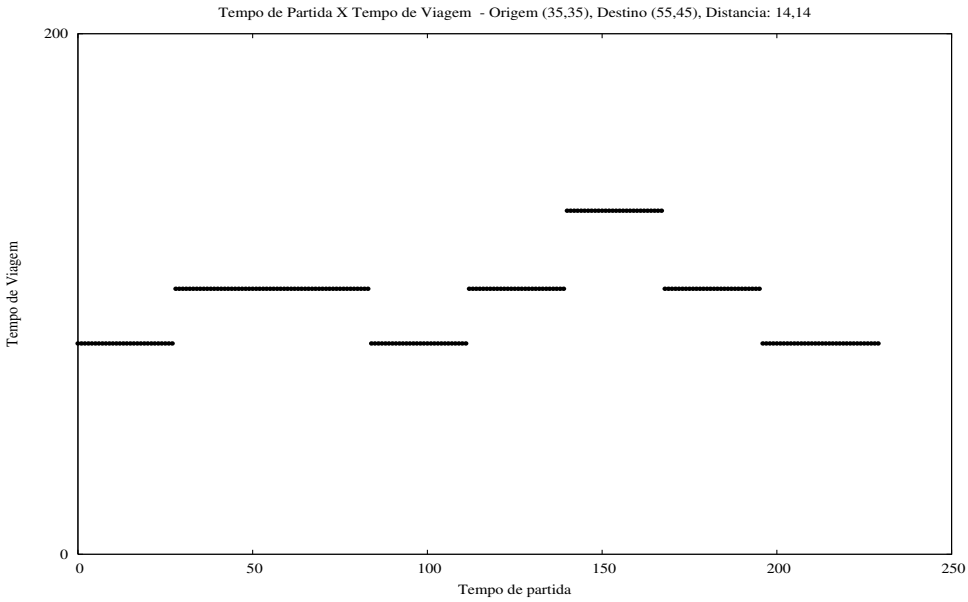

Figura 5.5: Variação dos tempos de viagem no decorrer do tempo

Na Figura 5.5, pode-se verificar que o período de operação dos veículos foi subdividido em vários intervalos de tempo. Cada um desses intervalos está associado a um patamar de velocidade média dos veículos. Percebe-se, portanto, que o tempo de viagem depende do momento da partida, bem como do trecho a ser percorrido.

5.12 Inserção de novas requisições no Algoritmo Evolucionário Dinâmico

Já foi definido que as políticas de roteamento dinâmico trazem consigo regras que determinam quando o Algoritmo Evolucionário Dinâmico deve incorporar as novas requisições

efetuadas, bem como quando deve ser realizada a operação de reajuste de rotas. Esta operação é necessária para que cada uma das novas requisições recebidas pelo algoritmo seja imediatamente atribuída a uma rota.

Nesta seção, discute-se como ocorre esta incorporação de novas requisições por parte do Algoritmo Evolucionário Dinâmico e qual o impacto dessa operação na estrutura de dados.

Em primeiro lugar, a estrutura de dados do grafo completo bi-direcionado precisa ser atualizada. A cada nova requisição incorporada pelo algoritmo, um novo vértice é inserido no grafo. Cada novo vértice inserido, por sua vez, implica a inserção de $n-1$ novas arestas, onde n é a quantidade de vértices (já acrescida do novo vértice) do grafo. Feito isto, os custos das novas arestas são calculados (distância entre os clientes).

A etapa seguinte consiste em adicionar, em cada indivíduo da população atual, cada uma das novas requisições em alguma rota. Esta inserção é feita avaliando-se o custo de inserção da nova requisição em cada posição de cada rota. Ao término da avaliação é verificado se a melhor alternativa é inserir a nova requisição na rota e na posição da rota que irá acarretar o menor custo de inserção ou se é mais vantajoso não inserir a requisição nas rotas existentes e sim, criar uma nova rota para atender a referida requisição.

Esta etapa é repetida até que todas as novas requisições estejam alocadas a alguma rota do indivíduo.

5.13 Conclusão

Neste capítulo, o Algoritmo Evolucionário Dinâmico, considerado como a etapa mais importante do procedimento geral de solução apresentado no capítulo 4 é explicado de forma mais detalhada. Cada fase do algoritmo é apresentada didaticamente por meio de pseudocódigo, facilitando a compreensão e a reprodução dos algoritmos que foram implementados.

Capítulo 6

Resultados Obtidos

Este capítulo apresenta os testes realizados com o arcabouço, considerando-se o problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis. Na seção 6.1, são apresentados os cenários de teste utilizados. Estes cenários foram gerados a partir das instâncias de Solomon (1987). Na seção 6.3, são listados os parâmetros do *arcabouço* que foram utilizados, bem como tabelas mostrando os valores referentes aos resultados obtidos. É apresentada uma análise estatística desses resultados através de médias amostrais, desvio padrão e intervalo de confiança assintótico de 95% de confiança.

6.1 Descrição dos Cenários de Teste

O *arcabouço* apresentado no capítulo 4 foi submetido a uma bateria de testes, nos quais foram utilizadas as intâncias clássicas de problemas de roteamento de veículos propostas por Solomon (1987). As instâncias de Solomon possuem 100 (cem) clientes cada uma.

Para adaptar essas instâncias ao contexto do problema de roteamento dinâmico de veículos, entretanto, foram necessários alguns ajustes. Assume-se que algumas requisições do arquivo de instância desejado sejam passadas ao algoritmo no tempo (0) zero, simulando, assim, as requisições conhecidas antes do início do período de operação dos veículos. No decorrer do tempo, as demais requisições contidas no arquivo de instância são repassadas ao algoritmo. O tempo no qual cada requisição é repassada é determinado por uma distribuição de probabilidade.

Um dos parâmetros do arcabouço é o grau de dinamismo, que indica qual o percentual de requisições do arquivo de instâncias que ficará reservado para ser repassado ao algoritmo no decorrer do período de operação.

Nas instâncias propostas por Solomon (1987), as localizações do depósito e dos clientes que efetuaram as requisições são dadas como valores inteiros entre 0 e 100, em um sistema de coordenadas cartesianas.

Os problemas testes são agrupados em seis tipos de instâncias. Nos conjuntos de problemas R1 e R2, as coordenadas dos clientes são geradas aleatoriamente em uma determinada área, segundo uma distribuição uniforme, conforme pode-se visualizar na Figura 6.1 e na Figura

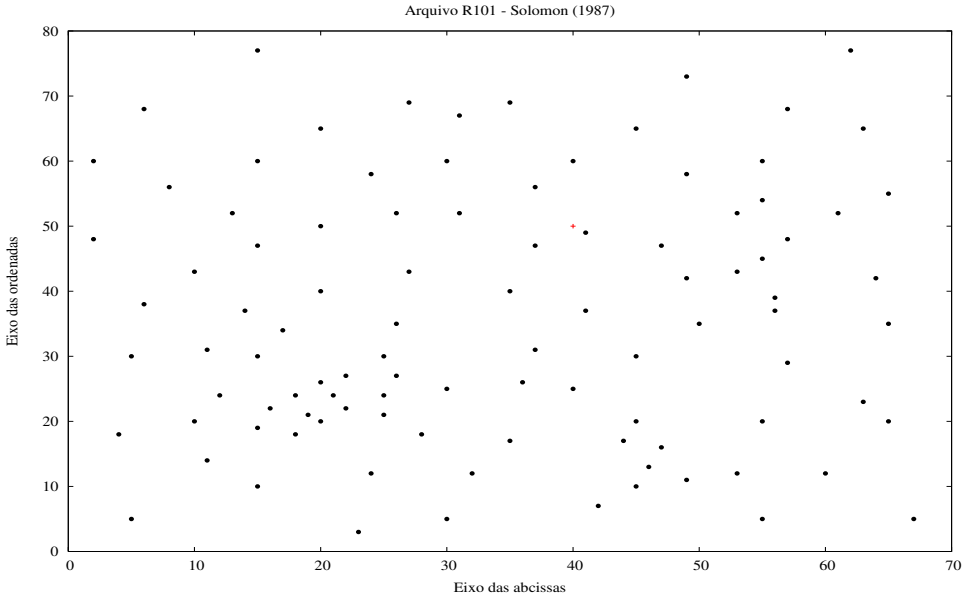

Figura 6.1: Instância R101

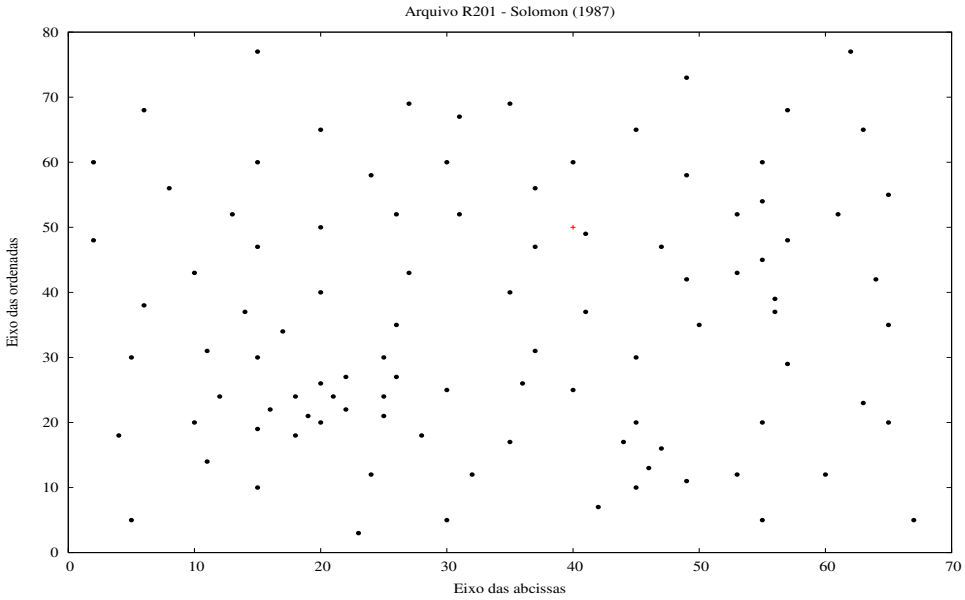

Figura 6.2: Instância R201

6.2.

A distribuição geográfica dos clientes nos conjuntos C1 e C2 é agrupada. Pode-se perceber, visualizando a Figura 6.3 e a Figura 6.4, que os clientes estão claramente divididos em grupos.

Nos conjuntos $RC1$ e $RC2$ (Figuras 6.5 e 6.6), a distribuição dos clientes é semi-agrupada, ou seja, partes das localizações dos clientes são geradas aleatoriamente e parte é gerada de forma que os clientes formem *clusters*. Clientes nos conjuntos $R1$, $C1$ e $RC1$ apresentam

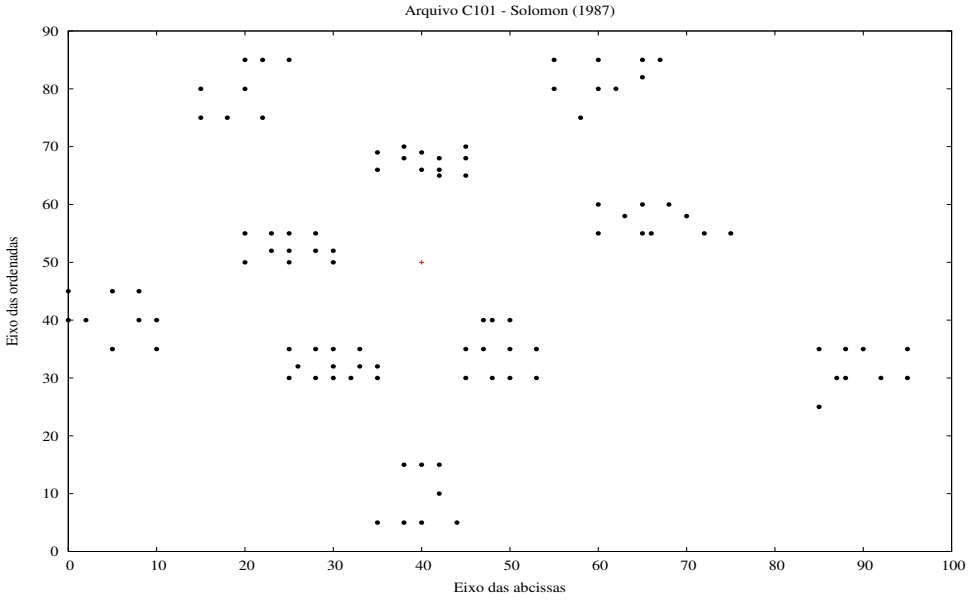

Figura 6.3: Instância C101

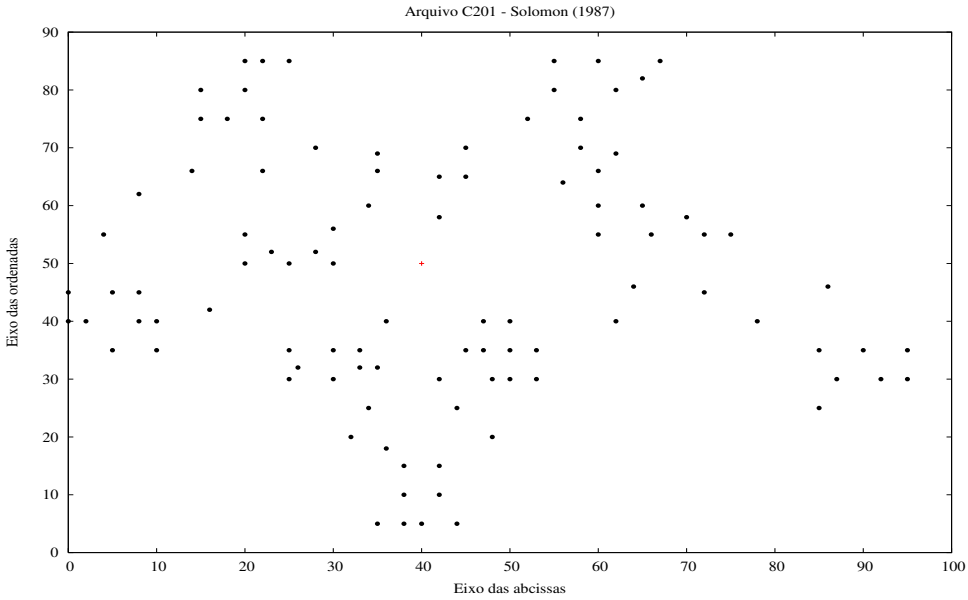

Figura 6.4: Instância C201

janelas de tempo com pouca folga. Como conseqüência disto, em geral, poucos clientes são visitados por rota, ao passo que, nos problemas $R2$, $C2$ e $RC2$, que possuem janelas de tempo maiores, as rotas geradas geralmente possuem mais clientes a serem visitados.

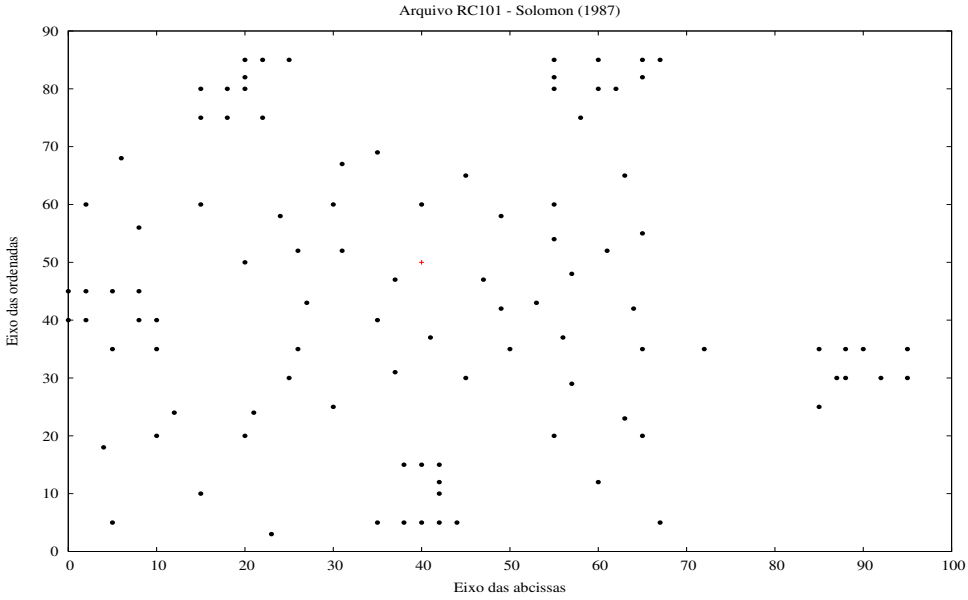

Figura 6.5: Instância RC101

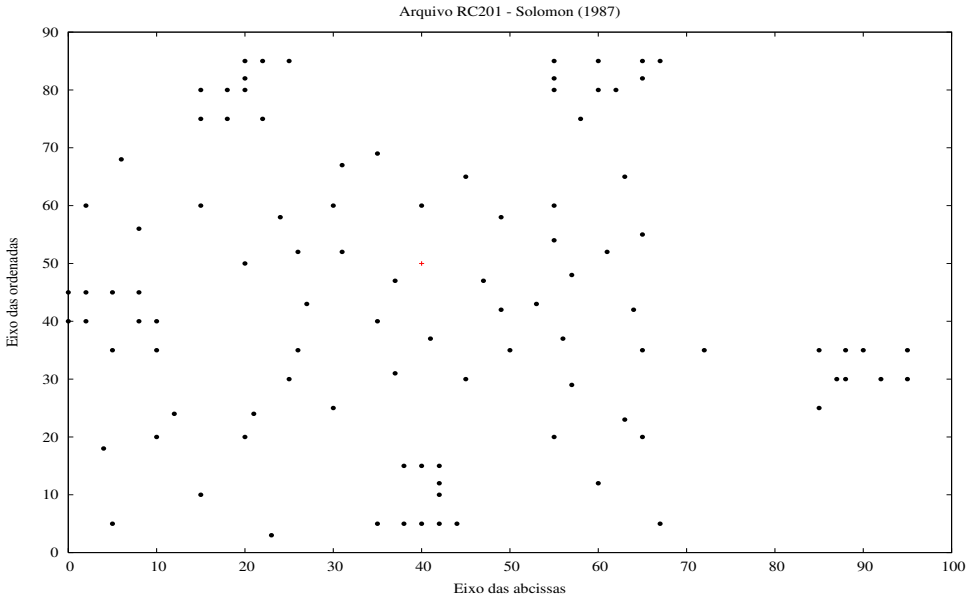

Figura 6.6: Instância RC201

6.2 A geração de requisições

Conforme foi apresentado no capítulo 4, o arcabouço foi testado utilizando-se as intâncias clássicas propostas por Solomon (1987). Para simular o caráter dinâmico do problema, foram definidas as seguintes regras:

- através de um parâmetro do arcabouço, sabe-se o percentual das requisições contidas

no arquivo de instância que deverá ser passado ao otimizador, previamente, a fim de que a geração da solução inicial ocorra antes do início do período de operação;

- através de uma distribuição aleatória, determina-se o tempo no qual cada requisição restante no arquivo de instância será gerada pela *Thread* produtora;
- quando o produtor verifica que o tempo atual é menor que ou igual ao tempo que foi determinado para se gerar uma requisição, ele insere na fila compartilhado a referida requisição, para que a *Thread* consumidora a repasse ao otimizador, de acordo com a política de roteamento dinâmico de veículos adotada.

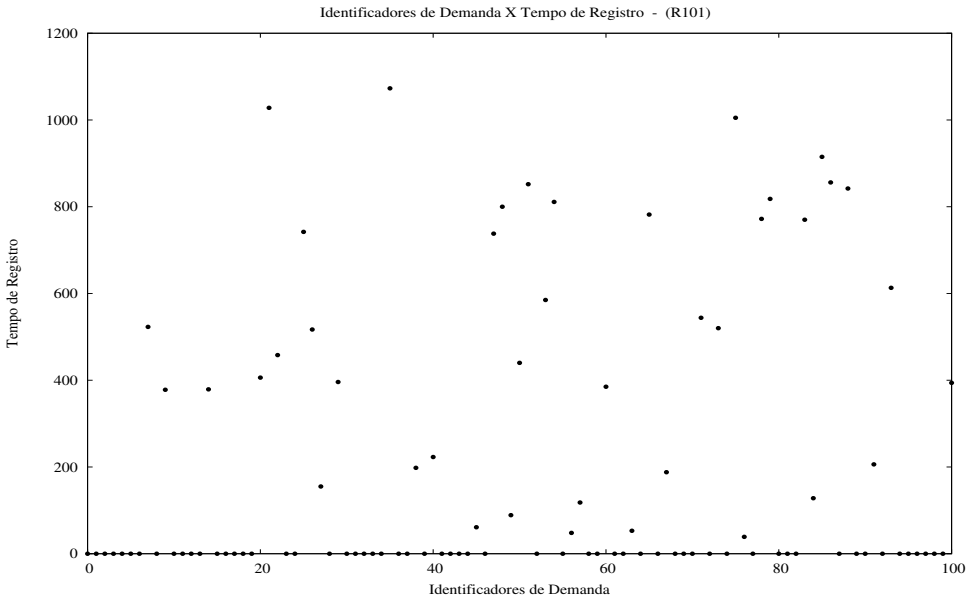

Figura 6.7: Horário de registro das requisições no decorrer do tempo com grau de dinamismo de 40%

As Figuras 6.7 e 6.8 ilustram os tempos nos quais as requisições são efetuadas no problema de roteamento dinâmico de veículos. Pode-se perceber que, na Figura 6.7, quando é considerado um grau de dinamismo de 40%, 60% das requisições são conhecidas no tempo 0 (zero). Observa-se também que esses 60% de requisições são selecionados aleatoriamente da totalidade das requisições listadas no arquivo da instância. Isto pode ser verificado observando-se que os pontos que estão sobre o eixo das abcissas ($time = 0$) estão espalhados de maneira uniforme (sabendo-se que o eixo das abcissas é o eixo que contém os identificadores das requisições da instância de Solomon)

De maneira análoga, verifica-se que, com o aumento do grau de dinamismo para 80%, a quantidade de requisições conhecidas *a priori*, cai para 20%, evidentemente. Comparando-se os dois gráficos, esta diferença é facilmente percebida.

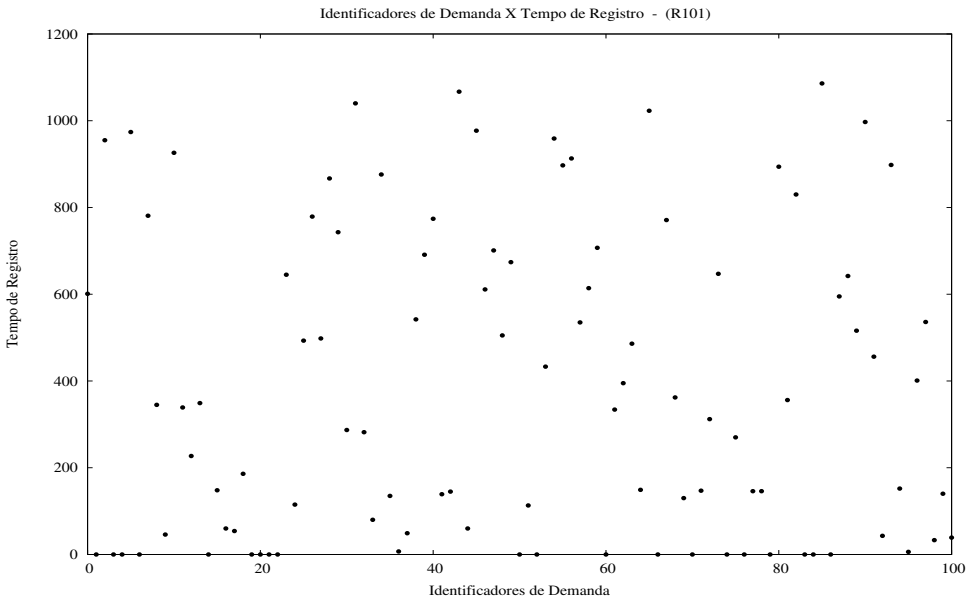

Figura 6.8: Horário de registro das requisições no decorrer do tempo com grau de dinamismo de 80%

6.3 Resultados Obtidos

Esta seção é destinada a listar os principais resultados obtidos através de uma bateria de testes à qual o arcabouço foi submetido, utilizando-se como base as instâncias propostas por Solomon (1987). O código fonte foi implementado na linguagem de programação JAVA, versão 1.4.2. O modelo matemático do Problema de Partição de Conjuntos foi executado a partir da API do sotware CPLEX [1], versão 9.1.

6.3.1 Parâmetros do arcabouço

Os testes foram realizados com os parâmetros listados na Tabela 6.1. Os quatro primeiros parâmetros são utilizados no cálculo do valor da função de avaliação. O custo fixo associado aos veículos utilizados diz respeito ao custo adicional que uma empresa teria, caso precisasse de mais um veículo para efetuar o roteamento. Este custo pode estar relacionado à contratação de novos motoristas ou à compra de novos veículos, por exemplo.

O custo adicional por unidade de distância percorrida pode estar relacionado ao custo com combustível ou ao tempo gasto nas viagens.

Neste estudo, optou-se por trabalhar com janelas de tempo flexíveis. Assim, quanto mais alta forem as penalidades por tempo de atraso/espera, menores as chances de uma solução que apresente muito atraso ou muita espera sobreviver durante as gerações do algoritmo evolucionário.

[1] CPLEX é um pacote de otimização da ILOG que resolve problemas de programação linear e problemas de programação linear inteira

Tabela 6.1: Listagem de parâmetros do arcabouço usados nos testes

Ítem	Parâmetro	Valor
1	Custo fixo por veículo utilizado	10
2	Custo por unidade de distância percorrida	2
3	Penalidade por atraso	40
4	Penalidade por espera	20
5	Número de candidatos por torneio (parâmetro usado no método de seleção	6
6	Taxa de geração de novas soluções viáveis	70 %
7	Taxa de mutação	90 %
8	Número de indivíduos da populacão	30
9	Quantidade de patamares de dependência de tempo	8
10	Prazo para atendimento de requisições efetuadas durante o período	80 %
11	Ilhas de Evolução	3
12	Capacidade dos Veículos	200
13	Tamanho da fila de requisições (parâmetro usado somente na política Por Demanda)	5

No método de seleção, apresentado no capítulo 5, a cada chamada ao método *competition*, são selecionados n indivíduos que irão disputar uma vaga entre os indivíduos selecionados. O valor de n é o parâmetro *número de candidatos por torneio*, mostrado na Tabela 6.1.

Taxa de geração de novas soluções viáveis e taxa de mutação são os percentuais de indivíduos que irão participar da fase de geração de novas soluções viáveis e da fase de *mutação*, respectivamente.

O número de indivíduos da população é a quantidade de soluções geradas pelo algoritmo da população inicial apresentado no capítulo 5. Esta quantidade de indivíduos mantém-se ao longo das gerações do algoritmo evolucionário.

A quantidade de patamares de dependência de tempo significa que o período de operação vai ser subdividido nessa quantidade de patamares e que, em cada um deles, será considerada uma velocidade média diferente para os veículos. Esta previsão de velocidade média é feita para cada trecho entre dois quaisquer pontos de demanda.

O prazo final para que sejam efetuadas requisições que se deseja que sejam atendidas no atual período de operação foi configurado com o valor 80%, ou seja, requisições efetuadas após serem decorridos 80% do período de operação são armazenadas para serem atendidas somente no período de operação seguinte.

Os valores dos parâmetros são definidos a critério das prioridades do usuário do *arcabouço*. Os valores listados na Tabela 6.1 foram definidos empiricamente e não têm nenhuma relação com casos reais. As penalidades definidas também não têm relação entre elas.

6.3.2 Tempo de Execução

O algoritmo evolucionário Dinâmico é executado durante todo o período de operação dos veículos. O período de operação é dado em unidades de tempo de simulação. A cada iteração

do algoritmo, o tempo de simulação é incrementado de u unidades, sendo u definido pelo usuário. Uma iteração gasta em média 13 (treze) milissegundos ($tempo_{iteracao}$). Assim, o tempo de execução($tempo_{execucao}$) pode ser calculado como:

$$tempo_{execucao} = tempo_{iteracao} * periodo/u \tag{6.1}$$

6.3.3 Análise Estatística

Da Análise de Desempenho de Algoritmos (Jaim (1991)), sabe-se que para uma amostra suficientemente grande, a distribuição das médias amostrais em torno da média populacional (μ) é uma *Normal* com desvio padrão $\sigma\ \sqrt{n}$, sendo n o tamanho da amostra. Esta razão é denominada de *erro padrão*. Quanto menor for o valor do erro padrão, mais próxima a média amostral estará da média populacional (μ).

6.3.3.1 Intervalo de Confiança

Neste trabalho, utiliza-se intervalo de confiança assintótico de 95% de confiança, em relação à média do valor das métricas analisadas: distância percorrida, número de veículos utilizados, tempo total de espera, tempo total de atraso e valor da função de avaliação. Isso significa que 95% de todas as médias amostrais estarão contidas neste intervalo de confiança.

Para se calcular o intervalo de confiança, com nível de confiança de 95%, utiliza-se a seguinte fórmula:

$$(\overline{x} - z_{1-\alpha/2}.S/\sqrt{(n)}, \overline{x} + z_{1-\alpha/2}.S/\sqrt{(n)}) \tag{6.2}$$

onde o valor $z_{1-\alpha/2}$ é obtido na tabela da distribuição *Normal*, S é o desvio-padrão da amostra e n é o tamanho da amostra. (Jaim (1991)). Nos testes realizados, considerou-se o tamanho da amostra como sendo de 33 (trinta e três) execuções.

6.3.4 Resultados

Nos testes listados nessa seção, variou-se o parâmetro grau de dinamismo, que assumiu 4 (quatro) valores: 20%, 40%, 60% e 80%. Para cada valor de grau de dinamismo, foram realizados testes considerando cada uma das 3 (três) políticas de roteamento dinâmico apresentadas no capítulo 4. Em cada um desses testes, foi analisada uma métrica denominada função de avaliação ou simplesmente *objetivo*. Esta função de avaliação é composta de 4 (quatro) métricas, a saber: distância total percorrida, veículos utilizados, tempo total de espera e tempo total de atraso. Na função de avaliação, um peso (prioridade) é atribuído a cada uma das métricas. Esta função foi apresentada no capítulo 5 e os pesos associados às métricas nesta função são parâmetros do arcabouço, conforme ilustra a Tabela 6.1. Os testes foram realizados tendo, como objetivo, minimizar esta função, considerando todas as quatro métricas simultaneamente. Testou-se, também, as quatro métricas citadas isoladamente.

6.3.4.1 Política Online

Avaliando-se a função de avaliação, verifica-se que, em geral, o valor da função cresce com o aumento do grau de dinamismo (Figura 6.10). Não se pode afirmar, todavia, com 95% de confiança que essa tendência sempre será verdadeira, visto que os intervalos de confiança não são disjuntos, como ilustra a Figura 6.9. Isto acontece devido à aleatoriedade do problema e dos parâmetros serem configurados empiricamente.

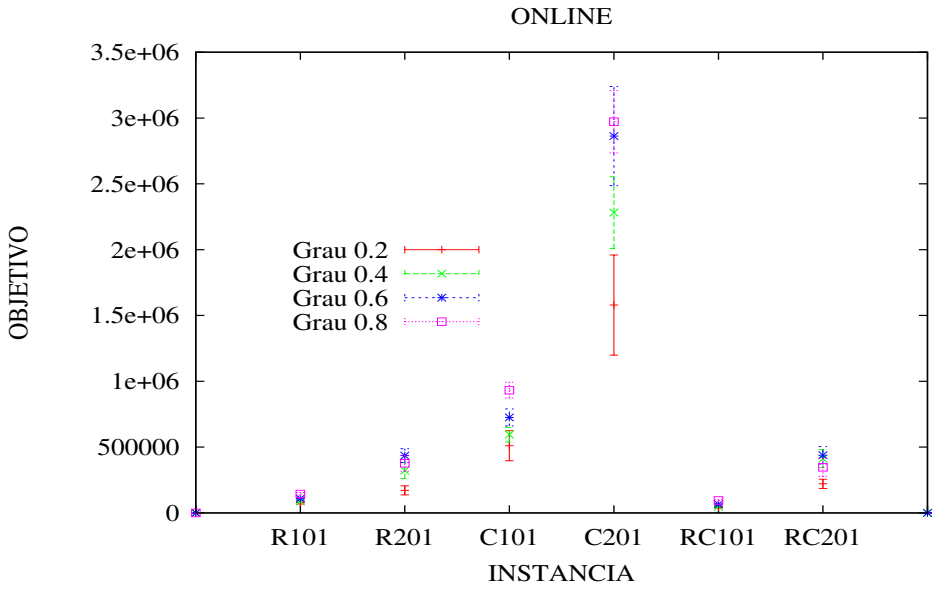

Figura 6.9: Valores da função de avaliação - Graus de dinamismo: 20%, 40%, 60% e 80%

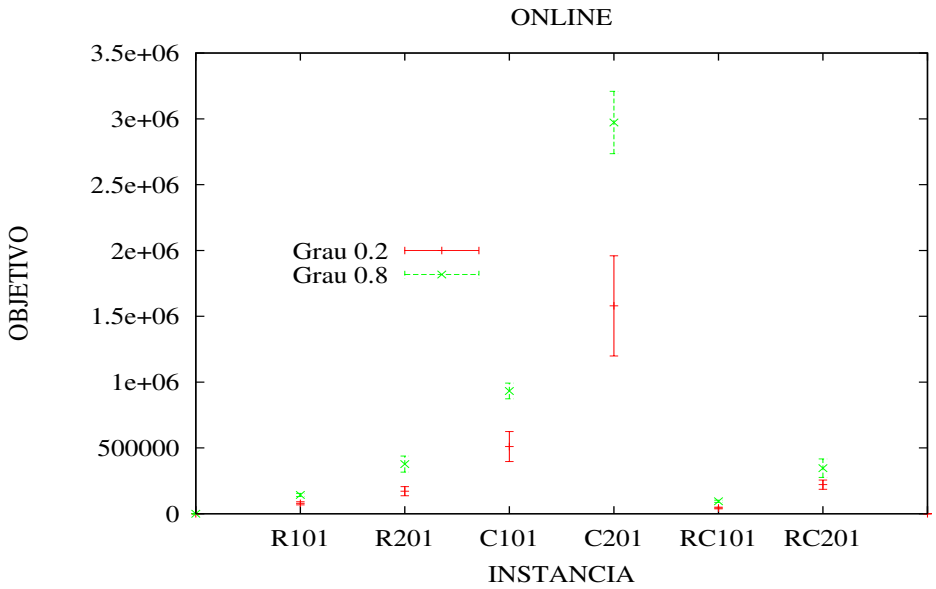

Figura 6.10: Valores da função de avaliação - Graus de dinamismo: 20% e 80%

Tabela 6.2: Política : Online - Métrica: distância - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	1220.39	1060.84 - 1379.93
R201.txt	618.69	390.38 - 847.01
C101.txt	1925.44	1614.41 - 2236.47
C201.txt	1394.03	1185.56 - 1602.50
RC101.txt	1434.82	1284.50 - 1585.15
RC201.txt	885.72	599.39 - 1172.05

Tabela 6.3: Política : Online - Métrica: distância - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1099.84	912.58 - 1287.10
R201.txt	909.52	699.79 - 1119.24
C101.txt	1502.44	1307.91 - 1696.97
C201.txt	1454.88	1121.75 - 1788.01
RC101.txt	1373.35	1234.24 - 1512.47
RC201.txt	1519.87	1180.68 - 1859.05

Tabela 6.4: Política : Online - Métrica: distância - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	957.79	824.45 - 1091.13
R201.txt	858.05	622.75 - 1093.36
C101.txt	1761.36	1534.16 - 1988.55
C201.txt	1734.15	1369.85 - 2098.45
RC101.txt	1329.34	1160.00 - 1498.69
RC201.txt	1873.22	1514.97 - 2231.48

Observa-se, pelos resultados listados na Tabela 6.2, na Tabela 6.3, na Tabela 6.4 e na Tabela 6.5 que, quando foram testadas as instâncias que distribuem os clientes de forma totalmente aleatória, em geral, os intervalos de confiança estão sobrepostos, considerando-se a distância total percorrida. Este fato que não possibilita afirmar-se categoricamente que a distância irá crescer com o aumento do grau de dinamismo, pois, nestes testes, buscou-se minimizar a função de avaliação como um todo.

O mesmo pode-se dizer a respeito dos testes realizados com as instâncias agrupadas e com as instâncias semi-agrupadas.

No que diz respeito à quantidade de veículos utilizados no roteamento, observa-se que é uma métrica que não sofreu muita influência da variação da política de roteamento dinâmico utilizada. Observa-se também que alterações no grau de dinamismo utilizado interfere mais nas janelas de tempo de atendimento, de modo que a carga total transportada praticamente não se altera. Sendo assim, alterações na quantidade de veículos (Tabela 6.6, Tabela 6.7,

Tabela 6.5: Política : Online - Métrica: distância - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	913.38	790.63 - 1036.13
R201.txt	615.19	432.54 - 797.83
C101.txt	1449.91	1271.25 - 1628.57
C201.txt	1858.52	1428.47 - 2288.57
RC101.txt	1464.73	1349.56 - 1579.89
RC201.txt	2135.84	1696.11 - 2575.57

Tabela 6.6: Política : Online - Métrica: veículos - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	16.06	14.97 - 17.15
R201.txt	10.67	7.28 - 14.05
C101.txt	22.18	19.57 - 24.79
C201.txt	27.42	21.40 - 33.45
RC101.txt	14.64	13.95 - 15.32
RC201.txt	20.64	16.26 - 25.02

Tabela 6.7: Política : Online - Métrica: Veículos - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	17.64	15.89 - 19.38
R201.txt	13.61	10.48 - 16.73
C101.txt	27.30	24.80 - 29.81
C201.txt	27.33	22.98 - 31.69
RC101.txt	15.82	14.95 - 16.68
RC201.txt	19.97	17.08 - 22.86

Tabela 6.8 e Tabela 6.9) podem ocorrer em momentos nos quais o algoritmo verifica ser mais compensador criar uma nova rota do que pagar o preço da penalidade referente a infringência de uma janela de tempo de atendimento. Dependendo da penalidade associada ao não cumprimento da janela de tempo e do custo fixo vinculado à adição de um novo veículo, podem ser criadas mais rotas (baixa penalidade para adição de novo veículo e alta penalidade de janela de tempo) ou menos rotas (alto custo fixo de veículos e baixa penalidade de janela de tempo).

Com relação ao tempo total de espera, verifica-se um equilíbrio entre os números obtidos quando foi variado o grau de dinamismo nos testes realizados (Tabela 6.10, Tabela 6.11, Tabela 6.12 e Tabela 6.13). À medida que o grau de dinamismo aumenta, poucas requisições são conhecidas a priori, fato que pode gerar muito tempo de espera no início da jornada. Como são poucos clientes a serem visitados, pode ser que, a princípio, as janelas de tempo sejam muito flexíveis. Com o decorrer da jornada, entretanto, pode acontecer que muitas

Tabela 6.8: Política : Online - Métrica: Veículos - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	21.91	19.80 - 24.02
R201.txt	16.97	13.25 - 20.69
C101.txt	26.03	23.28 - 28.78
C201.txt	27.76	22.14 - 33.37
RC101.txt	16.76	15.65 - 17.87
RC201.txt	17.21	13.55 - 20.87

Tabela 6.9: Política : Online - Métrica: veiculos - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	24.67	22.25 - 27.09
R201.txt	11.18	7.78 - 14.58
C101.txt	32.39	28.30 - 36.48
C201.txt	25.76	22.42 - 29.09
RC101.txt	20.73	18.89 - 22.57
RC201.txt	12.36	8.46 - 16.26

Tabela 6.10: Política : Online - Métrica: espera - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	1691.83	1411.25 - 1972.41
R201.txt	8756.88	5514.53 - 11999.23
C101.txt	15560.26	12055.25 - 19065.28
C201.txt	80752.39	61509.50 - 99995.28
RC101.txt	882.22	734.45 - 1029.99
RC201.txt	15314.03	11789.51 - 18838.55

requisições sobrecarreguem os veículos, de modo que o tempo total de espera venha a cair, aumentando, inclusive, o tempo total de atraso.

Em relação ao atraso, pode-se afirmar, com 95% de confiança, que ele cresce proporcionalmente ao aumento do grau de dinamismo, considerado os extremos (20% e 80% de dinamismo), como mostra a Tabela 6.14 e a Tabela 6.17. Quando se considera um acréscimo menor no grau de dinamismo, verifica-se que a média do tempo de atraso também acompanha o crescimento do grau de dinamismo, mas os intervalos de confiança podem sobrepor-se parcialmente (conforme Tabelas 6.14, 6.15, 6.16 e 6.17). É fácil perceber que, se existem mais requisições que não são conhecidas no tempo zero, os prazos ficam mais rígidos e isto pode implicar atraso.

6.3.4.2 Política Por Demanda

A função de avaliação também tende a crescer com o aumento do grau de dinamismo, como ilustram as Figuras 6.11 e 6.12. Este resultado é natural, visto que o valor da função de

Tabela 6.11: Política : Online - Métrica: espera - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	2118.93	1799.79 - 2438.07
R201.txt	11131.32	8024.49 - 14238.16
C101.txt	18343.62	15877.25 - 20809.99
C201.txt	79914.99	64368.72 - 95461.27
RC101.txt	1239.34	1006.31 - 1472.38
RC201.txt	16636.08	13182.04 - 20090.11

Tabela 6.12: Política : Online - Métrica: espera - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	2756.76	2344.87 - 3168.66
R201.txt	12683.61	9783.34 - 15583.87
C101.txt	17776.40	15421.02 - 20131.77
C201.txt	90787.13	71951.56 - 109622.69
RC101.txt	1656.54	1423.62 - 1889.46
RC201.txt	14980.59	11418.07 - 18543.11

Tabela 6.13: Política : Online - Métrica: espera - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	3248.27	2807.96 - 3688.58
R201.txt	9045.87	6419.08 - 11672.66
C101.txt	22051.01	19407.42 - 24694.59
C201.txt	76456.70	68657.19 - 84256.22
RC101.txt	2589.90	2216.35 - 2963.45
RC201.txt	10370.37	7101.95 - 13638.78

Tabela 6.14: Política : Online - Métrica: atraso - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	486.00	402.48 - 569.52
R201.txt	912.85	588.97 - 1236.73
C101.txt	3530.74	2989.71 - 4071.78
C201.txt	9029.76	6838.97 - 11220.55
RC101.txt	308.44	242.88 - 373.99
RC201.txt	650.73	311.23 - 990.24

Tabela 6.15: Política : Online - Métrica: atraso - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1048.02	918.09 - 1177.94
R201.txt	2441.23	1915.28 - 2967.17
C101.txt	5605.08	4965.59 - 6244.57
C201.txt	16994.05	13150.53 - 20837.57
RC101.txt	468.27	400.64 - 535.90
RC201.txt	1889.27	1320.35 - 2458.19

Tabela 6.16: Política : Online - Métrica: atraso - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	1289.38	1183.78 - 1394.97
R201.txt	4489.44	3126.82 - 5852.05
C101.txt	9165.13	8159.38 - 10170.88
C201.txt	26092.22	22687.09 - 29497.34
RC101.txt	752.74	643.96 - 861.52
RC201.txt	3379.45	2191.89 - 4567.02

Tabela 6.17: Política : Online - Métrica: atraso - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	1886.10	1704.81 - 2067.39
R201.txt	4872.38	3834.95 - 5909.80
C101.txt	12186.21	10820.97 - 13551.46
C201.txt	36004.91	31530.24 - 40479.57
RC101.txt	995.55	883.59 - 1107.50
RC201.txt	3417.69	2411.45 - 4423.94

avaliação deriva dos valores das demais métricas analisadas. Os intervalos de confiança, entretanto, possuem intersecção, fato que impede que se dê uma garantia de que a função de avaliação sempre irá crescer com o aumento do grau de dinamismo.

Nos testes realizados com a Política Por Demanda, em relação à distância total percorrida, pode-se verificar que, com o aumento do grau de dinamismo, a tendência é que a distância percorrida também cresça. Não é possível afirmar, todavia, com 95% de confiança que o aumento do grau de dinamismo de 20% para 40%, 60% e 80%, acarretará sempre uma maior distância total percorrida (Tabelas 6.18, 6.19, 6.20, 6.21). Há um equilíbrio, comparando-se a distância percorrida para os mesmos cenários, variando-se a política de roteameto dinâmico de *Online* para a política *Por Demanda*.

Analisando-se as Tabelas 6.22, 6.23, 6.24 e 6.25, pode-se visualizar que, à medida que o grau de dinamismo aumenta, a tendência é que a quantidade de veículos utilizada no roteamento também cresça. Esta evidência, entretanto, não se pode garantir com 95% de confiança,

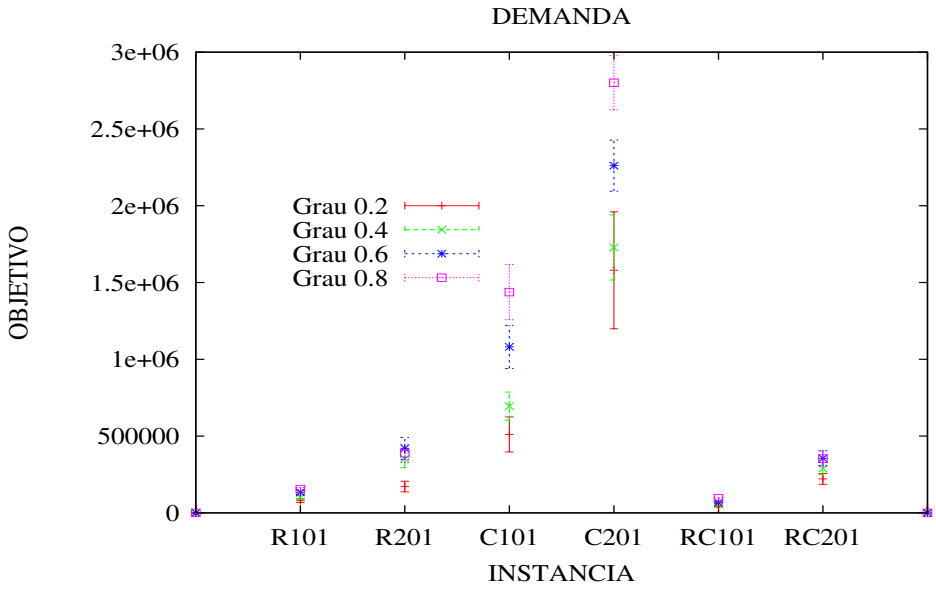

Figura 6.11: Valores da função de avaliação - Graus de dinamismo: 20%, 40%, 60% e 80%

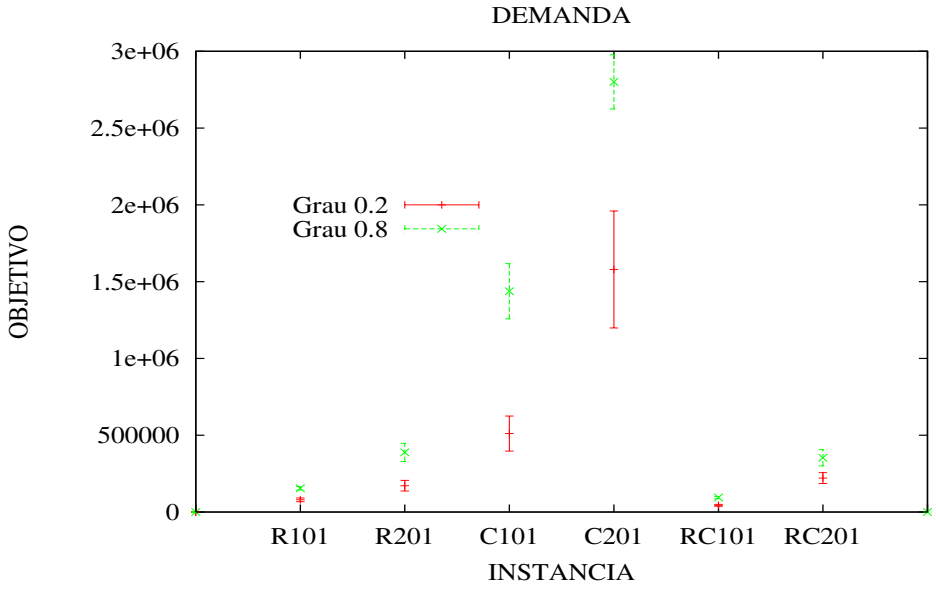

Figura 6.12: Valores da função de avaliação - Graus de dinamismo: 20% e 80%

visto que os valores dos intervalos de confiança sobrepõem-se parcialmente.

No tocante ao tempo total de espera, observa-se que o seu valor tende a subir com o aumento do grau de dinamismo. Isto ocorre em virtude de que, no início da jornada, há cada vez menos clientes a serem visitados e, por conseguinte, mais folga nas janelas de tempo de atendimento, acarretando, assim, um maior tempo de espera, conforme pode-se visualizar nas Tabelas 6.26, 6.27, 6.28 e 6.29.

Tabela 6.18: Política : Por Demanda - Métrica: distância - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	1239,00	1110,34 - 1367,66
R201.txt	872,31	657,83 - 1086,80
C101.txt	2013,26	1801,20 - 2225,31
C201.txt	1554,21	1197,17 - 1911,24
RC101.txt	1833,32	1677,38 - 1989,26
RC201.txt	1455,54	1187,59 - 1723,49

Tabela 6.19: Política : Por Demanda - Métrica: distância - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1192,92	1040,09 - 1345,76
R201.txt	858,18	634,87 - 1081,48
C101.txt	1683,08	1445,88 - 1920,29
C201.txt	1364,73	1055,19 - 1674,27
RC101.txt	2028,04	1830,24 - 2225,84
RC201.txt	1392,58	1036,21 - 1748,94

Tabela 6.20: Política : por Demanda - Métrica: distância - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	1085,73	966,10 - 1205,37
R201.txt	895,74	686,27 - 1105,21
C101.txt	1685,46	1496,70 - 1874,22
C201.txt	1668,01	1400,32 - 1935,71
RC101.txt	1708,09	1531,60 - 1884,58
RC201.txt	1375,08	1189,25 - 1560,90

Tabela 6.21: Política : Por Demanda - Métrica: distância - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	981,31	874,74 - 1087,88
R201.txt	582,14	457,40 - 706,88
C101.txt	1522,87	1304,91 - 1740,82
C201.txt	1113,85	890,16 - 1337,54
RC101.txt	1686,49	1520,42 - 1852,56
RC201.txt	1215,16	1054,58 - 1375,73

Tabela 6.22: Política : Por Demanda - Métrica: Veículos - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	17,42	16,25 - 18,60
R201.txt	8,03	6,53 - 9,53
C101.txt	21,09	18,86 - 23,32
C201.txt	14,61	12,10 - 17,11
RC101.txt	16,79	15,97 - 17,61
RC201.txt	11,58	10,11 - 13,04

Tabela 6.23: Política : Por Demanda - Métrica: Veículos - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	20,09	18,76 - 21,42
R201.txt	13,88	10,68 - 17,07
C101.txt	25,64	23,19 - 28,08
C201.txt	24,33	20,95 - 27,72
RC101.txt	18,76	17,86 - 19,65
RC201.txt	12,33	10,53 - 14,13

Tabela 6.24: Política : Por Demanda - Métrica: Veículos - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	23,06	21,16 - 24,96
R201.txt	14,52	10,65 - 18,38
C101.txt	28,64	25,61 - 31,66
C201.txt	21,39	17,56 - 25,22
RC101.txt	22,82	21,79 - 23,84
RC201.txt	14,30	11,32 - 17,29

Tabela 6.25: Política : Por Demanda - Métrica: Veículos - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	27,09	25,31 - 28,87
R201.txt	13,70	10,61 - 16,78
C101.txt	35,30	31,84 - 38,76
C201.txt	28,18	22,76 - 33,61
RC101.txt	25,88	24,63 - 27,13
RC201.txt	14,06	11,50 - 16,62

Tabela 6.26: Política : Por Demanda - Métrica: espera - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	1873,81	1556,09 - 2191,53
R201.txt	6832,82	5145,20 - 8520,45
C101.txt	12295,72	9413,48 - 15177,96
C201.txt	46379,78	36819,28 - 55940,28
RC101.txt	883,85	751,36 - 1016,33
RC201.txt	9254,68	7627,72 - 10881,64

Tabela 6.27: Política : Por Demanda - Métrica: espera - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	2488,50	2120,47 - 2856,54
R201.txt	11585,35	8353,28 - 14817,42
C101.txt	15405,00	12658,60 - 18151,41
C201.txt	60662,38	53027,17 - 68297,59
RC101.txt	1278,68	1068,54 - 1488,82
RC201.txt	9202,98	7290,43 - 11115,53

Tabela 6.28: Política : Por Demanda - Métrica: espera - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	2587,68	2143,36 - 3031,99
R201.txt	12405,92	8446,59 - 16365,26
C101.txt	18943,01	15755,38 - 22130,63
C201.txt	60801,44	49314,58 - 72288,31
RC101.txt	1523,11	1284,85 - 1761,36
RC201.txt	11225,05	8685,20 - 13764,90

O tempo de atraso também tende a aumentar de acordo com o aumento do grau de dinamismo (Tabelas 6.30, 6.31, 6.32, 6.33). Pode parecer estranho que tanto a espera quanto o atraso acompanhem o crescimento do grau de dinamismo, entretanto, percebe-se que a espera é grande no início da jornada quanto se tem poucos clientes a serem visitados e o atraso aumenta nos momentos de pico, quando chegam muitas requisições com janelas de tempo muito rígidas.

6.3.4.3 Política Periódica

As Figura 6.10 mostra que o valor da função de avaliação cresce com o aumento do grau de dinamismo de 20% para 80%. A exemplo do que aconteceu na análise das demais políticas, era esperado tal resultado, pois a função de avaliação é a métrica que de fato busca-se minimizar e o seu valor é composto pelas demais métricas associadas aos seus respectivos multiplicadores. Todavia, quando se considera o grau de dinamismo variando entre os valores 20%, 40%, 60%

Tabela 6.29: Política : Por Demanda - Métrica: espera - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	3366,89	2862,91 - 3870,86
R201.txt	10758,80	8352,48 - 13165,13
C101.txt	25537,08	21140,18 - 29933,99
C201.txt	70135,45	59245,69 - 81025,20
RC101.txt	2317,84	1980,22 - 2655,47
RC201.txt	10178,75	8231,77 - 12125,73

Tabela 6.30: Política : Por Demanda - Métrica: atraso - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	995,32	774,74 - 1215,91
R201.txt	838,68	547,99 - 1129,37
C101.txt	6539,07	4575,13 - 8503,02
C201.txt	16234,27	8886,48 - 23582,06
RC101.txt	559,99	434,26 - 685,72
RC201.txt	836,43	541,08 - 1131,78

Tabela 6.31: Política : Por Demanda - Métrica: atraso - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1243,10	1084,12 - 1402,09
R201.txt	2962,41	2151,42 - 3773,40
C101.txt	9578,22	7669,59 - 11486,85
C201.txt	12788,53	7887,80 - 17689,27
RC101.txt	695,35	589,27 - 801,43
RC201.txt	2552,71	1894,66 - 3210,75

Tabela 6.32: Política : Por Demanda - Métrica: atraso - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	1983,43	1756,27 - 2210,59
R201.txt	4250,66	3012,58 - 5488,73
C101.txt	17460,58	14745,05 - 20176,10
C201.txt	26055,20	22341,90 - 29768,51
RC101.txt	837,03	700,87 - 973,19
RC201.txt	3203,72	2402,42 - 4005,03

Tabela 6.33: Política : Por Demanda - Métrica: atraso - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	2108,47	1940,73 - 2276,21
R201.txt	4271,62	3199,99 - 5343,25
C101.txt	23060,36	19843,17 - 26277,56
C201.txt	34864,35	29654,17 - 40074,53
RC101.txt	1098,82	993,57 - 1204,07
RC201.txt	3667,39	2499,99 - 4834,79

e 80%, pode-se constatar ques os intervalos de confiança estão sobrepostos, como ilustra a Figura 6.9.

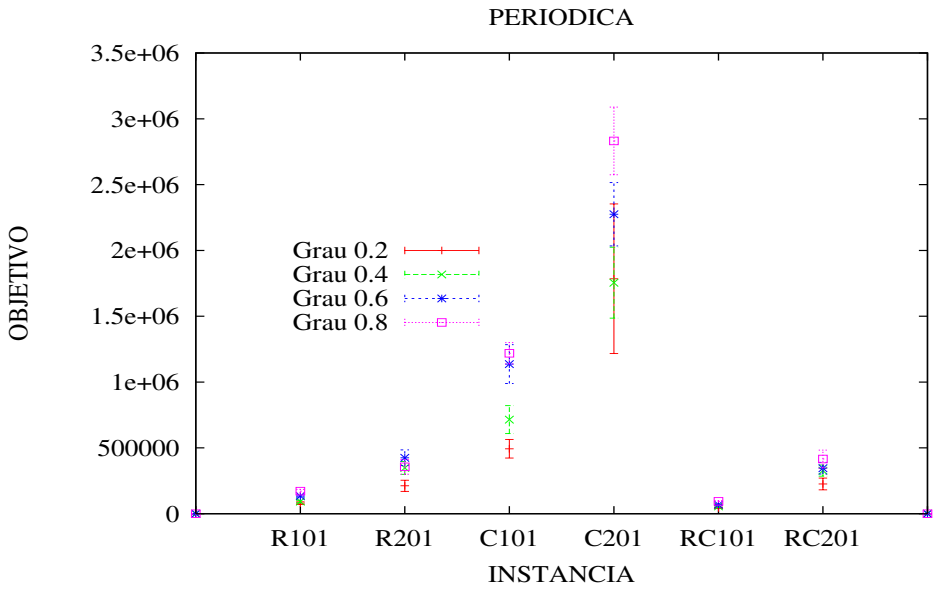

Figura 6.13: Valores da função de avaliação - Graus de dinamismo: 20%, 40%, 60% e 80%

Na Política *Periódica*, variando-se o grau de dinamismo, observa-se que os valores das métricas seguem a mesma lógica da política *Por Demanda*, visto que esperar encher a fila de requisições pode ser visto também como uma espera por um período de tempo. Embora as duas políticas aguardem um tempo entre momentos de realimentações do algoritmo com novas requisições recebidas, existe diferença entre elas. Enquanto na *Periódica* o tempo entre uma realimentação e outra é fixo e definido *a priori*. Na política Por Demanda, o tempo entre os reajustes varia, pois depende do momento em que a fila atinge o limite da sua capacidade de armazenamento de requisições.

A exemplo do que ocorre na política *Por Demanda*, na política *Periódica* a distância total percorida acompanha o aumento do grau de dinamismo do problema, conforme pode-se verificar nas Tabelas 6.34, 6.35, 6.36 e 6.37. Isto se deve ao fato de que as soluções que percorrem distância menores são encontradas quando se tem um grau de dinamismo baixo.

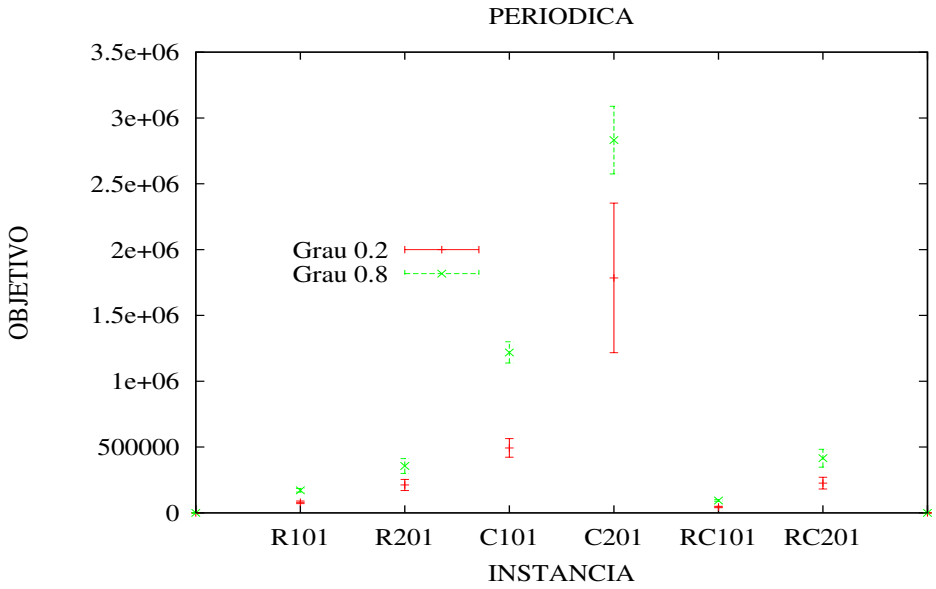

Figura 6.14: Valores da função de avaliação - Graus de dinamismo: 20% e 80%

Tabela 6.34: Política : Periódica - Métrica: distância - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	1448,60	1245,29 - 1651,91
R201.txt	710,27	564,51 - 856,03
C101.txt	1805,24	1621,51 - 1988,96
C201.txt	1275,57	1003,85 - 1547,29
RC101.txt	1949,00	1787,12 - 2110,88
RC201.txt	1423,44	1120,98 - 1725,90

Tabela 6.35: Política : Periódica - Métrica: Distância - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	1270,81	1108,09 - 1433,53
R201.txt	908,42	725,41 - 1091,43
C101.txt	1982,27	1664,14 - 2300,40
C201.txt	1760,61	1407,45 - 2113,77
RC101.txt	1800,85	1631,25 - 1970,45
RC201.txt	1401,07	1108,84 - 1693,31

Com o aumento do grau de dinamismo, as opções de alocação de novas requisições vão se reduzindo.

O número de veículos utilizados no roteamento também tende a acompanhar o aumento do grau de dinamismo do problema, como mostram as Tabelas 6.38, 6.39, 6.40 e 6.41, mas os intervalos de confiança estão sobrepostos.

Tabela 6.36: Política : Periódica - Métrica: distância - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1079,36	937,01 - 1221,70
R201.txt	804,09	654,01 - 954,17
C101.txt	1777,27	1562,44 - 1992,09
C201.txt	1285,51	1036,59 - 1534,43
RC101.txt	1651,26	1462,93 - 1839,60
RC201.txt	1460,82	1220,51 - 1701,13

Tabela 6.37: Política : Periódica - Métrica: distância - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	1071,98	931,59 - 1212,37
R201.txt	669,51	547,08 - 791,93
C101.txt	1477,36	1246,11 - 1708,61
C201.txt	1340,06	1050,41 - 1629,70
RC101.txt	1700,33	1587,02 - 1813,64
RC201.txt	1395,97	1147,66 - 1644,28

Tabela 6.38: Política : Periódica - Métrica: Veículos - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	18,21	16,75 - 19,67
R201.txt	11,15	9,05 - 13,26
C101.txt	21,88	19,38 - 24,38
C201.txt	18,91	14,80 - 23,02
RC101.txt	17,36	16,70 - 18,03
RC201.txt	13,88	11,52 - 16,23

Tabela 6.39: Política : Periodic - Métrica: veículos - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	20,45	18,93 - 21,98
R201.txt	12,27	9,61 - 14,94
C101.txt	28,00	24,87 - 31,13
C201.txt	20,30	17,05 - 23,55
RC101.txt	19,39	18,61 - 20,18
RC201.txt	13,03	10,95 - 15,11

Tabela 6.40: Política : Periodic - Métrica: Veículos - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	24,82	23,06 - 26,57
R201.txt	14,97	11,74 - 18,20
C101.txt	30,33	27,19 - 33,47
C201.txt	26,36	21,68 - 31,04
RC101.txt	23,30	22,42 - 24,19
RC201.txt	16,24	13,57 - 18,92

Tabela 6.41: Política : Periódica - Métrica: veículos - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	29,15	26,94 - 31,37
R201.txt	13,03	9,55 - 16,51
C101.txt	33,09	30,69 - 35,49
C201.txt	21,94	18,33 - 25,55
RC101.txt	26,79	25,71 - 27,87
RC201.txt	15,82	12,36 - 19,28

Tabela 6.42: Política : Periodic - Métrica: espera - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	2213,84	1853,10 - 2574,58
R201.txt	9262,55	7105,15 - 11419,94
C101.txt	14115,04	11441,76 - 16788,33
C201.txt	58931,98	42989,06 - 74874,91
RC101.txt	984,19	850,73 - 1117,66
RC201.txt	9635,15	7512,99 - 11757,31

O tempo total de espera também segue a mesma linha. As Tabelas 6.42, 6.43, 6.44 e 6.45 mostram que a espera aumenta de acordo com o aumento do grau de dinamismo. Não se pode afirmar, entretanto, com 95% de confiança, que essa tendência sempre irá confirmar-se, pois os intervalos de confiança não são disjuntos.

O tempo total de atraso, a exemplo do que ocorre com as outras métricas, também mostrou-se crescente com o aumento do grau de dinamismo do problema, como ilustram as Tabelas 6.46, 6.47, 6.48 e 6.49. Os intervalos de confiança, todavia, não são disjuntos.

6.4 Métricas testadas isoladamente

Além dos testes realizados utilizando como métrica a função de avaliação, foram realizados testes considerando-se isoladamente a métrica distância total percorrida, bem como testes considerando somente a métrica quantidade de veículos utilizados no roteamento.

Tabela 6.43: Política : Periódica - Métrica: espera - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	2053,88	1719,95 - 2387,82
R201.txt	10627,63	8158,45 - 13096,81
C101.txt	17349,30	14081,92 - 20616,69
C201.txt	56983,46	46867,41 - 67099,52
RC101.txt	1312,25	1120,93 - 1503,56
RC201.txt	10697,12	8453,35 - 12940,88

Tabela 6.44: Política : Periódica - Métrica: espera - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	2868,08	2509,76 - 3226,40
R201.txt	12244,33	8972,45 - 15516,21
C101.txt	18943,98	15955,23 - 21932,74
C201.txt	71886,17	60453,30 - 83319,04
RC101.txt	1669,97	1427,08 - 1912,85
RC201.txt	11378,16	9390,51 - 13365,82

Tabela 6.45: Política : Periódica - Métrica: espera - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	3826,13	3249,50 - 4402,76
R201.txt	9262,35	6865,81 - 11658,88
C101.txt	20646,19	17651,48 - 23640,91
C201.txt	64301,15	52934,03 - 75668,27
RC101.txt	2304,89	2020,73 - 2589,06
RC201.txt	13024,79	9203,72 - 16845,86

Tabela 6.46: Política : Periódica - Métrica: atraso - Grau de Dinamismo: 20%

Instância	Média	Intervalo de Confiança
R101.txt	844,80	699,61 - 989,99
R201.txt	634,14	401,78 - 866,50
C101.txt	5198,59	4105,52 - 6291,65
C201.txt	15088,86	6398,08 - 23779,64
RC101.txt	532,62	439,42 - 625,81
RC201.txt	761,57	382,43 - 1140,71

Tabela 6.47: Política : Periódica - Métrica: atraso - Grau de Dinamismo: 40%

Instância	Média	Intervalo de Confiança
R101.txt	1222,26	1066,88 - 1377,63
R201.txt	3343,16	2503,21 - 4183,11
C101.txt	9104,46	6846,05 - 11362,86
C201.txt	15305,82	11247,97 - 19363,67
RC101.txt	649,53	546,16 - 752,91
RC201.txt	2766,02	2192,60 - 3339,44

Tabela 6.48: Política : Periódica - Métrica: atraso - Grau de Dinamismo: 60%

Instância	Média	Intervalo de Confiança
R101.txt	1855,35	1713,10 - 1997,59
R201.txt	4452,52	3340,36 - 5564,67
C101.txt	18836,01	15878,05 - 21793,97
C201.txt	20839,92	17138,04 - 24541,80
RC101.txt	771,75	675,48 - 868,02
RC201.txt	2884,04	2051,03 - 3717,05

Tabela 6.49: Política : Periódica - Métrica: atraso - Grau de Dinamismo: 80%

Instância	Média	Intervalo de Confiança
R101.txt	2291,78	2141,58 - 2441,98
R201.txt	4234,46	3332,70 - 5136,23
C101.txt	20047,09	18826,86 - 21267,31
C201.txt	38582,50	33953,74 - 43211,26
RC101.txt	1099,83	977,19 - 1222,48
RC201.txt	3792,14	2787,54 - 4796,74

6.4.1 Distância Percorrida

Observando-se a Figura 6.15, verifica-se que, quando é aplicada a política Online com uma função de avaliação que considera somente a métrica da distância total percorrida, a tendência é que a distância cresça com o aumento do grau de dinamismo.

A Figura 6.16, destaca os resultados obtidos considerando-se o grau de dinamismo variando de 20% para 80%, reforçando e ilustrando, de forma mais clara, esta tendência.

No que se refere à Política Por Demanda, observou-se que a espera por uma determinada quantidade de novas requisições para que se faça o reajuste de rotas ofuscou o efeito da variação do parâmetro grau de dinamismo, de modo que esta variação não se retratou em diferenças marcantes nos resultados obtidos, considerando-se de forma isolada, a métrica da distância total percorrida, como ilustra a Figura 6.17. Mesmo destacando os testes realizados com 20% e com 80% de grau de dinamismo, pode-se verificar que os intervalos de confiança não são disjuntos, fato que ratifica o raciocínio exposto (Figura 6.18).

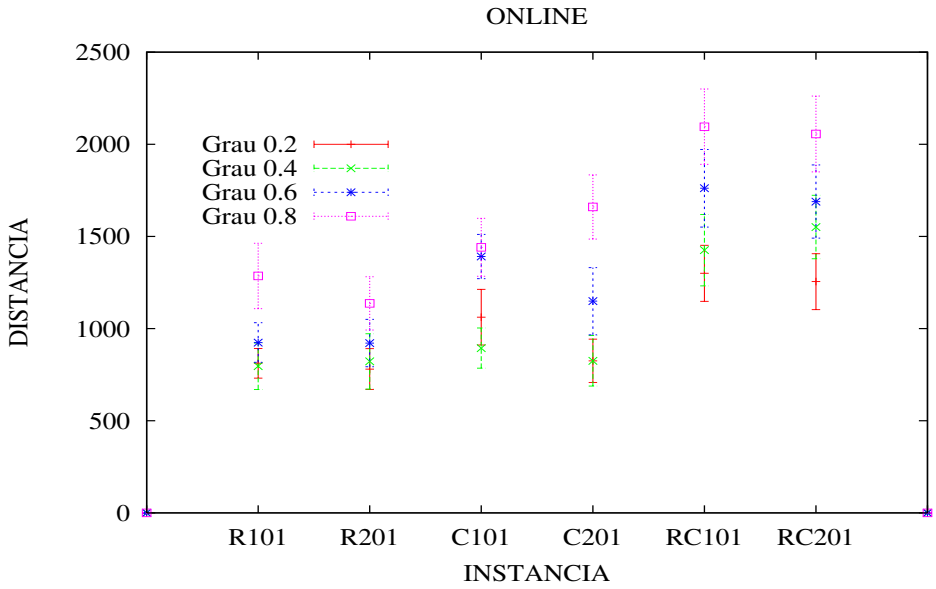

Figura 6.15: Distância percorrida - Graus de dinamismo: 20%, 40%, 60% e 80%

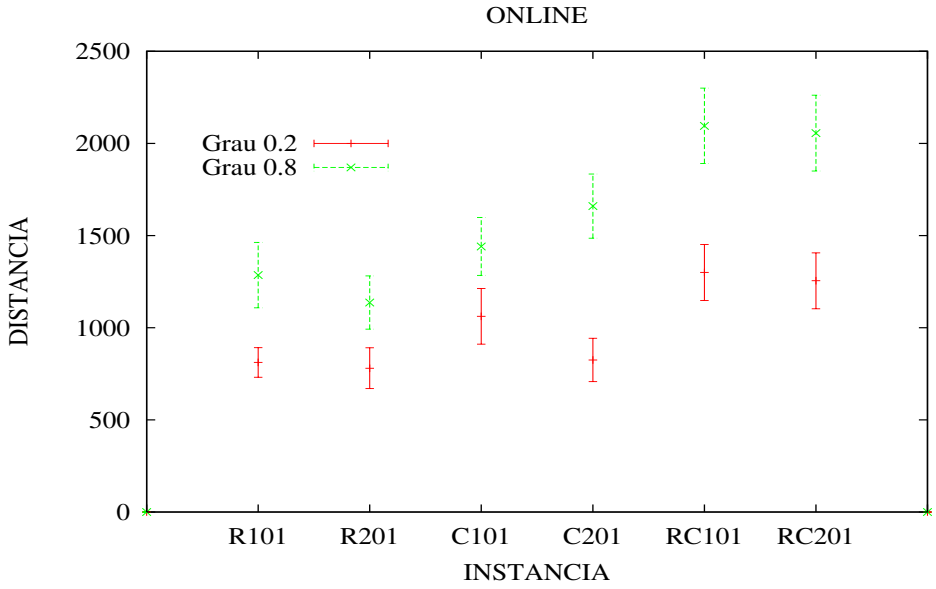

Figura 6.16: Distância percorrida - Graus de dinamismo: 20% e 80%

No tocante à Política Periódica, acontece algo muito semelhante ao que foi observado nos testes da Política Por Demanda. Isso occore devido a um fator que está presente nas duas políticas, que é a espera por um evento para que se inicie o processo de reajuste de rotas para incorporar as novas requisições. Se na Política Por Demanda esse evento é representado pelo momento no qual a fila de novas requisições está cheio, na Política Periódica, este evento ocorre quando expira uma fatia de tempo, definida previamente, que se espera entre dois

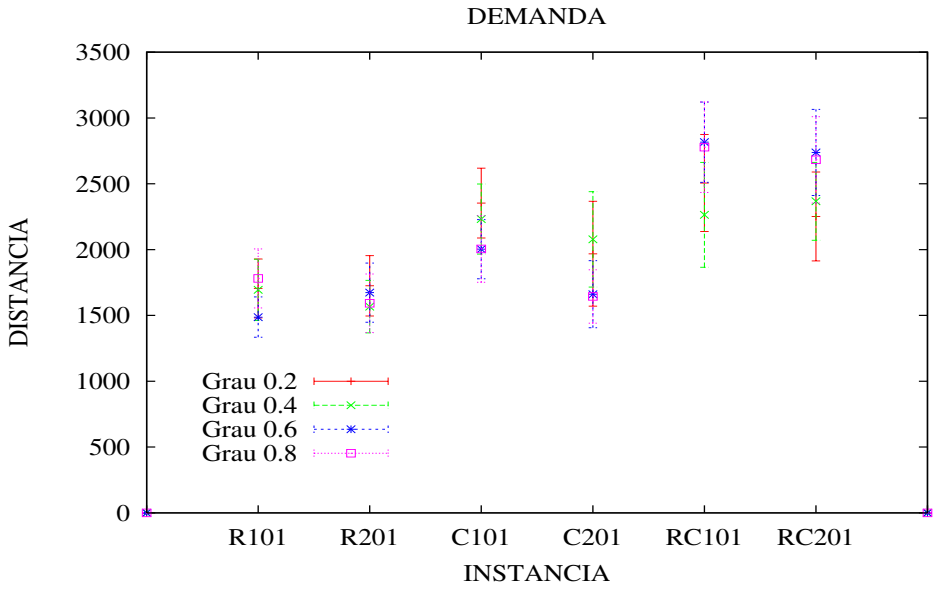

Figura 6.17: Distância percorrida - Graus de dinamismo: 20%, 40%, 60% e 80%

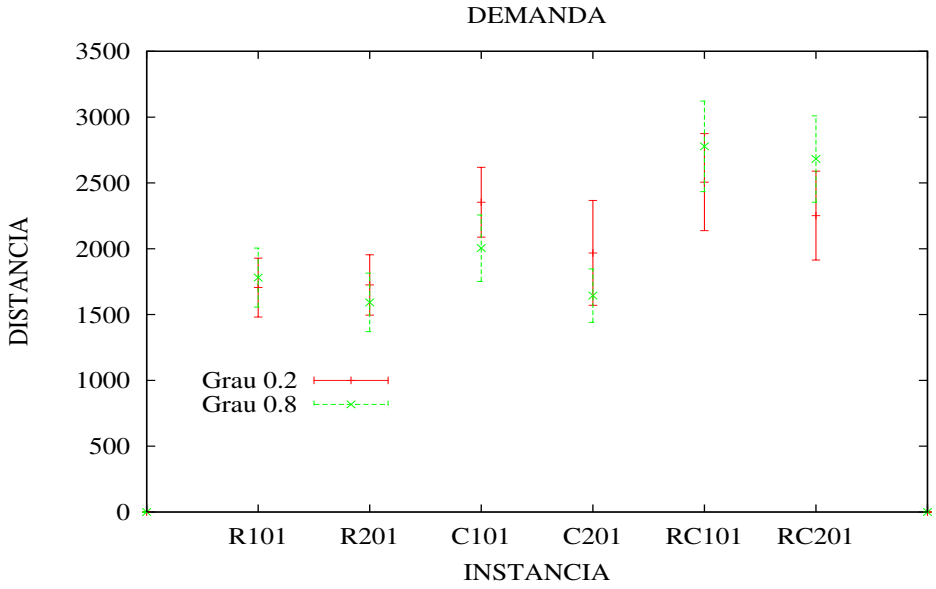

Figura 6.18: Distância percorrida - Graus de dinamismo: 20% e 80%

momentos de reajustes de rotas.

Essa semelhança reflete em resultados bastante semelhantes no que diz respeito à análise da métrica distância percorrida, como ilustram as Figuras 6.19 e 6.20

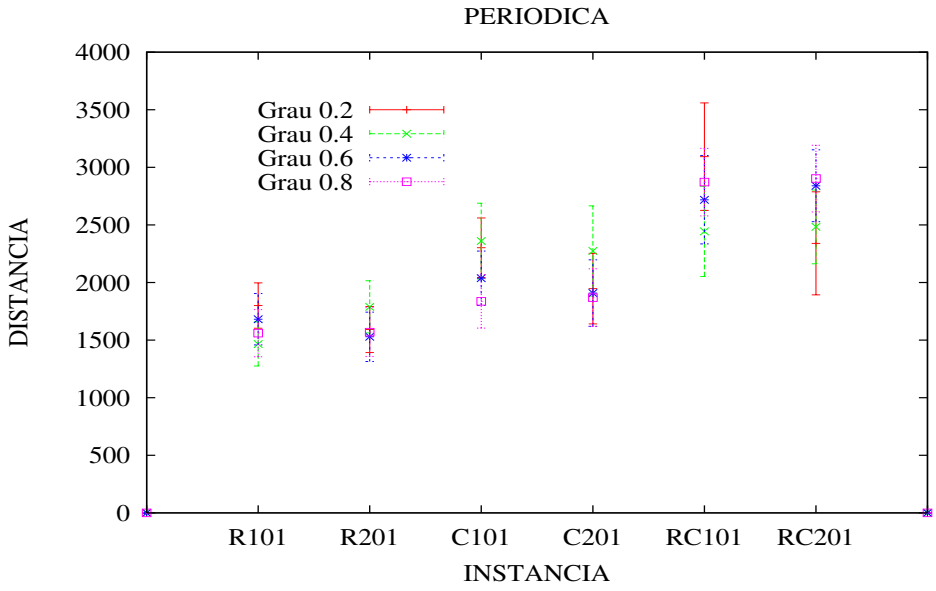

Figura 6.19: Distância percorrida - Graus de dinamismo: 20%, 40%, 60% e 80%

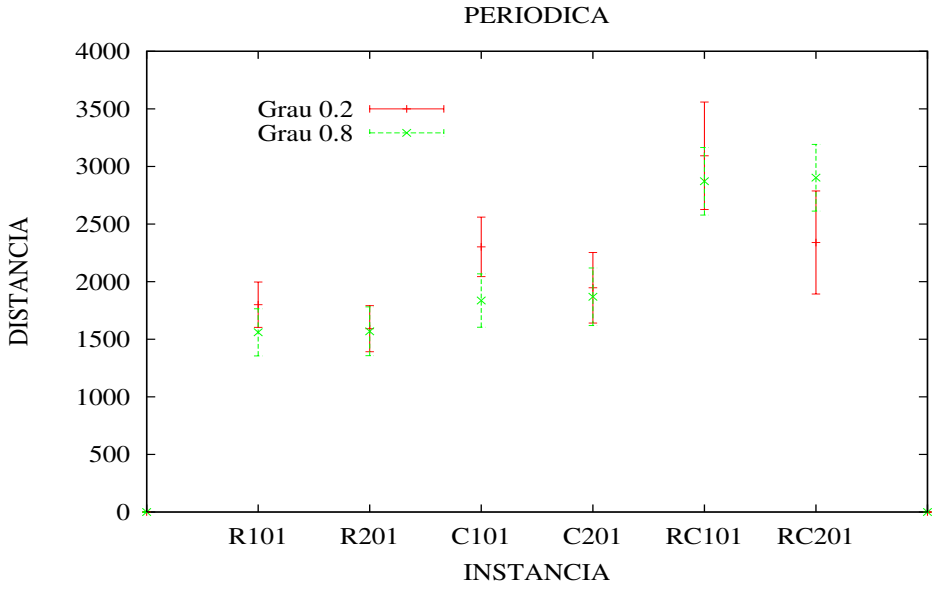

Figura 6.20: Distância percorrida - Graus de dinamismo: 20% e 80%

6.4.2 Veículos Utilizados

Observando-se os resultados obtidos quando foi considerada, de forma isolada, a métrica número de veículos utilizados no roteamento, verificou-se que a variação do parâmetro grau de dinamismo provoca uma variação muito pequena na quantidade de veículos usados em virtude desta variação não interferir na carga total a ser transportada. No que se refere à Política Online, observa-se que o número de veículos utilizados não varia muito, mas, em

geral, quando há variação, o número de veículos é maior quanto maior é o grau de dinamismo, como ilustram as Figuras 6.21 e 6.22.

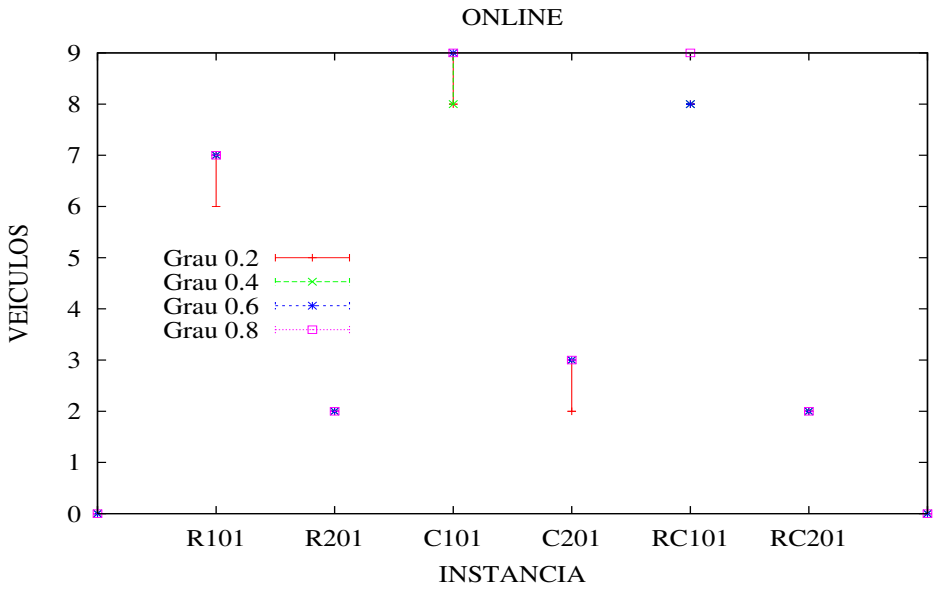

Figura 6.21: Veículos Utilizados - Graus de dinamismo: 20%, 40%, 60% e 80%

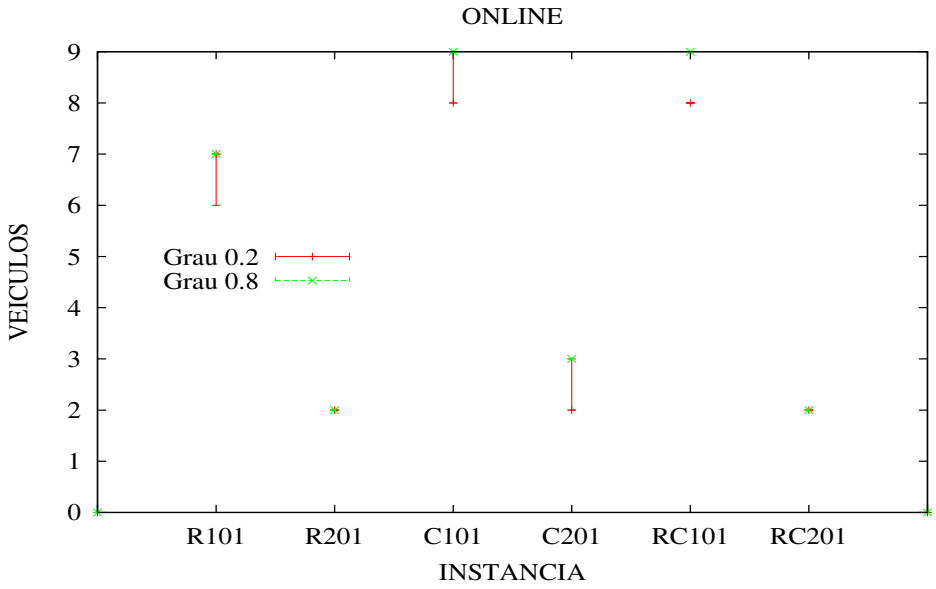

Figura 6.22: Veículos Utilizados - Graus de dinamismo: 20% e 80%

No tocante à Política Por Demanda, observou-se que o número de veículos geralmente cresce com o aumento do grau de dinamismo, como ilustram as Figuras 6.23 e 6.24. Isto ocorre porque esta política causa uma certa demora no repasse das novas requisições ao Otimizdor,

de modo que, podem ser necessários mais veículos para atender ao volume de requisições que são incorporadas, pricipalmente quando se aproxima o *deadline* do período.

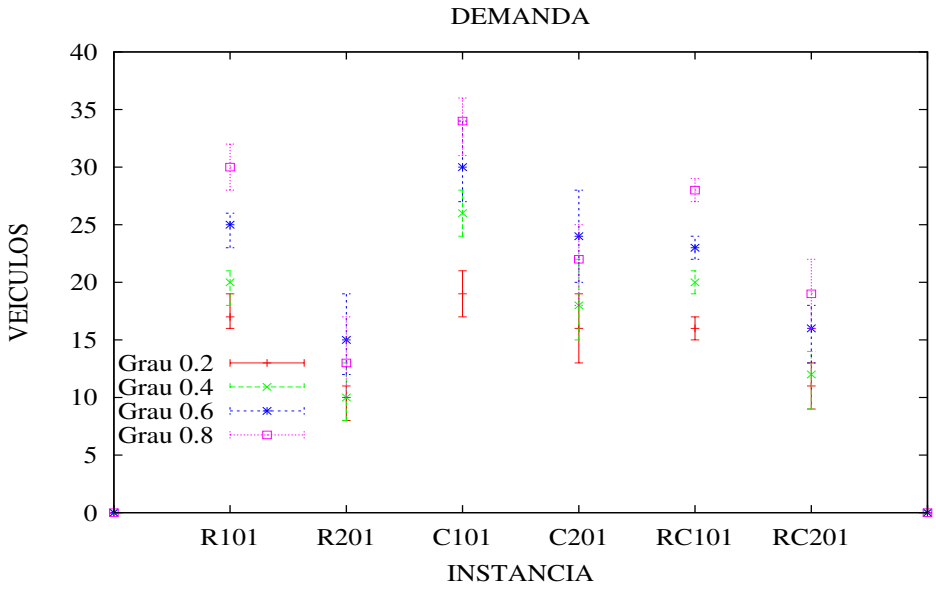

Figura 6.23: Veículos Utilizados - Graus de dinamismo: 20%, 40%, 60% e 80%

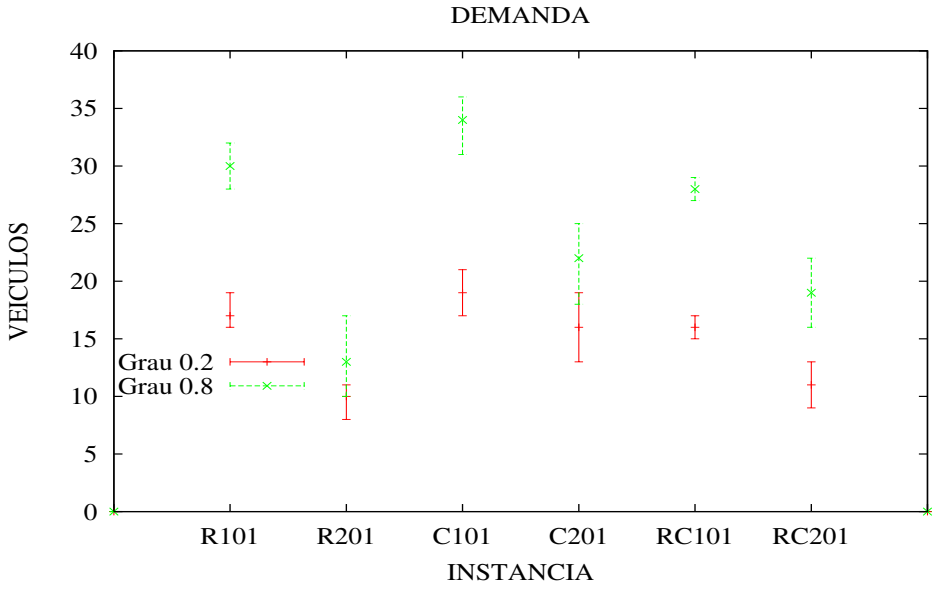

Figura 6.24: Veículos Utilizados - Graus de dinamismo: 20% e 80%

De maneira análoga ao que ocorre na Política Por Demanda, na Política Periódica, o número de veículos tende a crescer com o aumento do grau de dinamismo, conforme pode ser visualizado graficamente nas Figuras 6.25 e 6.26. A exemplo da Política Por Demanda, a Política Periódica também acarreta uma espera de uma fatia de tempo pré-determinada

entre os reajustes de rotas, fato que pode implicar necessidade de mais veículos para que seja efetuado o roteamento.

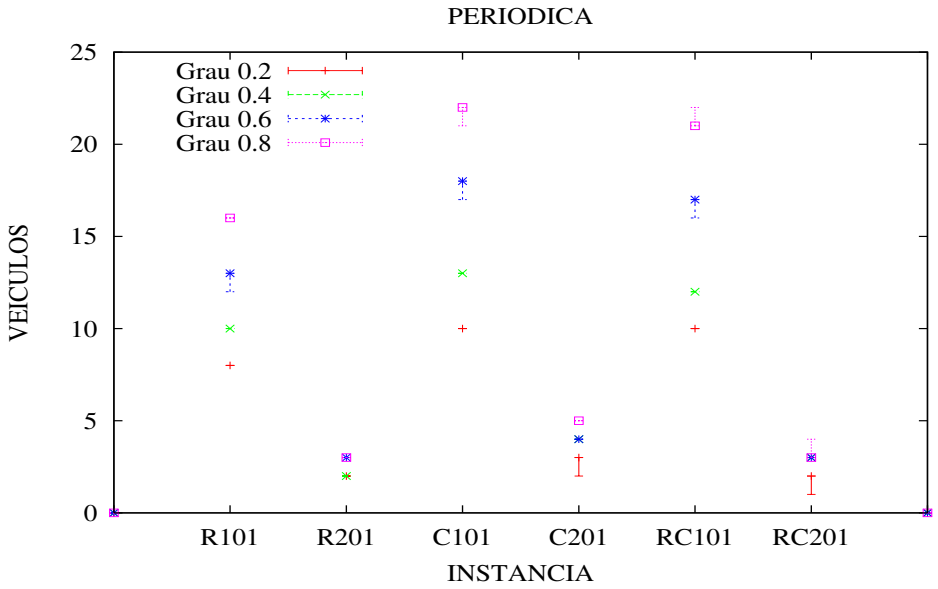

Figura 6.25: Veículos Utilizados - Graus de dinamismo: 20%, 40%, 60% e 80%

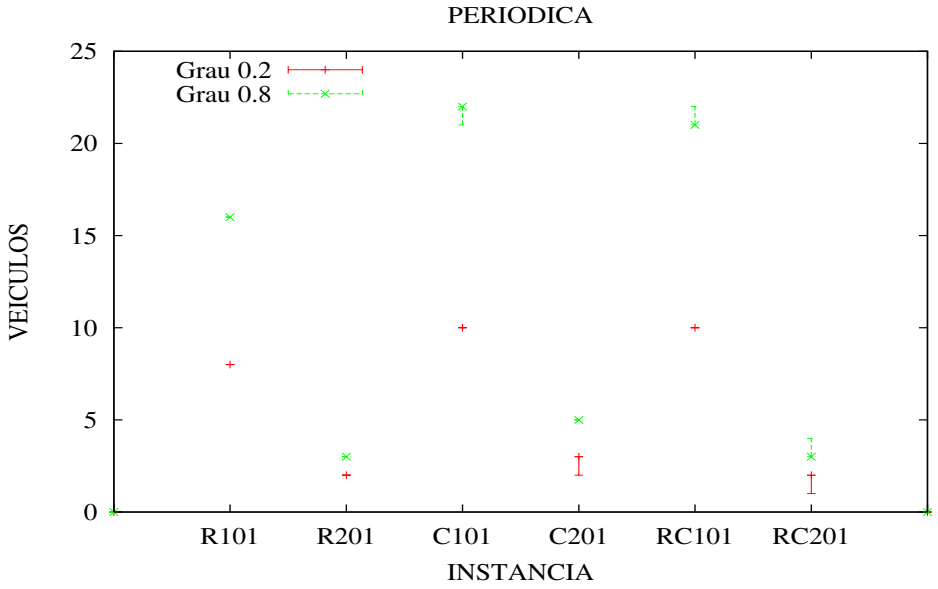

Figura 6.26: Veículos Utilizados - Graus de dinamismo: 20% e 80%

6.5 Conclusão

Neste capítulo, foram apresentados os cenários de teste, utilizando as instâncias clássicas de Solomon (1987), juntamente com os parâmetros com os quais o arcabouço foi configurado. Foram listados os resultados obtidos, variando-se o grau de dinamismo do problema, bem como a política de roteamento dinâmico utilizada.

Pode-se verificar, pela análise estatística apresentada, que as políticas demonstram resultados muito semelhantes, conforme mostra o intervalo de confiança das médias amostrais da função de avaliação, bem como das 4 (quatro) métricas que a compõem: distância total percorrida, número de veículos utilizados, tempo total de espera e tempo total de atraso. Nos testes, buscou-se minimizar o valor da função de avaliação como um todo, considerando-se as quatro métricas concomitantemente e os dados apresentados nas tabelas referentes a cada uma dessas métricas são os valores que compõem a função de avaliação juntamente com os multiplicadores definidos nos parâmetros do arcabouço.

Por ser um problema com características muito específicas, como o fato dos tempos de viagem serem variáveis e dependentes de um modelo de tráfego que estima a sua variação, não é viável comparar os resultados obtidos com os existentes na literatura para problemas de roteamento de veículos, considerando-se o problema proposto nessa dissertação (*Problema de Roteamento Dinâmico de Veículos com Janelas de Tempo e Tempos de Viagem Variáveis*). Foram utilizadas as instâncias clássicas propostas por Solomon (1987) em virtude dessas instâncias serem já conhecidas pela sua eficácia em testar algoritmos propostos para problemas de roteamento de veículos.

Capítulo 7

Considerações Finais e Trabalhos Futuros

Neste trabalho foi proposto um arcabouço que utiliza uma heurística de geração de colunas para o problema de roteamento dinâmico de veículos com janelas de tempo e tempos de viagem variáveis. As requisições a serem atendidas são retiradas dos arquivos de instâncias de Solomon (1987) e são geradas em tempo real. As requisições geradas são armazenadas em uma fila e repassadas ao algoritmo evolucionário dinâmico, de acordo com a política de roteamento dinâmico de veículos que estiver configurada. A política *Online* repassa as requisições imediatamente após serem geradas. A política *Por Demanda* repassa as requisições somente quando a fila atinge a sua capacidade limite de armazenamento e a política *Periódica* de tempos em tempos realimenta o algoritmo evolucionário dinâmico com as novas requisições. As ilhas de evolução, que são algoritmos evolucionários executando em paralelo, após gerarem suas soluções, as repassam ao *Problema de Partição de Conjuntos* encarregado de selecionar a melhor combinação de rotas (aquela que minimize os custos). Com o intuito de explorar-se melhor o espaço de busca de soluções, este processo é repetido com um problema reduzido. Como as colunas (rotas) geradas não são muitas, o Problema de Partição de Conjuntos executa na ordem de milissegundos.

A principal contribuição deste trabalho diz respeito à abordagem dinâmica do problema, associada ao fato do algoritmo buscar, durante todo o período de operação, melhores soluções para o prolema. Essas características, acrescidas ao fato dos tempos de viagem serem variáveis, dificultam ainda mais o problema. A revisão bibliográfica possibilita a constatação de que são ainda escassos trabalhos que consideram todas essas características.

Pode-se verificar que as políticas de roteamento dinâmico (Online, Por Demanda e Periódica) apresentam resultados bem próximos, todavia, em cada uma dessas políticas analisadas, podem ser encontradas vantagens e desvantagens.

A política Online repassa as novas requisições para o Otimizador e, por conseguinte para o algoritmo evolucionário dinâmico no momento em que as requisições são efetuadas. A vantagem do algoritmo dinâmico receber as novas requisições o quanto antes é que ele tem mais opções de posições nas rotas para inserir tais requisições., enquanto que as políticas

Períodica e Por Demanda não apresentam tal vantagem, pois, como elas esperam um período de tempo (Política Periódica) ou esperam encher a fila (Política Por Demanda), podem perder a oportunidade de inserir novas requisições entre requisições já atendidas no momento que o algoritmo recebe as novas requisições.

Por outro lado, as Políticas Por Demanda e Periódica tiram proveito de, no momento no qual fazem o reajuste de rotas, terem a tendência de conhecerem mais do que apenas uma requisição, como ocorre na Política Online. Isto facilita o processo de otimização das rotas.

Como trabalhos futuros, pretende-se desenvolver uma relaxação lagrangeana para o problema, de modo a gerar um limite inferior que, combinado com o limite superior fornecido pelo arcabouço apresentado neste trabalho, servirá de base para a implementação de um algoritmo *Branch-and-Bound* dinâmico que seja capaz de incorporar novas requisições durante o processo de varredura da árvore de soluções.

Outras heurísticas gulosas podem ser adaptadas para o contexto dinâmico, a exemplo do que foi feito com a heurística *Stochastic PFIH*. Tais heurísticas podem ser utilizadas para gerar-se a população inicial de indivíduos.

Pode-se propor uma nova política de roteamento dinâmico de veículos na qual, após cada repasse de novas requisições ao *Otimizador*, seja selecionado aleatoriamente o comportamento a ser adotado até o próximo repasse de requisições. Este tempo entre os repasses de requisições é denominado *round*. De acordo com a escolha aleatória, esta política denominada de *Política Híbrida*, pode comportar-se por um *round* como Online, Por Demanda ou Periódica.

Pretende-se utilizar uma estrutura de grafo considerando clientes previamente cadastrados, de modo que, quando for necessário adicionar ao problema novas requisições efetuadas por tais clientes, não será necessário criar um novo vértice no grafo e uma aresta conectando este vértice a todos os já existentes. Este grafo já será previamente gerado em uma fase de pré-processamento. Assim, reajustes no grafo serão necessários somente quando surgir uma nova requisição que não tenha sido efetuada por nenhum cliente cadastrado.

Pretende-se realizar novos testes com o objetivo de avaliar, de forma isolada, ou seja, uma a uma, as seguintes métricas: distância total percorrida, número de veículos utilizados, tempo total de espera e tempo total de atraso. Pretende-se também analisar o ganho que se tem ao utilizar as ilhas de evolução, o problema de Partição de Conjuntos e o problema reduzido.

Avaliações da relação custo/benefício de algumas fases do arcabouço, tais como a fase do Problema Reduzido e a fase do Problema de Partição de Conjuntos são também necessárias e fazem parte das próximas atividades a serem desenvolvidas.

O problema abordado também pode vir a ser adaptado para dar suporte ao cancelamento de requisições, no decorrer do período de operação da frota.

Referências Bibliográficas

Alvarenga, G.; Mateus, G. e Tomi, G. (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34:1561–1584.

Alvarenga, G. B.; Mateus, G. R. e Tomi, G. (2003). Finding near optimum solution for vehicle routing problem with time windows. *Second International Workshop on Freight Transportation - ODYSSEUS 2003*, pp. 1–13.

Bent, R. e Hentenryck, P. (2006). A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research*, 33:875–893.

Bertsimas, D. e Ryzin, G. (1993). Stochastic and dynamic vehicle routing in the euclidean plane with multiple capacitated vehicles. *Operations Research*, 41:60–76.

Bertsimas, D. e Ryzin, G. V. (1991). A stochastic and dynamic vehicle routing problem in the euclidean plane. *Operations Research*, 39:601–615.

Chabini, I. (1997). A new algorithm for shortest paths in discrete dynamic networks. *Proceedings of the 8th IFAC Symposium on Transportation Systems, Chania, Greece*, pp. 551–556.

Chen, H.; Hsueh, C. e Chang, M. (2006). The real-time time-dependent vehicle routing problem. *Transportation Research Part E*, 42:383–408.

Cheung, R. e Powell., W. (1996). An algorithm for multistage dynamic networks with random arc capacities, with an application to dynamic fleet management. *Operations Research*, 44:951–963.

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

Davis, L. (1987). *Genetic Algorithms and Simulated Annealing.* Pitman.

Davis, L. (1991). *Handbook of Genetic Algorithms.* Van Nostrand Reinhold.

Dondo, R. e Cerda, J. (2007). A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. *European Journal of Operational Research*, 176:1478–1507.

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

Grefenstette, J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Trans SMC*, 16:128.

Grefenstette, J. (1990). *Genetic algorithms and their applications.* A. Kent and J.G. Williams, editors Encyclopedia of Computer Science and Technology.

Hadley, G. (1964). *Nonlinear and Dynamic Programming.* Addison-Wesley, Reading, MA.

Haghani, A. e Jung, S. (2005). A dynamic vehicle routing problem with time-dependent travel times. *Comput. Oper. Res.*, 32(11):2959–2986.

Hill, A. e Benton, W. (1992). Modeling intra city time-dependent travel speeds for vehicle scheduling problems. *Journal of the Operation Research Society*, 43 (4):343–351.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* University of Michigan Press.

Ichoua, S.; Gendreau, M. e Potvin, J. (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, 144(2):379–396.

Jaim, R. (1991). *The Art of Computer Systems Performance Analysis.* John Wiley and Sons.

Kallehaugea, B.; Larsenb, J. e Madsen, O. (2006). Lagrangian duality applied to the vehicle routing problem with time windows. *Computers & Operations Research*, 33:1464–1487.

Lewis, H. F. e Sexton, T. R. (2007). On the tour partitioning heuristic for the unit demand capacitated vehicle routing problem. *Operations Research Letters, In Press.*

Longo, H.; Poggi, M. e Uchoa, E. (2006). Solving capacitated arc routing problems using a transformation to the cvrp. *Computers & Operations Research*, 33:1823–1837.

Lysgaard, J.; Letchford, A. e Eglese, R. (2003). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming.*

Malandraki, C. (1989). *Time dependent vehicle routing problems: Formulations, solution algorithms and computations experiments.* PhD thesis, Northwestern University, Evanston, III.

Malandraki, C. e Daskin, M. (1992). Time dependent vehicle routing problems: Formulations , properties and heuristic algorithms. *Transportation Science*, 26(3):185–200.

Malandraki, C. e Dial, R. (1996). A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90:45–55.

Mautora, T. e Naudina, E. (2007). Arcs-states models for the vehicle routing problem with time windows and related problems. *Computers & Operations Research*, 34:1061–1084.

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag.

Moghaddama, R. T.; Safaeib, N. e Rabbania, M. K. M. (2006). A new capacitated vehicle routing problem with split service for minimizing fleet cost by simulated annealing. *Journal of the Franklin Institute.*

Powell, W. (1986). A stochastic formulation of the dynamic vehicle allocation problem. *Transportation Science*, 20:117–129.

Powell, W. (1987). An operational planning model for the dynamic vehicle allocation problem with uncertain demands. *Transportation Research*, 21B:217–232.

Powell, W. (1988). *Comparative Review of Alternative Algorithm for the Dynamic Vehicle Allocation Problem.* In:Golden BL, Assad AA, editors., Amsterdam: North-Holland.

Psaraftis, H. (1988). Dynamic vehicle routing problems. *Vehicle Routing: methods and Studies. B.L. Golden and A.A. Assad editors*, pp. 223–248.

Regan, A.; Herrmann, J. e Lu, X. (2002). The relative performance of heuristics for the dynamic traveling salesman problem. *Proceedings of the 81st Annual Meeting of the Transportation Research Board,Washington, DC.*

Regan, A.; Mahmassani, H. e Jaillet., P. (1996). Dynamic decision making for commercial fleet operations using real-time information. *Transportation Research Record*, 1537:91–97.

Regan, A.; Mahmassani, H. e Jaillet, P. (1998). Evaluation of dynamic fleet management system: a simulation framework. *Transportation Research Record*, 1645:176–184.

Shen, Y. e Potvin, J. (1995). A computer assistant for vehicle dispatching with learning capabilities. *Operations Research*, 61:189–211.

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

Tarantilis, C. (2005). Solving the vehicle routing problem with adaptive memory programming methodology. *Computers & Operations Research*, 32:2309–2327.

Thompson, P. e Psaraftis, H. (1993). Cyclic transfer algorithms for multivehicle routing and scheduling problems. *INFORMS - Institute for Operations Research and the Management Sciences (INFORMS)*, 41:935–946.

Toth, P. e Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123:487–512.

Xu, H. (1994). Optimal policies for stochastic and dynamic vehicle routing problems. *Ph.D dissertation, Massachusetts Institute of Technology, Cambridge, MA*.