MONTAGEM DE INFRAESTRUTURA E PREDIÇÃO DE TRAJETÓRIA EM REDES VEICULARES

EVELLYN SOARES CAVALCANTE

MONTAGEM DE INFRAESTRUTURA E PREDIÇÃO DE TRAJETÓRIA EM REDES VEICULARES

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais. Departamento de Ciência da Computação. como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: ANTÔNIO ALFREDO FERREIRA LOUREIRO
COORIENTADOR: ANDRÉ LUIZ LINS DE AQUINO

Belo Horizonte
Julho de 2013

Cavalcante, Evellyn Soares
C376m Montagem de infraestrutura e predição de trajetória em redes veiculares / Evellyn Soares Cavalcante. — Belo Horizonte, 2013
xxvi, 68 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais. Departamento de Ciência da Computação.
Orientador: Antônio Alfredo Ferreira Loureiro

1. Computação – Teses. 2. Redes de Computadores – Teses. 3. Redes de Sensores Sem Fio – Teses.
I. Orientador. II. Título.

CDU 519.6*22 (043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Montagem de infraestrutura e predição de trajetória em redes veiculares

EVELLYN SOARES CAVALCANTE

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. ANTONIO ALFREDO FERREIRA LOUREIRO - Orientador
Departamento de Ciência da Computação - UFMG

PROF. ANDRÉ LUIZ LINS DE AQUINO - COORIENTADOR
Instituto de Computação - UFAL

PROFA. GISELE LOBO PAPPA
Departamento de Ciência da Computação - UFMG

PROFA. LUCIANA SALETE BURIOL
Instituto de Informática - UFRGS

Belo Horizonte, 03 de julho de 2013.

Dedico esse trabalho aos meus pais, Bete e Barros, que nunca mediram esforços para a minha felicidade, e à minha irmã, Jéssica, por ser minha parceira de vida.

Agradecimentos

Agradeço a Deus, primeiramente, por ter me dado saúde e iluminado minha cabeça e coração para perseverar.

Aos meus pais, Bete e Barros, por sempre me incentivarem a lutar pelos meus objetivos e terem uma palavra de conforto nos momentos mais difíceis. À minha irmã, Jéssica, por estar sempre por perto e ajudar no que eu precisava. E à toda a minha família, que de alguma forma contribuiu pro meu sucesso.

Aos meus orientadores, Loureiro e André, que foram responsáveis pelo meu amadurecimento acadêmico.

Aos meus amigos da vida toda, especialmente Mayara e Elen, que acompanharam todo meu percurso e torceram muito por mim.

Aos meus amigos belo horizontinos, pela companhia nos momentos de diversão que ajudaram a manter a sanidade. Agradecimentos especiais para o Samer, pela ajuda permanente, mesmo quando eu não pedia; ao Felipe, pelas melhores conversas e desabafos que me ensinaram tanto; à Izabela, pela confiança incondicional ao me apresentar sua casa como meu refúgio.

À família Soares Silveira, representada pela Nanda, que me acolheu como um dos seus e me ajudou a aguentar a saudade de casa.

E a todos aqueles que me ajudaram de alguma forma na realização desse trabalho.

"É preciso ter dúvidas. Só os estúpidos têm uma confiança absoluta em si mesmos."
(Orson Welles)

Resumo

Redes Veiculares Ad-Hoc (RVAH) são redes formadas por veículos que detêm a capacidade de sensoriamento e comunicação e trocam mensagens entre si e/ou entre pontos de acesso dispostos ao longo das estradas. Os pontos de acessos formam a infraestrutura das redes veiculares e têm grande importância na disseminação de informação pois ajudam a superar várias limitações de comunicação. Os serviços das RVAHs podem ser personalizados com a aplicação de técnicas de localização e rastreamento de veículos, o que dá condições para predizer trajetórias, por exemplo. Para atender a demanda de montagem de infraestrutura, esse trabalho propõe um algoritmo genético para distribuir pontos de acessos numa região de forma a alcançar a melhor cobertura de veículos. De forma complementar, considerando a personalização de dados disseminados, apresentamos uma modelagem, aplicável a algoritmos clássicos de aprendizado de máquina, para predizer trajetórias de veículos. Os resultados mostram que o algoritmo genético melhora em até 20.12 pontos percentuais a abordagem gulosa e que a árvore de decisão prever corretamente 0.85 das instâncias.

Abstract

Vehicular Ad-Hoc Networks (Vanets) are networks composed by vehicles within sensing and comunication capabilities and that exchange messages among themselves or among access points deployed in the region. Access points compose the Vanets infrastructure and are very important on the exchange of information because they facilitate and improve communication limitations. Vanet services can be customized applying localization and tracking techniques, therefore trajectory predictions can be done, for example. Concerning to the problem of installing infrastructure, this works proposes a genetic algorithm to distribute access points in a region to reache the best vehicle coverage. Regarding to the customization of the data dissemination, is presented a modeling, applicable to classic algorithms from machine learning, to predict trajectories of vehicles. Results show that the genetic algorithm presents solutions up to 20.12 percentual points better than the greedy solution and the decision tree classifies successfully 0.85 of the instances.

Resumo Estendido

Redes Veiculares Ad-Hoc (RVAHs) são redes formadas por veículos que detêm capacidade de sensoriamento e comunicação e trocam mensagens entre si e/ou entre pontos de acesso dispostos na região. Os dados coletados pelas RVAHs podem oferecer, dentre outros serviços, informações sobre as condições das estradas, do tráfego e do clima; o comportamento dos veículos e dos motoristas.

Além dos veículos, os principais agentes de disseminação de informações são os pontos de acessos, que ajudam a superar limitações de comunicação presentes nas redes veiculares. Dessa forma, é de grande importância um estudo para montar a infraestrutura da rede veicular, de maneira a facilitar e melhorar a qualidade da troca de informações.

A capacidade de conhecer a posição atual do veículo considerado, através de técnicas de localização e rastreamento, torna as aplicações mais interessantes, já que assim elas podem ser direcionadas e adaptadas ao ambiente envolvido e/ou ao usuário. Os dados provenientes dessa funcionalidade permitem que inferências sobre o comportamento do motorista sejam feitas, como, por exemplo, próximas posições, trajetórias a serem percorridas e mudanças de faixas.

Os dados de localização de veículos podem ser usados para montar a infraestrutura de uma rede veicular, pois se os dados pertencem a veículos que se movimentam numa região comum é possível identificar o comportamento global do tráfego e assim distribuir pontos de acesso de maneira a melhorar a qualidade da comunicação na rede. Da mesma maneira, esses dados são primordiais para realizar predições de trajetórias dos veículos, pois a partir de um histórico é possível identificar padrões dos motoristas e aplicar técnicas que tirem proveito dessa informação para prever um comportamento futuro.

Para atender a demanda de montagem de infraestrutura, esse trabalho propõe um algoritmo genético para distribuir pontos de acessos numa região de forma a alcançar a melhor cobertura de veículos. De forma complementar, visando a personalização na disseminação dos dados, é apresentada uma modelagem, aplicável a algoritmos clássicos de aprendizado de máquina, para predizer trajetórias de veículos.

O método genético, para a montagem da infraestrutura, foi aplicado em quatro cenários

com topologias reais de estradas na Suíça considerando um modelo de mobilidade veicular realista com duração de uma hora e meia. Os resultados mostraram que o algoritmo genético, com um método de inicialização otimizado que explora algumas das soluções encontradas pela abordagem gulosa e com operadores genéticos desenvolvidos com informações inerentes ao problema, apresenta soluções até 20.12 pontos percentuais melhores do que a abordagem gulosa. Outros resultados em que houve variação do número de pontos de acesso posicionados e o do tempo mínimo de recebimento da informação também mostrou que o algoritmo genético sempre supera a abordagem gulosa ou a gulosa aleatória.

Para o problema de predição, é utilizada uma base de dados real, coletada na cidade de Borlange, na Suécia, que possui 24 usuários, com diferentes características. Essa base foi adequada para aplicá-la a algoritmos de classificação utilizando o conceito de janela deslizante. Foi apresentado um estudo quantitativo da base e a análise do seu comportamento com quatro algoritmos de aprendizado implementados na biblioteca *scikit-learn*: (i) *k-Nearest Neighbors*, (ii) *Naive-Bayes*, (iii) SVM e (iv) Árvore de decisão. Cada rota do veículo é modelada como um grafo e o objetivo é, dada uma sequência de arestas, prever a próxima aresta. Os resultados mostraram que a árvore de decisão é o algoritmo que melhor classifica as instâncias, alcançando resultados com 0.85 de sucesso na predição.

Lista de Figuras

Lista de Tabelas

Lista de Siglas

AG	Algoritmo Genético
KNN	*k-Nearest Neighbors*
PA	Ponto de Acesso
PMCLT	Problema da Máxima Cobertura com Limite de Tempo
RSSF	Rede de Sensores sem Fio
RV	Rede Veicular
RVAH	Rede Veicular Ad-Hoc
STI	Sistema de Transporte Inteligente
SVM	*Support Vector Machines*
VANET	*Vehicular Ad-Hoc Network*

Sumário

Capítulo 1

Introdução

Com o advento de veículos com capacidade de sensoriamento e comunicação, a pesquisa em Sistema de Transporte Inteligente (STI) vem se consolidando e é uma área bastante promissora [Dimitrakopoulos & Demestichas, 2010]. Cada vez mais, aplicações que fazem uso das Rede Veicular Ad-Hoc (RVAH) , do inglês *Vehicular Ad-Hoc Network (VANET)* , tornam-se presentes e necessárias ao dia-a-dia [Hartenstein & Laberteaux, 2008].

Uma RVAH é uma rede *ad-hoc* dinâmica composta por veículos que possuem equipamentos de sensoriamento e comunicação sem-fio e, por isso, durante o seu deslocamento, são capazes de obter informações sobre o ambiente que os rodeiam e trocarem mensagens entre si ou entre pontos de acessos distribuídos pela cidade [Yousefi et al., 2006; Wang & Li, 2009]. Assim, são identificados dois paradigmas principais de comunicação: **veículo-para-veículo (V2V)**, quando a comunicação é entre veículos, e **veículo-para-infraestrutura (V2I)**, quando a comunicação é entre o veículo e a infraestrutura (algum ponto de acesso).

Os dados coletados pelas redes veiculares podem oferecer, dentre outros serviços, informações sobre as condições das estradas, do tráfego e do clima, sobre o comportamento dos veículos e dos motoristas. Tais informações são úteis para uma gama de aplicações: alerta de segurança, assistência ao motorista, roteamento de tráfego e transmissão de propagandas oportunísticas [Faezipour et al., 2012]. Além disso, essas informações podem ser usadas para criar um sistema de tráfego inteligente, que pode automaticamente atualizar os ciclos dos semáforos, indicar prováveis zonas de pedágio, medir a quantidade diária de veículos nas estradas, disseminar informações personalizadas aos motoristas, etc. [Barba et al., 2012; Vegni et al., 2013].

Alguns cenários em que as redes veiculares podem ser utilizadas são ilustrados na Figura 1.1 [Macedo et al., 2012]. O primeiro cenário representa uma área com a presença de poucos veículos como, por exemplo, rodovias. O segundo e terceiro cenário ilustram áreas urbanas que eventualmente possuem uma maior quantidade de veículos. Em todos os casos,

a informação é transmitida quando receptor e transmissor estão dentro de seus alcances de transmissão ou há uma infraestrutura de comunicação disponível para ser utilizada pelos veículos.

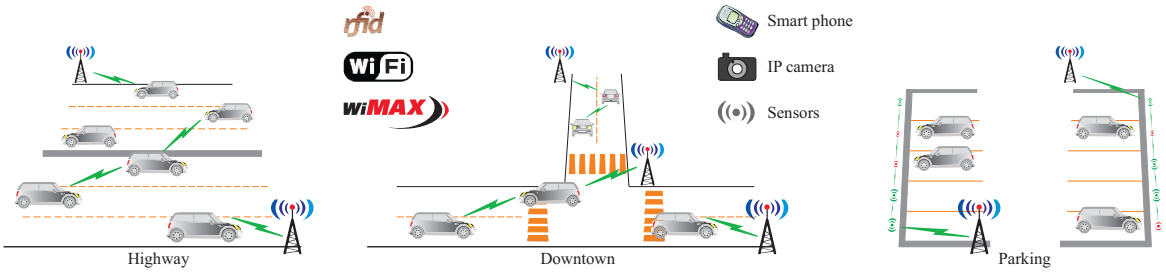


Figura 1.1: Sistema de transporte inteligente [Macedo et al., 2012]

Nos cenários mostrados na Figura 1.1, a disseminação de informação é um aspecto crucial. Além dos veículos, os principais agentes de disseminação de informações são os pontos de acessos, que ajudam a superar limitações de comunicação presentes nas redes veiculares, tais como: perda de dados; infraestrutura de roteamento dinâmica; e latência, presente em várias aplicações móveis sem fio [Lee et al., 2009]. Dessa forma, é de grande importância um estudo para montar a infraestrutura da rede veicular, de maneira a facilitar e melhorar a qualidade da troca de informações.

Outro estudo que vem adquirindo grande destaque para a área de transportes inteligentes refere-se aos métodos para localização e rastreamento de veículos. A capacidade de conhecer a posição atual do veículo considerado torna as aplicações mais interessantes, já que elas podem ser direcionadas e adaptadas ao ambiente envolvido e/ou ao usuário. Vários estudos aproveitam a oportunidade de comunicação para obter maior acurácia na localização e rastreamento por meio de cooperação entre veículos [Triplett et al., 2009; Fallah et al., 2010; Yao et al., 2011; Ramos et al., 2012].

Inserir técnicas de localização e rastreamento eficientes e eficazes nos STIs é essencial para o processo de entendimento do comportamento dos motoristas, pois torna-se possível oferecer serviços personalizados [Boukerche et al., 2008]. Os dados provenientes dessa funcionalidade permitem que inferências sobre o comportamento do motorista sejam feitas, como, por exemplo, próximas posições, trajetórias a serem percorridas e mudanças de faixas. A dificuldade dessa tarefa está relacionada, principalmente, à topologia extremamente dinâmica das redes veiculares, o que demanda adaptações para utilização dos métodos tradicionais existentes. Por outro lado, um fator facilitador é a limitação de movimentos a que os veículos estão sujeitos, já que existem demarcações bem definidas (estradas, rodovias, ruas, avenidas) por onde é permitida sua locomoção.

Sabe-se que, no mundo real, o comportamento dos motoristas obedece a um padrão: em determinado horário e dia da semana seguem um conjunto de avenidas/ruas, e de acordo

com as condições do tráfego, podem escolher caminhos alternativos [Axhausen et al., 2003]. Esse padrão pode ser modelado para que seja possível fazer previsões de rotas.

Os dados de localização de veículos podem ser usados para montar a infraestrutura de uma rede veicular, pois se os dados pertencem a veículos que se movimentam numa região comum é possível identificar o comportamento global do tráfego e assim distribuir pontos de acesso de maneira a melhorar a qualidade da comunicação na rede.

Da mesma maneira, esses dados são primordiais para realizar predições de trajetórias dos veículos, pois a partir de um histórico é possível identificar padrões dos motoristas e aplicar técnicas que tirem proveito dessa informação para prever um comportamento futuro.

Identificados esses dois problemas, os principais objetivos desse trabalho são:

- Montagem de Infraestrutura: dado um número limitado de pontos de acessos, busca-se identificar quais as melhores localizações para fixá-los, de maneira que o maior número possível de veículos consiga receber a informação transmitida por eles.

- Predição de Trajetória de Veículo baseada em Histórico do Motorista: verificar e analisar a aplicabilidade de algoritmos clássicos para aprendizado de máquina considerando uma modelagem proposta para o problema de predição de trajetória de veículos.

Para o problema da montagem de infraestrutura, foi desenvolvida uma abordagem genética enriquecida com informações do algoritmo guloso proposto na literatura e com características próprias do problema. Os resultados obtidos são avaliados tendo como base o algoritmo guloso e um guloso aleatório, também desenvolvido nesse trabalho.

O método genético foi aplicado em quatro cenários com topologias reais de estradas na Suíça considerando um modelo de mobilidade veicular realista [Naumov et al., 2006] com duração de uma hora e meia. Os quatro cenários estão dentro de uma área de $100\,km^2$, e apresentam diferentes características de tráfego: o centro da cidade de Zurique e Winterthur foram utilizados para representar o tráfego denso; as áreas rurais de Baden e Baar para caracterizar o tráfego esparso.

Os resultados mostraram que o algoritmo genético, com um método de inicialização otimizado que explora algumas das soluções encontradas pela abordagem gulosa e com operadores genéticos desenvolvidos com informações inerentes ao problema, apresenta resultados até 20.12 pontos percentuais melhores do que a abordagem gulosa. Nesse caso particular, o percentual de veículos cobertos aumentou de 54.01% para 74.13%. Outros resultados em que houve variação do número de pontos de acesso posicionados e o do tempo mínimo de recebimento da informação também mostrou que o algoritmo genético sempre supera a abordagem gulosa ou a gulosa aleatória.

Para o problema de predição, foram realizados experimentos de caráter exploratório em que é utilizada uma base de dados real, coletada na cidade de Borlange, na Suécia, que possui 24 usuários com diferentes características [Frejinger, 2008]. Essa base foi adequada para aplicá-la a algoritmos de classificação utilizando o conceito de janela deslizante. Foi apresentado um estudo quantitativo da base e a análise do seu comportamento com quatro algoritmos de aprendizado implementados na biblioteca *scikit-learn* [Pedregosa et al., 2011]: (i) *k-Nearest Neighbors*, (ii) *Naive-Bayes*, (iii) SVM e (iv) Árvore de decisão. Cada rota do veículo é modelada como um grafo e o objetivo é, dada uma sequência de arestas, prever a próxima aresta. Os resultados mostraram que a árvore de decisão é o algoritmo que melhor classifica as instâncias, alcançando resultados com 0.85 de sucesso na predição.

As principais contribuições desse trabalho são (i) apresentar um algoritmo genético para fixar pontos de acessos naquelas posições que fornecem a melhor cobertura de veículos, alguns resultados foram publicados em Cavalcante et al. [2012a,b]; e (ii) apresentar uma modelagem para o problema de predição de trajetória de veículos baseada em histórico do motorista.

O restante deste trabalho está organizado da seguinte forma. O Capítulo 2 apresenta o problema da distribuição de pontos de acesso em redes veiculares e os métodos desenvolvidos, enquanto o Capítulo 3 descreve o problema da predição de trajetórias de veículos, a modelagem proposta e os algoritmos utilizados. O Capítulo 4 apresenta e discute os resultados dos experimentos realizados. A conclusão e trabalhos futuros são discutidos no Capítulo 5.

Capítulo 2

Montagem de Infraestrutura de uma Rede Veicular Ad-Hoc (RVAH)

Como visto anteriormente, posicionar Pontos de Acesso (PAs) da infraestrutura de uma Rede Veicular (RV) para melhorar a troca de informação sob alguma perspectiva é um desafio. Uma abordagem para o problema é, primeiramente, encontrar uma modelagem adequada, em seguida desenvolver e aplicar estratégias inteligentes e eficazes. Nesse capítulo, serão apresentados alguns trabalhos relacionados, a modelagem adotada, o Problema da Máxima Cobertura com Limite de Tempo (PMCLT) e as técnicas utilizadas para solucioná-lo: a (i) solução gulosa proposta na literatura; e a (ii) solução genética, desenvolvida nesse trabalho.

2.1 Trabalhos Relacionados

Para o estudo de distribuição de PAs em RVAH, inicialmente, foram consideradas as abordagens para o problema de cobertura e conectividade em Redes de Sensores sem Fio (RSSFs). Nesse contexto, vários autores propuseram diferentes soluções baseadas em uma variedade de métodos [Meguerdichian et al., 2001; Li et al., 2003; Rosi et al., 2008; Lochert et al., 2008]. Os primeiros trabalhos visitados consideram a cobertura como uma métrica de qualidade de comunicação. Nesse sentido, Huang & Tseng [2003] formulam o problema de cobertura como um problema de decisão, em que é dado um número k e o objetivo é determinar se uma área é coberta por pelo menos k sensores que fazem parte da RSSF. Eles propõem algoritmos de tempo polinomial, em termos de número de sensores.

Outro trabalho, proposto por Habib & Safar [2007], modela o problema de posiciona-

mento de nós para melhorar a cobertura em RSSFs como dois sub-problemas: planejamento da planta e posicionamento, análoga à solução para construir placas de circuito integrados. Neste caso, a área é divida em células geométricas bem definidas (problema de planejamento) e os dispositivos sensores devem ser atribuídos em um conjunto de células (problema de posicionamento). Os autores solucionam esses sub-problemas como um único problema de otimização, utilizando uma abordagem evolucionária.

Ainda nesta direção, o objetivo em Jia et al. [2008] é ativar somente os sensores necessários num determinado momento para ter cobertura total de uma área com uma RSSF densa cujos sensores foram depositados aleatoriamente, para economizar energia e aumentar o tempo de vida da rede. A solução proposta pelos autores é baseada na seleção de conjuntos de cobertura e utiliza um algoritmo de busca baseado em algoritmo genético de ordenação não dominante.

Apesar de apresentarem boas soluções para o problema de cobertura, características inerentes a RVAHs, como por exemplo, a alta mobilidade dos elementos e a possibilidade para a utilização de uma infraestrutura, inviabilizam a aplicação das soluções para RSSFs nos cenários voltados a RVAHs e consequentemente ao trabalho aqui apresentado.

Buscando abordagens para o problema de cobertura em RVAHs, encontra-se o trabalho proposto por Kchiche & Kamoun [2009] que é uma abordagem gulosa baseada na centralidade de grupo para selecionar a melhor disposição dos PAs a fim de prover comunicação mais estável e regular entre os veículos. Eles buscaram alcançar o melhor desempenho possível em termos de atraso e sobrecarga de comunicação. Num trabalho posterior, Kchiche & Kamoun [2010] mostram por intermédio de simulações que o uso de PAs pode otimizar o desempenho de uma RVAH, especialmente em regiões esparsas e em casos de comunicação de longa distância. Além disso, propuseram estratégias para deposição de PAs baseadas em centralidade e equidistância e mostraram que essas características são importantes para melhorar a qualidade de serviço.

Outra abordagem interessante é proposta por Sou [2010] que estuda um modelo de posicionamento de PAs para reduzir o consumo desnecessário de energia. Ele considera que os PAs devem ser dispostos separados pela mesma distância ao longo de uma estrada e devem alternar entre os modos ativo e inativo. O objetivo do trabalho foi escolher quais e quantos PAs deveriam entrar em modo ativo de forma a maximizar a economia de energia para satisfazer os critérios de conectividade.

Por fim, Trullols et al. [2010] apresenta três maneiras diferentes de modelar o problema de deposição de PAs: como um Problema da Máxima Cobertura, Problema da Mochila ou um PMCLT. Para cada modelo, eles provêem uma solução gulosa e uma de divisão e conquista. Para o PMCLT a solução gulosa alcança os melhores resultados. Com base nesses resultados, o trabalho aqui apresentado modela o problema de cobertura como um PMCLT, propõe uma

abordagem com Algoritmo Genético (AG) e compara o método proposto com a solução gulosa.

2.2 Modelagem do Problema

O problema de posicionamento de PAs pode ser modelado como um PMCLT e pode ser definido como segue. Seja um cruzamento entre duas ou mais rodovias, chamado interseção i; a área limitada pelo alcance de transmissão de raio R do PA; o conjunto de veículos S_i que passam na interseção i e cada veículo, $v_j \in S_i$, que possui um peso T representando seu tempo de permanência na interseção. Existem m veículos circulando durante o período de observação, e τ é o tempo mínimo necessário para um veículo receber a informação com sucesso. A transmissão não precisa ser realizada por um único PD, um deles pode começar a transmissão e outros podem terminá-la, desde que o veículo permaneça em suas áreas de alcance durante o tempo mínimo exigido. Busca-se distribuir k PAs com alcance de transmissão R em uma malha rodoviária com área A e n interseções, a fim de obter a maior cobertura.

Formalmente, seja $V = \{v_1, \ldots, v_v\}$ o conjunto de veículos que passam pela região considerada e $S_i \subseteq V$ um subconjunto dos veículos que entram na interseção i. O objetivo é escolher k conjuntos, a fim de maximizar a cardinalidade de $S_1 \cup S_2 \cup \cdots \cup S_k$, considerando o tempo de permanência de cada veículo na interseção. Considere $T_{n,v}$ a matriz interseção $\times$ veículo, onde $T_{i,j} \geq 0$ representa o tempo total que o veículo j permanece na interseção i. Assim, o PMCLT pode ser formulado como abaixo:

$$\max \sum_{j=1}^{m} v_j, \tag{2.1}$$

sujeito a:

$$\sum_{i=1}^{n} \left(T_{i,j} \times y_i\right) \geq v_j \times \tau \forall j, \tag{2.2}$$

$$\sum_{i=1}^{n} y_i \leq k, \tag{2.3}$$

$$y_i \in \{0,1\} \forall i, \tag{2.4}$$

$$v_j \in \{0,1\} \forall j. \tag{2.5}$$

A Equação 2.1 representa a função objetivo do PMCLT. A restrição da Equação 2.2 força que v_j precisa atingir o tempo mínimo τ necessário nas interseções para ser coberto. A

restrição descrita na Equação 2.3 assegura que no máximo k interseções são selecionadas, ao passo que a restrição na Equação 2.4 indica a existência de um PA na interseção i. Por fim, a restrição da Equação 2.5 indica se o veículo é escolhido.

O PMCLT é NP-difícil [Hochbaum, 1996] e, por isso, é necessário utilizar heurísticas para alcançar soluções aproximadas da ótima. Existe, na literatura, um método guloso proposto por Trullols et al. [2010] que fornece soluções de maneira rápida e baseia-se no tempo total em que os veículos permanecem nas interseções. Esse método será descrito na próxima seção.

Para aplicar essa modelagem na prática, as seguintes premissas precisam ser consideradas:

1. Os PAs são colocados no centro dos cruzamentos.
2. Um veículo recebe os dados transmitidos por um PA se ele estiver no seu raio de transmissão.
3. A informação é totalmente recebida pelo veículo se a soma dos tempos de sua permanência no raio de transmissão de todos os PAs que atravessa for, pelo menos, o tempo mínimo determinado.

2.3 Algoritmo Guloso

Para resolver o PMCLT, Trullols et al. [2010] faz uso de uma heurística que objetiva maximizar o tempo total que cada veículo permanece nas interseções e cada um deles só contribui com no máximo τ de tempo para o objetivo. O modelo que representa essa heurística, é formulado formalmente como:

$$\max \sum_{j=1}^{m} \min \left(\tau, \sum_{i=1}^{n} T_{i,j}\, y_i \right), \tag{2.6}$$

sujeito a:

$$\sum_{i=1}^{n} y_i \leq k, \tag{2.7}$$

$$y_i \in \{0,1\} \forall i. \tag{2.8}$$

A função objetivo na Equação 2.6 representa a heurística que busca maximizar o tempo total de permanência dos veículos nas interseções. A restrição descrita na Equação 2.7 asse-

gura que no máximo k interseções são selecionadas, ao passo que a restrição na Equação 2.8 indica a existência de um PA na interseção i.

O Algoritmo 1 corresponde à essa heurística. Essa abordagem distribui os PAs na região e otimiza a cobertura dos veículos, considerando o tempo.

O algoritmo recebe, como entrada: o número k de interseções para serem posicionadas, o conjunto $\mathbf{S}$ de interseções, a matriz T, que indica o tempo no qual o veículo permaneceu na interseção, e o tempo mínimo τ para a transmissão de dados. O resultado do algoritmo é o conjunto $\mathbf{S}'$ das interseções que foram selecionadas.

Algoritmo 1 Abordagem gulosa para o PMCLT

Require: k, T, τ, $\mathbf{S}$
Ensure: $\mathbf{S}'$
1: $\mathbf{S}' \leftarrow \varnothing$
2: $t_j \leftarrow 0$, $j = \{1, \ldots, m\}$
3: **repeat**
4: $\quad W_i \leftarrow \sum_{j=1}^{m} \min(\tau - t_j, T_{ij})$, $i = \{1, \ldots, n\}$
5: $\quad$ Selecionar $S_i \in \mathbf{S}$ que maximiza W_i
6: $\quad t_j \leftarrow \min(\tau, t_j + T_{ij})$, $j = \{1, \ldots, m\}$
7: $\quad \mathbf{S}' \leftarrow \mathbf{S}' \cup S_i$
8: $\quad \mathbf{S} \leftarrow \mathbf{S} \setminus S_i$
9: $\quad k \leftarrow k - 1$
10: **until** $k = 0$ ou $\mathbf{S} = \varnothing$

Para cada interseção i, $W_i\,(i = 1, \ldots, n)$ denota o tempo de cobertura dos veículos nos PAs (linha 4), que é obtido somando-se o tempo em que cada veículo permanece na interseção. Quando o tempo de um veículo na interseção ultrapassa τ, o excesso é ignorado, uma vez que a transmissão é completada no tempo τ. Por outro lado, se o tempo de permanência do veículo na interseção não for suficiente para a transmissão, ele é salvo no vetor t_j (linha 6), e então na iteração seguinte o tempo necessário para completar a transmissão é calculado ($\tau - t_j$, linha 4).

Dados os valores de W_i, a interseção S_i que oferece o maior tempo de cobertura é selecionada, inserida no subconjunto $\mathbf{S}'$ (linha 7), e em seguida removida do conjunto $\mathbf{S}$ (linha 8). Este procedimento é executado até que k interseções sejam selecionadas ou $\mathbf{S}$ esteja vazio (linha 10).

2.4 Algoritmo Genético

O Algoritmo Genético (AG) [Forrest, 1993; Mitchell, 1998] tradicional possui um conjunto de componentes e um fluxo bem definido de operações, mas que são dependentes do problema e por isso podem ser adequados às necessidades. O algoritmo executa a mesma

sequência de passos repetidas vezes, cada uma chamada de geração. A Figura 2.1 mostra a sequência de operações básicas:

- Inicialmente, forma-se uma população de indivíduos, chamada **população inicial**, que representa soluções possíveis para o problema. Os elementos que formam os indivíduos são chamados genes e o conjunto dos genes são os cromossomos. Cada cromossomo é avaliado segundo uma função, recebe um valor que mede seu desempenho e o objetivo pode ser maximizar ou minimizar esse valor.

- Em seguida, no processo de **seleção**, indivíduos dessa população são selecionados probabilisticamente.

- Os indivíduos selecionados são recombinados entre si com uma probabilidade p_{cruz}. Essa recombinação é chamada de **cruzamento**.

- Os novos indivíduos podem ainda sofrer alterações espontâneas em seu cromossomo, ou seja, **mutação** com uma probabilidade p_{mut}.

- A **avaliação** é realizada novamente no cromossomo.

- Uma operação de **elitismo** poderá ser executada para preservar os melhores indivíduos da geração.

- Essa sequência de operações gera a **nova população** e o processo é realizado novamente.

Além disso, o processo de geração da nova população pode considerar um mecanismo elitista, em que os melhores indivíduos da população atual são preservados na seguinte.

Nesse trabalho, foram realizadas adaptações de várias etapas do AG para o problema aqui descrito, tais como, representação do indivíduo, inicialização da população, função objetivo e operadores genéticos, que serão descritos a seguir.

2.4.1 Representação do Indivíduo/Cromossomo

Um AG faz sua busca num espaço de indivíduos, também chamados cromossomos. Cada cromossomo representa uma solução candidata ao problema, é formado por genes e cada gene é uma instância de um alelo particular. Um cromossomo pode ser representado computacionalmente de diversas maneiras, por exemplo, como uma *string* de *bits* ou um vetor de inteiros.

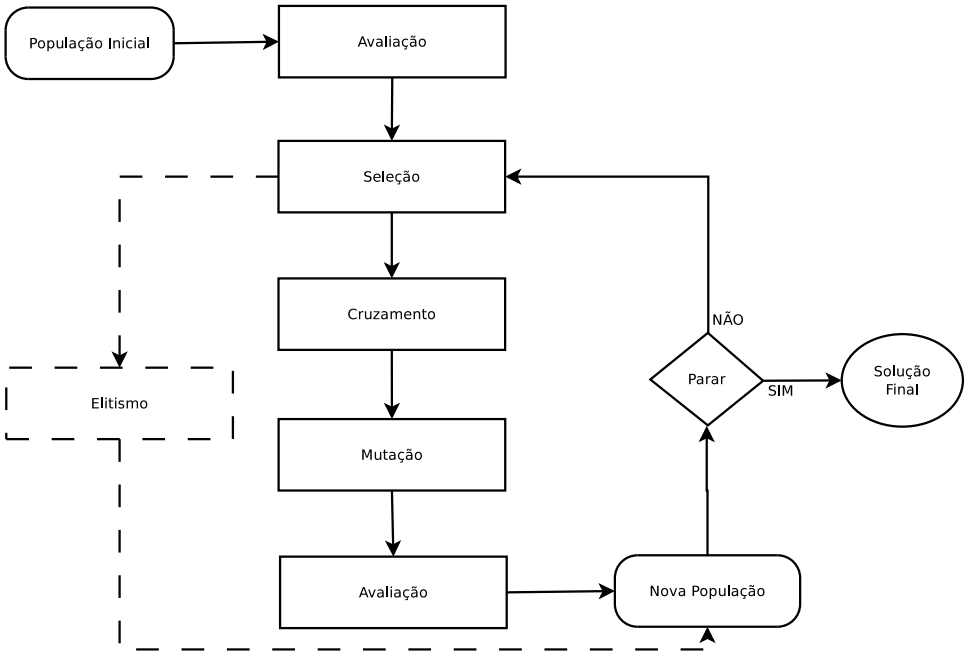


Figura 2.1: Fluxograma de um algoritmo genético.

Considerando o PMCLT, adotamos a representação inteira. Dado o mapa da malha viária de uma região com n interseções e k PAs a serem posicionados, cada indivíduo é representado por

$$\mathbf{I} = \{I_1 \leq I_2 \leq \cdots \leq I_x\},$$

onde $I_i \in \mathbb{N}^+$, $0 \leq I_i < n$ e $x \leq k$. Por exemplo, num cenário com $n = 10$ interseções para posicionar $k = 4$ PAs, um indivíduo válido é $\mathbf{I} = \{0, 4, 8, 9\}$, ou seja, os PAs serão colocados nas interseções de número 0, 4, 8 e 9.

Como cada gene é uma interseção, ele possui as características provenientes do problema, isto é, existe um número de veículos que circulam na interseção por uma determinada quantidade de tempo. Essas características foram utilizadas e incorporadas ao algoritmo com o objetivo de alcançar melhores soluções.

2.4.2 Função objetivo/*Fitness*

A função objetivo, também chamada de *fitness*, representa a pontuação de cada cromossomo, ou seja, quão bem cada cromossomo resolve o problema. Ela pode ser maximizada ou minimizada, dependendo do objetivo.

Para o PMCLT, a *fitness* f de um indivíduo $\mathbf{I}$ é definida como a porcentagem de veícu-

los cobertos na área considerada, isto é,

$$f_{\mathbf{I}} = \frac{|\widehat{V}|}{m},$$

onde $\widehat{V} \subseteq V$ e $\widehat{v}_j \in \widehat{V} \mid \widehat{v}_j = \sum_{i=1}^{n} T_{i,j} \geq \tau$. Como o objetivo é alcançar a maior cobertura de veículos possível, o AG procura maximizar a *fitness*.

2.4.3 Inicialização da População

A primeira operação do AG consiste na geração da população, que forma o espaço de busca inicial. A inicialização de população tradicional do AG é feita de forma aleatória, gerando n cromossomos que podem ser soluções para o problema. No entanto, pode-se utilizar técnicas para melhorar essa inicialização.

Para o PMCLT, foi feito um estudo com diversos tipos de inicialização da população buscando melhorar o desempenho do algoritmo, tanto computacionalmente, tornando o processo evolutivo mais rápido, quanto em termos da cobertura alcançada. Assim, soluções geradas pela abordagem gulosa foram inseridas na população inicial.

O algoritmo guloso (Algoritmo 1) foi modificado para retirar sua característica determinística e, por consequência, gerar soluções diferentes quando executado várias vezes. Isso foi realizado introduzindo aleatoriedade no algoritmo: em cada iteração, não só a melhor interseção (a de menor identificador com maior tempo acumulado) é selecionada, mas uma escolha aleatória entre as 10 melhores é feita. O objetivo dessa técnica é encontrar outras soluções que sejam boas o suficiente para ajudar na evolução do algoritmo genético, mesmo não sendo a aproximada da ótima fornecida pelo algoritmo guloso. Para isso, a linha 5 do Algoritmo 1 foi modificada da seguinte forma:

`Selecione` $S_i \in \mathbf{S}$, `onde` $i = rand(1:10)$ `é o` i-`ésimo que maximiza` W_i.

Cinco variações de inicialização da população foram utilizadas:

1. **A:** Totalmente aleatória;
2. **A + G:** Aleatória com o acréscimo da solução gulosa;
3. **A + G + GuM:** 50% aleatória complementando com a solução gerada pelo guloso e as soluções obtidas pelo algoritmo guloso aleatorizado;
4. **A + GuM:** 50% aleatória complementando com soluções geradas pelo algoritmo guloso aleatorizado;

5. **A + GuM + 0.2g:** 50% aleatória e 50% com soluções geradas pelo guloso aleatorizado, além disso, 20% dos genes de 1/4 dos indivíduos provenientes do guloso são as interseções por onde mais transitam veículos.

A população foi mantida aleatória nas estratégias (2), (3), (4) e (5) para evitar a introdução de vieses da solução gulosa, isto é, evitar que a evolução do algoritmo genético seja influenciada pelas soluções do algoritmo guloso.

2.4.4 Operador de Seleção

A operação de seleção do AG escolhe indivíduos que serão responsáveis por formar a população da próxima geração, sendo desejável que os melhores indivíduos sejam selecionados. Existem várias estratégias para realizar essa escolha e ela é crucial para a convergência do algoritmo [Miller & Goldberg, 1996].

No AG desenvolvido, foi escolhido o esquema de seleção por torneio, em que w indivíduos são sorteados aleatoriamente e apenas o melhor entre eles é selecionado. Para diminuir a pressão de seleção, isto é, impedir que o algoritmo convirja prematuramente optou-se por escolher $w = 2$. Foram selecionados metade dos indivíduos da população total. Esses foram preservados para a próxima geração, ou seja, aplicamos um mecanismo de elitismo que preserva os melhores indivíduos. Os cromossomos selecionados também foram utilizados para construir a outra metade da nova população, através de cruzamentos.

2.4.5 Operadores de Cruzamento

Esse operador recombina subsequências de genes de dois cromossomos para criar dois filhos. Várias estratégias diferentes de cruzamentos foram desenvolvidas e testadas no AG, em algumas utilizou-se informações sobre o problema. Os cruzamentos executados foram:

1. **Cruzamento de um ponto.** Uma posição única é escolhida nos cromossomos de ambos os pais. Todos os genes do cromossomo da mãe localizados antes do ponto escolhido são antepostos aos genes do pai localizados a partir daquele ponto em diante e vice-versa. São gerados, dessa forma, dois filhos. Se a mesma interseção estiver presente em ambas as metades, ela é considerada somente uma vez e o tamanho do indivíduo será menor que k. Por exemplo, considere o cromossomo mãe $P_1 = \{0,4,6,7,12\}$ e o cromossomo pai $P_2 = \{2,5,10,13,15\}$, o ponto de corte escolhido é a posição 4, assim os filhos gerados são: $F_1 = \{0,4,6,13,15\}$ e $F_2 = \{2,5,10,7,12\}$.

2. Cruzamento baseado na *fitness*, chamado de **operador de fusão** [Beasley & Chu, 1996]. Nesse operador, leva-se em consideração a *fitness* dos pais P_1 e P_2: quem

tem maior *fitness* tem maior probabilidade de transferir seu gene para o filho, segundo a fórmula $\Pr_{P_1} = \frac{f_{P_1}}{f_{P_1}+f_{P_2}}$. Para cada posição do cromossomo do novo indivíduo, é sorteada uma probabilidade, que é comparada com $\Pr_P$ de um dos pais, se o valor sorteado for menor, então o gene herdado é proveniente desse cromossomo, caso contrário o gene do outro cromossomo é transferido. Se os genitores compartilharem o mesmo gene, ele é imediatamente transferido para o filho. Por exemplo, considere os pais do item 1. Sejam $f_{P_1} = 50$ e $f_{P_2} = 80$ e as probabilidades sorteadas do filho $\Pr_F = \{0.4, 0.2, 0.6, 0.7, 0.9\}$, como $\Pr_{P_1} = 0.38$, então o cromossomo do filho será $F = \{2, 4, 10, 13, 15\}$.

3. Cruzamento baseado nas características do problema chamado **inserção dos melhores que a média**. Esse operador considera as características do problema para recombinar os genes dos pais. O filho herda primeiro os genes dos pais cujas interseções que representam atravessam mais veículos do que a média de todas as interseções da região. Em seguida, o novo indivíduo é preenchido utilizando o operador de fusão descrito no item 2. Intuitivamente, espera-se que aquelas interseções por ondem passam mais veículos influenciem positivamente na *fitness* do indivíduo. Considere novamente os genitores P_1 e P_2 do item 1, suas *fitness* f_{P_1} e f_{P_2} do item 2 e que nas interseções 4, 6 e 13 passam mais veículos que a média da região. Inicialmente, o filho é $F = \{4, 6, 13\}$. Sorteia-se os valores das probabilidades para as posições faltantes, $\Pr_F = \{0.4, 0.2\}$ e considera-se $P_1 = \{0, 7, 12\}$ e $P_2 = \{2, 5, 10, 15\}$. A operação final acarreta no filho $F = \{4, 6, 13, 0, 5\}$.

2.4.6 Operadores de Mutação

O operador de mutação é responsável por inserir diversidade genética na população, através da mudança de alguns genes do cromossomo, com o objetivo de diminuir a chance das soluções caírem em ótimos locais. Foram utilizados no PMCLT dois operadores de mutação.

1. **Mutação de um ponto.** O valor de um gene é substituído. Tanto a posição do gene quanto seu valor são escolhidos aleatoriamente. Por exemplo, considere o indivíduo $I = \{2, 4, 5, 6, 7\}$, a posição e o valor do gene são sorteados, 1 e 10, respectivamente. O novo indivíduo é $I = \{10, 4, 5, 6, 7\}$.

2. Mutação baseada nas características do problema, chamado **remoção do pior gene**. Esse operador é adaptado ao problema e remove a interseção do cromossomo por onde passam menos veículos e insere uma cujo número de veículos seja maior. Espera-se que interseções por ondem passam poucos veículos influenciem negativamente a *fitness* do indivíduo.

2.4.7 Operadores de Busca Local

A busca local explora regiões a procura de ótimos locais que o AG pode não alcançar. Com o objetivo de acelerar a convergência do algoritmo genético e verificar uma possível evolução das soluções para ótimos locais, vários operadores de busca local foram desenvolvidos. A busca local foi aplicada após a mutação em todos os indivíduos da população.

Para realizá-la, foi considerado o número de veículos que passam em cada interseção. Apesar do tempo em que esses veículos permanecem na interseção ser importante, não se optou por essa característica pois ela já é considerada na inicialização da população através do algoritmo guloso aleatorizado.

1. **Inserção de interseções com quantidade de veículos acima da média.** Esse operador, de forma aleatória, remove 20% dos genes do indivíduo e introduz aqueles que possuem valor maior que a média de todos os genes possíveis. Isto é, as interseções por onde passam mais veículos do que a média de todas as interseções são inseridas no cromossomo.

2. **Substituição de interseções mais densas por aquelas que flutuam ao redor da média global.** Essa operação remove 20% dos maiores genes do indivíduo e introduz aqueles que estão ao redor da média global, no intervalo entre $(\mu - \sigma, \mu + \sigma)$, onde σ é o desvio padrão da quantidade de veículos em cada interseção e μ é a média de veículos por interseção. Dessa forma, as interseções mais frequentadas no cromossomo são substituídas por aquelas cuja quantidade de veículos está próxima da média da região.

3. **Substituição de interseções densas por outras densas.** O operador substitui genes de valores altos por outros de valores altos também. Um gene é considerado com valor alto se, considerando o vetor ordenado de quantidade de veículos por interseção, sua posição está na primeira metade desse vetor. Isso significa que as interseções por onde passam muitos veículos são trocadas por outras que também são densas.

4. **Substituição de interseções pouco frequentadas por outras mais frequentadas.** A operação remove, de forma aleatória, 20% dos genes que possuem os menores valores no cromossomo e introduz aqueles considerados com valores altos. Logo, interseções por onde passam poucos veículos são substituídas por aquelas muito transitadas.

O Algoritmo 2 apresenta o AG implementado. A linha 1 representa a inicialização da população, que pode ser qualquer uma das descritas na Subseção 2.4.3. Depois que a população é inicializada, indivíduos são avaliados e selecionados por torneio (subseção 2.4.4) e

então submetidos a alguma estratégia de cruzamento (subseção 2.4.5) e operação de mutação (subseção 2.4.6). Um procedimento elitista mantém os melhores indivíduos na população seguinte, que é completada com os indivíduos produzidos pelos cruzamentos e operações de mutação. Após essas operações, uma das estratégias de busca local, apresentadas na Subseção 2.4.7, também pode ser executada. Este processo é realizado até que um número máximo de gerações seja atingido.

Esse algoritmo foi executado considerando várias combinações de operadores, que será discutida na seção de resultados 4.1.

Algoritmo 2 Algoritmo genético para o PMCLT

Require: k, T, τ, $\mathbf{S}$
Ensure: $\mathbf{S}'$
1: $\mathbf{P}_{1,p} \leftarrow$ inicialização da população
2: $melhor \leftarrow \max(0, f_{\mathbf{I}_i}), i = \{1, \ldots, p\}$
3: **repeat**
4: Avalia os indivíduos de acordo com a *fitness*
5: Realiza a seleção por torneio
6: Executa cruzamento probabilidade p_{cruz}
7: Executa mutação com probabilidade p_{mut}
8: [Executa busca local]
9: Inserção elitista na nova população *P*'
10: Insere novos indivíduos em *P*'
11: $melhor \leftarrow \max(melhor, f_{\mathbf{I}_i}), i = \{1, \ldots, p\}$
12: **until** atingir número máximo de gerações
13: $\mathbf{S}' \leftarrow \mathbf{I}_i$

Capítulo 3

Predição de Trajetória de Veículo Baseada em Histórico do Motorista

Nesse capítulo, serão discutidos alguns trabalhos relacionados à predição de trajetória de veículos na Seção 3.1. Será apresentado também o que compõe o histórico do usuário na Seção 3.2. A modelagem proposta para o problema será descrita na Seção 3.3. Por fim, a Seção 3.4 apresentará uma breve descrição dos algoritmos de aprendizado de máquina aplicados nas bases de dados construídas de acordo com a modelagem. A contribuição dessa parte do trabalho é a execução e avaliação dos algoritmos apresentados em uma base de dados para RVAH, que servirá de barema para proposição de novos algoritmos.

3.1 Trabalhos Relacionados

Com a acelerada evolução dos dispositivos com capacidade de sensoriamento e comunicação, o problema da predição de trajetória de veículos, apesar de antigo [Krozel & Andrisani, 1993], tem sido bastante estudado nos últimos anos. A seguir são destacados os principais trabalhos relacionados.

Caveney [2009] utiliza técnicas de fusão de dados que combinam informações de mapas digitais com modelos dinâmicos e de informações compartilhadas entre os veículos da Vanet para obter uma predição mais acurada. As predições realizadas são de curto prazo e dependem do ambiente em que se dirige, da qualidade dos sensores e da precisão do mapa.

Um outro método de rastreamento e predição de movimento a longo prazo de veículos foi proposto por Hermes et al. [2010]. O método de rastreamento baseia-se em imagens capturadas de duas câmeras acopladas aos veículos. A técnica utilizada na fase de predição depende do tamanho do histórico de movimentação do alvo, podendo ser baseado em fluxo, isto é, na velocidade com que as imagens são capturadas; baseado na cinemática do

veículo, velocidade e aceleração; e baseado em filtro de partículas, quando vários padrões de movimento foram observados. Essa última técnica foi superior às outras duas abordagens.

Lytrivis et al. [2011] mostram que a cooperação (troca de mensagens utilizando um canal de comunicação sem fio) entre os veículos aumenta a acurácia dos resultados do algoritmo de predição de trajetória. No sistema proposto, são consideradas posição, velocidade, aceleração, direção e variação angular de todos os veículos conectados para calcular suas trajetórias futuras. Além disso, utilizam também a geometria da estrada obtidas de um mapa digital. Os resultados observados são testados em um sistema de frenagem automática e avaliados como satisfatórios. No entanto, os autores colocam como desafio situações ocorridas em cruzamentos, quando é necessária a sincronização entre os vários veículos vindos de diversas direções.

O problema da predição nos cruzamentos é considerado por Lefevre et al. [2011]. Os autores propõem um arcabouço probabilístico que combina informações de mapa digital e do estado atual do veículo para estimar qual manobra o motorista pretende realizar quando se aproxima de cruzamentos. São utilizadas distribuições de probabilidade para inicializar uma rede Bayesiana baseada em contexto que modela o processo de decisão do veículo em um cruzamento. A sinalização do motorista também é considerada. O modelo proposto pode gerar incertezas na predição, que também são tratadas pelos autores.

O trabalho de Shan et al. [2011] diferencia-se dos anteriores, pois foca na previsão de percurso a longo prazo em áreas grandes e com terrenos acidentados. O modelo de veículos apresentado não se limita às técnicas clássicas, que utilizam equações de dinâmica e cinemática, mas incorpora propriedades de contexto e geometria conhecidas do ambiente e dados coletados por pontos observadores distribuídos na região. O método inclui diferentes modelos de movimentação para o veículo que podem ser intercalados: mover-se na estrada; parar na estrada; e atravessar cruzamentos. Os resultados mostraram ser satisfatórios e o estudo abre caminho para a pesquisa em rastreamento cooperativo, mesmo quando os veículos fornecem informações sobre seu posicionamento com atraso.

Na área de mineração de dados, existe um ramo que trata de problemas relacionados à trajetória, buscando identificar padrões. Giannotti et al. [2007], por exemplo, fornecem diferentes abordagens disponíveis para minerar trajetórias e as avaliam empiricamente em bases de dados reais e *benchmarks* sintéticos. Em Morzy [2007], técnicas de mineração são utilizadas nos dados de localização de objetos móveis para descobrir trajetórias frequentes e regras de movimentação, que são casadas com a localização atual do objeto para construir um modelo probabilístico de predição. Eisner et al. [2011] apresentam um novo algoritmo para predição de trajetória que faz uso do padrão da malha rodoviária e do caminho que o usuário já percorreu e obtém ótimos resultados com pouco custo computacional.

3.2 Histórico do Usuário

Considera-se que os veículos são rastreados através de um GPS, por exemplo, e sua localização é gravada em intervalos de tempo (iguais ou variáveis). Dessa forma, os dados que formam o arquivo com o histórico do motorista são: longitude, latitude e instante de tempo em que a posição foi coletada, como mostra a Tabela 3.1.

Tabela 3.1: Arquivo do histórico do motorista

Id	Instante de Tempo	Latitude	Longitude
1	2001-01-29 13:01:08	60.477410	15.440701
2	2001-01-29 13:01:11	60.476698	15.438019
3	2001-01-29 13:01:12	60.475159	15.441260
4	2001-01-29 13:01:14	60.476698	15.438019
5	2001-01-30 09:57:30	60.473678	15.461589
6	2001-01-30 09:57:31	60.472793	15.462871
7	2001-01-30 09:57:33	60.472793	15.462871
8	2001-01-30 09:57:35	60.472247	15.463700
9	2001-01-30 09:57:36	60.471881	15.464233
...			

Dois registros consecutivos da posição do veículo formam um trecho t_i. Uma trajetória, viagem ou rota completa $\mathbf{T_j}$ consiste em um conjunto de n trechos, cujas primeira e última posições representam o momento e o lugar em que o veículo foi ligado e desligado, respectivamente. O histórico de um usuário $\mathscr{H}$, por sua vez, representa um conjunto de m trajetórias que o motorista realizou durante o período de observação e que é atualizado a cada nova observação.

A Equação 3.1 define um trecho t_i:

$$t_i = \{(\text{lat}_i, \text{lon}_i), (\text{lat}_{i+1}, \text{lon}_{i+1})\}, \text{ tal que } 1 \leq i \leq n. \tag{3.1}$$

A igualdade entre trechos é definida pela Equação 3.2:

$$t_i = t_j, \text{ se} \begin{cases} \text{lat}_i = \text{lat}_j, \\ \text{lon}_i = \text{lon}_j, \\ \text{lat}_{i+1} = \text{lat}_{j+1}, \text{ e} \\ \text{lon}_{i+1} = \text{lon}_{j+1}. \end{cases} \tag{3.2}$$

Uma trajetória T_j é definida segundo a Equação 3.3:

$$\mathbf{T_j} = \{t_1, t_2, \ldots, t_n\}, \text{ tal que } 1 \leq j \leq m. \tag{3.3}$$

A Equação 3.4 define o histórico do usuário:

$$\mathscr{H} = \{\mathbf{T_1}, \mathbf{T_2}, \ldots, \mathbf{T_m}\}. \tag{3.4}$$

No exemplo de histórico da Tabela 3.1, os registros 1 e 2 formam o trecho t_1, os 2 e 3 formam t_2 e os registros 3 e 4 formam o trecho t_3. Considerando que o carro foi ligado no registro 1 e desligado no 4, então $\mathbf{T_1} = \{t_1, t_2, t_3\}$ é uma trajetória completa do motorista. Seguindo o mesmo raciocínio e considerando que o carro foi ligado no registro 5 e desligado no 9, então $\mathbf{T_2} = \{t_4, t_5, t_6, t_7\}$. Nota-se que, nessa modelagem, cada trajetória $\mathbf{T_j}$ pode ser vista como um grafo, cujos trechos são as arestas. Assim, o histórico do usuário passa a ser um conjunto de grafos.

A partir das formalizações apresentadas, o arquivo original do histórico, com informações de GPS, pode ser transposto em um arquivo composto somente por identificadores de trechos. Isto é, cada linha do novo arquivo é uma sequência de identificadores de trechos e representa uma trajetória completa. Ao transformar o exemplo dados na Tabela 3.1 nesse formato, o registro 1 uni-se ao 2 e transforma-se no trecho t_1, os registros 2 e 3 formam o trecho t_2, os registros 3 e 4 formam o trecho t_3. Esses três trechos formam, então a trajetória T_1 do usuário. O mesmo processo é seguido para formar a próxima trajetória, os registros 5, 6, 7, 8 e 9 são convertidos na trajetória $T_2 = \{t_4, t_5, t_6, t_7\}$. O arquivo gerado a partir dos registros da Tabela 3.1 pode ser visto na Tabela 3.2. Desse arquivo, serão derivadas as bases de dados que servirão de entrada para os algoritmos de aprendizado.

Tabela 3.2: Arquivo gerado a partir do histórico do usuário da Tabela 3.1

t_1	t_2	t_3	
t_4	t_5	t_6	t_7
	...		

3.3 Modelagem do Problema

O problema da predição de trajetória de veículos considerado nesse trabalho, consiste em dado o histórico do usuário $\mathscr{H}$ e os trechos, $\{t_1, t_2, \ldots, t_i\}$, que esse motorista percorreu até o momento atual, qual é o próximo trecho t_{i+1} que ele irá percorrer? Essa previsão deve ser

feita dinamicamente em tempo real, ou seja, a cada novo trecho realizado, busca-se prever o próximo que o motorista irá percorrer.

Nesse trabalho é feita uma análise exploratória do problema para resolvê-lo, utilizando algoritmos de classificação que são muito difundidos em aprendizado de máquina [Kotsiantis, 2007]. Esses algoritmos exigem uma entrada específica, em que cada instância é formada por um ou mais atributos e pertence a uma determinada classe. Para isso, é necessário adaptar o arquivo do histórico do usuário descrito na Seção 3.2, em que cada linha representa uma trajetória completa, como mostrado na Tabela 3.2.

A base de dados foi construída utilizando janelas deslizantes e é mostrada no algoritmo 3. $\mathscr{B}$ representa a base de dados que será gerada; w representa o tamanho da janela, que corresponde à quantidade mínima de trechos que o usuário precisa ter para dar início à previsão; $\mathbf{A}$ é o conjunto dos atributos de uma instância; $\mathbf{C}$ contém a classe correspondente ao conjunto $\mathbf{A}$ de atributos.

O Algoritmo 3 executa os seguintes passos:

- Na linhas 2 a 8, todas as trajetórias $\mathbf{T_j}$ do histórico do usuário $\mathscr{H}$ são consideradas. Nessa iteração, as trajetórias serão particionadas consecutivamente em um tamanho w.
- Nas linhas 3 a 7, o algoritmo percorre os trechos de t_1 a t_{n-w} da trajetória $\mathbf{T_j}$. É nessa iteração que ocorre o deslizamento da janela.
- Nas linhas 4 e 5, os conjuntos dos atributos $\mathbf{A}$ e a classificação $\mathbf{C}$ desses atributos são preenchidos respectivamente por $\{t_i, \ldots, t_{i+w-1}\}$ e t_{i+w}.
- Na linha 6, a instância definida nas linhas 4 e 5 é inserida na base.

Algoritmo 3 Transposição do histórico $\mathscr{H}$ do usuário para a base de dados $\mathscr{B}$ com tamanho de janela w

Require: w, $\mathscr{H}$
Ensure: $\mathscr{B}$
1: $\mathscr{B} \leftarrow \varnothing$
2: **for all** $\mathbf{T_j} \subset \mathscr{H}$ **do**
3: **for all** $t_i \in \mathbf{T_j} \setminus \{t_{n-w+1}, \ldots, t_{n-1}, t_n\}$ **do**
4: $\mathbf{A} \leftarrow \{t_i, t_{i+1}, \ldots, t_{i+w-1}\}$
5: $\mathbf{C} \leftarrow \{t_{i+w}\}$
6: $\mathscr{B} \leftarrow \mathscr{B} \cup \{\mathbf{A}, \mathbf{C}\}$
7: **end for**
8: **end for**

Dessa forma, cada linha do arquivo da base de dados $\mathscr{B}$, ou seja, cada instância, conterá sempre $w+1$ trechos, em que o algoritmo considerará os w primeiros trechos como os

atributos da instância e o $(w+1)$-ésimo trecho como a classe, que será prevista. Para os casos particulares em que $w \geq |\mathbf{T_j}|$, os primeiros $w - |\mathbf{T_j}|$ atributos são preenchidos com 0, que não é utilizado para identificar nenhum trecho existente. A base de dados proveniente da Tabela 3.2 com tamanho de janela $w = 1$ está apresentada na Tabela 3.3. Cada linha desse arquivo é uma instância, a primeira coluna corresponde aos atributos e a segunda à classe.

Tabela 3.3: Base de dados referente ao histórico da Tabela 3.2 com $w = 1$

t_1	t_2
t_2	t_3
t_4	t_5
t_5	t_6
t_6	t_7
...	

3.4 Algoritmos de Classificação

O tipo de aprendizado aplicado nesse trabalho é chamado supervisionado [Caruana & Niculescu-Mizil, 2006], que consiste em associar dois conjuntos: os dados observados X e uma variável externa y, que se quer prever, chamada de classe ou rótulo. Para o problema de previsão de trajetória do veículo os dados de localização observados do usuário, ou seja, os trechos, formam o conjunto das possibilidades para a variável y.

Os algoritmos de aprendizado de máquina buscam extrair padrões a partir de observações, isto é, dos dados de treinamento, para aplicar corretamente esses padrões em exemplos não vistos, chamados dados de teste. Os algoritmos escolhidos são bastante utilizados na área de aprendizado de máquina e serão descritos nessa seção.

3.4.1 Classificador *k-Nearest Neighbors*

k-Nearest Neighbors (KNN) [Cover & Hart, 1967] é um classificador onde o aprendizado é baseado na analogia, ele não tenta construir um modelo geral interno, mas simplesmente armazena as instâncias dos dados de treinamento. Para determinar a classe de um elemento que não pertença ao conjunto de treinamento, o classificador KNN procura k elementos do conjunto de treinamento que estejam mais próximos deste elemento desconhecido. Estes k elementos são chamados de k-vizinhos mais próximos. A classe mais frequente dentre esses k-vizinhos mais próximos é atribuída ao ponto de consulta. O valor ótimo de k é dependente

dos dados. No geral, um valor alto diminui os efeitos de ruídos, mas torna a fronteira da classificação menos notável.

Para utilizar o KNN é necessário definir uma função de distância entre dois exemplos em um espaço de atributos $\mathbb{R}^n$. Um exemplo de distância geralmente utilizada é a Distância Euclidiana. Um problema com este tipo de algoritmo é que algumas funções de distância podem ter problemas com entradas de alta dimensionalidade.

3.4.2 Naive Bayes

O *Naive Bayes* [Friedman et al., 1997], e suas variações, são algoritmos de classificação probabilísticos. A predição de classes é baseada na aplicação da Regra de Bayes. A probabilidade de ocorrência de uma classe y_i, dado o valor dos atributos da entrada X_i, isto é, $\Pr(y_i|X_i)$ é calculada a partir dos dados de treinamento. Um ponto fraco do Naive Bayes é que ele assume uma forte independência dos atributos, o que pode não ser verdade em muitos casos. Nesse trabalho, foram utilizados os algoritmos que implementam as distribuições Gaussiana e Multinomial [Murphy, 2012] para classificação.

3.4.3 Support Vector Machines (SVM)

Os *Support Vector Machines* (SVMs) são algoritmos que representam os dados em um espaço $\mathbb{R}^n$ e tentam encontrar o hiperplano que melhor separa os dados nesse espaço [Cristianini & Shawe-Taylor, 2000]. O modelo gerado é o que maximiza a margem: a distância entre o hiperplano separador e os "vetores de suporte", que é o espaço que o hiperplano pode ser movido sem causar erros nos dados de treino. A ideia pode ser estendida para dados não linearmente separáveis através de hiperplanos com margens flexíveis, hiperplanos que permitem erros nos dados de treino, ou por intermédio do uso de funções *kernel* que mapeiam a entrada para espaços de maior dimensionalidade em que os dados são linearmente separáveis, isto é, que podem ser separados por uma reta em um hiperplano. O *kernel* escolhido nesse trabalho foi o RBF [Daqi & Tao, 2007].

3.4.4 Árvores de Decisão

Algoritmos baseados em Árvores de Decisão utilizam árvores para representar o relacionamento entre um exemplo X_i e uma classe y_i [Quinlan, 1996]. Cada nó interno da árvore representa um atributo e os nós folhas representam as classes. Uma árvore de decisão é construída selecionando os atributos mais discriminativos dos dados de treino utilizando abordagens relacionadas ao conceito de ganho de informação e entropia da Teoria da Informação [Gray, 1990]. Para classificar os dados, o valor dos atributos da entrada é testado contra os nós da

árvore até alcançar um nó folha. A classe representada por esse nó é utilizada para realizar a predição. Árvores de decisão têm a vantagem de ter resultados interpretáveis por humanos, e por lidar bem com atributos irrelevantes. Porém, árvores de decisão tendem a sofrer de *overfitting*, treinamento excessivo. Esse problema pode ser amenizado realizando poda de nós que não são muito significativos, melhorando assim a generalização da árvore.

Capítulo 4

Resultados

Nesse capítulo, serão discutidos os resultados alcançados a partir dos experimentos realizados para o problema de montagem de infraestrutura e predição de trajetória em RVs, que foram apresentados nos capítulos anteriores.

4.1 Montagem da Infraestrutura

Nessa seção, será apresentada a avaliação realizada sobre o algoritmo genético proposto na Seção 2.4 para organizar a infraestrutura de uma RVAH. A solução gerada através do algoritmo guloso de Trullols et al. [2010], bem como as soluções geradas pela modificação inserida no guloso descrita na Subseção 2.4.3 são utilizadas como base para avaliar o desempenho do AG.

Na Subseção 4.1.1, serão caracterizados os cenários em que foram aplicados os experimentos. Na Subseção 4.1.2, será discutida a escolha dos parâmetros do AG. Nas Subseções 4.1.3 e 4.1.4, é apresentado o comportamento do AG para diferentes combinações de inicialização de população e operadores genéticos, respectivamente. Nas Subseções 4.1.5 e 4.1.6, será estudado o comportamento do AG quando são variados o tempo de recebimento da informação e o número de PAs disponíveis.

4.1.1 Cenários

O AG proposto foi avaliado usando quatro conjuntos de dados extraídos de um registro de mobilidade veicular realística, coletados durante 1 h e 30 min, a partir de uma rede rodoviária urbana da Suíça [Nagel, 2012]. Os conjuntos de dados correspondem a quatro regiões diferentes, localizadas dentro de uma área de 100 km^2, centradas nas cidades de Zurich e Winterthur, que caracterizam o tráfego pesado, e nas áreas rurais de Baden e Baar, que ca-

racterizam o tráfego leve. Cada região tem sua própria topologia e densidade, tal como descrito na Tabela 4.1.

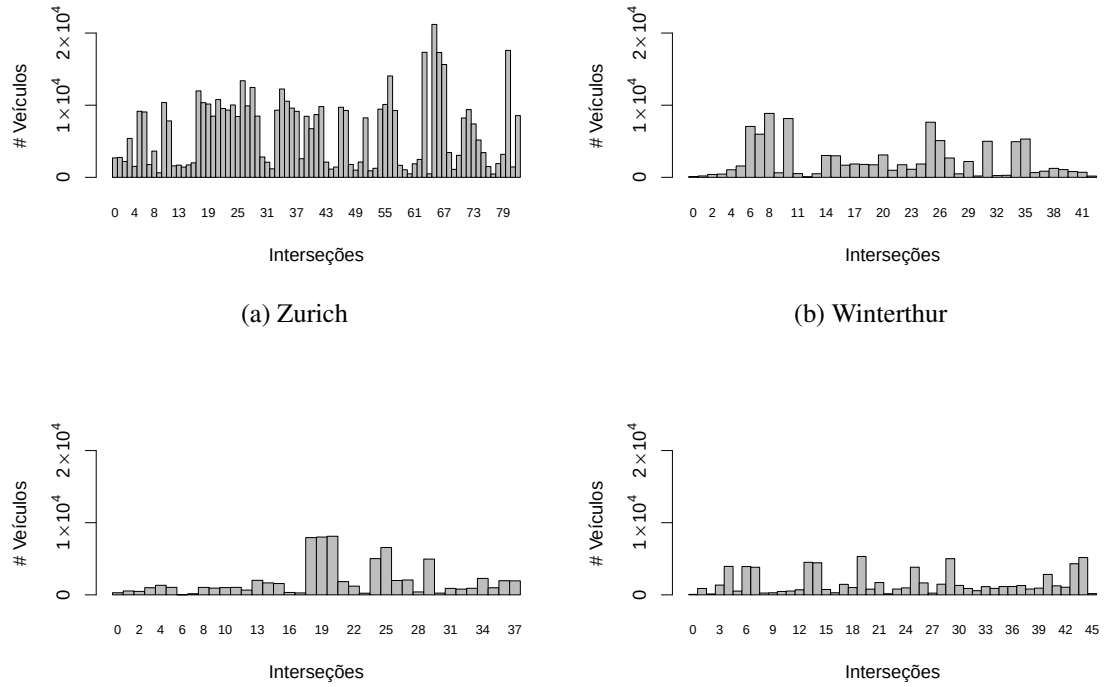
Tabela 4.1: Características do cenário

	Zurich	Winterthur	Baden	Baar
Interseções	83	43	38	46
Veículos	70.537	13.578	11.632	9.876

Cada base dessas foi processada por Trullols et al. [2010] com o objetivo de gerar a matriz T, com dimensões $n \times m$, em que n é o número de interseções e m é o número de veículos que transitam na região por pelo menos 60 s. Para isso, as interseções são identificadas por um número no intervalo $\{i \in \mathbb{Z}^+ | 1 \leq i \leq n\}$ e os veículos, no intervalo $\{j \in \mathbb{Z}^+ | 1 \leq j \leq m\}$. Cada elemento $T_{i,j}$ em T é a diferença entre os tempos inicial e final correspondentes a entrada e a saída do veículo i na fronteira da interseção j, que é limitada pelo raio de transmissão R do PA. No trabalho aqui descrito, foram utilizadas essas bases processadas, que impede de recuperar as posições reais da localização dos PAs no mapa das regiões, mas, no entanto, não compromete a avalização realizada.

Com o objetivo de caracterizar as bases de dados utilizadas, foram plotadas na Figura 4.1 a quantidade de veículos que atravessa cada interseção, no eixo x encontram-se os identificadores das interseções e no eixo y a quantidade de veículos. A Figura 4.1a de Zurich confirma a densidade da região, onde muitas interseções são frequentadas por vários veículos. Nos cenários de Winterthur, Baar e Baden, nos gráficos 4.1b, 4.1c, 4.1d respectivamente, em que circulam menos veículos na região, pode-se notar que uma pequena quantidade de interseções são muito frequentadas, enquanto que o comportamento global é ter uma concentração menor de veículos circulando. Essa informação é importante pois se espera que a cobertura da região seja influenciada pela quantidade de veículos que passa pela interseção, e consequentemente pela escolha das interseções onde serão colocados os PAs.

Além do número de veículos que transitam em cada interseção, uma informação importante a considerar nos cenários é a média de tempo que os veículos gastam para atravessar cada interseção. Isso é importante pois, no problema tratado, o tempo em que cada veículo permanece na interseção é determinante para definir se o veículo foi coberto. Essa informação é extraída para cada interseção i a partir da fórmula $\frac{\sum_{j=1}^{m} T_{ij}}{m_i}$, em que m_i é o número de veículos que passam pela interseção i. A Figura 4.2 apresenta essa informação para cada cenário, no eixo x estão os identificadores das interseções e no eixo y a média de tempo calculada. No cenário de Zurich, no gráfico 4.2a, verifica-se que, na grande maioria das interseções, um veículo demora em torno de 10 a 20 segundos para atravessar a interseção. Já no cenário de Winterthur, um veículo custa de 5 a 15 segundos para cruzar a maioria das interseções, o mesmo acontece no cenário de Baar, no gráfico 4.2d. No cenário de Baden,



(a) Zurich

(b) Winterthur

(c) Baden

(d) Baar

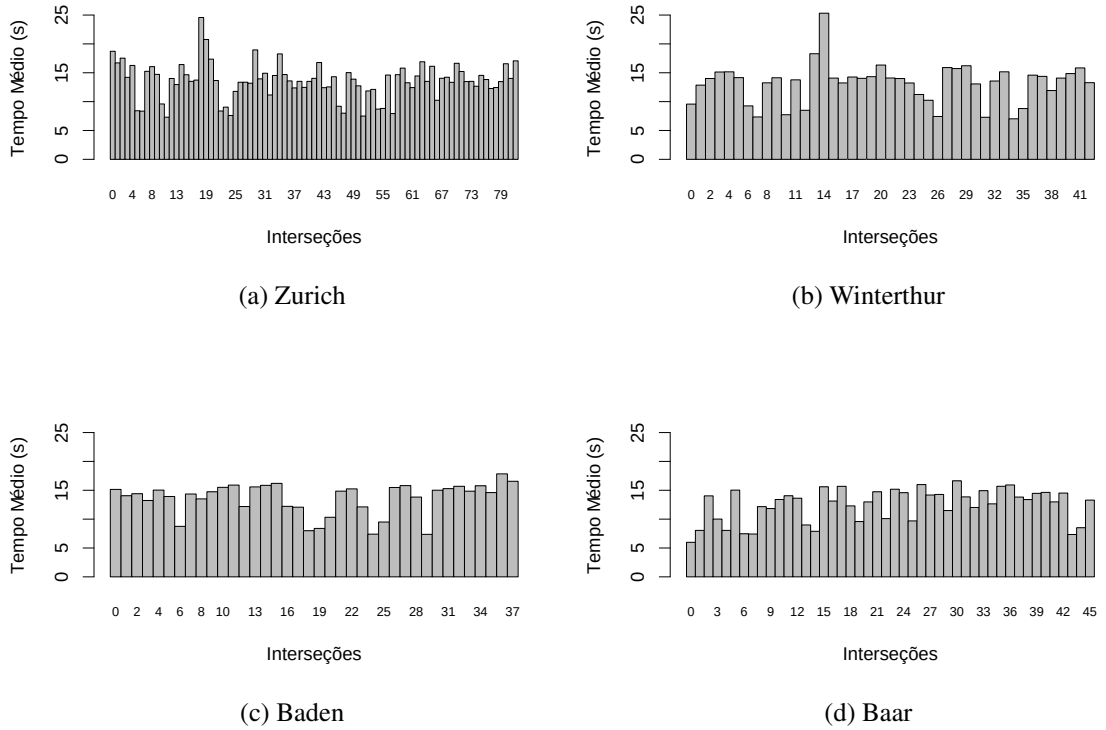
Figura 4.1: Número de veículos que passam em cada interseção

o tempo médio que um veículo gasta para atravessar as interseções está no intervalo de 10 a 15 segundos.

4.1.2 Ajuste dos parâmetros

O algoritmo genético possui vários parâmetros que precisam ser avaliados e ajustados para aumentar a qualidade das soluções atingidas. Esses parâmetros são número de gerações, tamanho da população, tamanho do torneio na seleção e probabilidades de cruzamento e mutação. Além disso, o problema também possui parâmetros que precisam ser ajustados: o número k de PAs e o tempo mínimo τ necessário para transmissão da informação. Assim, o espaço paramétrico do problema são todas as combinações de valores para essas variáveis, que resulta em um número grande de possibilidades, por isso, foi necessário fixar alguns valores.

Para calibrar as variáveis do algoritmo genético, fixamos os parâmetros inerentes ao problema. O valor de k foi determinado em 30% do número de interseções no cenário considerado, para tornar justa a quantidade disponível de PAs mesmo em cenários diferentes. O limite do tempo τ de transmissão da informação, foi definido em 30 s em todos os cenários.



(a) Zurich

(b) Winterthur

(c) Baden

(d) Baar

Figura 4.2: Gráficos de barras da média de tempo que os veículos permanecem nas interseções

O número de gerações foi fixado em 200, determinando o critério de parada do AG e o tamanho do torneio em 2, por ser um valor consolidado na literatura. Para os demais parâmetros, foram escolhidos alguns valores de testes, que estão apresentados na Tabela 4.2. Os valores para o tamanho da população no cenário de Zurich foi diferenciado, por possuir um espaço de busca consideravelmente maior que os outros cenários e por isso ele está separado dos outros cenários na tabela. Para Zurich foram os usados os valores 200, 300 e 400, já para os demais foram utilizados os valores 50, 100 e 200. Para probabilidade de cruzamento e mutação em todos os cenários foram testados os mesmos valores, para cruzamento 0.80, 0.90, 0.95 e para a mutação 0.001, 0.010, 0.100. Os valores vencedores estão destacados com uma formatação de fonte diferente para cada cenário.

Os valores em `máquina de escrever` correspondem aos escolhidos para o cenário de `Zurich`, os valores em **negrito** correspondem a **Wintherthur**, os sublinhados foram adotados para o cenário de Baden e os em *itálicos* para *Baar*. Na tabela, verifica-se que para o cenário de Zurich a faixa de valores utilizados para o tamanho da população foi diferente. Isso é justificado pela disparidade das características de Zurich, número de interseções e quantidade de veículos, em comparação com os outros três cenários.

Tabela 4.2: Valores de parâmetros testados durante calibração do AG

Cenário (# PAs)	Tam. da População	Prob. de Cruzamento	Prob. de Mutação
	200	0.80	0.001
Zurich (25)	300	0.90	0.010
	400	0.95	0.100
Winterthur (13)	50	*0.80*	0.001
Baden (11)	100	0.90	**0.010**
Baar (14)	***200***	**0.95**	*0.100*

Para os valores definidos, são possíveis 27 combinações de experimentos para cada cenário, ou seja, 4×27 experimentos e devido à natureza não determinística do AG é necessário realizar replicações para cada experimento. Para reduzir a carga de experimentos, utilizou-se uma metodologia de experimentação na qual um parâmetro é variado enquanto os demais são fixados, a cada etapa aquele valor que obteve o melhor resultado é escolhido para o parâmetro que foi variado. Inicialmente, o parâmetro variado foi tamanho da população, os outros foram fixados no nível mais baixo, assim o valor de tamanho da população que obteve as melhores soluções foi estabelecido. Em seguida, utilizando o valor de tamanho da população determinado, o processo foi repetido para decidir a probabilidade de cruzamento e, posteriormente, com os valores de tamanho da população e probabilidade de cruzamento ajustados, a probabilidade de mutação.

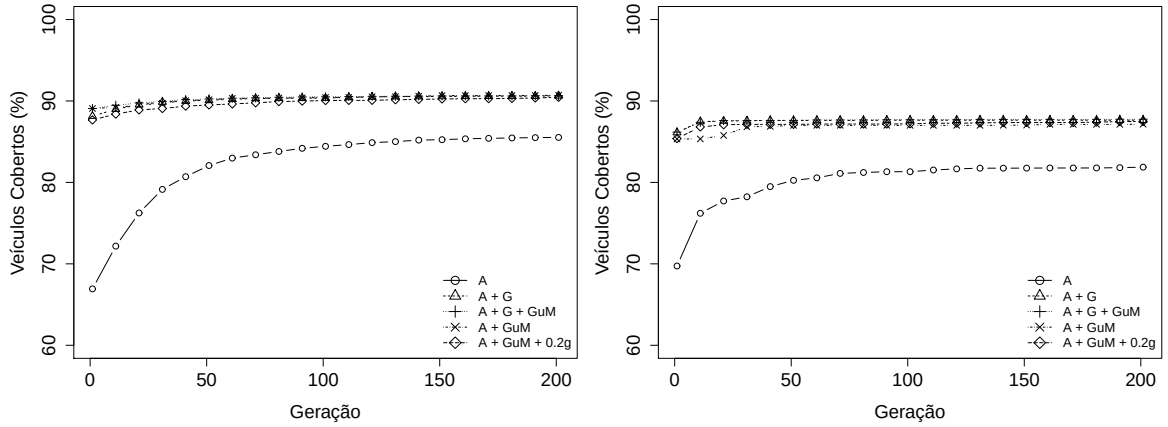
Nessa fase de ajustes, para cada um dos experimentos, foram realizadas cinco replicações, elas foram aumentadas para dez na fase final de análise dos resultados. A qualidade da solução foi avaliada segundo o teste-t [Jain, 1991] das médias da melhor aptidão nas replicações com 99% de confiança. Em situações de impasse do teste-t, isto é, em que não foi possível concluir que um valor fosse estatisticamente maior que os outros, foram analisados também o tempo de evolução e a convergência do algoritmo.

As próximas subseções relatam os experimentos realizados para avaliar o AG nos quatro cenários mencionados. Eles foram divididos em quatro fases: (i) na Subseção 4.1.3, verifica-se o comportamento do algoritmo com diferentes tipos de inicialização de população; (ii) na Subseção 4.1.4, analisa-se o comportamento do AG com os combinações de diversos operados genéticos; (iii) na Subseção 4.1.5, apresenta-se o resultado quando há variação no tempo τ de transmissão da mensagem; e (iv) na Seção 4.1.6, mostra-se o efeito na cobertura de veículos quando há uma variação no número PAs posicionados.

4.1.3 Inicialização da população

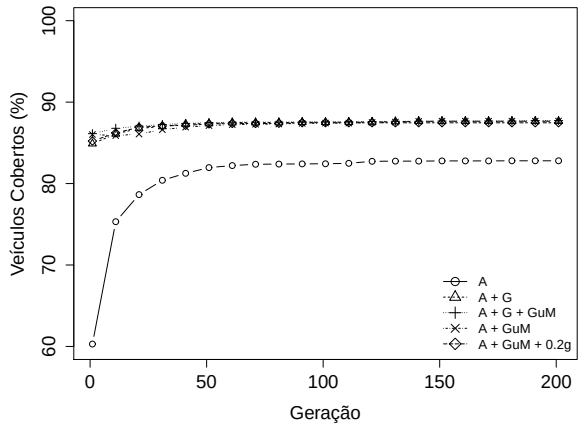
Uma maneira de agilizar o processo de convergência de um AG é construir uma inicialização da população mais otimizada, para que o tempo exploração do algoritmo seja reduzido. Baseado nisso, foram testados quatro procedimentos de inicialização da população que se aproveitam das boas soluções geradas pelo algoritmo guloso. Eles foram comparados com a inicialização totalmente aleatória, presente na abordagem inicial de um algoritmo genético. Esses procedimentos de inicialização foram descritos no Capítulo 2, na Subseção 2.4.3. Para evitar a forte influência da estratégia gulosa nas soluções do genético, metade da população foi mantida aleatória.

Os gráficos da Figura 4.3 mostram o comportamento do AG com as diferentes estratégias de inicialização, em cada cenário.

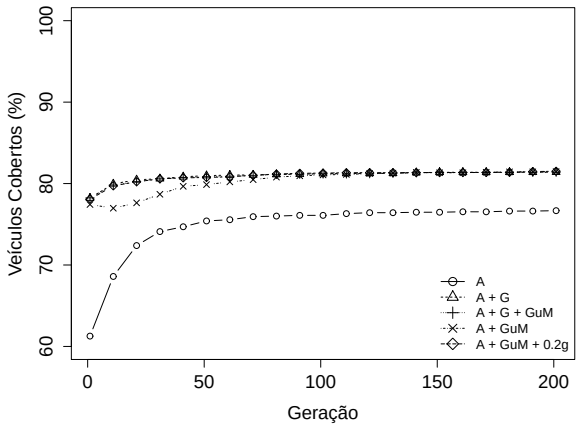


(a) Zurich

(b) Winterthur



(c) Baden



(d) Baar

Figura 4.3: Inicialização de população por cenários

Para aumentar a confiança dos resultados, optou-se por executar dez replicações. As

curvas representam a média da melhor *fitness* em cada geração. Para cada ponto, ou seja, para cada geração, calculamos o intervalo com 99% de confiança, aplicando o teste-t. Esses intervalos não foram apresentados nos gráficos para não polui-los e dificultar seu entendimento, no entanto, os intervalos das gerações 1, 50, 100, 150 e 200 podem ser observados na Tabela 4.3. Os melhores resultados estão em **negrito**.

Observa-se que em todos os cenários a convergência é mais lenta quando a população é totalmente aleatória, além disso, quando as soluções gulosas são utilizadas a qualidade da solução é estatisticamente superior à aleatória. É possível perceber também que o comportamento dos AGs que utilizam soluções gulosas é bastante similar. Mais ainda, aplicando o teste-t pareado nas médias da geração 200 nas inicializações que utilizam a solução gulosa, conclui-se que elas são equivalentes estatisticamente, já que seus intervalos de confiança sobrepõem-se.

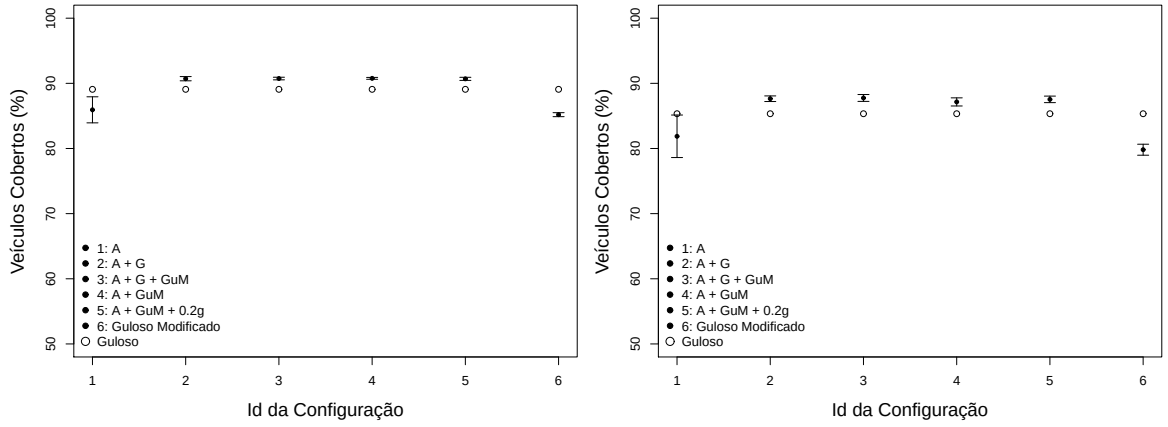
Tabela 4.3: Média e intervalo de confiança para cada cenário nas gerações 1, 50, 100, 150 e 200, nas diferentes inicializações da população

	Zurich		Winterthur		Baden		Baar		Ger
	I.C.	Média	I.C.	Média	I.C.	Média	I.C.	Média	
A	(64.09, 69.77)	66.93	(63.54, 75.92)	69.73	(49.10, 71.48)	60.29	(59.52, 63.02)	61.27	1
	(79.59, 84.55)	82.07	(77.34, 83.16)	80.25	(79.22, 84.70)	81.96	(72.68, 78.14)	75.41	50
	(81.98, 86.88)	84.43	(78.09, 84.53)	81.31	(79.48, 85.38)	82.43	(73.56, 78.64)	76.10	100
	(83.13, 87.37)	85.25	(78.52, 85.00)	81.76	(79.78, 85.78)	82.78	(73.87, 79.09)	76.48	150
	(83.45, 87.61)	85.53	(78.61, 85.13)	81.87	(79.79, 85.81)	82.80	(74.20, 79.14)	76.67	200
A + G	(87.65, 88.63)	88.14	(85.11, 87.13)	86.12	(83.81, 85.97)	84.89	(77.25, 79.19)	78.22	1
	(89.72, 90.44)	90.08	(87.14, 88.08)	87.61	(86.95, 87.79)	87.37	(80.65, 81.29)	80.97	50
	(90.06, 90.66)	90.36	(87.21, 88.07)	87.64	(87.24, 87.80)	87.52	(80.95, 81.49)	81.22	100
	(90.24, 90.86)	90.55	(87.21, 88.07)	87.64	(87.46, 87.84)	87.65	(81.20, 81.56)	81.38	150
	(90.35, 90.93)	90.64	(87.21, 88.07)	87.64	(87.46, 87.84)	87.65	(81.22, 81.58)	81.40	200
A + G + GuM	(89.08, 89.08)	89.08	(85.64, 86.72)	86.18	(85.78, 86.48)	86.13	(77.65, 78.83)	78.24	1
	(89.87, 90.63)	90.25	(87.11, 88.13)	87.62	(87.18, 87.78)	87.48	(80.43, 81.25)	80.84	50
	(90.32, 90.70)	90.51	(87.15, 88.25)	87.70	(87.32, 87.82)	87.57	(80.71, 81.65)	81.18	100
	(90.38, 90.86)	90.62	(87.18, 88.30)	87.74	(87.45, 87.91)	87.68	(81.17, 81.51)	81.34	150
	(90.41, 90.89)	90.65	**(87.23, 88.29)**	**87.76**	**(87.59, 87.87)**	**87.73**	(81.21, 81.57)	81.39	200
A + GuM	(89.08, 89.08)	89.08	(85.34, 85.34)	85.34	(86.02, 86.02)	86.02	(77.43, 77.43)	77.43	1
	(89.73, 90.59)	90.16	(86.42, 87.58)	87.00	(86.76, 87.48)	87.12	(78.95, 80.79)	79.87	50
	(90.11, 90.75)	90.43	(86.42, 87.58)	87.00	(87.13, 87.71)	87.42	(80.87, 81.25)	81.06	100
	(90.49, 90.83)	90.66	(86.44, 87.64)	87.04	(87.50, 87.82)	87.66	(81.09, 81.57)	81.33	150
	(90.53, 90.89)	**90.71**	(86.53, 87.77)	87.15	(87.52, 87.82)	87.67	(81.22, 81.72)	81.47	200
A + GuM + 0.2g	(87.03, 88.31)	87.67	(84.42, 86.34)	85.38	(84.08, 86.36)	85.22	(76.96, 79.20)	78.08	1
	(88.76, 90.26)	89.51	(86.63, 87.63)	87.13	(86.58, 88.06)	87.32	(80.07, 81.43)	80.75	50
	(89.57, 90.53)	90.05	(86.76, 87.62)	87.19	(86.71, 88.21)	87.46	(80.97, 81.59)	81.28	100
	(89.81, 90.69)	90.25	(86.99, 87.71)	87.35	(86.71, 88.21)	87.46	(81.10, 81.06)	81.35	150
	(90.09, 90.81)	90.45	(86.96, 88.02)	87.49	(86.71, 88.21)	87.46	**(81.23, 81.79)**	**81.51**	200

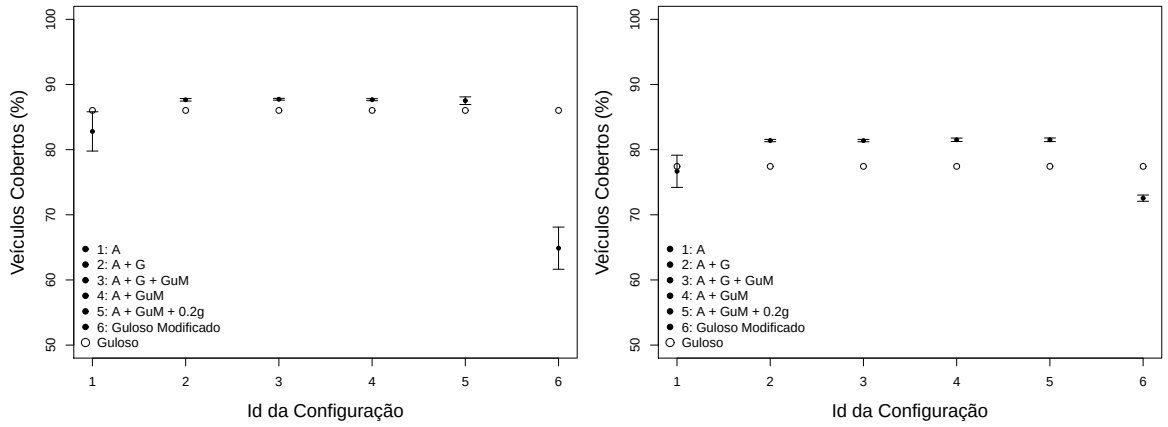
Nas execuções do AG para cada configuração de inicialização da população, diferentes soluções foram geradas, isto é, mais de um conjunto de interseções foram apresentados como resposta. A Figura 4.4 mostra as médias das aptidões de cada inicialização e de 200 execuções do algoritmo guloso modificado, e seus intervalos com 99% de confiança. Nos

gráficos também é apresentada a cobertura alcançada da solução fornecida pelo algoritmo guloso.

A Figura 4.4 mostra que a inicialização da população totalmente aleatória apresenta a maior variação nas aptidões das soluções entre todas as abordagens, isso ocorre pois o AG não é direcionado inicialmente para gerar boas soluções. Além disso, fica claro que a inserção de soluções gulosas na inicialização da população aumenta a cobertura de veículos com relação ao guloso e ao modificado.



(a) Zurich (b) Winterthur



(c) Baden (d) Baar

Figura 4.4: Média das aptidões finais de cada inicialização da população nas dez replicações por cenários

A Tabela 4.3 apresenta os resultados das melhores aptidões, as médias e os intervalos de confiança em cada cenário. Em **negrito**, estão os melhores resultados e sublinhado estão as melhores médias.

As inicializações que utilizam soluções gulosas são equivalentes estatisticamente, já que seus intervalos de confiança se sobrepõem. A configuração A+GuM+0.2g alcança em

todos os cenários o melhor resultado, isso indica que selecionar as interseções mais frequentadas melhora a solução. No entanto, essa mesma configuração não é a melhor considerando a média dos resultados, em que ganha somente em Baar, isso indica uma maior variação nos resultados nas replicações.

A cobertura é melhorada em relação ao guloso em 1.79, 2.94, 1.79 e 4.44 pontos percentuais para os cenários de Zurich, Winterhur, Baden e Baar, respectivamente. Em suma, considerando a inicialização que utiliza o algoritmo guloso é possível melhorar os resultados obtidos pelo AG, evidenciando assim a qualidade da nossa solução.

Nos experimentos realizados posteriormente, foi utilizado para inicialização da população a configuração A+G+GuM.

Tabela 4.4: Análise das aptidões das soluções finais obtidas pelos algoritmo genético considerando as inicializações da população, algoritmo guloso e guloso modificado

		A	A + G	A + G + GuM	A + GuM	A + GuM + 0.2g	GuM	G
Zurich	Melhor	89.45	90.81	**90.87**	90.84	90.84	88.04	89.08
	Média	85.53	90.64	90.65	90.71	90.45	85.19	
	I.C.	(83.45, 87.61)	(90.35, 90.93)	(90.41, 90.89)	(90.53, 90.89)	(90.09, 90.81)	(84.88, 85.50)	
Winter-thur	Melhor	85.99	**88.28**	**88.28**	87.73	**88.28**	86.43	85.34
	Média	81.87	87.64	87.76	87.15	87.54	79.81	
	I.C.	(78.61, 85.13)	(87.21, 88.07)	(87.23, 88.29)	(86.53, 87.77)	(87.04, 88.04)	(78.97, 80.65)	
Baden	Melhor	85.44	87.78	87.78	87.78	**87.81**	85.75	86.02
	Média	82.80	87.65	87.73	87.67	87.52	64.88	
	I.C.	(79.79, 85.81)	(87.46, 87.84)	(87.59, 87.87)	(87.52, 87.82)	(86.93, 88.11)	(61.64, 68.12)	
Baar	Melhor	79.65	81.66	81.66	81.76	**81.87**	78.50	77.43
	Média	76.68	81.40	81.39	81.47	81.51	72.55	
	I.C.	(74.21, 79.15)	(81.22, 81.58)	(81.21, 81.57)	(81.25, 81.77)	(81.23, 81.79)	(72.07, 73.03)	

4.1.4 Operadores Genéticos

Nessa subseção, serão apresentados os resultados dos experimentos considerando os operadores genéticos descritos na Seção 2.4. Esses operadores foram desenvolvidos com o objetivo de melhorar a qualidade das soluções alcançadas pelo AG.

Foram combinados três operadores de cruzamento, dois operadores de mutação e quatro operadores de busca local, o que resultou em oito experimentos. A Tabela 4.5, contém apenas as permutações executadas. Em todos os experimentos, as configurações apresentadas na Subseção 4.1.2 foram mantidas. Além disso, a configuração A+G+GuM foi utilizada para inicializar a população.

Na Figura 4.5, são apresentadas as curvas de convergência de cada cenário considerando as oito configurações apresentadas. Mais uma vez, foram executadas dez replicações para cada configuração e as curvas apresentadas correspondem a média dos melhores indivíduos de cada geração. A escala dos gráficos está diferente em cada caso para mostrar melhor o comportamento do AG.

Tabela 4.5: Combinações dos experimentos para análise do impacto dos operadores genéticos no AG

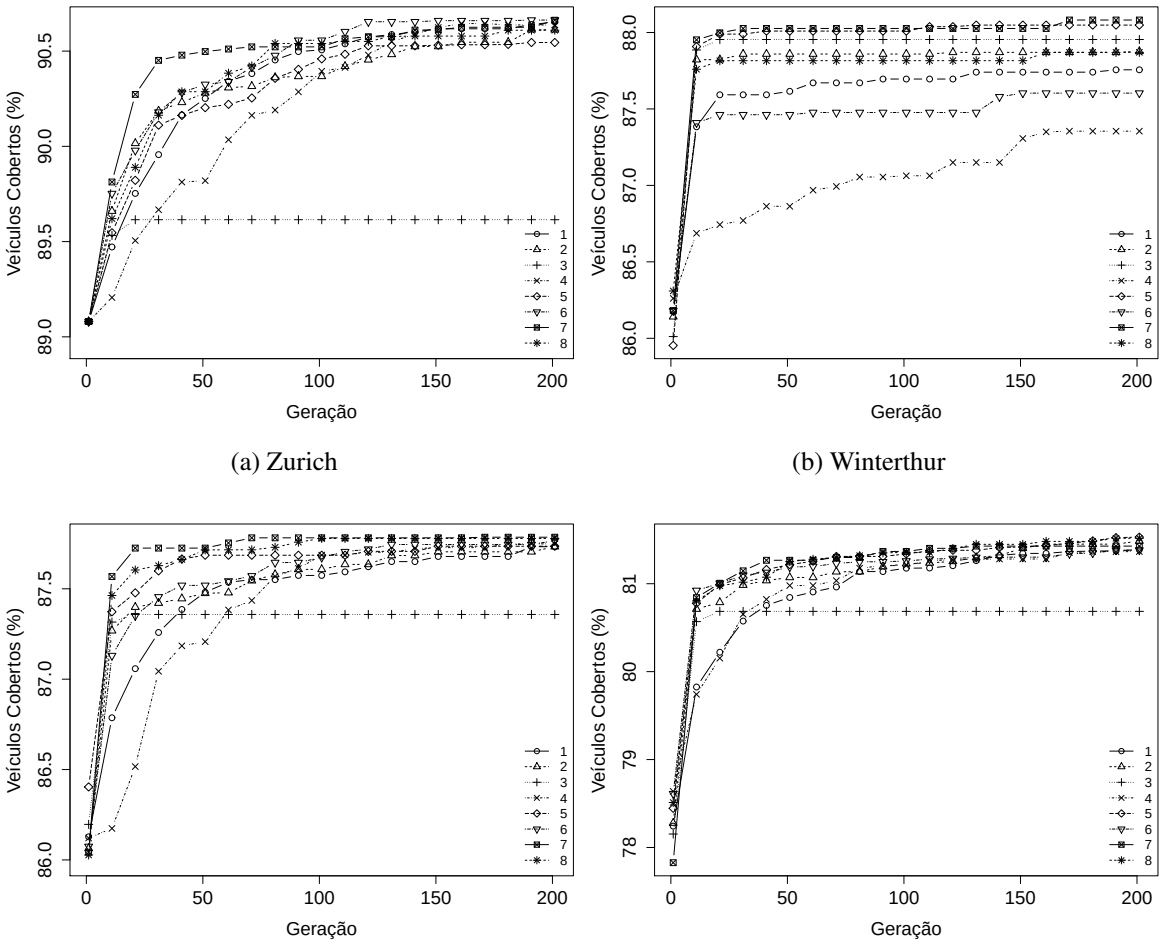
Id.	Op. de Cruzamento	Op. de Mutação	Busca Local
1	Um ponto	Um ponto	–
2	Fusão	Um ponto	–
3	Fusão	Remoção do pior gene	–
4	Inserção dos melhores que a média	Um ponto	–
5	Fusão	Um ponto	Substituição de genes densos por densos
6	Fusão	Um ponto	Substituição de genes não densos por densos
7	Fusão	Um ponto	Substituição de genes densos por acima da média
8	Fusão	Um ponto	Substituição de genes densos por ao redor da média

No gráfico 4.5b pode-se observar que a estratégia de número 4 obteve o pior resultado em Winterthur, diferente dos outros, em que a pior estratégia foi a de número 3. Isso pode indicar que inserir as interseções por onde passam mais veículos que a média de todas as interseções em Winterthur afeta negativamente o tempo em que os veículos permanecem nelas, isto é, nas interseções mais frequentadas os veículos passam mais rapidamente.

Os gráficos 4.5a, 4.5c e 4.5d mostram que a pior configuração é a 3. Isso indica que a pior interseção, ou seja, aquela por onde passam menos veículo, influencia fortemente no tempo em que os veículos permanecem na interseção e ajuda a melhorar a solução. Ademais, percebe-se nos quatro cenários que o comportamento da estratégia 7 é superior aos demais em algum momento da evolução. Na Tabela 4.6, é possível confirmar que essa estratégia alcança ótimos resultados, tanto considerando o melhor indivíduo quanto a média das soluções nas 10 replicações. Dessa forma, a configuração 7 foi a selecionada para os próximos experimentos realizados.

Na Figura 4.6 estão apresentados os intervalos com 99% confiança dos resultados finais em cada uma das configurações do AG e do algoritmo guloso modificado, identificada pelo número 9, além do resultado do algoritmo guloso. A Tabela 4.4 complementa a figura e apresenta os melhores resultados, as médias e o intervalos de confianças para os algoritmos genético e guloso modificado, além da solução fornecida pelo algoritmo guloso. Os melhores indivíduos estão em **negrito**, as melhores médias estão sublinhadas.

Na Figura 4.6, nota-se o algoritmo guloso modificado, por sua vez, não consegue obter uma melhor cobertura que o AG em nenhum dos cenário. Isso confirma que os operadores genéticos funcionam para melhorar a solução. E em comparação com o algoritmo guloso, verifica-se que as soluções fornecidas pelo AG também são superiores.



(a) Zurich

(b) Winterthur

(c) Baden

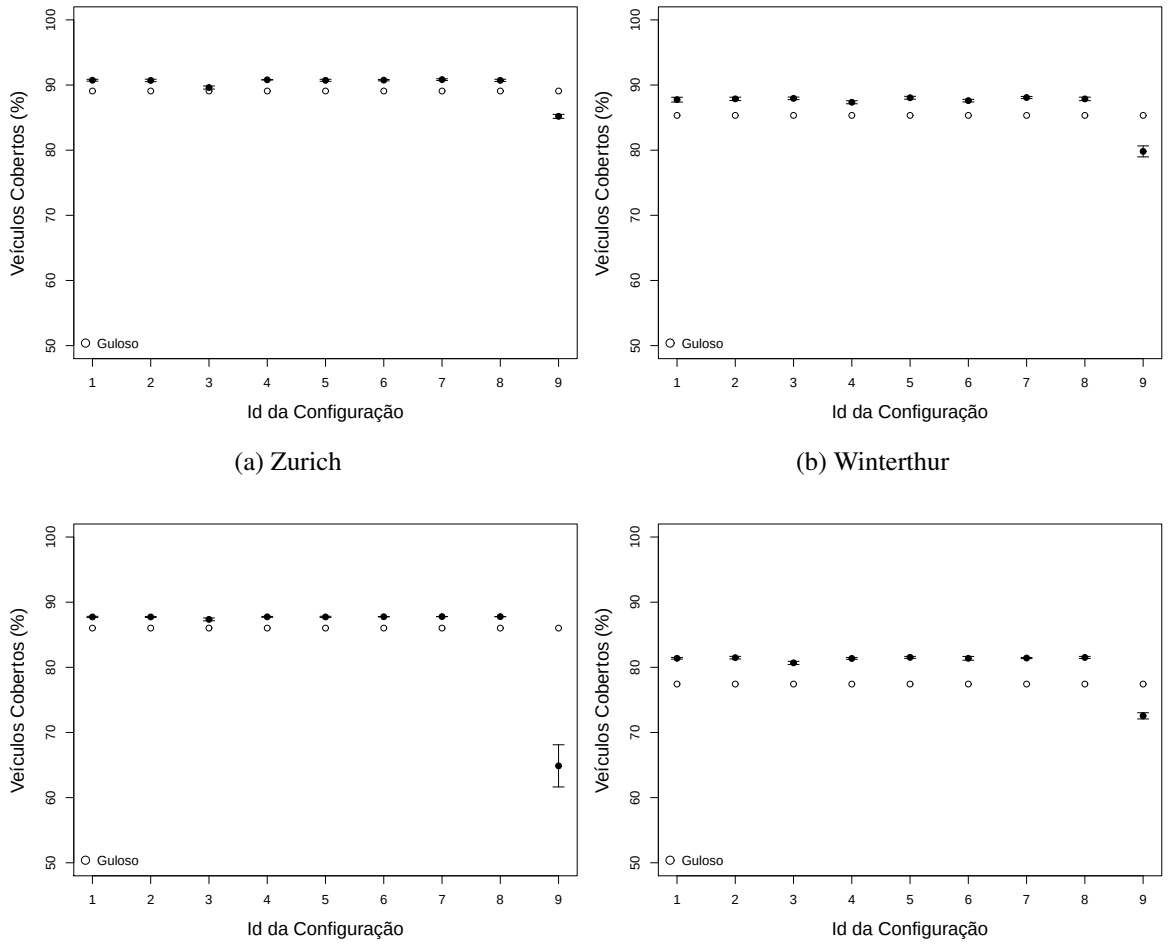
(d) Baar

Figura 4.5: Análise da convergência das configurações do AG

4.1.5 Variação no tempo τ

Nesta seção apresentamos uma análise na variação do tempo mínimo τ que o veículo precisa permanecer no raio de transmissão R dos PAs para receber com sucesso a informação transmitida. Nos cenários anteriores foram utilizados somente $\tau = 30\,\text{s}$. Nos próximos experimentos, esse número foi variado de 10 s a 60 s em intervalos de 10 unidades. Foram consideradas a inicialização A+G+GuM da Subseção 4.1.3 e a configuração 7 do AG apresentada na Subseção 4.1.4.

Para cada experimento com o AG foram executadas 10 replicações e calculado intervalos com 99% de confiança, dos resultados. O algoritmo guloso modificado foi replicado 200 vezes e seu intervalo de confiança também foi calculado. A Figura 4.7 e a Tabela 4.7 sintetizam os resultados. Elas apresentam os resultados com intervalos de confiança dos algoritmos genético, guloso modificado e o resultado do guloso.



(a) Zurich

(b) Winterthur

(c) Baden

(d) Baar

Figura 4.6: Aptidões finais com intervalos de confiança para cada configuração do AG, GuM e guloso

A partir dos gráficos da Figura 4.7, é possível perceber em todos os cenários que quanto menor o tempo mínimo τ mais os algoritmos apresentam resultados semelhantes, pois fica mais fácil encontrar combinações de interseções que atendam ao tempo determinado. Além disso, a cobertura alcançada é bem elevada aproximando-se de 100%. Isso é justificado, observando a média que os veículos gastam para atravessar as interseções apresentadas na Figura 4.2, que é em sua maioria é menos que 15 s.

A partir de $\tau = 20$ s, a solução fornecida pelo algoritmo guloso modificado é estatisticamente inferior aos resultados tanto do algoritmo guloso quanto do genético, em todos os cenários. Os resultados do AG tornam-se maiores estatisticamente que o algoritmo guloso a partir de $\tau = 30$ s, no entanto, esse ganho não é muito perceptível. Já quando $\tau \geq 40$ s, o ganho é considerável em todos os cenários, principalmente em Baden, em que o chega a 20.12 pontos percentuais.

Tabela 4.6: Análise das soluções finais obtidas pelo algoritmo genético considerando as combinações de operadores, algoritmo guloso e guloso modificado

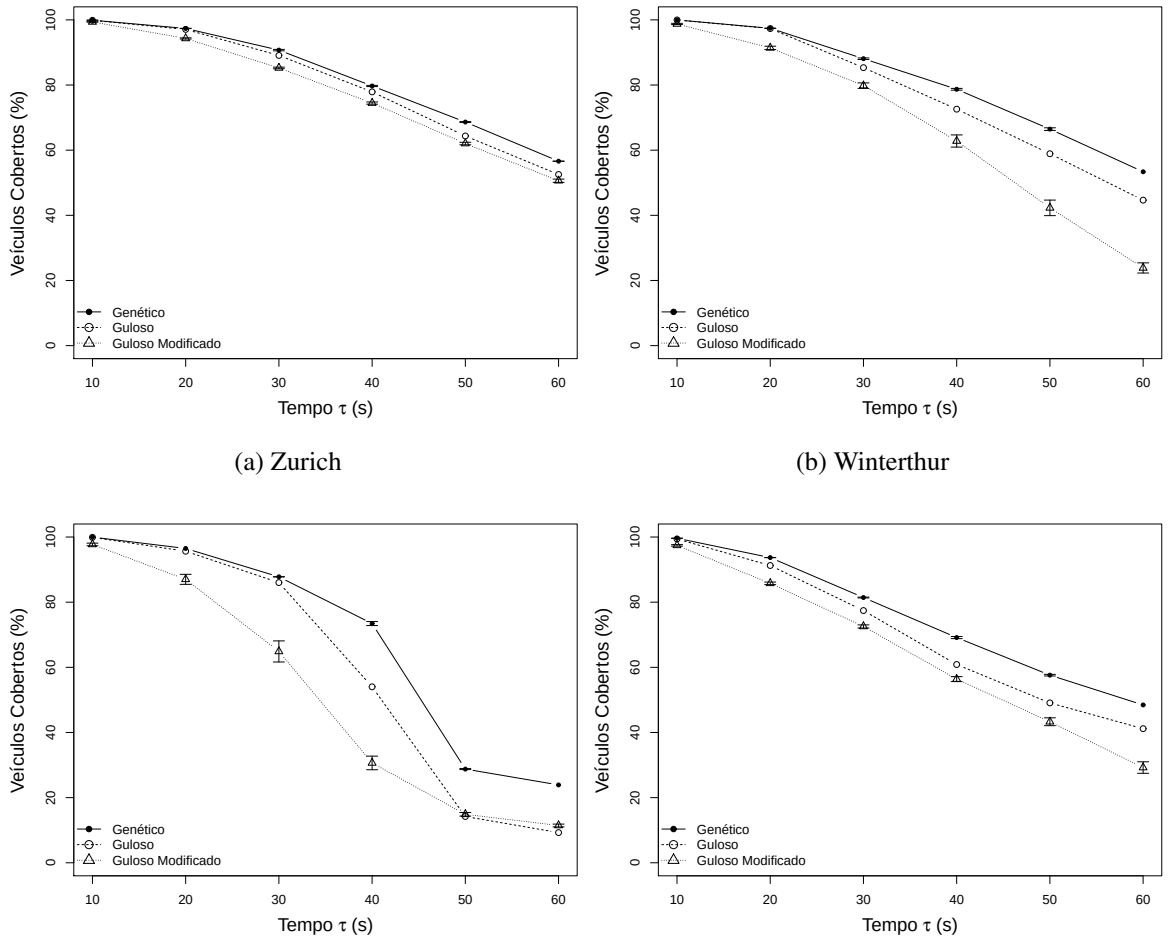
		Configuração				
		1	2	3	4	5
Zurich	Melhor	90.88	**91.09**	90.08	90.84	**91.09**
	Média	90.74	90.70	89.61	90.80	90.72
	I.C.	(90.60, 90.88)	(90.51, 90.89)	(89.37, 89.85)	(90.76, 90.84)	(90.57, 90.87)
Winterthur	Melhor	**88.28**	**88.28**	**88.28**	87.66	88.28
	Média	87.76	87.88	87.95	87.35	88.05
	I.C.	(87.39, 88.13)	(87.62, 88.14)	(87.75, 88.15)	(87.12, 87.58)	(87.83, 88.27)
Baden	Melhor	87.78	87.78	87.78	**87.81**	**87.81**
	Média	87.73	87.73	87.36	87.75	87.74
	I.C.	(87.63, 87.83)	(87.65, 87.81)	(87.14, 87.58)	(87.68, 87.82)	(87.64, 87.84)
Baar	Melhor	81.66	81.87	81.09	**81.87**	**81.87**
	Média	81.39	81.49	80.69	81.37	81.53
	I.C.	(81.26, 81.52)	(81.29, 81.69)	(80.45, 80.93)	(81.23, 81.51)	(81.38, 81.68)

		Configuração				
		6	7	8	GuM	G
Zurich	Melhor	90.84	**91.09**	**91.09**	88.04	
	Média	90.75	90.83	90.72	85.19	89.08
	I.C.	(90.64, 90.86)	(90.70, 90.96)	(90.55, 90.89)	(84.88, 85.50)	
Winterthur	Melhor	87.94	**88.28**	**88.28**	86.43	
	Média	87.60	88.08	87.87	79.81	85.34
	I.C.	(87.40, 87.80)	(87.92, 88.24)	(87.60, 88.14)	(78.97, 80.65)	
Baden	Melhor	87.78	**87.81**	**87.81**	85.75	
	Média	87.77	87.79	87.78	64.88	86.02
	I.C.	(87.75, 87.79)	(87.78, 87.80)	(87.77, 87.79)	(61.64, 68.12)	
Baar	Melhor	**81.87**	81.66	**81.87**	78.50	
	Média	81.38	81.43	81.52	72.55	77.43
	I.C.	(81.09, 81.67)	(81.34, 81.52)	(81.36, 81.68)	(72.07, 73.03)	

O cenário mais difícil de Zurich, gráfico 4.7a, é o que apresenta comportamento mais homogêneo comparando os três algoritmos. O algoritmo genético obtém melhor resultados em todos os casos e seu ganho máximo com relação ao algoritmo guloso é 4.39 pontos percentuais, como mostra a Tabela 4.7.

O cenário de Baden, gráfico 4.7c é o que apresenta comportamento menos homogêneo quando comparado com os outros cenários. Há uma queda abrupta na cobertura de veículos ao reduzir o tempo τ de 40 para 50 s. Isso indica que a dificuldade do problema ao aumentar o tempo de recebimento da informação pelos veículos cresce consideravelmente e implantar PAs em somente 30% das interseções em Baden torna a troca de informações na infraestrutura ineficaz.

Analisando a cobertura alcançada com somente 30% das interseções preenchidas por PAs, percebe-se que, em todos os cenários, se τ for maior que 30 s a cobertura não alcança pelo menos 80% dos veículos. Assim existe um forte compromisso a ser considerado entre o número de PAs que será implantado e o tempo em que a informação demora para ser



(a) Zurich

(b) Winterthur

(c) Baden

(d) Baar

Figura 4.7: Variação do tempo τ para cada cenário

transmitida ao veículo.

Por fim, nota-se que a dificuldade do problema aumenta à medida que aumentamos o tempo de recebimento da informação e para resolver problemas mais difíceis, o AG gera soluções melhores.

4.1.6 Variações no número k de PAs

Nesta seção apresentamos uma análise na variação do número de PAs. Nos cenários anteriores eram utilizados apenas 30% do número de cruzamentos. Aqui esse número foi variado de 10 a 35 em intervalos de 5 unidades. A Figura 4.8 e a Tabela 4.8 sintetizam os resultados.

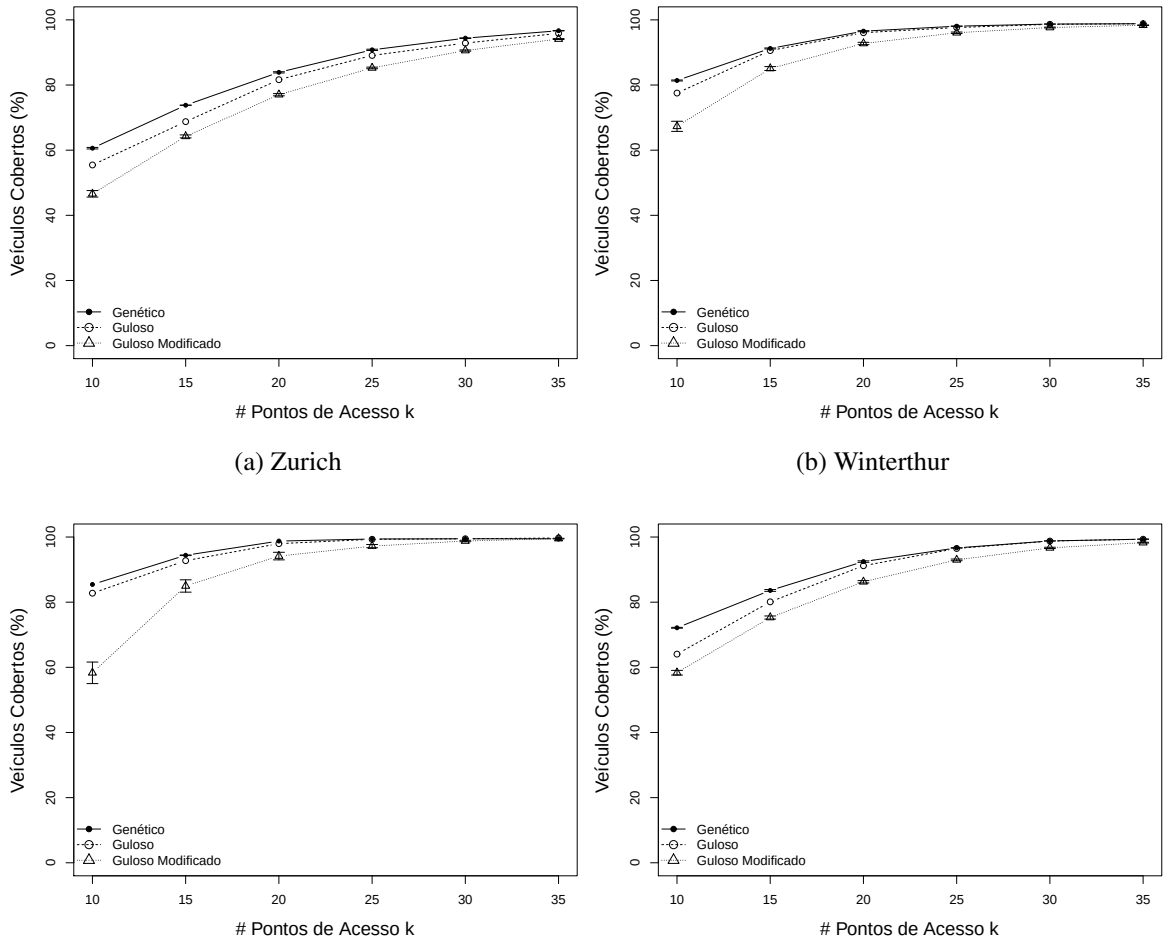
Como esperado, com o aumento do número de PAs, o percentual de áreas cobertas também aumenta. Nota-se que, em muitos casos, o AG obtém estatisticamente os melhores resultados que o algoritmo guloso e guloso modificado.

Tabela 4.7: Resultado da cobertura para a variação no tempo τ de contato entre o veículo e os PAs para transmissão da informação

Cenário	Algoritmo		Tempo τ					
			10	20	30	40	50	60
Zurich	GA	Melhor	**99.92**	**97.43**	**91.03**	**79.89**	**68.75**	**56.65**
		Média	99.90	97.37	90.72	79.71	68.64	56.62
		I.C.	(99.89, 99.91)	(97.33, 97.41)	(90.50, 90.94)	(79.52, 79.90)	(68.51, 68.77)	(56.58, 56.66)
	GuM	Melhor	99.83	96.47	89.02	77.84	68.10	55.44
		Média	99.37	94.30	85.25	74.46	62.04	50.62
		I.C.	(99.32, 99.42)	(94.12, 94.48)	(85.00, 85.5)	(74.12, 74.8)	(61.65, 62.43)	(50.12, 51.12)
	Gu		99.86	97.08	89.08	77.89	64.36	52.53
	Melhor GA − Gu (p.p.)		0.06	0.35	1.95	2.00	4.39	4.12
Winterthur	GA	Melhor	**99.99**	**97.45**	**88.28**	**78.80**	**66.88**	**53.37**
		Média	99.98	97.43	88.08	78.69	66.46	53.37
		I.C.	(99.98, 99.98)	(97.41, 97.45)	(87.84, 88.32)	(78.42, 78.96)	(66.05, 66.87)	(53.37, 53.37)
	GuM	Melhor	99.87	95.82	86.43	77.43	**66.88**	**53.37**
		Média	98.75	91.40	79.81	62.81	42.29	23.85
		I.C.	(98.56, 98.94)	(90.88, 91.92)	(78.97, 80.65)	(60.94, 64.68)	(39.9, 44.68)	(22.29, 25.41)
	Gu		99.98	97.33	85.34	72.59	58.90	44.66
	Melhor GA − Gu (p.p.)		0.01	0.12	2.94	6.21	7.98	8.71
Baden	GA	Melhor	**99.97**	**96.49**	**87.81**	**74.13**	**28.86**	**23.90**
		Média	99.97	96.49	87.79	73.44	28.76	23.90
		I.C.	(99.97, 99.97)	(96.49, 96.49)	(87.77, 87.81)	(72.82, 74.06)	(28.6, 28.92)	(23.90, 23.90)
	GuM	Melhor	99.85	95.58	85.75	68.05	24.30	19.48
		Média	97.72	87.01	64.88	30.65	14.88	11.44
		I.C.	(97.32, 98.12)	(85.47, 88.55)	(61.64, 68.12)	(28.56, 32.74)	(14.33, 15.43)	(11.01, 11.87)
	Gu		99.91	95.64	86.02	54.01	14.21	9.27
	Melhor GA − Gu (p.p.)		0.06	0.85	1.79	20.12	14.65	14.63
Baar	GA	Melhor	**99.64**	**93.70**	**81.66**	**69.35**	**57.95**	**48.45**
		Média	99.63	93.69	81.43	69.14	57.60	48.45
		I.C.	(99.62, 99.64)	(93.65, 93.73)	(81.3, 81.56)	(68.81, 69.47)	(57.35, 57.85)	(48.45, 48.45)
	GuM	Melhor	99.12	90.86	78.50	65.88	52.18	45.53
		Média	97.44	85.78	72.55	56.38	43.30	29.23
		I.C.	(97.21, 97.67)	(85.39, 86.17)	(72.07, 73.03)	(55.62, 57.14)	(42.09, 44.51)	(27.46, 31)
	Gu		99.40	91.26	77.43	60.86	49.08	41.17
	Melhor GA − Gu (p.p.)		0.24	2.44	4.23	8.49	8.87	7.28

Para o conjunto de Zurich, em particular, quando o número máximo de PAs é implantado, há um acréscimo de 10 PAs na região, com relação ao número dos experimentos anteriores, o que levou a uma cobertura de 96.81% de veículos. Para os outros três conjuntos de dados, o número de PAs foi aumentado consideravelmente. Para Winterthur, onde 43 interseções estão disponíveis, 35 PAs correspondem a cerca de 81% das interseções cobertas. Neste caso, o número de veículos cobertos pelo AG aumentou para 98.82%. Para Baden, a abrangência é ainda maior, 92% das interseções possuem PAs, e o número de veículos cobertos foi para 99.47%. Esse caso merece destaque, pois a melhor solução algoritmo guloso randômico resultou em 100% de cobertura, única configuração que levou à cobertura total da região. Finalmente, para Baar, 76% das interseções foram equipadas com antenas, levando a 96.35% de cobertura.

Entre os resultados apresentados, Baar apresenta a cobertura de veículos mais baixa



(a) Zurich

(b) Winterthur

(c) Baden

(d) Baar

Figura 4.8: Variação do número k de PAs a serem implantados

mesmo comparado a Zurich, em que a porcentagem de PAs nas interseções é bem inferior. Isto pode ser explicado pelas características deste cenário, que possui tráfego leve. Observando os gráficos 4.1a e 4.1d e os dados da Tabela 4.1, nota-se que o percentual de veículos que circulam é altamente correlacionado com as densidades das regiões. Como esperado, é mais fácil cobrir veículos em regiões mais densas do que com tráfegos leves. Assim, atingir o máximo de cobertura em Zurich é mais fácil do que fazer o mesmo em Baar, embora Zurich possua mais interseções.

4.2 Predição de Trajetória de Veículos

Nessa seção, serão apresentados os resultados obtidos dos experimentos realizados na predição de trajetória de veículos utilizando algoritmos de aprendizado de máquina e a modelagem

Tabela 4.8: Resultado da cobertura para a variação do número k de PAs a serem implantados

Cenário	Algoritmo		# Pontos de Acesso k					
			10	15	20	25	30	35
Zurich	GA	Melhor	60.68	73.85	84.00	91.09	94.52	96.81
		Média	60.61	73.80	83.89	90.80	94.39	96.66
		I.C.	(60.38, 60.84)	(73.78, 73.82)	(83.70, 84.08)	(90.60, 91.00)	(94.29, 94.49)	(96.6, 96.72)
	GuM	Melhor	57.05	69.99	81.73	89.02	93.03	95.93
		Média	46.60	64.23	77.03	85.25	90.58	94.12
		I.C.	(45.60, 47.60)	(63.77, 64.69)	(76.69, 77.37)	(85.00, 85.50)	(90.39, 90.77)	(93.98, 94.26)
	Gu		55.45	68.76	81.64	89.08	92.88	95.86
Winterthur	GA	Melhor	81.60	91.41	96.68	98.10	98.72	98.82
		Média	81.39	91.23	96.53	98.06	98.71	98.82
		I.C.	(81.24, 81.54)	(91.08, 91.38)	(96.37, 96.69)	(98.03, 98.09)	(98.7, 98.72)	(98.82, 98.82)
	GuM	Melhor	79.08	90.19	96.12	97.87	98.64	98.84
		Média	67.32	85.06	92.75	96.05	97.65	98.36
		I.C.	(65.77, 68.87)	(84.46, 85.66)	(92.39, 93.11)	(95.83, 96.27)	(97.51, 97.79)	(98.27, 98.45)
	Gu		77.54	90.58	96.07	97.67	98.64	98.82
Baden	GA	Melhor	85.44	94.45	98.76	99.40	99.47	99.47
		Média	85.44	94.40	98.76	99.39	99.47	99.47
		I.C.	(85.44, 85.44)	(94.31, 94.49)	(98.76, 98.76)	(99.38, 99.40)	(99.47, 99.47)	(99.47, 99.47)
	GuM	Melhor	82.70	94.14	97.81	99.21	99.97	100.00
		Média	58.31	84.97	94.15	97.21	98.78	99.52
		I.C.	(55.01, 61.61)	(83.08, 86.86)	(92.99, 95.31)	(96.71, 97.71)	(98.56, 99.00)	(99.42, 99.62)
	Gu		82.76	92.74	97.97	99.30	99.47	99.70
Baar	GA	Melhor	72.32	83.83	92.56	96.74	98.83	99.35
		Média	72.14	83.60	92.46	96.65	98.80	99.35
		I.C.	(72.01, 72.27)	(83.32, 83.88)	(92.23, 92.69)	(96.60, 96.70)	(98.79, 98.81)	(99.34, 99.36)
	GuM	Melhor	66.25	81.48	90.72	96.12	98.36	99.24
		Média	58.35	75.29	86.26	92.99	96.67	98.27
		I.C.	(57.63, 59.07)	(74.81, 75.77)	(85.9, 86.62)	(92.73, 93.25)	(96.47, 96.87)	(98.15, 98.39)
	Gu		64.05	80.10	91.19	96.49	98.76	99.32

proposta no capítulo 3. Foi feita uma avaliação do desempenho dos algoritmos e um estudo da influência do tamanho da janela na modelagem da base de dados. Os experimentos e análises realizados para esse problema possuem caráter exploratório e, apesar de apresentarem resultadores promissores, necessitam de um estudo mais aprofundado.

4.2.1 Base de Dados

A base de dados utilizada para realizar os experimentos contém registros reais da mobilidade de motoristas que circulam na cidade de Börlange [Frejinger, 2008]. Börlange é uma cidade localizada na Suécia que possui 3.077 cruzamentos interconectados por 7.459 estradas, seu mapa viário completo pode ser visto na Figura 4.9.

Os dados foram coletados durante dois anos (1999 – 2001) para um experimento de análise de congestionamento de tráfego. Aproximadamente 200 carros particulares (com um motorista por carro) foram equipados com um dispositivo GPS. Em intervalos de tempo regulares (cada 5 segundos aproximadamente), a posição e a hora do veículo foram gravadas e armazenadas. Devido a questões da acurácia do GPS, muitas rotas observadas não corres-

pondiam ao mapa da cidade, assim os dados de 24 usuários foram manualmente verificados e corrigidos, o que resultou na base de dados disponível com um total de 420.814 rotas.

Figura 4.9: Mapa viário da cidade de Börlange, na Suécia

Cada instância da base original representa um trecho percorrido pelo veículo. Os atributos considerados são: identificador do usuário, identificador do dia, identificador da viagem, instante de tempo inicial, instante de tempo final, longitude inicial e final, latitude inicial e final. Esse formato foi convertido para ficar compatível com o modelo de janelas deslizantes descrito no Capítulo 3. Isso foi feito da seguinte forma: atribuiu-se um identificador único para cada par de longitude, latitude iniciais e longitude, latitude finais de cada trecho, ou seja, cada trecho foi identificado, o que resultou num total de 5761 trechos diferentes. Posteriormente, cada viagem de usuário foi particionada no número n de janelas determinado, em que os primeiros $n-1$ identificadores correspondem aos atributos e o iden-

tificador *n* corresponde à classe daquela instância. Cada usuário possui sua própria base de dados, com o número total de trechos que acessa.

A Tabela 4.9 apresenta as características de cada usuário, considerando o número de dias cujos dados foram coletados, o número de viagens totais realizadas, o número de trechos totais percorridos, a média de viagens realizadas por dia e média de trechos percorridos por viagens. O usuário 8 possui menos registros e o 22 mais registros. Essa informação é importante, pois considera-se a hipótese de que quanto mais registro um usuário possui, mais acurada será a previsão do seu comportamento. A quantidade de viagens por dia varia bastante entre os usuários, no entanto, cada usuário faz, no mínimo 3 viagens por dias. Com relação a quantidade de trechos que percorrem por viagem, há usuários que possuem viagens bem curtas, com cerca de 4 trechos percorridos, como também viagens longas, com mais de 16 trechos percorridos.

Tabela 4.9: Características dos registros das viagens de cada usuário

Id.	# Dias	# Viagens	# Trechos	Média $\frac{Viagens}{Dia}$	Média $\frac{Trechos}{Viagens}$
1	121	461	15383	3.81	12.71
2	186	1291	31861	6.94	8.56
3	145	783	19296	5.40	6.05
4	151	813	22749	5.38	9.42
5	50	303	8258	6.06	9.18
6	55	305	10792	5.55	16.35
7	134	722	22548	5.39	12.94
8	15	162	4286	10.80	14.29
11	92	542	14739	5.89	7.63
15	88	470	10462	5.34	4.40
17	58	342	12681	5.90	14.58
22	208	2292	46517	11.02	4.14
24	117	1197	24733	10.23	9.61
28	142	927	24043	6.53	11.29
68	89	421	12704	4.73	10.98
88	76	736	17437	9.68	9.98
102	73	392	6529	5.37	4.47
131	154	772	20820	5.01	7.51
154	94	598	16402	6.36	10.26
155	74	486	12442	6.57	9.89
164	71	710	23007	10.00	7.54
175	67	633	14601	9.45	12.11
194	78	631	15645	8.09	6.27
210	75	486	12879	6.48	13.21

Foram executados experimentos com tamanho de janela iguais a 1, 5, 10, 15. Dessa forma, foram geradas 4 bases de dados para cada usuário. Cada base possui suas próprias características quanto ao número de classes e instâncias, essa informação está apresentada na Tabela 4.10, juntamente com o número de trechos únicos em que cada usuário passa. Fica explícito que o tamanho da janela e a quantidade de classes e instâncias da base é

inversamente proporcional, comparando as colunas # Classes e # Instância do tamanho de janela 1 com as do tamanho 15, por exemplo. Isso ocorre pois aumentando-se o tamanho da janela, a quantidade de partições cuja trajetória do usuário é dividi da diminui. Observa-se também que a quantidade de classes é bem próxima da quantidade de trechos que o usuário percorre, analisando a coluna # Trechos e # Classes. Isso é claro, pois o número de classes da base de um usuário pode ser, no máximo, o número de trechos por onde ele passa. A tabela também apresenta, na coluna % Classes Únicas, a porcentagem de instâncias cuja classe aparece somente uma vez na base de dados, o que impede a predição. Para contornar esse problema, replicamos cinco vezes, o número de *folds* da validação cruzada, cada uma das instâncias de classes únicas antes de aplicar os algoritmos.

Se a classe de uma base de dado for vista como uma variável aleatória discreta X, é possível medir o grau de incerteza para cada base através do cálculo de entropia [Shannon, 2001]. Para o problema de predição do trecho, seja $p(x) = \Pr\{X = x\}$ e $x \in T$, onde T é o conjunto de todos os trechos possíveis. A entropia $H(X)$ da variável aleatória discreta X é definida por

$$H(X) = -\sum_{x \in T} p(x) \log_2 p(x) \tag{4.1}$$

Uma entropia alta significa que X é proveniente de uma distribuição uniforme, já uma baixa entropia significa que X é proveniente de uma distribuição variada. Com isso, quanto maior a entropia da distribuição mais difícil ela é de ser predita.

A Tabela 4.12 apresenta a entropia associada à cada base de dados. Na coluna Base Aleatória, a entropia foi calculada forçando cada instância da base de dados do usuário a possuir uma classe diferente, ou seja, se a base possuir 20 instâncias, cada uma dessas instância pertencerá a uma classe diferente, a base terá 20 classes. Isso leva a uma distribuição uniforme da classe, a uma entropia alta e com isso a uma maior dificuldade na previsibilidade do comportamento do motorista. Na coluna Base Real, a entropia foi calculada considerando a real classificação de cada instância, e, como esperado, tornando a entropia menor e aumentando a previsibilidade do comportamento. Assim, é possível perceber o ganho de informação obtido com a partir do histórico do usuário, que está apresentado na coluna Ganho. Percebe-se que a entropia das bases possuem valores similares, concentram-se entre 8 e 9 bits. Esse valor pode ser considerado alto, já que o ganho adquirido partindo da base artificialmente desordenada para a base considerando o histórico do usuário não é alto. Isso pode indicar que as predições podem não alcançar resultados satisfatórios.

Nas próximas sub-seções serão descritos os experimentos e resultados obtidos.

Tabela 4.10: Características das quatro bases de dados criadas para cada usuário

Id.	# Trechos	Tamanho da Janela					
		1			5		
		# Classes	# Instâncias	% Classes Únicas	# Classes	# Instâncias	% Classes Únicas
1	1230	1216	14898	22.31	1113	13127	23.13
2	2436	2406	30566	55.81	2259	25732	58.06
3	2307	2272	18526	67.78	2103	15709	68.48
4	1990	1963	21917	47.18	1818	18876	49.77
5	1191	1166	7953	34.62	1077	6819	36.41
6	1274	1253	10493	30.60	1153	9469	29.77
7	2023	1989	21823	51.97	1840	19123	54.29
8	882	877	4096	31.71	815	3486	34.65
11	1474	1456	14234	33.68	1351	12619	35.58
15	1432	1421	9993	41.54	1315	8274	45.07
17	1546	1536	12311	33.68	1456	11064	35.30
22	2160	2127	44246	50.17	1973	36154	54.29
24	1496	1477	23563	36.50	1373	19206	38.71
28	1636	1621	23122	39.40	1525	19621	44.79
68	1779	1758	12289	49.06	1652	10697	52.63
88	1780	1749	16692	37.35	1632	14080	42.03
102	1667	1621	6160	58.97	1440	4998	59.17
131	2057	2013	20016	63.76	1874	17202	66.73
154	1146	1133	15762	28.80	1041	13507	28.76
155	1428	1412	11967	38.21	1289	10202	38.71
164	2122	2096	22308	44.19	1935	19754	44.42
175	1400	1364	13967	28.89	1264	11646	31.71
194	1560	1534	15027	39.66	1410	12801	39.45
210	1201	1170	12350	29.32	1085	10651	29.03

Id.	# Trechos	Tamanho da Janela					
		10			15		
		# Classes	# Instâncias	% Classes Únicas	# Classes	# Instâncias	% Classes Únicas
1	1230	991	11059	22.31	863	9193	21.15
2	2436	2105	20186	62.19	1947	15355	64.06
3	2307	1946	12642	72.46	1747	10081	68.12
4	1990	1680	15360	52.62	1550	12267	53.44
5	1191	983	5463	38.30	898	4268	40.94
6	1274	1068	8307	30.21	1007	7222	33.23
7	2023	1683	15911	54.69	1556	12948	55.62
8	882	725	2772	34.06	644	2128	35.10
11	1474	1234	10753	35.93	1134	9016	35.31
15	1432	1175	6326	44.52	1068	4680	44.38
17	1546	1348	9606	36.43	1255	8282	37.60
22	2160	1810	27667	57.35	1652	20727	56.67
24	1496	1207	14436	38.30	1066	10372	37.29
28	1636	1403	15595	48.08	1296	11825	49.58
68	1779	1523	8853	54.10	1421	7206	56.15
88	1780	1510	11073	45.31	1408	8537	48.44
102	1667	1225	3833	58.54	1013	2930	50.00
131	2057	1728	14077	67.72	1594	11340	67.60
154	1146	946	10790	28.04	865	8240	28.54
155	1428	1142	8097	39.29	1005	6128	37.71
164	2122	1802	16898	45.61	1725	14310	46.56
175	1400	1165	9158	34.65	1070	7114	34.48
194	1560	1305	10311	40.67	1215	8221	43.44
210	1201	1013	8679	29.22	960	6952	31.25

Tabela 4.12: Entropia associada às bases de dados

Id.	Janela 1			Janela 5		
	Base Aleatória	Base Real	Ganho	Base Aleatória	Base Real	Ganho
1	13.86	9.04	4.82	13.68	8.93	4.75
2	14.90	9.56	5.34	14.65	9.53	5.12
3	14.18	9.86	4.32	13.94	9.82	4.12
4	14.42	9.45	4.97	14.20	9.39	4.81
5	12.96	9.08	3.88	12.74	8.94	3.80
6	13.36	9.13	4.23	13.21	9.04	4.17
7	14.41	9.35	5.06	14.22	9.28	4.94
8	12.00	8.95	3.05	11.77	8.85	2.92
11	13.80	9.18	4.62	13.62	9.11	4.51
15	13.29	9.12	4.17	13.01	9.08	3.93
17	13.59	9.35	4.24	13.43	9.28	4.15
22	15.43	9.13	6.30	15.14	9.10	6.04
24	14.52	8.80	5.72	14.23	8.71	5.52
28	14.50	8.70	5.80	14.26	8.64	5.62
68	13.59	9.61	3.98	13.38	9.54	3.84
88	14.03	9.30	4.73	13.78	9.23	4.55
102	12.59	9.79	2.80	12.29	9.68	2.61
131	14.29	9.33	4.96	14.07	9.28	4.79
154	13.94	8.15	5.79	13.72	8.11	5.61
155	13.55	8.94	4.61	13.32	8.81	4.51
164	14.45	9.74	4.71	14.27	9.66	4.61
175	13.77	9.05	4.72	13.51	9.04	4.47
194	13.88	9.07	4.81	13.64	9.01	4.63
210	13.59	8.90	4.69	13.38	8.90	4.48

Id.	Janela 10			Janela 15		
	Base Aleatória	Base Real	Ganho	Base Aleatória	Base Real	Ganho
1	13.43	8.77	4.66	13.17	8.59	4.58
2	14.30	9.45	4.85	13.91	9.38	4.53
3	13.63	9.77	3.86	13.30	9.69	3.61
4	13.91	9.32	4.59	13.58	9.26	4.32
5	12.42	8.83	3.59	12.06	8.75	3.31
6	13.02	8.91	4.11	12.82	8.78	4.04
7	13.96	9.22	4.74	13.66	9.18	4.48
8	11.44	8.70	2.74	11.06	8.54	2.52
11	13.39	9.04	4.35	13.14	9.01	4.13
15	12.63	9.04	3.59	12.19	9.08	3.11
17	13.23	9.15	4.08	13.02	9.03	3.99
22	14.76	9.00	5.76	14.34	8.91	5.43
24	13.82	8.50	5.32	13.34	8.39	4.95
28	13.93	8.57	5.36	13.53	8.48	5.05
68	13.11	9.48	3.63	12.81	9.44	3.37
88	13.43	9.21	4.22	13.06	9.23	3.83
102	11.90	9.49	2.41	11.52	9.32	2.20
131	13.78	9.21	4.57	13.47	9.14	4.33
154	13.40	8.08	5.32	13.01	8.14	4.87
155	12.98	8.69	4.29	12.58	8.55	4.03
164	14.04	9.61	4.43	13.80	9.62	4.18
175	13.16	9.06	4.10	12.80	9.08	3.72
194	13.33	8.96	4.37	13.01	8.95	4.06
210	13.08	8.91	4.17	12.76	8.92	3.84

4.2.2 Métricas de Avaliação

A métricas que serão utilizada para avaliar o desempenho do algoritmo baseia-se na análise da matriz de confusão [Gu et al., 2009]. Cada coluna da matriz de confusão representa as instâncias na classe prevista, enquanto cada linha representa as instâncias da classe real. Cada elemento $e_{k,j}$ dessa matriz fornece o número de exemplos, cuja classificação foi C_j e que a verdadeira classe era C_k. Dessa forma, os elementos da diagonal principal dessa matriz correspondem às instâncias que foram corretamente classificadas: número de verdadeiros negativos (VN) e de verdadeiros positivos (VP); enquanto os outros elementos representam os erros cometidos: número de falsos positivos (FP) e falsos negativos (FN). A Tabela 4.14 mostra uma matriz de confusão para uma variável binária.

Tabela 4.14: Matriz de confusão para um classificador binário

		Predição	
		$y = 0$	$y = 1$
Real	$y = 0$	VN	FP
	$y = 1$	FN	VP

A partir dessa matriz da Tabela 4.14 é possível extrair 4 métricas que avaliam, de forma independente, o desempenho sobre as classes positiva e negativa [Castro & Braga, 2011]:

- Taxa de Falsos Positivos:

$$FPr = \frac{FP}{VN + FP} \tag{4.2}$$

- Taxa de Falsos Negativos:

$$FNr = \frac{FN}{VP + FN} \tag{4.3}$$

- Taxa de Verdadeiros Positivos:

$$VPr = \frac{VP}{VP + FN} \tag{4.4}$$

- Taxa de Verdadeiros Negativos:

$$VNr = \frac{VN}{VN + FP} \tag{4.5}$$

Essas taxas derivam outras métricas que são muito utilizadas em aprendizado de máquina, como *Recall* e *Precision* [Melamed et al., 2003]. *Recall* (R) é equivalente à taxa de verdadeiros positivos (VPr) e denota a razão entre o número de instâncias positivas classificadas corretamente e o número total de instâncias positivas originais,

$$R = VPr = \frac{VP}{VP+FN} \tag{4.6}$$

A métrica *Precision* (P) corresponde à razão entre o número de instâncias positivas corretamente classificadas e o número total de instâncias classificadas como positivas,

$$P = \frac{VP}{VP+FP} \tag{4.7}$$

Finalmente, essas duas métricas servem para calcular a *F-measure*, que será utilizada para avaliação dos algoritmos. Ela considera somente o desempenho da classe positiva. *F-measure* é a média harmônica de *precision* e *recall* e alcança seu valor máximo em 1 e o pior em 0:

$$F = 2 \times \frac{P \times R}{P+R} \tag{4.8}$$

4.2.3 Ajuste dos Parâmetros dos Algoritmos

Existem diversas implementações disponíveis para os algoritmos de aprendizado escolhidos. Nesse trabalho, a implementação utilizada foi a da biblioteca escrita na linguagem Python, *scikit-learn*[1].

Cada algoritmo, com exceção do *Naive Bayes* Gaussiano, possui parâmetros que precisam ser ajustados para que os melhores resultados sejam obtidos. A calibração de parâmetros foi realizada empiricamente para todas as base de dados. Foi utilizada a técnica de validação cruzada com 5-*folds* [Kohavi, 1995]. Nessa técnica a base de dados é particionada em k subconjuntos de mesmo tamanho mutualmente exclusivos, em que $k-1$ destes subconjuntos são utilizados para construção do modelo, ou seja, como dados de treinamento, e um subconjunto é utilizado para validação do modelo, como dados de teste. Esse processo é realizado k vezes, cada uma com um subconjunto de teste diferente.

A combinação de parâmetros que obteve na média o melhor resultado em cada base foi considerada para execução dos próximos experimentos.

[1] Disponível em: `http://scikit-learn.org`

Tabela 4.15: Parâmetros selecionados para o algoritmo k-NN

Parâmetro	Valor	Identificador dos Usuários			
		Janela 1	Janela 5	Janela 10	Janela 15
Função de Pesos	Distance	102, 155	3, 4, 15, 22, 88, 102, 131, 154, 155, 210	3, 7, 8, 11, 15, 17, 24, 28, 88, 155, 175	6, 7, 11, 17, 22, 28, 88, 194, 210
Algoritmo	Ball Tree				
Função de Pesos	Distance	1, 3, 4, 5, 8, 15, 17, 68	1, 2, 5, 6, 7, 8, 11, 17, 24, 28, 68, 164, 175, 194	1, 2, 4, 5, 6, 22, 68, 102, 131, 154, 164, 194, 210	1, 2, 3, 4, 5, 8, 15, 24, 68, 102, 131, 154, 155, 164, 175
Algoritmo	KD Tree				
Função de Pesos	Uniform	28, 164, 175	—	—	—
Algoritmo	Ball Tree				
Função de Pesos	Distance	2, 6, 7, 11, 22, 24, 88, 131, 154, 194, 210	—	—	—
Algoritmo	KD Tree				

4.2.3.1 K-Nearest Neighbors

Para o algoritmo de classificação *k-nearest neighbors*, 5 parâmetros precisaram ser ajustados. Os valores selecionados estão expostos na Tabela 4.15.

1. **Número de Vizinhos** a ser considerados: 5, 6, 7, 10, 15. Para todos os usuários e tamanhos de janelas a melhor escolha foi igual a 5.

2. **Função de Pesos** utilizada na predição:

   - **Uniform:** Os pesos são uniformes, todos os pontos na vizinhança possuem pesos iguais.
   - **Distance:** Os pesos dos pontos são o inverso de suas distâncias, nesse caso, pontos que estão próximos do ponto de consulta, terão influência maior que aqueles distantes.

3. **Algoritmo** utilizado para computar os vizinhos mais próximos. O *scikit-learn* fornece dois algoritmos baseados em árvore e que diminuem o tempo de processamento na escolha desses parâmetros: **Ball Tree** [Omohundro, 1989] e **KD Tree** [Bentley, 1975].

4.2.3.2 Naive Bayes Multinomial

O algoritmo *naive-bayes* multinomial possui dois parâmetros a ser ajustados e os resultados encontram-se na Tabela 4.17.

1. **Alpha:** 0, 0.1, 0.2, 0.5, 1, 2, 3. Parâmetro aditivo de suavização.

2. **Ajute à Priori:** *true*, *false*. Se deve aprender a probabilidade da classe previamente ou não. Se *false* é utilizada uma distribuição uniforme.

4.2.3.3 Árvore de Decisão

Os parâmetros que foram calibrados para o algoritmo da árvore de decisão estão descritos a seguir e a Tabela 4.19 mostra os valores selecionados.

1. **Critério:** Função para medir a qualidade de uma partição na árvore

   - **Gini:** mede a impureza do atributo em relação a classe [D'Ambrosio & Tutore, 2011].
   - **Entropia:** mede o ganho de informação.

2. **Número máximo de atributos** a considerar ao realizar a partição:

   sqrt: o número máximo de atributos é $\sqrt{num\_atributos}$.

   log2: o número máximo de atributos é $\log_2 num\_atributos$.

   None: o número máximo de atributos é $num\_atributos$.

3. **Profundidade máxima da árvore:** 5, 10, 15, 20, 30, 40, None, os nós são expandidos até todas as folhas serem puras ou até que todas as folhas contenham menos que o número mínimo de amostras necessário para dividir um nó interno.

4. **Número mínimo de amostras de divisão:** 2, 3, 5, 10, 15, 20, 30, 40, 50. É o número mínimo de amostras necessário para dividir um nó interno. Para todos os usuários e tamanhos de janelas o número mínimo de amostras de divisão escolhido foi 2.

5. **Número mínimo de amostras da folha:** 1, 2, 3, 4, 5, 10, 15, 20. É o número mínimo de amostras necessário para estar em um nó folha. Para todos os usuários e tamanhos de janelas o número mínimo de amostras da folha foi 1.

Tabela 4.17: Parâmetros selecionados para o algoritmo *naive bayes* multinomial

Parâmetro	Valor	Identificador dos Usuários			
		Janela 1	Janela 5	Janela 10	Janela 15
Alpha	0	1, 2, 3, 11, 15, 24, 28, 68, 131, 154	—	—	—
Ajuste à Priori	true				
Alpha	0.1	4, 5, 6, 7, 8, 17, 22, 88, 102, 164, 175, 194, 210	—	88, 102	28
Ajuste à Priori	true				
Alpha	0.5	—	—	154	—
Ajuste à Priori	true				
Alpha	1	—	—	—	11, 102
Ajuste à Priori	true				
Alpha	2	—	—	—	131
Ajuste à Priori	true				
Alpha	3	—	—	131	17, 88
Ajuste à Priori	true				
Alpha	0	—	—	24	8
Ajuste à Priori	false				
Alpha	0.1	—	1, 2, 3, 4, 5, 7, 8, 11, 15, 17, 22, 24, 28, 68, 88, 131, 154, 155, 175, 210	2, 3, 5, 6, 7, 8, 15, 17, 22, 68, 175, 194	1, 5, 7, 15, 24, 68, 155, 164, 175, 194, 210
Ajuste à Priori	false				
Alpha	0.2	—	6, 102, 164, 194	1, 4, 11, 155	2, 4
Ajuste à Priori	false				
Alpha	0.5	—	—	—	3, 22
Ajuste à Priori	false				
Alpha	1	—	—	164	6
Ajuste à Priori	false				

Tabela 4.19: Parâmetros selecionados para o algoritmo árvore de decisão

Parâmetro	Valor	Identificador dos Usuários			
		Janela 1	Janela 5	Janela 10	Janela 15
Critério	gini	1, 2, 3, 4, 5, 6, 7, 8, 11, 15, 17, 22, 24, 28, 68, 88, 102, 131, 155, 164, 175, 194, 210	—	—	—
Num. De Atributos	sqrt				
Pronf. Árvore	None				
Critério	entropy	154	—	—	—
Num. De Atributos	sqrt				
Pronf. Árvore	None				
Critério	gini	—	5, 8, 102, 154	5, 6, 8	3, 8, 154, 210
Num. De Atributos	None				
Pronf. Árvore	None				
Critério	entropy	—	1, 2, 3, 4, 6, 11, 15, 17, 22, 24, 28, 68, 88, 155, 175, 194, 210	1, 3, 4, 7, 11, 15, 17, 22, 24, 28, 68, 88, 102, 131, 155, 164, 175, 194, 210	1, 4, 5, 6, 11, 15, 17, 22, 24, 28, 68, 88, 102, 131, 155, 164, 175, 194
Num. De Atributos	None				
Pronf. Árvore	None				
Critério	entropy	—	7, 131, 164	1, 154	2, 7
Num. De Atributos	None				
Pronf. Árvore	15				

4.2.3.4 SVM

Para o SVM dois parâmetros precisaram de calibração, os valores selecionados estão apresentados na Tabela 4.21.

C: 0.5, 1, 10, 100, 1000. Parâmetro de custo, que controla o equilíbrio entre permitir erros no treino e forçar margens rígidas.

Gamma: 0, 0.001, 0.0001. Parâmetro de controle do kernel RBF.

4.2.4 Avaliação dos Algoritmos

Para verificar o desempenho dos algoritmos e o seu comportamento com a variação do tamanho da janela, cada base de dados foi submetida a uma validação cruzada com 5-*folds*. A métrica de avaliação é o *F-measure*, Equação 4.8.

Tabela 4.21: Parâmetros selecionados para o algoritmo SVM

Parâmetro	Valor	Identificador dos Usuários			
		Janela 1	Janela 5	Janela 10	Janela 15
C	1	17, 154	88, 154, 175, 194, 210	—	—
Gamma	0				
C	10	1, 2, 3, 4, 5, 6, 7, 8, 15, 22, 28, 68, 88, 102, 131, 155, 164, 175, 194, 210	1, 2, 3, 4, 5, 7, 8, 11, 17, 22, 24, 68, 131, 164	4	—
Gamma	0				
C	100	24	—	—	—
Gamma	0				
C	100	—	—	—	88, 102
Gamma	0.0001				
C	100	—	—	8	—
Gamma	0.001				
C	1000	11	6, 15	—	—
Gamma	0				
C	1000	—	—	2, 3, 5, 6, 7, 11, 15, 17, 22, 68, 88, 102, 154, 155, 164, 175, 194, 210	1, 2, 3, 4, 5, 7, 8, 11, 15, 17, 22, 24, 68, 131, 154, 155, 164, 175, 194, 210
Gamma	0.0001				
C	1000	—	28, 102, 155	1, 24, 28, 131	28
Gamma	0.001				

Para cada base, foram reportadas as médias do *F-measure* nos 5-*folds* e são apresentadas nos gráficos das Figuras 4.10, 4.11, 4.12, 4.13 para cada usuário. Adicionalmente, foram calculados os intervalos com 99% de confiança para cada resultado, que também são apresentados nos gráficos. Os rótulo **NBM** é utilizado para *naive-bayes* utilizando a distribuição Multinomial; **NBG**, *naive-bayes* utilizando a distribuição Gaussiana; **KN** para o classificador *k-nearest neighbors*; **SVM** para *support vector machines*; e **AD** para árvores de decisão.

A partir dos gráficos, é possível perceber que o comportamento dos algoritmos são similares para os usuários, principalmente, considerando as versões do *naive-bayes* utilizando as distribuições Gaussiana e Multinomial. Em todo os casos, eles obtiveram os piores resultados, mostrando que para essa modelagem do problema da predição de trajetória de veículos o *naive-bayes* não é um algoritmo apropriado. Esses algoritmos possuem comportamento semelhante, com base no tamanho da janela, predominantemente quanto maior a

janela, melhor a classificação.

Já os classificadores *k-nearest neighbors*, SVM e árvore de decisão obtiveram os melhores resultados, de forma que, para algumas bases eles não diferem estatisticamente. Com relação ao tamanho da janela, verifica-se que o comportamento predominante é o resultado da classificação crescer, alcançar seu pico e decair, à medida que o tamanho aumenta.

Os melhores resultados são mostrados na Tabela 4.23. Ela apresenta, para cada usuário, a configuração cujas classes foram melhor preditas. Os resultados ficaram limitados ao intervalo de 0.60 a 0.85 de sucesso na classificação. Esses resultados podem ser satisfatórios em aplicações de redes veiculares que não sejam de caráter preventivo e emergencial, que necessitem de maior confiabilidade, como, por exemplo, aplicações de entretenimento.

O algoritmo que obteve melhores resultados na maioria foi a árvore de decisão, ganhando para 22 usuários. O SVM foi melhor nas outras 2 bases. O tamanho de janela 5 foi o que mais obteve melhores resultados, para 19 usuários. O tamanho de janela 10 venceu somente para o usuário 1, e o a janela de tamanho 1 venceu para 4 usuários. Com base nesses resultados, verifica-se que para a modelagem proposta a árvore de decisão com janela 5 tem o melhor comportamento e resulta em uma classificação mais acurada. Os usuários 1, 102 e 22 serão analisados com mais detalhes. O usuário 1 foi será analisado por ter sido o único que o melhor resultado foi obtido com tamanho de janela 10 e por obter a melhor predição entre todos os outros. O usuário 102 foi escolhido por ter o pior resultado, apesar de percorrer poucos trechos em comparação com os demais. Por fim, o usuário 22 por ter percorrido o maior número de trechos.

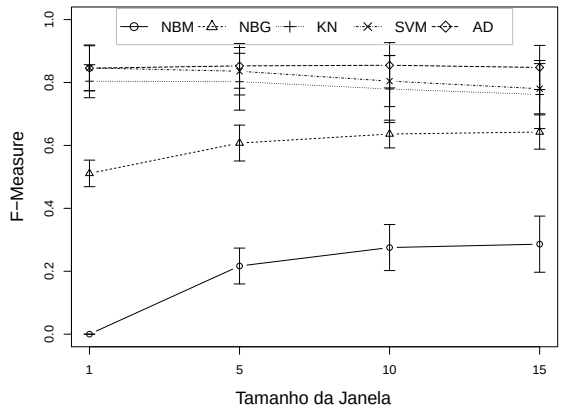
Verifica-se que a predição para o usuário 1 foi a melhor entre todas, com acurácia de 0.85. O usuário 1 foi observado durante 121 dias, em média, realiza 3.81 viagens por dia, e percorre, em média, 12.71 trechos por cada viagem realizada, como mostra a Tabela 4.9. A partir da Tabela 4.10, nota-se que esse usuário é o que possui a menor porcentagem de instâncias classificadas unicamente, cerca de 22.31%. Isso confirma que se o usuário possui um comportamento estável com relação às viagens que realiza, mantendo um padrão no seu trajeto, a predição pode ser realizada mais facilmente. Esse usuário foi o único cujo tamanho de janela 10 obteve as melhores predições.

O usuário 102 obteve o resultado mais baixo, com acurácia de 0.60. Esse usuário foi observado durante 73 dias, realiza, em média, 5.37 viagens por dia, e percorre, em cada viagem, cerca de 4.47 trechos, como mostra a Tabela 4.9. Na Tabela 4.12, é possível identificar que o ganho de informação considerando o comportamento aleatório do usuário e o dado pelo seu histórico é o menor dentre todos os usuários, indicando que esse usuário não mantem um padrão bem definido nas suas rotas. Além disso, a Tabela 4.10 mostra que 58.97% das instâncias da sua base tem classes únicas, o que confirma o comportamento desordenado do usuário.

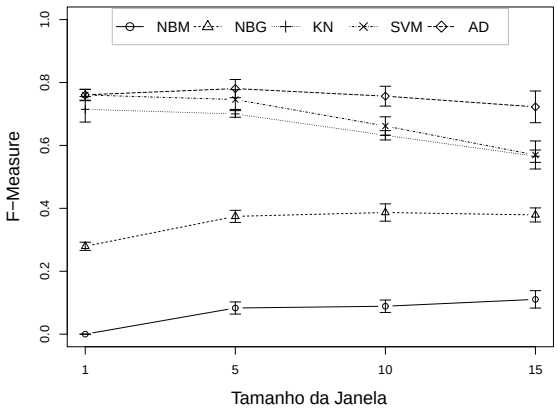
Tabela 4.23: Melhores resultados alcançados para cada usuário

Id.	Algo	Tam. Jan.	Média *F-Measure*	I.C. (99%)
1	AD	10	0.85	(0.78, 0.93)
2	AD	5	0.78	(0.75, 0.81)
3	AD	1	0.74	(0.70, 0.79)
4	AD	5	0.82	(0.79, 0.86)
5	AD	5	0.72	(0.68, 0.76)
6	AD	5	0.76	(0.65, 0.88)
7	AD	5	0.81	(0.74, 0.88)
8	AD	1	0.75	(0.61, 0.88)
11	AD	5	0.78	(0.69, 0.87)
15	AD	5	0.73	(0.60, 0.86)
17	AD	5	0.66	(0.57, 0.75)
22	AD	5	0.83	(0.79, 0.88)
24	AD	5	0.81	(0.74, 0.88)
28	AD	5	0.83	(0.76, 0.89)
68	SVM	1	0.74	(0.68, 0.81)
88	AD	5	0.68	(0.65, 0.71)
102	SVM	1	0.60	(0.51, 0.69)
131	AD	5	0.74	(0.66, 0.82)
154	AD	5	0.78	(0.72, 0.84)
155	AD	5	0.82	(0.73, 0.91)
164	AD	5	0.71	(0.67, 0.74)
175	AD	5	0.73	(0.66, 0.79)
194	AD	5	0.74	(0.68, 0.79)
210	AD	5	0.79	(0.73, 0.85)

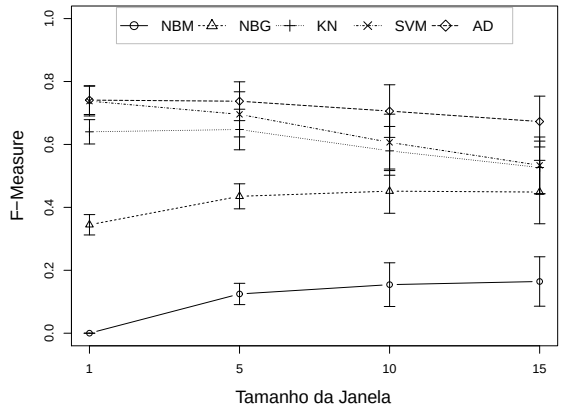
O usuário 22 destaca-se por corresponder àquele que possui um maior número de registros considerados. Ele foi observado durante 208 dias, com uma média de 10.23 viagens por dia e percorrendo, em média, 4.40 trechos por viagens. Esse obteve *f-measure* na classificação de 0.83. No entanto, observando a Tabela 4.12, nota-se que o usuário obteve o maior ganho entre todos os outros, o que indica que apesar de possuir muitas classes consideradas uma única vez, seu comportamento geral segue um padrão mais bem definido.



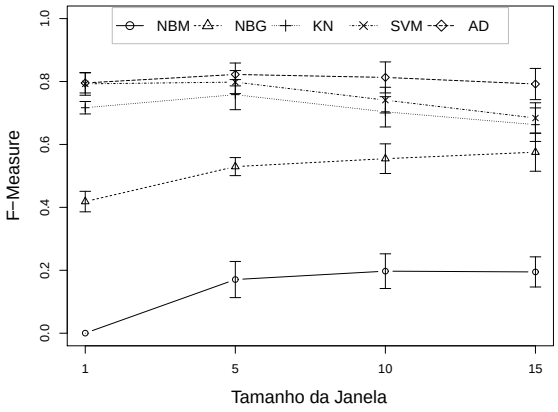
(a) Usuário 1



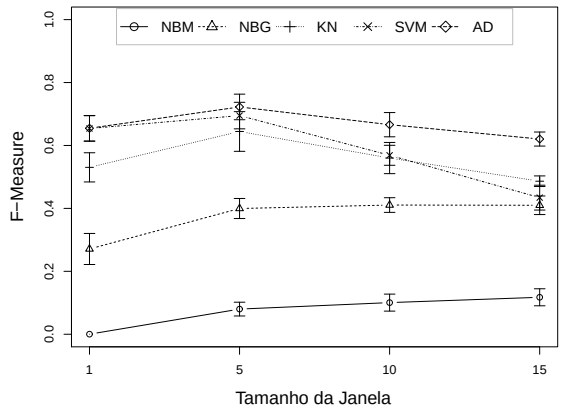
(b) Usuário 2



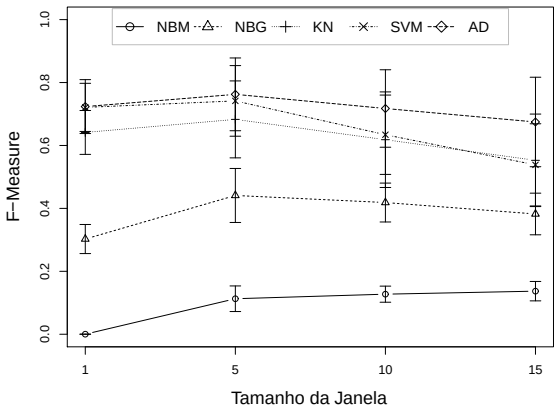
(c) Usuário 3



(d) Usuário 4

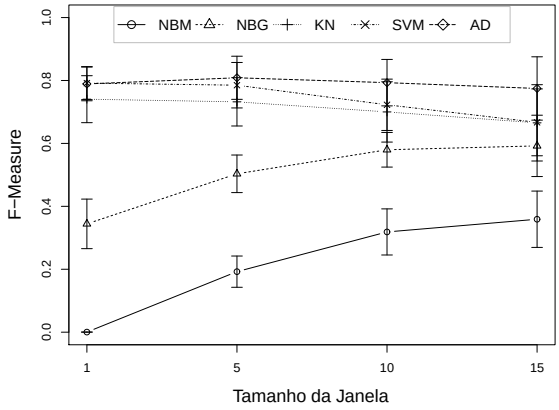


(e) Usuário 5

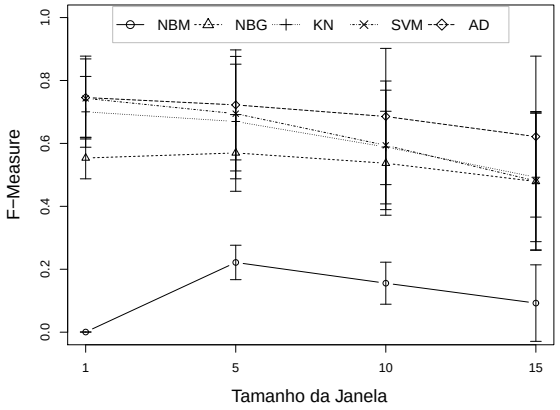


(f) Usuário 6

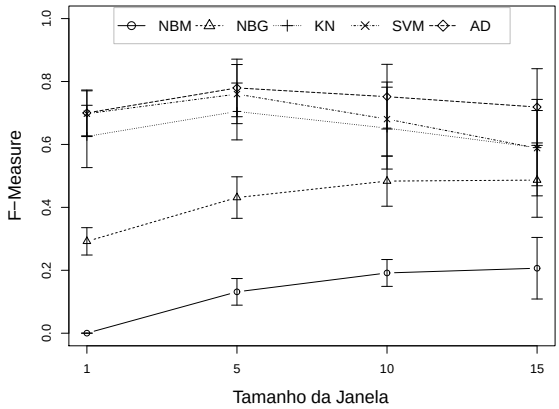
Figura 4.10: Média do *F-measure* para a classificação nos 5-*folds* 1, 2, 3, 4, 5 e 6



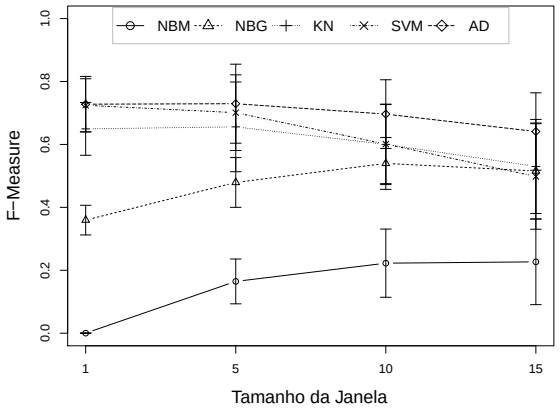
(a) Usuário 7



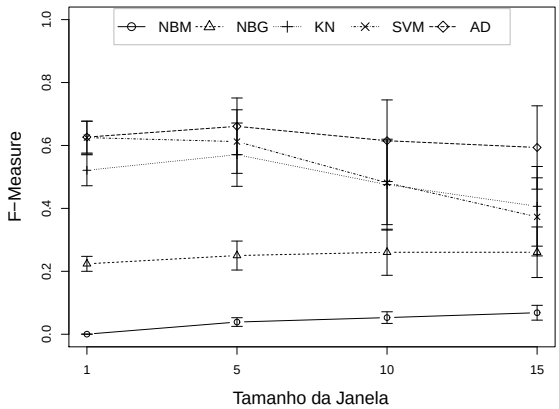
(b) Usuário 8



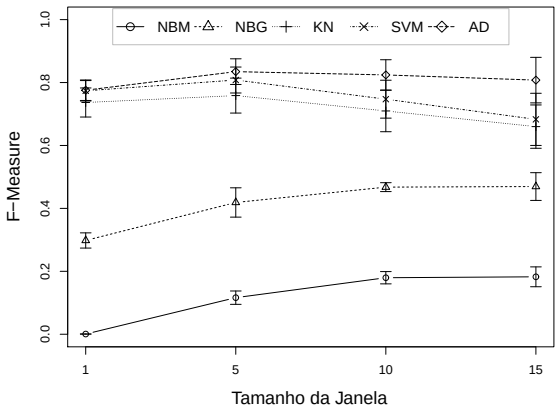
(c) Usuário 11



(d) Usuário 15

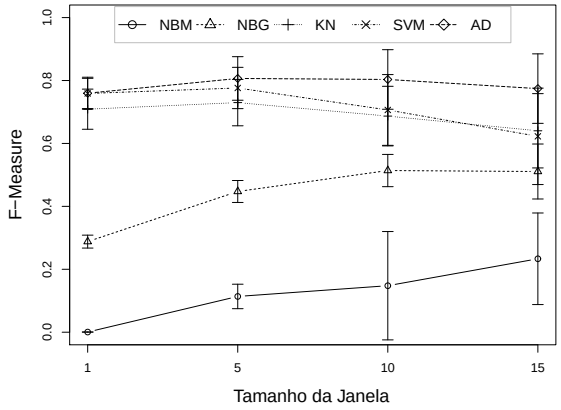


(e) Usuário 17

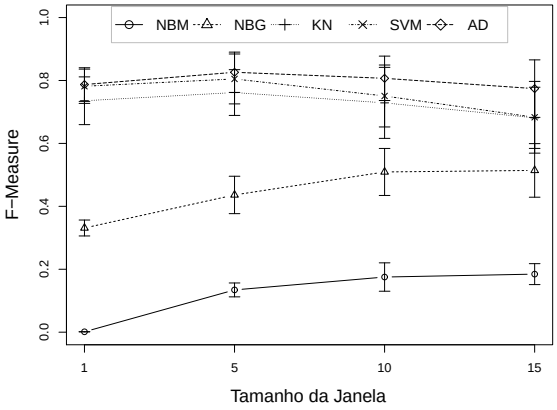


(f) Usuário 22

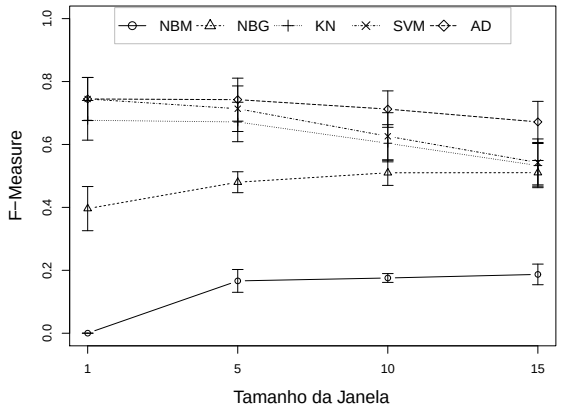
Figura 4.11: Média do *F-measure* para a classificação nos 5-*folds* 7, 8, 11, 15, 17 e 22



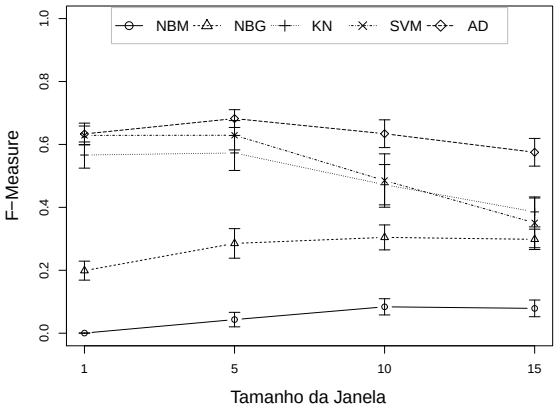
(a) Usuário 24



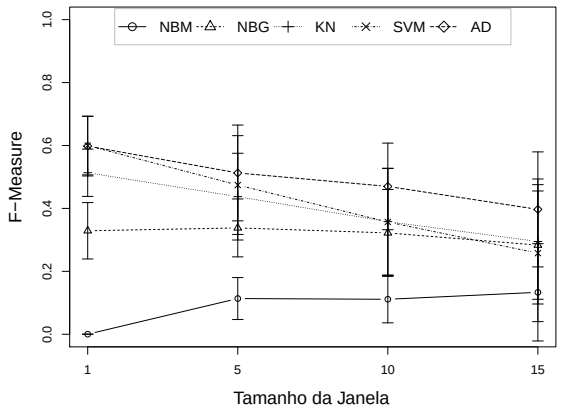
(b) Usuário 28



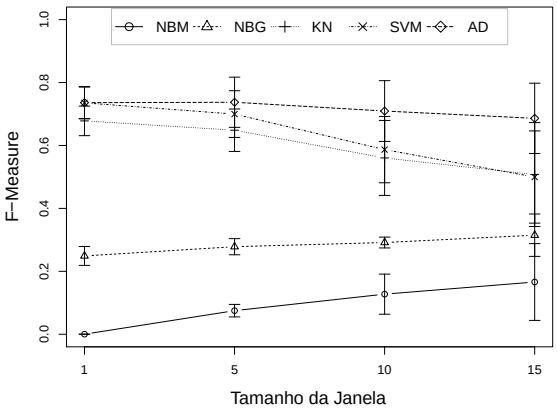
(c) Usuário 68



(d) Usuário 88

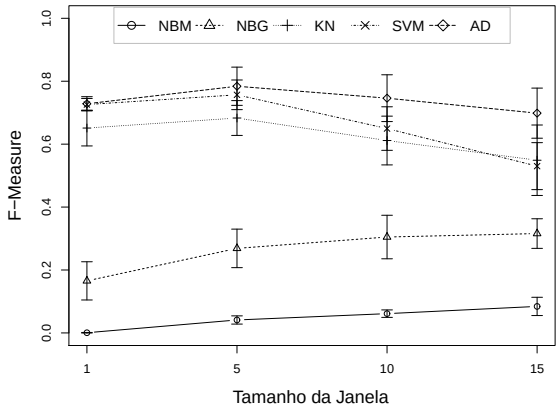


(e) Usuário 102

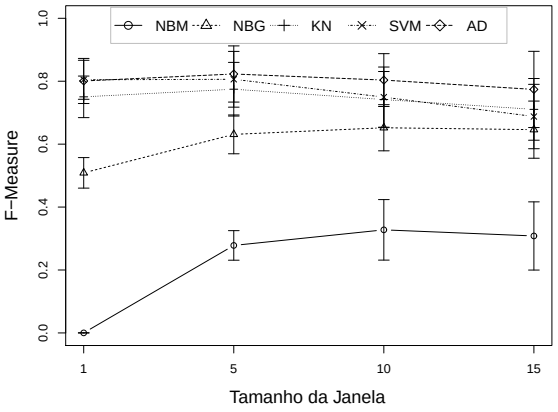


(f) Usuário 131

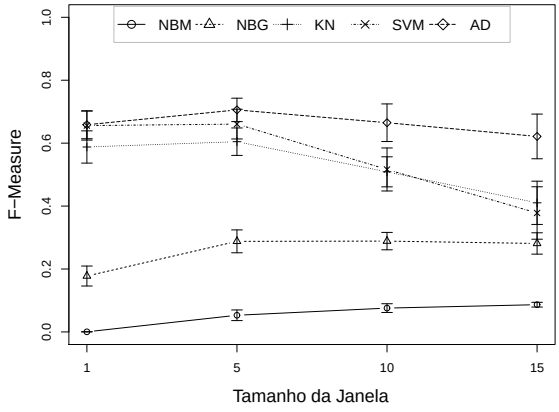
Figura 4.12: Média do *F-measure* para a classificação nos 5-*folds* 24, 28, 68, 88, 102 e 131



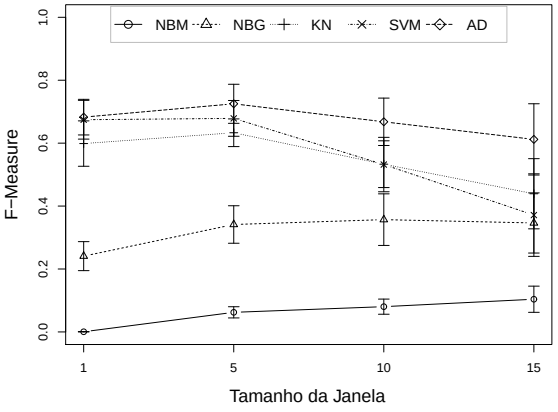
(a) Usuário 154



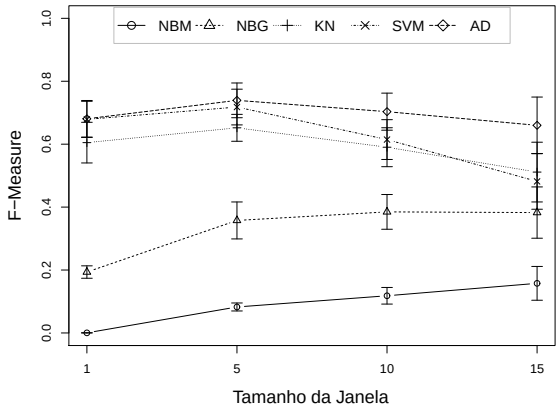
(b) Usuário 155



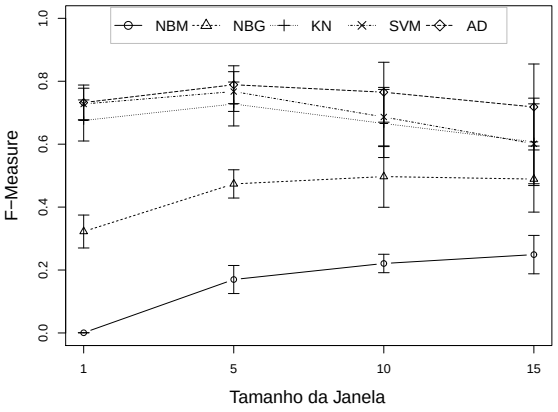
(c) Usuário 164



(d) Usuário 175



(e) Usuário 194



(f) Usuário 210

Figura 4.13: Média do *F-measure* para a classificação nos 5-*folds* de cada algoritmo dos usuários 154, 155, 164, 175, 194 e 210

Capítulo 5

Conclusão

Esse trabalho apresentou uma abordagem baseada em Algoritmo Genético (AG) para o problema de posicionamento de pontos de acesso em redes veiculares. Pontos de acesso são componentes fundamentais para ajudar a disseminação de informações. Outro estudo apresentado, correspondeu à predição de trajetória de veículos baseada no histórico do motorista.

A distribuição de pontos de acesso na região foi modelada utilizando uma variação do problema de cobertura de conjunto, Problema da Máxima Cobertura com Limite de Tempo (PMCLT) . Um AG foi proposto para o problema, seu procedimento de inicialização foi melhorado com dados provenientes de uma busca gulosa e alguns operadores foram propostos utilizando informações provenientes do próprio problema. Experimentos foram realizados combinando as diferentes inicializações de população, os diversos operadores genéticos e realizando variações no número de pontos de acesso disponíveis e o tempo mínimo de recebimento da informação.

Com pesquisa global realizada pelo AG, foi possível encontrar posições para os Pontos de Acesso (PAs) que levam a uma melhor cobertura de veículos do que os obtidos pela abordagem gulosa. Os resultados mostraram que o AG obteve ganhos de até 20 pontos percentuais.

Para o problema da predição da trajetória de veículos, foi proposta uma modelagem que transpõe os dados de maneira que algoritmos de classificação de aprendizado de máquina podem ser aplicados para realizar as predições. Para essa transposição os conceitos de trechos, trajetórias, histórico do usuário e janelas deslizantes foram definidos. Além disso, foi apresentada uma caracterização dos usuários disponíveis na base de dados utilizada para avaliação do método. Por fim, foram realizados experimentos variando o tamanho da janela e aplicou-se os algoritmos *naive bayes*, *k-nearest neighbors*, SVM e árvore de decisão, com o objetivo de verificar o comportamento desses algoritmos com a modelagem proposta.

Os resultados demonstraram que a janela de tamanho cinco alcança os melhores re-

sultados na maioria das bases, os algoritmos *k-nearest neighbors*, SVM e árvore de decisão possuem comportamento semelhante, no entanto, a árvore de decisão consegue o melhor resultado com 0.85 de pontuação para a predição na métrica de avaliação utilizada. Apesar dos resultados promissores, eles são exploratórios, o que faz necessário um estudo mais aprofundado e uma avaliação mais rígida do problema.

Como trabalho futuro, o AG pode ser personalizado utilizando funções multi-objetivos. Além disso, é aceitável utilizar o AG com outras técnicas para alcançar resultados melhores, mais rapidamente. O AG pode ser adaptado também para fornecer o número mínimo de pontos de acesso necessários para alcançar uma cobertura mínima. Outro aspecto interessante a ser estudado é o paralelismo da solução, aumentando o desempenho do algoritmo. Em adicional, é desejável aplicar o AG para cenários maiores, como a cidade de Belo Horizonte e com dados de tráfego real.

Já para o problema da predição de trajetória, pode-se aplicar outros algoritmos de aprendizado que sejam mais adequados ao problema, como algoritmos próprios para prever sequências, por exemplo. Além disso, é interessante desenvolver um algoritmo próprio baseado nas características desse problema. Considera-se importante também, realizar um estudo do quanto é necessário coletar de dados para a previsão alcançar uma acurácia mínima desejável, ou seja, quantos dias o usuário precisa ser observado para seus trechos serem preditos acuradamente e em qual intervalo de tempo o seu histórico precisa ser atualizado. Faz-se necessário também aplicar outros algoritmos propostos na literatura que resolvem o mesmo problema para efeito de comparação de resultados.

Uma possível aplicação que pode utilizar a junção de alocação dos PAs e a predição de trajetória dos veículos é a disseminação de informações personalizadas para motoristas. Com os PAs otimamente posicionados, disseminando propagandas de entretenimento ou produtos/estabelecimentos em geral e com um mecanismo capaz prever os próximos caminhos que o motorista fará, essas informações disseminadas podem ser filtradas e transmitidas de acordo com a previsão de cada usuário. Desenvolver e simular essa aplicação combinando as melhores soluções para os dois problemas está também entre trabalhos futuros.

Alguns resultados do problema de montagem de infraestrutura estão publicados em Cavalcante et al. [2012a,b].

Referências Bibliográficas

Axhausen, K. W.; Schönfelder, S.; Wolf, J.; Oliveira, M. & Samaga, U. (2003). 80 weeks of gps-traces: approaches to enriching the trip information. Relatório técnico.

Barba, C. T.; Mateos, M. Á.; Soto, P. R.; Mezher, A. M. & Aguilar-Igartua, M. (2012). Smart city for VANETs using warning messages, traffic statistics and intelligent traffic lights. Em *Intelligent Vehicles Symposium*, pp. 902–907.

Beasley, J. E. & Chu, P. C. (1996). A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2):392--404.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509--517.

Boukerche, A.; Oliveira, H. A. B. F.; Nakamura, E. F. & Loureiro, A. A. F. (2008). Vehicular ad hoc networks: A new challenge for localization-based systems. *Comput. Commun.*, 31:2838–2849.

Caruana, R. & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. Em *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pp. 161--168, New York, NY, USA. ACM.

Castro, C. L. d. & Braga, A. P. (2011). Aprendizado supervisionado com conjuntos de dados desbalanceados. *Sociedade Brasileira de Automatica: Controle & Automação*, 22:441 – 466.

Cavalcante, E. S.; Aquino, A. L.; Pappa, G. L. & Loureiro, A. A. (2012a). Roadside unit deployment for information dissemination in a vanet: an evolutionary approach. Em *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion*, GECCO Companion '12, pp. 27--34, New York, NY, USA. ACM.

Cavalcante, E. S.; Cavalcante, L. P. A.; Aquino, A. L. L.; Pappa, G. L. & Loureiro, A. A. F. (2012b). Uma abordagem evolutiva para posicionamento de pontos de disseminação em vanets. Em *Anais do XLIV Simpósio Brasileiro de Pesquisa Operacional*, pp. 2791--2802.

Caveney, D. (2009). Vehicular path prediction for cooperative driving applications through digital map and dynamic vehicle model fusion. Em *IEEE Vehicular Technology Conference Fall*, pp. 1–5.

Cover, T. & Hart, P. (1967). Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27.

Cristianini, N. & Shawe-Taylor, J. (2000). *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge University Press, New York, NY, USA.

D'Ambrosio, A. & Tutore, V. (2011). Conditional classification trees by weighting the gini impurity measure. Em Ingrassia, S.; Rocci, R. & Vichi, M., editores, *New Perspectives in Statistical Modeling and Data Analysis*, Studies in Classification, Data Analysis, and Knowledge Organization, pp. 273–280. Springer Berlin Heidelberg.

Daqi, G. & Tao, Z. (2007). Support vector machine classifiers using rbf kernels with clustering-based centers and widths. Em *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pp. 2971–2976.

Dimitrakopoulos, G. & Demestichas, P. (2010). Intelligent transportation systems. *Vehicular Technology Magazine, IEEE*, 5(1):77–84.

Eisner, J.; Funke, S.; Herbst, A.; Spillner, A. & Storandt, S. (2011). Algorithms for matching and predicting trajectories. Em Müller-Hannemann, M. & Werneck, R. F. F., editores, *ALENEX*, pp. 84–95. SIAM.

Faezipour, M.; Nourani, M.; Saeed, A. & Addepalli, S. (2012). Progress and challenges in intelligent vehicle area networks. *Commun. ACM*, 55(2):90--100.

Fallah, Y. P.; Huang, C.; Sengupta, R. & Krishnan, H. (2010). Design of cooperative vehicle safety systems based on tight coupling of communication, computing and physical vehicle dynamics. Em *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, p. 159, New York, New York, USA. ACM Press.

Forrest, S. (1993). Genetic algorithms: principles of natural selection applied to computation. *Science*, 261(5123):872–878.

Frejinger, E. (2008). *Route Choice Analysis: Data, Models, Algorithms and Applications*. Tese de doutorado, EPFL, Lausanne.

Friedman, N.; Geiger, D. & Goldszmidt, M. (1997). Bayesian network classifiers. *Mach. Learn.*, 29(2-3):131--163.

Giannotti, F.; Nanni, M.; Pinelli, F. & Pedreschi, D. (2007). Trajectory pattern mining. Em *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pp. 330--339, New York, NY, USA. ACM.

Gray, R. M. (1990). *Entropy and information theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Gu, Q.; Zhu, L. & Cai, Z. (2009). Evaluation measures of the classification performance of imbalanced data sets. Em Cai, Z.; Li, Z.; Kang, Z. & Liu, Y., editores, *Computational Intelligence and Intelligent Systems*, volume 51 of *Communications in Computer and Information Science*, pp. 461–471. Springer Berlin Heidelberg.

Habib, S. & Safar, M. (2007). Sensitivity study of sensors' coverage within wireless sensor networks. Em *Proceedings of 16th International Conference on Computer Communications and Networks*, pp. 876--881.

Hartenstein, H. & Laberteaux, K. (2008). A tutorial survey on vehicular ad hoc networks. *Communications Magazine, IEEE*, 46(6):164–171.

Hermes, C.; Einhaus, J.; Hahn, M.; Wohler, C. & Kummert, F. (2010). Vehicle tracking and motion prediction in complex urban scenarios. Em *2010 IEEE Intelligent Vehicles Symposium*, pp. 26--33. IEEE.

Hochbaum, D. S. (1996). *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company.

Huang, C.-F. & Tseng, Y.-C. (2003). The coverage problem in a wireless sensor network. Em *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, WSNA '03, pp. 115--121, New York, NY, USA. ACM.

Jain, R. K. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, 1 edição.

Jia, J.; Chen, J.; Chang, G.-R. & Wen, Y.-Y. (2008). Efficient cover set selection in wireless sensor networks. *Acta Automatica Sinica*, 34(9):1157--1162.

Kchiche, A. & Kamoun, F. (2009). Access-points deployment for vehicular networks based on group centrality. Em *3rd International Conference on New Technologies, Mobility and Security*, pp. 207--2012, Cairo, Egypt.

Kchiche, A. & Kamoun, F. (2010). Centrality-based access-points deployment for vehicular networks. Em *17th International Conference on Telecommunications (ICT)*, pp. 700--706. IEEE.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. Em *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pp. 1137--1143, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. Em *Proceedings of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pp. 3--24, Amsterdam, The Netherlands, The Netherlands. IOS Press.

Krozel, J. & Andrisani, D. (1993). Intelligent path prediction for vehicular travel. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(2):478--487.

Lee, U.; Cheung, R. & Gerla, M. (2009). *Emerging Vehicular Applications*, capítulo 8, pp. 207 -- 215. Chapman and Hall/CRC.

Lefevre, S.; Ibanez-Guzman, J. & Laugier, C. (2011). Context-based estimation of driver intent at road intersections. Em *IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems*, pp. 67--72.

Li, X.-Y.; Wan, P.-J. & Frieder, O. (2003). Coverage in wireless ad hoc sensor networks. *IEEE Transactions on Computers*, 52(6):753--763.

Lochert, C.; Scheuermann, B.; Wewetzer, C.; Luebke, A. & Mauve, M. (2008). Data aggregation and roadside unit placement for a vanet traffic information system. Em *5th ACM International Workshop on VehiculAr Inter-NETworking*, San Francisco, California, USA.

Lytrivis, P.; Thomaidis, G.; Tsogas, M. & Amditis, A. (2011). An advanced cooperative path prediction algorithm for safety applications in vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):669--679.

Macedo, D. F.; de Oliveira, S.; Teixeira, F. A.; Aquino, A. L. L. & Oliveira, R. R. (2012). $(CIA)^2$-ITS: Interconnecting mobile and ubiquitous devices for intelligent transportation systems. Em *IEEE Pervasive Computing and Communication*, Lugano, Switzerland.

Meguerdichian, S.; Koushanfar, F.; Potkonjak, M. & Srivastava, M. (2001). Coverage problems in wireless ad-hoc sensor networks. Em *Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3 of *INFOCOM 2001*, pp. 1380--1387. IEEE.

Melamed, I. D.; Green, R. & Turian, J. P. (2003). Precision and recall of machine translation. Em *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003–short papers - Volume 2*, NAACL-Short '03, pp. 61--63, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miller, B. L. & Goldberg, D. E. (1996). Genetic algorithms, selection schemes, and the varying effects of noise. *Evolutionary Computation*, 4(2):113--131.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA.

Morzy, M. (2007). Mining frequent trajectories of moving objects for location prediction. Em Perner, P., editor, *Machine Learning and Data Mining in Pattern Recognition*, volume 4571 of *Lecture Notes in Computer Science*, pp. 667–680. Springer Berlin Heidelberg.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. The MIT Press.

Nagel, K. (2012). Eth zurich traces. On line: `http://www.lst.inf.ethz.ch/research/ad-hoc/car-traces`.

Naumov, V.; Baumann, R. & Gross, T. (2006). An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces. Em *7th ACM international symposium on Mobile ad hoc networking and computing*, Florence, Italy.

Omohundro, S. M. (1989). Five balltree construction algorithms. Relatório técnico, International Computer Science Institute.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M. & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825--2830.

Quinlan, J. R. (1996). Learning decision tree classifiers. *ACM Comput. Surv.*, 28(1):71--72.

Ramos, H.; Boukerche, A.; Pazzi, R.; Frery, A. & Loureiro, A. (2012). Cooperative target tracking in vehicular sensor networks. *Wireless Communications, IEEE*, 19(5):66–73.

Rosi, U.; Hyder, C. & hoon Kim, T. (2008). A novel approach for infrastructure deployment for vanet. Em *Second International Conference on Future Generation Communication and Networking*, volume 1 of *FGCN '08*, pp. 234--238.

Shan, M.; Worrall, S. & Nebot, E. (2011). Long term vehicle motion prediction and tracking in large environments. Em *14th International IEEE Annual Conference on Intelligent Transportation Systems*, pp. 1978--1983, Washington, DC, USA.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3--55.

Sou, S. L. (2010). A power-saving model for roadside unit deployment in vehicular networks. *IEEE Communications Letters*, 14(7):623--625.

Triplett, B. I.; Klein, D. J. & Morgansen, K. A. (2009). Cooperative estimation for coordinated target tracking in a cluttered environment. *Mobile Networks and Applications*, 14(3):336--349.

Trullols, O.; Fiore, M.; Casetti, C.; Chiasserini, C. & Ordinas, J. B. (2010). Planning roadside infrastructure for information dissemination in intelligent transportation systems. *Computer Communications*, 33(4):432 -- 442.

Vegni, A. M.; Biagi, M. & Cusani, R. (2013). *Smart Vehicles, Technologies and Main Applications in Vehicular Ad hoc Networks*, capítulo 1. InTech.

Wang, Y. & Li, F. (2009). Vehicular ad hoc networks. Em Misra, S.; Woungang, I. & Chandra Misra, S., editores, *Guide to Wireless Ad Hoc Networks*, Computer Communications and Networks, pp. 503–525. Springer London.

Yao, J.; Balaei, A. T.; Hassan, M.; Alam, N. & Dempster, A. G. (2011). Improving cooperative positioning for vehicular networks. *IEEE Transactions on Vehicular Technology*, 60(6):2810--2823.

Yousefi, S.; Mousavi, M. & Fathy, M. (2006). Vehicular ad hoc networks (VANETs): Challenges and perspectives. Em *6th International Conference on ITS Telecommunications Proceedings*, pp. 761--766, Chengdu, China.