MAURÍCIO RANGEL GUIMARÃES SERRA

APLICAÇÕES DE APRENDIZAGEM POR REFORÇO EM CONTROLE DE TRÁFEGO VEICULAR URBANO

FLORIANÓPOLIS
2004

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

APLICAÇÕES DE APRENDIZAGEM POR REFORÇO EM CONTROLE DE TRÁFEGO VEICULAR URBANO

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a obtenção
do grau de Mestre em Engenharia Elétrica.

MAURÍCIO RANGEL GUIMARÃES SERRA

Florianópolis, Maio de 2004.

APLICAÇÕES DE APRENDIZAGEM POR REFORÇO EM CONTROLE DE TRÁFEGO VEICULAR URBANO

MAURÍCIO RANGEL GUIMARÃES SERRA

"Esta Dissertação foi julgada adequada para a obtenção do título de **Mestre em Engenharia Elétrica**, área de concentração em **Controle, Automação e Informática Industrial**, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina."

Prof. Eduardo Camponogara, Ph.D.
Orientador

Prof. Jefferson Luiz Brum Marques, Ph.D.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Prof. Eduardo Camponogara, Ph.D.
Presidente

Prof. Werner Kraus Júnior, Ph.D.

Profa. Ana Lúcia C. Bazzan, Ph.D.

Prof. Ricardo J. Rabelo, Dr.

Prof. Jomi Fred Hübner, Dr.

Dedico esta dissertação aos meus pais, Montgomery Serra e Tereza Serra. Tendo a certeza de que mesmo distantes estamos sempre unidos pelo amor.

AGRADECIMENTOS

Aos meus pais pelo grande apoio, reconhecimento e incentivos que me deram em mais uma etapa da minha caminhada.

Aos meus tios Jenner Serra, Maria Dulce Serra, Maria Liz Serra, Maria José Guimarães e Juraci Guimarães que me deram um grande apoio pessoal e incentivos para continuar no caminho certo.

Ao meu orientador, prof. Eduardo Camponogara, pelos ensinamentos, orientação, apoio moral, dedicação e amizade durante estes dois anos de pós-graduação na UFSC. Parte do sucesso desse trabalho é devido à sua colaboração.

Ao meu co-orientador, prof. Werner Kraus Jr., pelos conselhos, amizade, orientação e ajuda para a conclusão deste trabalho.

Aos membros da banca pelas ótimas contribuições na geração da versão final dessa dissertação.

À prof[a]. Lenise Grando Goldner pelas orientações e ajuda indispensáveis nos tópicos referentes à Engenharia de Tráfego.

Aos professores da PPGEEL que acrescentaram um pouco mais de conhecimento à minha vida profissional.

À terapeuta Aymara Penna pelo apoio e ajuda incomensurável nos momentos difíceis.

Aos grandes amigos que fiz dentro do mestrado que compartilharam comigo, direta ou indiretamente, este período de muito trabalho, alegrias e tristezas. Em especial aos amigos Adriana Postal, Allan Magri, André Leal, Daniel Lopes, Eduardo Cambruzzi, Georges Bruel e Marcos Linhares.

Aos integrantes do Projeto SINCMobil que estiveram comigo, me ajudando, motivando e apoiando no desenvolvimento desta dissertação. Em especial aos amigos Karen Farfan, Flávio Cuareli, Ricardo Schmidt, Rodrigo Carlson, Rodrigo Berti e Silvia Galvão.

À CAPES pela concessão de uma bolsa de mestrado, que foi de extrema importância para a realização deste trabalho de dissertação.

A todos que contribuíram de certa forma para a conclusão deste documento.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia Elétrica.

APLICAÇÕES DE APRENDIZAGEM POR REFORÇO EM CONTROLE DE TRÁFEGO VEICULAR URBANO

MAURÍCIO RANGEL GUIMARÃES SERRA

Maio/2004

Orientador: Prof. Eduardo Camponogara, Ph.D.
Co-Orientador: Prof. Werner Kraus Júnior, Ph.D.
Área de Concentração: Controle, Automação e Informática Industrial.
Palavras-chave: Controle de Tráfego, Inteligência Artificial, Otimização, Aprendizagem por Reforço, Aprendizagem de Máquina.
Número de Páginas: xii + 83

Essa dissertação investiga a aplicação de técnicas de aprendizagem por reforço, mais precisamente o uso do algoritmo *Q-learning* distribuído, como uma nova ferramenta de controle para o tráfego veicular urbano a custos menos onerosos que os apresentados pelos métodos de controle de tráfego responsivo e adaptativo, que necessitam de dispositivos complexos, além da dependência de especialistas para operá-los. Visando melhorar o desempenho das redes de tráfego, esse trabalho sugere o desenvolvimento e implementação de agentes inteligentes distribuídos como uma forma de controlar o fluxo da via, sendo a tarefa modelada como um jogo estocástico, onde múltiplos agentes distribuídos sobre a rede, cada um com uma visão parcial do estado da mesma buscam resolver os seus problemas locais, dando origem a um conjunto de problemas de aprendizagem por reforço (um para cada agente controlador). Esses agentes tentam através de experiências adquiridas maximizar seus ganhos com suas ações, ou seja, tentam minimizar o número médio de veículos esperando na fila nas intersecções que atuam, obtendo assim um desempenho eficaz na via, minimizando os congestionamentos e o tempo de viagem sob a rede em estudo. O desempenho é avaliado através de simulações computacionais realizadas com um simulador protótipo para jogos dinâmico distribuídos, projetado especificamente para o domínio de redes de tráfego, onde nele foi modelada uma sub-rede de tráfego da cidade de Florianópolis, sendo os resultados das diferentes políticas utilizadas no trabalho analisados segundo variações de densidades de fluxos na rede (5% até 75% da sua capacidade). Os resultados das simulações mostram que o algoritmo proposto apresenta um desempenho superior aos induzidos pelas políticas aleatória uniforme e de melhor esforço, indicando assim a tecnologia de aprendizagem por reforço como uma possível alternativa no uso do controle de tráfego veicular urbano.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

APPLICATIONS OF REINFORCEMENT LEARNING IN URBAN TRAFFIC CONTROL

MAURÍCIO RANGEL GUIMARÃES SERRA

May/2004

Advisor: Prof. Eduardo Camponogara, Ph.D.
Co-Advisor: Prof. Werner Kraus Júnior, Ph.D.
Area of Concentration: Control, Industrial Automation.
Key words: Traffic Control, Artificial Intelligence, Optimization, Reinforcement Learning, Machine Learning.
Number of Pages: xii + 83

This dissertation investigates the application of reinforcement-learning techniques, specifically a modified distributed *Q-learning* algorithm, in urban traffic control as an alternative to the costly in-place control technology: traffic-responsive and traffic-adaptive control, which demand costly networks of sensors and the periodic intervention of traffic control engineers. To improve the performance of a traffic network this dissertation suggests the modelation of the traffic network control problem as a stochastic game, distributed among distributed control agents, each with a partial view of the traffic network, that seek the best control responses by learning from their past experience working with the local network conditions. The avaliation of this method was performed in a simulator of stochastic games arising from the distributed control of traffic networks and the test-bed was a stochastic game modelled from a section of the city of Florianópolis' traffic network, taking into account statistics provided by the city's road and transportation department. The performance of the proposed algorithm was compared to the performance attained by other control strategies, providing evidence that the proposed algorithms can outperform standard control algorithms. This indicates that the reinforcement-learning technology is a possible alternative in the traffic network control.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

"A vida do homem é como um jogo de dados; se você não consegue a jogada que esperava, pode mostrar sua habilidade tirando o máximo da jogada que conseguiu". (Terêncio)

1.1 Introdução e Motivação

As grandes cidades do Brasil assim como as dos demais países em desenvolvimento apresentam graves problemas de transporte e qualidade de vida, como queda da mobilidade e acessibilidade, degradação das condições ambientais, atrasos desnecessários e altos índices de acidentes de trânsito. Não é por menos, pois vários são os problemas associados ao tráfego veicular urbano, onde podemos destacar o sistema de sinalização que pode ser mais ou menos adequado e/ou eficaz; o aumento do volume de veículos no tráfego que cresceu cerca de 1080% (GEIPOT) nos últimos 30 anos; a administração da engenharia de tráfego; a pavimentação das vias públicas; a legislação de trânsito; o sistema de transporte coletivo e a conduta dos que trafegam pela cidade (motoristas e pedestres) [30].

A partir de tal analogia podemos perguntar. É possível um outro comportamento no trânsito?

É difícil responder a esta pergunta, pois o trânsito é uma resultante complexa de inúmeros fatores complexos em si por sua vez. Mas é consenso ao menos entre os especialistas, de que não bastaria diminuir a quantidade de veículos nos espaços urbanos para melhorar o trânsito, tampouco apenas uma questão de engenharia de tráfego. Por outro lado, há uma convergência interessante entre especialistas e o senso-comum de que um sistema de transporte coletivo eficaz melhoraria o desempenho, assim como uma sinalização suficiente e adequada com bons ajustes dos tempos nos controladores de tráfego, como por exemplo, a adoção de sistemas automatizados que efetuam contagens em tempo real. Mas, a adoção desses sistemas em nossa realidade, esbarra em altos custos que impedem a adoção em larga escala, além da inexistência de um padrão de comunicação nesse setor, criando assim um monopólio dos fornecedores que impossibilitam a interoperabilidade dos equipamentos.

Acreditando que técnicas de aprendizagem de máquina possam desempenhar um papel relevante no aumento da eficácia das tecnologias de controle de tráfego, esse trabalho se propõe a investigar a possibilidade de se gerenciar o tráfego veicular urbano através de agentes inteligentes distribuídos, mais especificamente, modelar a tarefa de operar uma rede de tráfego como um jogo estocástico distribuído. As oportunidades para o desenvolvimento dessa técnica são enormes, variando desde a concepção de modelos inteligentes de predição do fluxo de tráfego, passando pelo planejamento de redes de tráfego, até o controle inteligente em tempo real.

1.2 Objetivos e Contribuições

O objetivo dessa dissertação é investigar o potencial da modelagem das redes de tráfego como um jogo estocástico distribuído e poder provar que o desenvolvimento de uma estratégia de controle responsivo usando um algoritmo de aprendizagem pode apresentar um desempenho superior as técnicas de controle aleatório uniforme e de melhor esforço. Fornecendo assim uma nova visão ao problema de gerenciamento de tráfego.

Os sistemas responsivos de controle de sinais de tráfego permitem fazer um melhor uso da capacidade das vias através de ajustes das configurações dos semáforos, baseados em estimativas de desempenho e condições do tráfego obtidas através do uso da teoria de filas. Com esse pensamento de um melhor desempenho do tráfego nas vias urbanas, é implementado nessa dissertação um algoritmo que busca através do aprendizado a partir de experiências passadas (estimativas apresentadas em estados sucessivos) uma otimização.

No caso do algoritmo implementado é adotado a quantidade média do número de veículos que se encontram nas intesecções semafóricas como parâmetro a ser melhorado, onde o algoritmo em estudo busca as ações que maximizam o retorno, que é um número mínimo na média de veículos aguardando nas filas segundo diversos comportamentos na via. Tendo como fator decisivo na sua implementação vantagens como, uma habilidade de aprender sem conhecimento prévio ou uma capacidade de se adaptar a novas situações.

Considerou-se como problema de estudo uma pequena rede veicular da cidade de Florianópolis, que foi modelada em um simulador como um problema de aprendizagem por reforço, o qual naturalmente incorpora a natureza estocástica e a dinâmica do fluxo de veículos. Através desta, diversas simulações foram realizadas, variando-se a densidade do fluxo de entrada nas vias e comparando os resultados obtidos para as diferentes formas de controle, onde o algoritmo em estudo obteve um ótimo resultado.

1.3 Estrutura da Dissertação

Esta dissertação tem sua estrutura organizada em um texto composto por seis capítulos e um apêndice, como forma de orientar uma melhor leitura, como descrito a seguir:

Capítulo 2: Redes de Tráfego Veicular Urbano - É feita uma introdução ao sistema de tráfego urbano, enfocando o uso das técnicas de controle utilizadas no Brasil, bem como uma explanação de alguns problemas que o gerenciamento e controle de tráfego podem apresentar; quais os avanços obtidos e quais os desafios que os gerenciadores de tráfego poderão enfrentar futuramente, além de apresentar os modelos para redes veiculares; os autômatos celulares e uma primeira explanação do problema que essa dissertação irá tratar.

Capítulo 3: Inteligência Artificial - Apresenta conceitos sobre Inteligência Artificial, partindo da origem formal da IA como ciência; um breve histórico da IA e suas subáreas. Fala-se também sobre os sistemas especialistas, que se utilizam de conhecimento especializado para as resoluções dos seus problemas; sobre aprendizado de máquina, que estuda métodos computacionais como forma de adquirir novos conhecimentos bem como meios de organizar os conhecimentos já existentes; sobre agentes inteligentes, que são capazes de aprender e de se adaptar ao ambiente com maior capacidade de sucesso; e sobre Inteligência Artificial Distribuída, que se utiliza de sistemas de concepções descentralizadas ou distribuídas onde vários componentes trabalham conjuntamente dividindo tarefas com o intuito de alcançar seus objetivos.

Capítulo 4: Aprendizagem por Reforço - Apresenta alguns tópicos específicos relacionados a teoria de Aprendizagem por Reforço; a propriedade de Markov; e o processo de decisão Markoviano. São apresentados também os três métodos de resolução para os problemas de AR, que são: Programação Dinâmica, método de Monte Carlo e Diferença Temporal, onde através de um exemplo avaliamos e comparamos suas eficiências, com o objetivo de salientar similaridades e diferenças entre estas teorias.

Capítulo 5: Aprendizagem por Reforço Aplicada ao Controle de Tráfego Urbano - Apresenta a modelagem e implementação do modelo no problema de estudo proposto. São apresentadas a formalização teórica e o desenvolvimento do algoritmo *Q-learning* distribuído, bem como as técnicas de controle que servirão como comparação, assim como as análises comparativas dos resultados computacionais obtidos.

Capítulo 6: Conclusões - São apresentadas as conclusões do trabalho realizado, estado atual da pesquisa e propostas de possíveis estudos futuros.

Apêndice: Configuração do Simulador - Contém as listas com as configurações do simulador gerado e implementado em linguagem C++. No Apêndice A.1 é apresentado a topologia da rede de tráfego; no Apêndice A.2 é mostrado a configuração dos movimentos factíveis e as probabilidade das conversões; e no Apêndice A.3 os parâmetros que definem a via, tal como a quantidade de veículos, quantidade e tamanho das células.

Capítulo 2

Redes de Tráfego Veicular Urbano

2.1 Introdução

Procurando minimizar os problemas urbanos associados à circulação de veículos, diversas formas de controle de tráfego vem sendo utilizadas. Dentre estas, os semáforos podem ser uma das mais eficientes maneiras de controle em uma intersecção. No entanto, quando ineficientemente operados, estes podem ocasionar grandes transtornos para os usuários da via, acarretando atrasos nos tempos de viagem, consumo excessivo de combustíveis, poluição, etc.

Os semáforos são equipamentos que alternam o direito de passagem de veículos e/ou pedestres em intersecções de duas ou mais vias. Os semáforos visam assegurar, principalmente, dois atributos operacionais: fluidez e segurança. A fluidez pode ser definida como a facilidade com que é realizado o escoamento das correntes de tráfego. Já a segurança pode ser compreendida como a garantia de um escoamento isento de perigos para os usuários.

O semáforo, apesar de organizar, disciplinar e gerenciar os movimentos em conflito no espaço viário, evidentemente contribui para eventuais atrasos e paradas de veículos e pedestres. Assim, torna-se necessário o uso de técnicas de controle semafórico que reduzam tais prejuízos, melhorando a fluidez e a segurança do tráfego.

Visando minimizar o congestionamento urbano, os gestores públicos têm buscado alternativas mais eficientes e eficazes para gestão do tráfego, ao invés do simples aumento da oferta viária, viabilizando benefícios operacionais para o usuário do tráfego urbano.

Para um entendimento mais claro proposto por este trabalho, este capítulo traz alguns conceitos e temas importantes relacionados ao tráfego como, um breve histórico da utilização do semáforo dentro do Brasil; a atual contextualização do gerenciamento e controle do tráfego urbano; a problemática apresentada; o que foi feito para resolver esses problemas; quais os desafios futuros; é mostrado modelos matemáticos presentes no tráfego e é apresentado conceitos de autômatos celulares.

2.2 Sistemas de Tráfego Urbano

2.2.1 Histórico

Segundo, Bonetti Jr. e Pietroantonio em [5], a evolução do controle semafórico no Brasil ocorreu na década de 70 por linhas definidas pelas experiências de outros países e foi liderada pelas iniciativas tomadas pela Companhia de Engenharia de Tráfego do Município de São Paulo (CET/SP).

Em 1976, quando aqui ainda imperavam os equipamentos eletromecânicos, ocorreu a introdução de controladores multiplanos de tempo fixo, estes consistem na utilização de planos de controle semafóricos determinados previamente por meio de dados históricos do tráfego, onde a escolha do plano a ser usado é feita por programação horária e necessitam de constante levantamento de dados em campo para atender às variações da demanda de tráfego.

Já em 1980, incluiu-se a implantação da centralização de equipamentos com programações em tempos fixos, projeto batizado de SEMCO em São Paulo, utilizando controladores eletrônicos importados da Inglaterra. Segundo João Neto em [29], a diferença do SEMCO é que ele trabalha com grupos de intersecções semafóricas, chamadas sub-áreas. Dentro de uma mesma sub-área, todos os cruzamentos têm o mesmo ciclo ou múltiplos do ciclo, pois é fundamental que um eixo com várias intersecções semafóricas consecutivas e próximas, trabalhem no mesmo ciclo, senão seria impossível garantir o sincronismo. Outra vantagem é a possibilidade de se poder ajustar os tempos de um ou mais cruzamentos que possam vir a sofrer situações inesperadas (veículos quebrados, acidentes, etc), através dos técnicos que operam na central [27].

Na década de 90 foram desenvolvidos controladores eletrônicos mais modernos (SEMIN - Semáforos Inteligentes) e implantados os sistemas centralizados de tráfego (CTA's) com técnicas de controle de tráfego em tempo real, que tem como objetivo a centralização e cuja programação semafórica é determinada dinamicamente por sistemas dedicados, com bases em dados do tráfego coletados por laços detectores em campo, tendo como característica a operação de forma isolada ou coordenada, buscando determinar dinamicamente seu tempo de ciclo [40].

Esse controle pode ser aplicado tanto na forma atuada como seleção dinâmica de planos. Na forma atuada, os tempos semafóricos (tempos de verde, vermelho e amarelo) não são fixos, variando de um tempo mínimo a um máximo, não existindo um cálculo de otimização que vise minimizar os atrasos na área sob controle, ele simplesmente prolonga o verde à medida que detecta a aproximação de mais veículos até atingir o verde máximo. Já na seleção dinâmica de planos ocorre uma variação do sistema de controle em tempos fixos, mas ao invés dos planos serem selecionados por uma tabela horária, os mesmos são selecionados por uma tabela de decisão previamente definida por programas específicos de dimensionamento de plano semafóricos.

Os sistemas centralizados podem ser definidos em quatro linhas básicas, de acordo com sua geração, ou seja:

i) **supervisão e operação** - são aqueles em que o software permite a supervisão do estado da

operação, isto é, qual estágio ou intervalo do plano de tráfego a tempos fixos vigente está em operação;

ii) **monitoração** - onde, através de detectores que colhem dados do tráfego é possível mensurar e monitorar as condições das correntes de tráfego, acumulando dados em tempo real ou por processo de comparação com parâmetros pré-fixados, tais como: saturação, comprimento de fila e ocupação;

iii) **seleção automática de planos** - consiste na seleção de planos pré-definidos, com base em dados coletados previamente por programas específicos de dimensionamento de plano semafóricos;

iv) **controle em tempo real** - é aquele onde o processo de informação (coleta de dados) acontece de uma maneira suficientemente rápida de forma que os resultados são disponíveis a tempo de influenciar o processo de controle ou monitoração, no próprio ciclo ou adiante.

2.2.2 Contextualização

O uso do controle semafórico atuado pelo tráfego (ver sub-seção 2.2.3) não é disseminado no Brasil tal como é em países como os Estados Unidos, Inglaterra e Austrália. As vantagens e desvantagens da utilização desse sistema pelo tráfego são pouco estudadas no Brasil, o que pode ser citado como causa e/ou conseqüência de seu pequeno uso.

Até os dias de hoje, o controle semafórico em modo atuado não tem sido utilizado em larga escala no Brasil, exceto para estágios específicos de travessia de pedestres acionados por botoeira (semáforos semi-atuados), talvez pela ausência de recursos mais avançados nos equipamentos nacionais.

Os controladores nacionais apresentam recursos de operação de planos a tempos fixos nos modos isolado e coordenado e atuação no modo isolado. Segundo Bonetti e Pietroantonio em [5], o modo atuado coordenado, normalmente oferecido, ainda não foi suficientemente testado para ter-se uma avaliação dos resultados. Por sua vez, os controladores nacionais dispõem do recurso de centralização em supervisão e operação, distinguindo-se entre si pela característica de interfaces mais ou menos amigáveis para tanto.

As iniciativas em São Paulo seguem a tradição inglesa, que contrasta com a observada nos EUA, onde os sistemas centralizados evoluíram de forma pouco significativa e o uso de controladores atuados com ajustes de tempos e seleção de estágios, combinada com a coordenação *off-line* foi o caminho predominante. A evolução das técnicas de controle semafórico na Austrália também teve uma relação mais próxima com o uso da atuação, influenciando os sistemas de controle em tempo real.

Segundo Bonetti e Pietroantonio [5], há uma tendência pela centralização em tempo real como forma de controle semafórico nas cidades brasileiras, mas que gera um impasse estrutural, pois os controladores de tráfego em operação nas cidades brasileiras são na sua grande maioria de fabricação nacional e os softwares disponíveis permitem apenas centralização de supervisão e operação.

Para se chegar ao controle em tempo real há necessidade da troca dos controladores existentes por equipamentos que suportem esse tipo de controle e além disso, o controle em tempo real exige a instalação de detectores, fato que deve ser considerado na análise. Em qualquer controle semafórico centralizado sempre há necessidade da implementação de rede de comunicação envolvendo os controladores de tráfego e a central de controle o que gera custos de implantação e manutenção.

No Brasil, aparentemente, a pequena evolução da atuação nos controladores eletrônicos nacionais pode ser citada como causa e conseqüência desse processo. A dificuldade de implantar sistemas centralizados também limita o desenvolvimento da tecnologia no setor.

2.2.3 Problemas no Gerenciamento e Controle

Segundo Luna em [14], quando os fluxos de tráfego são conflitantes entre si, é necessário o emprego de um controle de tráfego adequado para reduzir atrasos e acidentes. Essa é a filosofia básica por trás do controle semafórico.

Definir qual operação adotar e qual programação executar são duas das dificuldades encontradas no gerenciamento e controle do tráfego, pois o fluxo de tráfego varia ao longo do tempo, requerendo que o tempo de verde (tempo durante o qual a luz verde permanece acesa) também varie de forma a se adequar a essa demanda variável. Com isso tenta-se ao máximo otimizar uma via utilizando critérios como: 1) maximizar a capacidade de fluxo da rede; 2) minimizar o impacto negativo do tráfego no meio ambiente; 3) incrementar a segurança no tráfego; 4) minimizar o tempo de viagem; entre outros.

O Denatran [17] destaca que em uma boa operação semafórica três variáveis devem ser bem ajustadas ao logo do tempo: 1) tempo de ciclo (tempo total para uma completa seqüência de sinalização numa intersecção); 2) percentual de verdes de cada fase (razão entre o tempo de verde e o ciclo); e 3) defasagem (retardo entre o início do sinal verde das intersecções a montante e a jusante). Assim, o controle semafórico pode ser classificado em:

Sistemas de tempo fixo: onde os planos são pré-calculados, baseados em dados históricos do fluxo do tráfego e são programados para entrarem em operação numa determinada hora do dia. Essa técnica exige que se faça uma previsão de médio prazo do comportamento do tráfego, aproximando os tempos próximos das piores condições encontradas dentro da operação de cada plano.

Sistemas atuados: respondem às variações de curto prazo do tráfego, identificadas por meio de detectores. Seus planos continuam sendo pré-calculados e implementados em função do horário e do dia da semana, com a diferença do pré-estabelecimento de valores mínimos e máximos de tempos de verde, permitindo assim que o tempo varie dentro desse intervalo fornecido. Funciona partindo de um tempo mínimo, e esse é acrescido a cada nova detecção veicular até o tempo máximo, tendo a vantagem de poder suprimir automaticamente estágios que não apresentam demanda.

Sistemas adaptativos: apresentam uma quantidade muito diversificada de tecnologias e técnicas utilizadas, variam também quanto a responsividade do sistema, automatização e autonomia em implementação de estratégias de controle. Esses sistemas devem variar os tempos dos semáforos a partir de medições diretas de variáveis de tráfego ou por estimação dessas variáveis de desempenho por coleta de dados empregando detectores. Sendo os mais usados os sistemas SCATS (Austrália) e SCOOT (Inglaterra) [19].

Segundo Karen [8], as análises sobre os sistemas de tráfego são realizadas, na maioria das vezes, sobre as características dos elementos que os compõem, ou seja, sobre características físicas e operacionais das vias, intersecções, assim como sobre o próprio movimento. Entretanto, no contexto urbano algumas peculiaridades devem ser consideradas, já que muitos problemas surgem oriundos da elevada demanda de transporte. Os dispositivos de controle do tráfego desempenham um papel fundamental no escopo das atividades de gerência do comportamento do sistema como um todo.

2.3 Problemática

Associado ao crescimento dos grandes centros urbanos, na maioria das vezes sem planejamento, surgiu o problema referente ao aumento da taxa de motorização para uma malha viária que não conseguiu se expandir na mesma proporção e ritmo do aumento de veículos e conseqüentemente da demanda [15].

Uma alternativa para minimizar o problema do excesso de veículos nos grandes centros urbanos é a utilização de tecnologia computacional não somente no controle e gerenciamento de tráfego, como também no projeto, planejamento, simulações e manutenção dos sistemas de controle do tráfego, originando termos como CATE (Computer Aided Traffic Engineering) e ITS (Intelligent Transportation System), no qual os chamados Sistemas de Controle de Tráfego Urbano se incluem [15].

2.3.1 Quais Problemas Existem?

Schmitz em [39], relata que o ritmo da indústria automobilística nacional acarreta uma rápida saturação das ruas e avenidas dos centros urbanos mal projetados e de capacidade já bastante limitada pelo número de novos veículos que entram em circulação. Em função deste crescimento, graves problemas afligem o trânsito como: congestionamentos constantes, falta de coordenação dos semáforos, segurança dos motoristas, entre outros que repercutem seriamente na economia como um todo.

Outro problema verificado é na tentativa de reduzir os congestionamentos nas vias urbanas, onde é feito o uso de semáforos automatizados que sincronizam mudanças de estado, mas essas sincronizações são realizadas, na maioria das vezes, em apenas alguns trechos. Conseqüentemente, o problema é solucionado somente nesses trechos e eventualmente compromete a qualidade do trânsito nas outras regiões. Verifica-se ainda que na maioria dos centros urbanos o trânsito não possui um fluxo uniforme

em todo o seu período de tempo, dificultando assim a gerência dos tempos necessários na configuração dos semáforos. Enfim, a dúvida que normalmente surge é que dadas as várias estratégias de sincronização de semáforos, qual utilizar?

Contudo, como vimos na seção 2.2.2, os controladores de tráfego em operação nas cidades brasileiras são, na sua grande maioria, de fabricação nacional e os softwares disponíveis permitem apenas centralização de supervisão e operação. E para se chegar ao controle em tempo real há necessidade da troca dos controladores existentes por equipamentos que suportem esse tipo de controle, e além disso, o controle em tempo real exige a implantação de detectores. Além da necessidade da implementação de uma rede de comunicação envolvendo os controladores de tráfego e a central de controle, o que gera custos de implantação e manutenção. E essa dificuldade de implantar sistemas centralizados também limita o desenvolvimento da tecnologia no setor.

Outros problemas estão relacionados aos pacotes de softwares de gerenciamento de tráfego, onde os dados coletados em campo, as especificações dos dispositivos de controle de tráfego, seus protocolos de comunicação, sua topologia ao longo da malha viária e seu grau de interoperabilidade são mantidos em sigilo. Pois do ponto de vista financeiro, interoperar significa preservar investimentos realizados e para interoperar deve-se adotar padrões [15].

2.3.2 Quais Avanços Foram Obtidos?

A implantação de tecnologias computacionais avançadas no controle do tráfego urbano é um dos avanços apresentados na tentativa de resolver ou pelo menos minimizar os problemas no trânsito, onde através de sincronismo e eficiência, tentam gerenciar o fluxo de veículos na malha viária minimizando o problema de congestionamento urbano e suas conseqüências, como: um maior consumo de combustível; maiores índices de poluição (sonora e atmosférica); tempo de espera maior para veículos específicos como ambulâncias, bombeiros, transportes coletivos entre outros [15].

Para minimizar atrasos a esses veículos específicos, diversos métodos foram ou ainda estão sendo adotados, como por exemplo: i) corredores exclusivos de tráfego; ii) implantação de terminais de integração para ônibus coletivos; iii) sensores instalados em postes próximos às intersecções semafóricas identificam a presença desses veículos e dão prioridade de passagem aos mesmos, entre outros.

O desenvolvimento de simuladores de malha viária afim de simular as mais diversas situações reais é outro avanço obtido, onde estes visão a obtenção de informações importantes para a gerência e controle do tráfego e, mais precisamente, dos semáforos nos cruzamentos.

Outra forma de tentar acabar com os problemas no trânsito é a adoção de componentes importantes em um controle automatizado, como os dispositivos de controle de tráfego: semáforos, sensores, etc. Entre os sensores adotados no gerenciamento do tráfego, temos: detectores por botão; por laço indutivo; por radar; por ultra-som; por rádio freqüência; por emissão de luz; etc [43].

A introdução dos microprocessadores trouxe a possibilidade de aumento da capacidade computacional para os equipamentos controladores de tráfego, e com isso a utilização de algoritmos de

controle ótimo para intersecções isoladas, monitoração do funcionamento dos equipamentos, etc. Os controladores eletrônicos microprocessados são os que existem atualmente de mais evoluído em controle de tráfego.

Outro avanço obtido foi a implantação dos sistemas ITS que consistem num conjunto de tecnologias aplicadas ao gerenciamento de sistemas de transportes para melhorar a eficiência e segurança viária, reduzindo custos de uma rede de transportes. Onde é feito através do uso de tecnologias de informática, de telecomunicações e de controle automático [23].

Enfim, vários trabalhos estão sendo realizados no Brasil e no mundo para sanar os problemas relacionados ao tráfego, como por exemplo, estudos de análises do tipo de equipamento a ser utilizado, tendo relação direta com o modo de operação do controle, estudos sobre a quantificação das vantagens e desvantagens das diversas possibilidades de sinalização, estudos sobre os custos na implantação de possíveis soluções, entre outros.

2.3.3 Quais são os Desafios?

Como observou-se na seção 2.3.1, tem-se um problema do ponto de vista de um sistema de computação, onde existem dois fatores relevantes, que são a interoperabilidade entre esses dispositivos de controle e os padrões a serem adotados, de forma que os sistemas de controle de tráfego urbano possam evoluir através da agregação de novas facilidades e recursos ou ainda desativar ou modificar os já existentes. Então esse tipo de comunicação deveria ser feito num padrão único, para que os dispositivos pudessem funcionar em qualquer tipo de controlador.

Faltam investimentos na aplicação de diversos estudos relacionados ao tráfego. Como por exemplo, nos avanços rápidos e contínuos na eletrônica e tecnologia da computação que são um grande elemento impulsionador dos novos conceitos em controle de tráfego viário.

Com investimentos em sistemas ITS, os veículos poderiam adotar computadores de bordo e comunicadores, onde esses poderiam receber do controle de tráfego central, instruções sobre o melhor caminho até o destino final. O computador de bordo também poderia informar ao computador central o seu tempo de viagem e a velocidade para serem adotadas como parte da informação a ser processada. Em sistemas ainda mais avançados, a temporização dos semáforos seria coordenada instantaneamente pelas informações recebidas dos veículos próximos. Mais do que simplesmente acomodar o trânsito de veículos por meio de uma rede local, o sistema poderia redefinir padrões de circulação.

Já para os usuários de transportes urbanos, o sistema ITS poderia fornecer informações mais confiáveis sobre o as condições do tempo de viagem, informações essas que poderiam ser obtidas em casa ou no escritório, assim o sistema poderia evitar picos de sobrecarga, fazendo com que os serviços sejam mais adequados e a viagem mais confortável para o passageiro.

Finalmente, no futuro poderá ser implantado um sistema de controle automático para condução de veículos, no qual o motorista ao chegar em uma estação autorizada, entrará na via que determinará seus espaços frontais e laterais a serem respeitados (minimização do espaço inter-veicular), limites de

velocidade e demais parâmetros (estratégias de prevenção de colisões) diretamente ao computador de bordo, além do controle da entrada de veículos por acessos laterais. Sistema este que encontra-se em estudo na Inglaterra (Leeds), batizado com o nome de Prometheus, e nos Estados Unidos (Berkeley), batizado com o nome de IVHS [3] e [31].

2.3.4 Qual é o Problema de Interesse na Presente Dissertação?

O problema a ser apresentado e tratado por esse documento é mostrar que a utilização de técnicas de aprendizagem por reforço, mas especificamente o uso do algoritmo *Q-learning*, em um ambiente de simulação real de tráfego, venha a ser eficaz na minimização da quantidade média do número de veículos esperando nas filas das intersecções, e com isso minimizar o congestionamento e melhorar o tempo de viagem na rede em estudo.

Assim usando a estratégia de decisão aplicada (tamanho da fila de espera), o objetivo é mostrar que o algoritmo *Q-learning* distribuído utilizado corresponde às expectativas de não deixar os motoristas esperando por tempos maiores que os obtidos pelas técnicas de controle uniformemente variada e de melhor esforço e assim não comprometer as vias, esgotando a capacidade de sua ocupação.

Como essa técnica simula um ambiente real as variações de fluxo nas vias também são observadas, fazendo com que os agentes do algoritmo adotem padrões para diferentes densidades nas vias.

2.4 Modelos para Redes Veiculares

Freqüentemente os volumes de veículos observados variam muito ao longo do dia, gerando demanda diferenciada em função do horário. O modo de obter-se melhor rendimento é através do uso de controladores que possam alterar a duração das fases ao longo do dia. Esses ajustes podem ser feitos por detecção, onde a duração das fases varia em função da existência ou não de veículos na área de atuação de dispositivos detectores.

O método mais básico de concepção para escolha de um plano semafórico é a modelagem de formação e descarga de filas, que ocorrem nas intersecções em duas situações: 1) durante o vermelho, quando ocorre o bloqueio para o avanço dos veículos e 2) quando a razão de chegada de fluxo é maior do que a capacidade de descarga da via durante o tempo de verde do semáforo. Este último caso ocorre em situações de sobre-saturação. Este tipo de modelagem busca responder questões do tipo: quanto um usuário vai esperar ou qual o número de unidades aguardando em uma fila. Essa orientação se adapta bem ao tipo de análise feita para escolha do melhor plano semafórico para determinado período do dia [14]. Essa modelagem pode ser feita de duas formas, uma baseada em filas verticais e uma outra baseada em filas horizontais, como veremos a seguir.

2.4.1 Modelos Baseados em Fila Vertical

2.4.1.1 Descrição Temporal de Chegadas

O modelo baseado no fluxo de veículos detectado pelo sistema projeta um perfil de como esses veículos chegam ao longo do tempo na linha de retenção. Esse perfil não é nada mais que um histograma de chegada na linha de retenção [24]. O conceito de fila no ciclo apresenta um processo de formação e desmanche de fila durante um ciclo visualizado nas Figuras 2.1 (a) e 2.2 (a).

A Figura 2.1 representa uma situação de não congestionamento, onde todos os veículos que chegam na aproximação num ciclo conseguem sair do cruzamento durante o tempo de verde desse ciclo e a Figura 2.2 representa uma situação de congestionamento, onde nem todos os veículos conseguem sair do cruzamento durante o tempo de verde do ciclo.

Na Figura 2.1 (a), o fluxo que chega (suposto constante) está representado pelo segmento OE, enquanto que o fluxo que sai do cruzamento está representado pelo segmento CB. A tangente do ângulo X representa, numericamente (em veíc./s), o fluxo que chega, enquanto que a tangente do ângulo Y representa, numericamente (em veíc./s), o fluxo de saturação (que é o maior fluxo possível que sai da aproximação supondo que o semáforo esteja sempre verde). O segmento OA representa os veículos que chegam durante o tempo de vermelho. O segmento AB (intervalo de tempo t_1 a t_2) representa os veículos que chegam quando o semáforo já está verde. Salienta-se que estes veículos, apesar de o semáforo já estar verde, ainda "aguardam" na fila. Só não "aguardam" na fila os veículos que chegam após o instante t_2 (intervalo de tempo t_2 a t_3, onde t_3 é o fim do ciclo), pois estes veículos passam diretamente pelo cruzamento sem sofrer nenhum retardamento. O ponto de intersecção dos dois segmentos é o ponto onde a fila é desmanchada completamente, ou seja, no ponto B (instante t_2) a fila é zerada.

Em caso de congestionamento (fluxo de chegada maior que o fluxo de saturação) os dois segmentos não se interceptarão por maior que seja o tempo de verde, significando que a fila não se desmanchará no ciclo, acumulando veículos para o ciclo seguinte representado pelo segmento BD na Figura 2.2 (a). Se isso ocorrer continuamente, a fila crescerá de ciclo em ciclo, caracterizando a ocorrência de congestionamento [25].

Na Figura 2.1 (b) a fila é representada durante o tempo de vermelho pelo segmento OC e durante o tempo de verde, a fila é representada pelo segmento compreendido pelo segmento $OB - OC$. Pode-se perceber que a fila vai aumentando durante o tempo de vermelho, chegando ao seu máximo no instante t_1 (início do verde). A partir daí, a fila começa a diminuir até zerar-se no instante t_2. A partir do instante t_2 os veículos que chegam no cruzamento, saem sem sofrer atraso até o instante t_3 (fim do ciclo). Assim, a maior fila ocorrida no ciclo é aquela verificada no instante t_1 (segmento AC).

É importante ainda ressaltar que a fila poderá não ser desmanchada totalmente no ciclo se não houver um tempo de verde suficiente. A Figura 2.2 (b) mostra uma situação em que o tempo de verde não foi suficiente para desmanchar a fila, sobrando portanto uma fila excedente para o ciclo seguinte (segmento BD).

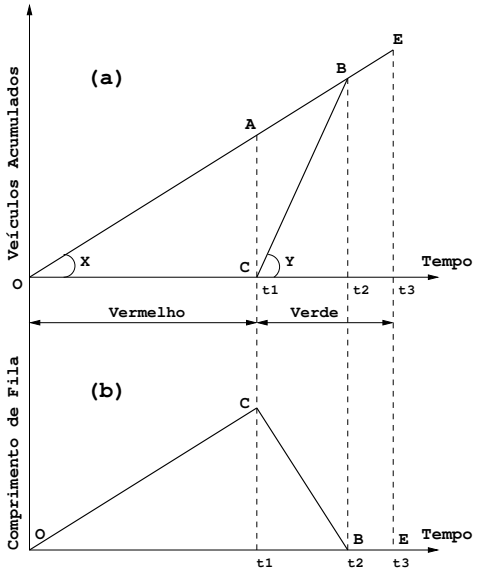

Figura 2.1: (a) Formação de fila numa situação de não congestionamento e (b) Atraso total por ciclo

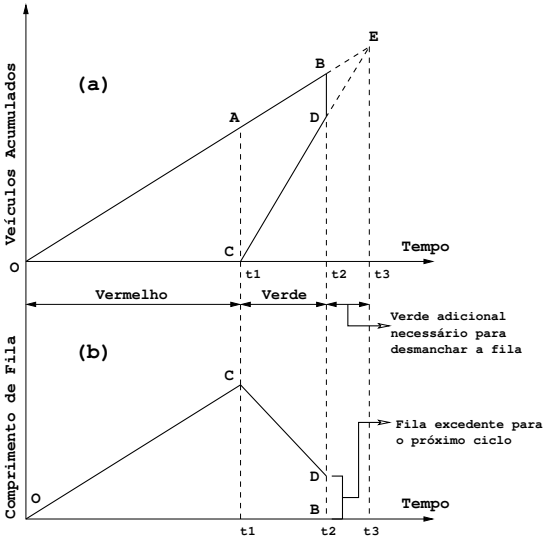

Figura 2.2: (a) Formação de fila numa situação de congestionamento e (b) Atraso total por ciclo

O atraso total no ciclo (soma do tempo de espera de todos os veículos que "aguardaram" na fila no ciclo) é igual à área compreendida entre os 2 eixos (*OB* e *OC*) e os segmentos que representam fluxo que chega e o fluxo de saturação. Na Figura 2.1 (b), é apresentado o atraso total no ciclo que corresponde à área do triângulo *OBC*.

2.4.1.2 Descrição Espacial de Chegadas

Para uma melhor ilustração, utilizamos o desenho de uma via imaginária conforme a Figura 2.3, onde esta foi dividida em várias seções, onde cada uma delas pode ser ocupada por um número x de veículos.

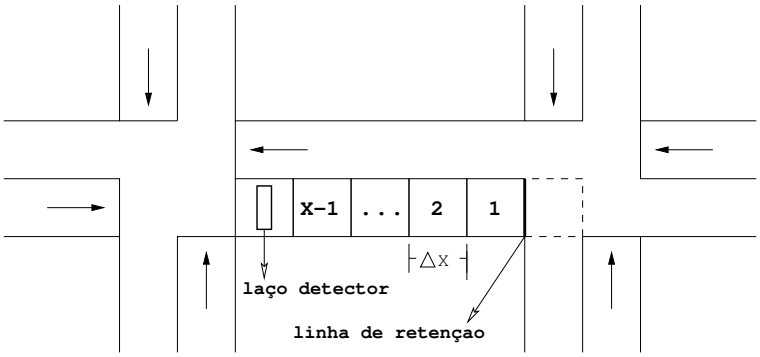

Figura 2.3: Via de tráfego

Na descrição espacial de chegadas podemos observar que os veículos chegam a uma razão constante como pode ser observado no segmento *OE* da Figura 2.1 (a). Durante o tempo de vermelho,

os veículos são distribuídos na via, pelas seções, conforme Figura 2.3 e vão acumulando-se verticalmente na seção 1 ao lado da linha de retenção, a quantidade de veículos nessa seção vai aumentando durante o tempo de vermelho até alcançar a maior fila do ciclo, que é representada pelo segmento *AC* (início de verde). Os veículos que chegam durante o tempo de verde, ou seja, segmento *AB* (intervalo de tempo t_1 e t_2) ocupam seções anteriores à seção 1 (ex: $2,3,\ldots,x$) sofrendo um "atraso" e quando estes chegam a seção 1 a fila vertical já esta sendo descarregada. A partir do instante t_2 os veículos que chegam ao cruzamento saem sem sofrer atraso.

Uma das desvantagens do modelo baseado em filas verticais é o modelamento de congestionamentos, pois como observamos, os veículos se acumulam na seção 1 ao lado da linha de retenção enquanto as outras seções não são preenchidas e o acúmulo de veículos na extensão da via não é caracterizado.

2.4.2 Modelos Baseados em Fila Horizontal

Utilizando a ilustração adotada para o modelo de filas verticais (Figura 2.3), demonstraremos o modelo baseado em filas horizontais. Nos quais, os veículos apresentam um fluxo de chegada a uma razão constante observado pelo segmento *OE* da Figura 2.1 (a). Durante o tempo de vermelho, os veículos são distribuídos na via pelas seções conforme Figura 2.3 e vão acumulando-se horizontalmente nas seções $1, \ldots, x$. A quantidade de veículos nessas seções é definida pelo engenheiro de tráfego, e esta vai aumentando durante o tempo de vermelho até alcançar a maior fila do ciclo, que é representada pelo segmento *AC*. Quando inicia o tempo de verde, as seções descarregam seus veículos na via. Os veículos que chegam durante o tempo de verde, ou seja, segmento *AB* (intervalo de tempo t_1 e t_2) vão apresentar um suposto "atraso" por ainda pegar formação de fila. A partir do instante t_2 os veículos que chegam ao cruzamento, saem sem sofrer atraso.

Diferentemente do modelo baseado em fila vertical, a fila horizontal pode calcular congestionamentos na via (Figura 2.2 (a)). Mas, temos que salientar que o congestionamento detectado pelo sistema é sempre relativo, dependendo da posição do laço detector. Se este estiver muito próximo da linha de retenção, a fila atingirá a Fila Máxima (representa a maior quantidade de veículos que a via pode comportar desde a linha de retenção até o laço detector) a todo momento e todos os intervalos congestionados serão consistentes, dando um alto índice de congestionamento para a via. Por outro lado, se estiver localizado a uma grande distância da linha de retenção e o sistema estiver indicando um alto grau de congestionamento na via, isto poderá ser um indicador de uma situação crítica [25].

Isto ilustra a necessidade do bom conhecimento que o engenheiro de tráfego tem que ter das vias para poder interpretar corretamente os dados apresentados pelo sistema. E para que este engenheiro tenha uma melhor compreensão das limitações de capacidade dos sistemas viários e a avaliação das conseqüências de ocorrências que provoquem pontos de estrangulamento nos mesmos, este precisa conhecer as relações entre as grandezas básicas da via, que veremos a seguir.

2.4.2.1 Relações entre as Grandezas Básicas

As análises macroscópicas exigem a definição das três grandezas básicas que são: fluxo, concentração e velocidade. Que serão apresentadas a seguir, mais estas podem ser vistas com mais detalhes em [13].

Fluxo de tráfego (q), também chamado de volume de tráfego, é uma variável temporal e significa o número de veículos que cruzam uma determinada seção de via considerada dentro de um dado intervalo de tempo. Assim, durante o intervalo de tempo T são contados os $n(x)$ veículos que cruzam a seção, o fluxo $q(x)$ em veíc/h, é então definido por: $q(x) = \frac{n(x)}{T}$, onde x é o comprimento da seção.

Concentração (k), também chamada densidade, é uma grandeza espacial, significando o número de veículos presentes numa determinada extensão de via. Imagine que num determinado instante t uma fotografia é tirada e nela é possível contar os N veículos que se encontram naquele trecho de via. A concentração $k(t)$ em veíc/km, é dada pela expressão: $k(t) = \frac{N(t)}{X}$, onde X é um comprimento limitado entre seções.

Velocidade (v) é uma grandeza definida dividindo a expressão do fluxo pela da concentração: $V = \frac{q(x)}{k(t)} = \frac{X}{T} \times \frac{n(x)}{N(t)}$, definida em km/h. Ao invés do que ocorre com o fluxo (variável temporal) e a concentração (variável espacial), a velocidade é uma variável cuja média pode ser obtida espacialmente (velocidade média espacial) ou temporalmente (velocidade média temporal).

É interessante observar que, uma grandeza temporal é medida no espaço infinitesimal (uma seção da via) e uma grandeza espacial é medida no tempo infinitesimal (um instante t).

Para apresentarmos as relações entre as grandezas básicas do tráfego, primeiramente devemos apresentar alguns valores:

- V_f é a velocidade de fluxo livre, corresponde a média das velocidades desejadas pelos motoristas dos veículos numa corrente de tráfego;
- k_J é a concentração máxima (completo congestionamento);
- q_{max} é o máximo fluxo que pode ser atendido por uma via (representa a capacidade da via);
- V_o é a velocidade "ótima", correspondente ao ponto em que se alcança q_{max};
- k_o é a concentração "ótima", correspondente ao ponto em que se alcança q_{max}.

Modelos de velocidade $\times$ concentração: Os modelos lineares de velocidade $\times$ concentração têm a representação gráfica que aparece na Figura 2.4. Este modelo tem a vantagem da simplicidade, mas observações de campo revelaram que o comportamento linear da curva velocidade $\times$ concentração acontece apenas nas faixas intermediárias de v e k, como mostra a figura 2.5.

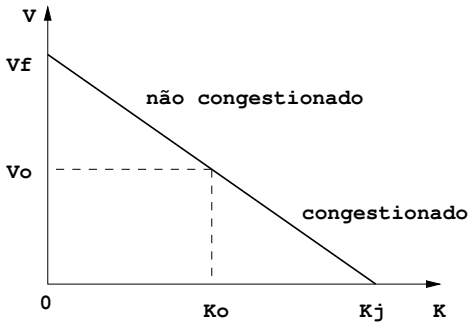

Figura 2.4: Representação gráfica do modelo linear de velocidade × concentração

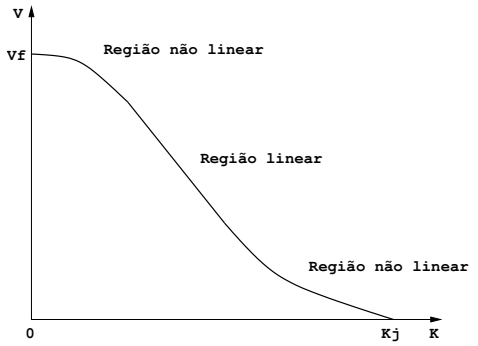

Figura 2.5: Comportamento observado em campo velocidade × concentração

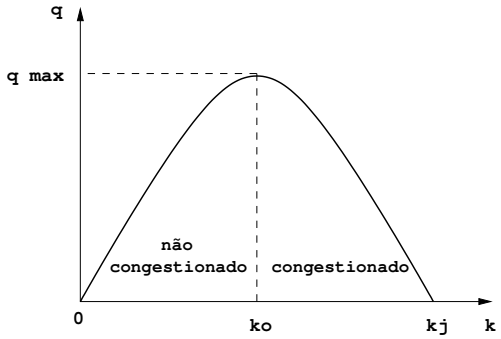

Figura 2.6: Diagrama representando a relação fluxo × concentração

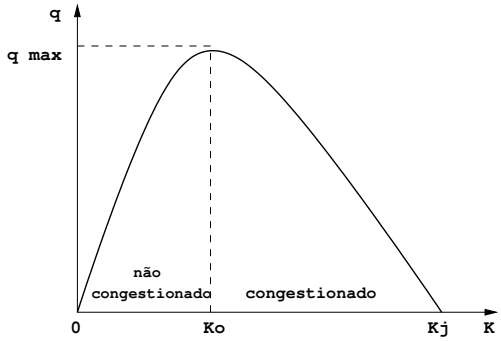

Figura 2.7: Diagrama fluxo × concentração observada em campo

Modelos de fluxo × concentração: A curva sugerida pelos teóricos que primeiro estudaram a relação entre estas variáveis macroscópicas do tráfego está representada na Figura 2.6. Observações de campo demonstraram que a curva não era simétrica, estando mais próxima daquela representada na figura 2.7.

Modelos de fluxo × velocidade: Também para esta relação foi proposto o modelo parabólico, derivado do modelo de Greenshields para a relação velocidade × concentração e correspondente ao diagrama da Figura 2.8.

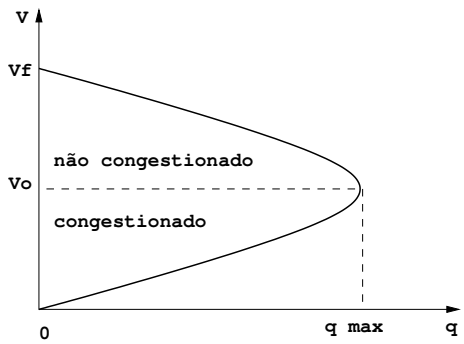

Figura 2.8: Diagrama da relação parabólica entre velocidade e fluxo

2.4.3 Autômatos Celulares (ACs)

Os Autômatos Celulares são exemplos de sistemas dinâmicos discretos, de implementação extremamente simples, que nos permitem a manipulação direta de seus parâmetros para o estudo de sua dinâmica. Por isso, os ACs se tornaram ferramentas importantes para o estudo e modelagem de sistemas complexos reais nas mais diversas áreas [16].

Um Autômato Celular é definido por seu espaço celular e por sua regra de transição. Onde o espaço celular é um reticulado de N células idênticas dispostas em um arranjo d-dimensional, algo como um tabuleiro de xadrez. Tais células possuem um conjunto finito de estados predefinidos e um conjunto de condições necessárias para a mudança dos mesmos. Os estados das células são alterados conforme um conjunto de regras de transição. Tais regras de transição são baseadas no estado atual da célula e de suas vizinhas, isto é, a cada lapso-de-tempo, todas as células do reticulado atualizam seus estados de acordo com a regra [16]. É válido ressaltar que os estados são alterados simultaneamente no tempo de acordo com as regras idênticas para todas as células.

Segundo Chávez em [12], os Autômatos Celulares apresentam três características fundamentais: 1) **Paralelismo** (os estados dos elementos são atualizados simultaneamente); 2) **Localidade** (o novo estado de uma célula é determinado pelo seu estado anterior e pelo estado dos vizinhos); e 3) **Homogeneidade** (todas as células aplicam as mesmas regras de evolução).

Um exemplo simples seria a propagação de um pulso numa corda. Dividimos a corda em pequenos pedaços, que chamaremos de células. Temos que para uma célula sofrer alteração no seu estado, ou seja, para um pulso ser propagado, é preciso que algum de seus vizinhos seja estimulado no tempo anterior. Então podemos definir um conjunto de duas regras básicas: 1) se uma célula está estimulada no instante t, ela não estará no instante $t+1$; e 2) se uma célula não está estimulada no instante t, ela ficará no instante $t+1$ se pelo menos um vizinho seu estiver estimulado no instante t.

O conjunto destas regras consiste no que podemos e vamos chamar de função de transição (f). Na Figura 2.9 podemos encontrar um esquema deste exemplo, onde é mostrado como funciona a transição de estados das células.

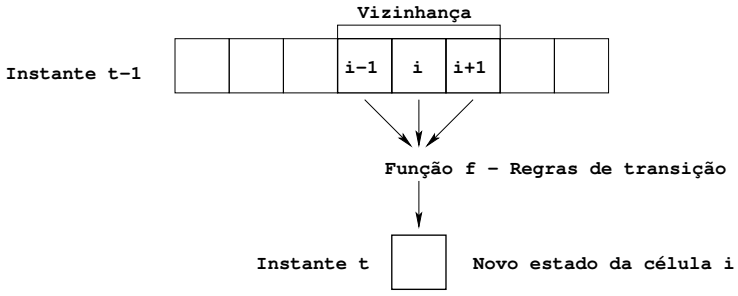

Figura 2.9: Transição dos estados das células

Ao se aplicar um sistema de AC a um dado problema, é preciso especificar o sistema em relação a vários pontos básicos, como por exemplo: a) geometria do sistema, que define a dimensão e o formato das células; b) tamanho da vizinhança, que especifica quais serão os vizinhos; c) condições de limite

ou contorno, que determina condições de limites do autômato; d) condições iniciais, que mostra a condição em que o sistema se encontra no instante inicial t_0; e) conjunto de estados, que mostra os possíveis estados em que as células poderá se encontrar; e f) regras de transição, que definem como o sistema vai evoluir.

Falaremos a respeito de Autômatos Celulares novamente no Capítulo 5, pois o simulador utilizado neste trabalho, é baseado nesse sistema.

2.5 Considerações Finais

Neste capítulo, fez-se uma introdução sobre sistemas de tráfego urbano, onde mostrou-se um breve histórico sobre o uso das técnicas de controle utilizadas no Brasil, bem como uma explanação de alguns problemas que o gerenciamento e controle de tráfego possuem. Fez-se também uma abordagem sobre a problemática que envolve o tráfego urbano, dando ênfase à apresentação de alguns problemas, mostrando os avanços obtidos e os desafios que os gerenciadores de tráfego irão enfrentar (perspectivas para o futuro), além de fazer uma primeira apresentação do problema que essa dissertação irá tratar (minimizar a quantidade média do número de veículos esperando nas filas das intersecções através do uso do algoritmo *Q-learning*, e com isso minimizar o congestionamento e o tempo de viagem na rede). Viu-se também nesse capítulo uma abordagem sobre modelos para redes veiculares, onde foi apresentado modelos baseados em fila vertical e fila horizontal, apresentou-se as grandezas básicas exigidas em uma análise macroscópica, as relações entre estas grandezas de tráfego e por último falou-se de autômatos celulares.

O próximo capítulo apresentará uma breve introdução sobre Inteligência Artificial para que o leitor entenda o principio de solução proposto nessa dissertação.

Capítulo 3

Inteligência Artificial (IA)

3.1 Introdução

Nos dias atuais é muito comum ouvirmos, lermos, escrevermos, falarmos a respeito de Inteligência Artificial. Mas saberemos nós o que é na verdade esta ciência? O que estuda? Que aplicações práticas têm?

Mas, antes de tentarmos responder a essas perguntas, deveríamos primeiro tentar responder a outras questões. Como, o que é inteligência? O que é, um ser inteligente? O que entendemos por aprendizado? Apesar de termos uma noção básica do que significam estas palavras, temos uma grande dificuldade de defini-las em termos práticos e de forma bastante precisa.

Infelizmente, ninguém sabe ao certo o que é inteligência, ou como estar certo de que alguém está se comportando de maneira inteligente, apesar de estudos cada vez mais avançados de vários profissionais, das mais diversas áreas. Será que ser inteligente é apenas adquirir e conservar conhecimento em modelos e responder depressa a certas situações? Nesse caso, poderão os computadores ser inteligentes?

O estudo da inteligência é uma das disciplinas mais antigas, que por mais de 2000 anos, os filósofos tem se esforçado por compreender como se vê, aprende, lembra e raciocina, assim como a maneira com que estas atividades deveriam se realizar.

O conceito de aprendizado também é de difícil definição e de grande importância para que possamos partir para a construção de sistemas inteligentes. De uma maneira sucinta, Osório em [32], diz que poderíamos definir aprendizado como a capacidade de se adaptar, de modificar e melhorar o comportamento e respostas, sendo uma das propriedades mais importantes dos seres ditos inteligentes.

Os esforços do campo da Inteligência Artificial, se concentram em chegar à compreensão de entidades inteligentes. Por isto, uma das razões de seu estudo é o de aprender mais acerca de nós mesmos. De modo diferente da filosofia e da psicologia, que também se ocupam da inteligência, os esforços da

IA procuram tanto a construção de entidades inteligentes como também a sua compreensão, tentando assim construir um corpo de explicações algorítmicas dos processos mentais humanos [34].

As seções desse capítulo apresentarão quesitos que tornarão mais claro o significado de Inteligência Artificial, a sua história, a sua situação atual, os seus subdomínios, além de abordar também o Teste de Turing, a Inteligência Artificial Distribuída e por fim os Agentes Inteligentes.

3.2 O que é Inteligência Artificial?

Inteligência Artificial (IA) é um ramo da Ciência da Computação que reúne uma extensa gama de disciplinas relacionadas com a reprodução, em sistemas artificiais, do pensamento e das características normalmente associadas à inteligência humana. O escopo da IA é portanto muito amplo e abrange desde estudos epistemológicos e filosóficos até a robótica e sistemas adaptativos, passando por modelos formais de raciocínio, redes neurais artificiais, teoria do caos, algoritmos genéticos e assim por diante [33].

Segundo Bittencourt [4] a IA pode ser entendida através de duas linhas principais de pesquisa para a construção de sistemas inteligentes, que são: **Linha Simbólica:** que dá ênfase aos processos cognitivos, ou seja, a forma como o ser humano raciocina; e **Linha Conexionista:** que dá ênfase no modelo de funcionamento do cérebro, dos neurônios e das conexões neurais.

Segundo Russel e Norvig [38] existem várias definições para designar esse termo, pois IA foi e continua sendo uma noção que dispõe de múltiplas interpretações. E a partir dessas definições, tem-se adotado quatro enfoques diferentes, que são possíveis objetivos a se alcançar. Existem 2 enfoques centrados no comportamento humano, que constitui uma ciência empírica; e 2 enfoques racionalistas que combinam matemática e engenharia. Ambos têm dado valiosas contribuições. A organização desses enfoques pode ser vista na Tabela 3.1, sendo descritos de maneira breve a seguir.

	Eficiência Humana	Inteligência Ideal
Processos mentais e raciocínio	Sistemas que pensam como humanos	Sistemas que pensam racionalmente
Conduta	Sistemas que agem como humanos	Sistemas que agem racionalmente

Tabela 3.1: Interpretações da IA

Sistemas que pensam como humanos

Esses sistemas se baseiam na criação do modelo cognitivo, que tinha como enfoque fazer com que um determinado programa pensasse como humanos, mas primeiramente teria que ser definido como pensam os seres humanos, teria que penetrar no funcionamento da mente humana. Existem duas formas de se fazer isto: uma é mediante a introspecção (apanhando nossos próprios pensamentos conforme estes vão acontecendo) ou mediante a realização de experimentos psicológicos. Se

uma teoria bastante precisa da mente fosse obtida, poder-se-ia proceder a expressar tal teoria em um programa de computador.

Sistemas que agem como humanos

Tentam oferecer uma definição satisfatória operativa do que é a inteligência. Turing definiu uma conduta inteligente como sendo a capacidade de alcançar eficiência a nível humano em todas as atividades de tipo cognitivo, suficiente para enganar a um avaliador, chamada de Teste de Turing. (mais detalhes na seção 3.4).

A necessidade de agir como os humanos, se apresenta basicamente quando os programas de IA devem interagir com pessoas. Esses programas deverão se comportar de acordo com certas convenções próprias das interações humanas com a finalidade de se poder entendê-los. Por outro lado, a maneira de elaborar representações e de raciocinar em que estão baseados estes sistemas poderá ou não conformar-se de acordo com um modelo humano.

Sistemas que pensam racionalmente

Tem como enfoque as leis do pensamento. Tentando codificar a "maneira correta de pensar", quer dizer, codificar os processos irrefutáveis de pensamento, através de silogismos (esquemas de estruturas de argumentação através das quais sempre se chega a conclusões corretas quando se parte de premissas verdadeiras). Tais leis do pensamento devem governar a maneira com que a mente opera. Um exemplo de silogismo seria: **premissa maior:** uma pessoa pontual é uma pessoa responsável; **premissa menor:** você é pontual; **conclusão:** logo, você é responsável.

Porém este enfoque apresenta dois obstáculos. Um primeiro que é a formalização do conhecimento, onde não é fácil através de um conhecimento informal expressá-lo em termos formais exigidos pela notação lógica. E um segundo é o processo de inferência, que é o responsável por encontrar regras que sejam adequadas à situação corrente que se deseja resolver. As regras tem um formato genérico, onde uma condição antecedente é seguida por uma ação conseqüente. Se a condição apresentada pelo antecedente é satisfeita, então a ação especificada pelo conseqüente é executada. Assim, a inferência é um processo pelo qual se chega a uma proposição, afirmada na base de uma ou outras mais proposições aceitas como ponto de partida do processo. Como exemplo temos:

> Se: A temperatura de um forno for $\geq 800^{o}C$ e o tempo de operação for $\geq 15seg$.
>
> Então: Fechar Válvula e mostrar alerta de "excedida a temperatura".

Mas esse conhecimento não ajuda muito o sistema na resolução de casos para os quais ele não está programado. Há uma grande dificuldade em lidar com os casos não previstos, as exceções, as situações incomuns.

Sistemas que agem racionalmente

Agir racionalmente implica em agir de maneira tal que se alcance os objetivos desejados, com base em certas suposições. Um exemplo seria um agente. De acordo com este enfoque, se considera a IA como o estudo e construção de agentes racionais. Esse enfoque baseado no planejamento de um agente racional oferece duas vantagens. Primeiro, é mais geral que o enfoque das "leis do pensamento", dado que efetuar inferências corretas é só um mecanismo útil para garantir a racionalidade, porém não é um mecanismo necessário. E segundo, por estar mais de acordo com o ponto de vista do avanço científico que os enfoques baseados na conduta ou pensamento humano, toda vez que se define claramente o que será a norma da racionalidade, norma que é de aplicação geral.

3.3 Histórico da IA

Provavelmente, a Inteligência Artificial nasceu na antiga Grécia onde se pretendia criar autômatos que procuravam simular formas e habilidades do ser humano.

Apesar de relativamente recente como Inteligência Artificial, esta ciência é a realização de um sonho do homem que remonta à Antiguidade Clássica. No Renascimento, Descartes introduz a idéia de que somos máquinas que pensam, cujos músculos são comandados pelo cérebro através do sistema nervoso. Assim Descartes aborda a problemática da IA ao definir o Homem como um corpo capaz de movimento resultante do engenho divino e um Autômato como um corpo capaz de movimento resultante do engenho humano.

No Século XIX surge a figura de Alan Turing, que propôs o chamado jogo da imitação, segundo o qual, se uma máquina fosse capaz de ganhar não restariam as mínimas dúvidas quanto à evidência da existência de máquinas inteligentes. Assim, as teorias de Turing são de certa forma precursoras dos posteriores sistemas especialistas, que procuram imitar os procedimentos dos peritos humanos.

Nesta época o debate era sobretudo teórico e abstrato, pois só com o aparecimento dos primeiros computadores, depois da segunda guerra mundial, é que as noções de automatismo e inteligência se revolucionaram tornando possível a sua discussão em torno da criação do modelo da inteligência humana e da construção de autômatos à imagem do homem.

No entanto só em 1956, no Summer Workshop do Dartmount College (USA), é que a Inteligência Artificial começa a ser reconhecida como ciência, onde um grupo de jovens cientistas se reuniu para discutir uma nova e revolucionaria idéia: "como construir máquinas inteligentes". No entanto o seu objeto de estudo continua incerto, no sentido em que o homem ainda não possui uma definição suficientemente satisfatória de inteligência para compreender os processos da inteligência artificial e da representação do conhecimento.

Assim a história da IA é povoada de diferentes paradigmas que se contrapõem, de teorias que se defendem e abandonam, e que são consecutivamente retomadas. Segundo Caeiro em [6], podemos dividir a história da IA em períodos como segue.

Primeiro Período

Chamado de “euforia”, caracterizou-se pelas previsões que se abeiraram do berço da nova ciência após a criação, em 1956 por Newell e Simon, do primeiro programa de IA, o Logic Theorist (LT). O sucesso do LT foi seguido pelo General Problem Solver (GPS), cujos detalhes podem ser obtidos em [28]. Onde, seus criadores previram que esta ciência, no espaço de 10 anos, iria acabar por criar uma entidade computacional capaz de ultrapassar os desempenhos mentais próprios do cérebro humano, em qualquer domínio. Previsão que não chegou a concretizar-se.

Os primeiros anos da IA se caracterizaram pelo debate entre aqueles que achavam oportuno simular a estrutura do cérebro humano e aqueles que, pelo contrário, concentravam o interesse da ciência mais sobre as suas funções do que sobre o seu procedimento.

A investigação dividia-se então em duas áreas, por um lado, a pesquisa sobre a simulação dos processos cognitivos do ser humano e por outro, a pesquisa sobre o rendimento tão eficiente quanto possível dos programas através de processos não humanos.

Segundo Período

Nos anos 60 difundiu-se o interesse pelos programas capazes de compreender a linguagem humana, mas estes programas não respondiam às expectativas dos investigadores, sendo ainda muito limitados e sendo o seu conhecimento fornecido totalmente pelo programador e em linguagens de baixo nível. No entanto, estes programas tiveram o mérito de introduzir aquele que seria o pilar das posteriores investigações em IA.

No final da década de 60 e início da década de 70 a euforia veio a dar lugar a um visão mais realista da IA, contemplando um conjunto de dificuldades e limitações inerentes ao mundo real, que não tinham sido ainda tidas em conta. Dentre as várias dificuldades, destaca-se a tomada de consciência da existência de problemas pertencentes à classe dos NP-Completos, sendo necessárias heurísticas concretas para os atacar. Deslocou-se o interesse da pesquisa no sentido de temas mais ligados a conhecimentos relativos a domínios específicos (sistemas especialistas).

Terceiro Período

Caracterizado pelos sistemas especialistas, também chamados de agentes inteligentes, que são programas que possuem um vasto e específico conhecimento sobre um determinado assunto. Constituíram num grande sucesso em várias áreas de aplicação.

Os sistemas especialistas pretendem simular o pensamento de um perito humano. Dentre os muitos sistemas desse gênero destacaram-se: o DENDRAL, que foi o primeiro sistema especialista. Depois surgiu o MYCIN, que se diferenciava do DENDRAL por não haver regras gerais teóricas as quais tinham que refletir incerteza.

Quarto Período

Caracterizado pela maturidade da IA, é o período das redes conexionistas (inspirado na rede neurológica humana) e da cibernética (estuda o modelo do cérebro humano). Assim, as suas investigações são no sentido de através da criação de modelos dos processos cognitivos do homem, construir autômatos que à semelhança do sistema neural humano, se encontrem interligados por um conjunto de conexões.

Anos 80

Chamada "a década de ouro da IA", é onde se dá a grande explosão. As evoluções das pesquisas e das concretizações em IA sucedem-se, e formam-se várias IA's servidas por uma atividade científica dotada de um entusiasmo impulsionador de novas descobertas e avanços nesta promissora área da ciência e tecnologia.

Assim, a par da engenharia do conhecimento, da programação em lógica, do raciocínio qualitativo, e da Inteligência Artificial Distribuída, surgem e aprofundam-se os estudos em redes neurais em processamento de imagens, em vida artificial, em lingüística e visão computacional.

A IA procurava e procura ainda compreender a mente através de modelos computacionais, construir sistemas computacionais capazes de realizar ações tradicionalmente consideradas mentais e encontrar novos meios de representação baseados em computação, através dos quais o intelecto humano se poderia expressar de modo diferente, mas com clareza e força.

3.4 Teste de Turing

Em 1950, Alan Turing publicou um artigo chamado "Computing Machine and Intelligence" [45]. Neste artigo, Turing apresentou pela primeira vez, o que hoje é conhecido por Teste de Turing. Com este pretendia-se descobrir se uma máquina pode ou não pensar.

O Teste de Turing funciona da seguinte forma: um interrogador (humano) manterá um "diálogo" de forma indireta com duas entidades ocultas; uma delas sendo um computador (A da Figura 3.1) e outra sendo um humano (B da Figura 3.1). O interrogador tentará, através do "diálogo" realizado decidir qual dos dois é o humano. Se ao final do teste o interrogador não conseguir distinguir quem é o humano, então conclui-se que o computador pode pensar, segundo o Teste de Turing. Porém, nenhuma máquina conseguiu passar consistentemente pelo Teste de Turing. Alguns computadores, devidamente programados, conseguiram passar por versões simplificadas do teste; contudo, sempre esteve ausente o atributo mental do entendimento. Como Marvin Minsky, do MIT, diz: "O maior desafio é dar bom senso às máquinas, e bom senso é essencial para passar no Teste de Turing".

O Teste de Turing envolve pelo menos quatro grandes subáreas da IA [38]: 1) Processamento de Linguagem Natural; 2) Representação do Conhecimento; 3) Raciocínio Automático; e 4) Aprendizagem Automática.

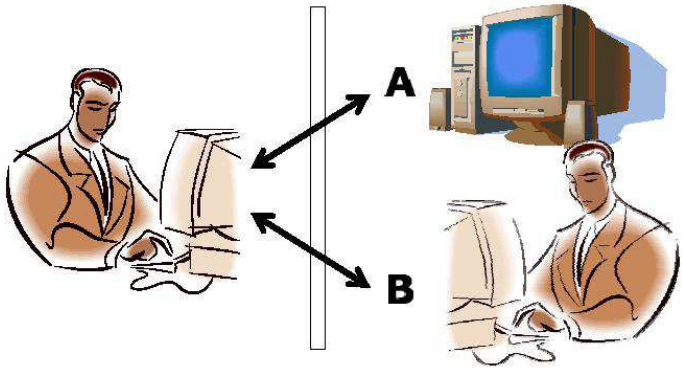

Figura 3.1: Representação do modelo de Turing

3.5 Situação Atual da IA

Segundo Caeiro [6], os estudos em IA atualmente dividem-se em quatro ramos fundamentais.

a) área ligada ao estudo das redes neurais e ao conexionismo que se relaciona também com a capacidade dos computadores aprenderem e reconhecerem padrões;

b) área ligada à biologia molecular na tentativa de construir vida artificial;

c) área relacionada com a robótica, ligada à biologia e procurando construir máquinas que alojem vida artificial;

d) e finalmente o ramo clássico da IA que se liga desde o início à psicologia, desde os anos 70 à epistemologia e desde os anos 80 à sociologia e que tenta representar na máquina os mecanismos de raciocínio e de procura.

3.6 Subdomínios da IA

A IA é dividida em áreas e suas principais aplicações são descritas no que segue:

3.6.1 Sistemas Especialistas (SE)

Este ramo da IA utiliza técnicas que fazem extensivo uso de conhecimento especializado para resolver problemas no nível de um especialista humano, problemas estes suficientemente difíceis para requererem para a sua solução significativa experiência humana, por isso sua atuação é em um domínio restrito. Usa também um complexo encadeamento de inferências para desempenhar tarefas, as quais um especialista poderia executar.

A parte mais sensível no desenvolvimento de um SE é a aquisição de conhecimento. É necessário integrar novos conhecimentos ao já disponível, através da definição de relatos entre os elementos que

constituem o novo conhecimento e os elementos já armazenados na base. Outro ponto importante na aquisição de conhecimento é o tratamento de incoerências, pois dependendo da forma como o novo conhecimento é adquirido, pode haver erros de aquisição.

Segundo Bittencourt em [4], a parte mais importante no projeto de um SE é a escolha do método de representação do conhecimento. No entanto, problemas de eficiência, facilidade de uso e a necessidade de expressar conhecimento incerto e incompleto levaram ao desenvolvimento de diversos tipos de formalismos de representação do conhecimento. Por exemplo: **Lógica**, que é a base para a maioria dos formalismos de representação do conhecimento; as **Redes Semânticas**, que são conjuntos heterogêneos de sistemas; os **Quadros ou "Frames"**, que permitem a representação de relações hierárquicas entre conceitos e através dessas relações podemos inferir propriedades e responder a certas questões.

3.6.2 Aprendizado de Máquina (*Machine Learning)*

Podemos dizer que uma máquina aprende à medida que esta modifica a sua estrutura, o programa ou os dados de tal maneira que o desempenho futuro esperado da máquina melhore. Como por exemplo, quando uma máquina de reconhecimento de voz progride depois de ouvir várias amostras da fala de uma pessoa, trata-se de um caso em que a máquina está aprendendo. Aprendizado de máquina normalmente refere-se a modificações em sistemas que desempenham tarefas associadas com a Inteligência Artificial. Estas tarefas envolvem reconhecimento, diagnóstico, planejamento, controle de robô, e outros [20].

Em [35] é dito que, máquinas que aprendem são distintas de programação exatamente da mesma forma que aprendizagem humana é distinta do instinto animal. Máquinas que são capazes de aprender no sentido desta definição serão chamadas *"Learning Machines"*. Mesmo que esta definição seja imperfeita, ela captura o ponto fundamental: aprendizagem é uma questão de comportamento, não de composição. Aprendizagem deve ser mais que apenas o acúmulo de volume de informação; um pequeno sistema especialista com capacidade de criar automaticamente novas e úteis regras deverá ser chamado de máquina que aprende, um banco de dados convencional, de qualquer tamanho provavelmente não será.

Mas por que usar máquinas quem aprendem? Porque há uma quantidade de problemas para os quais soluções algorítmicas exatas não existem ou não são práticas, e máquinas que aprendem podem desenvolver soluções úteis para alguns destes casos. Máquinas que são apenas programadas freqüentemente não tratam adequadamente ambigüidade ou informação incompleta, máquinas que aprendem podem ser mais robustas. Se máquinas podem ser treinadas para agir como assistentes para especialistas humanos sobrecarregados, elas multiplicam os recursos humanos.

Como as máquinas aprendem? Segundo Osório em [32], exite 3 tipos de aprendizagem, que são:

Supervisionada: O princípio básico desta forma de aprendizagem é que deve-se conhecer, através de pares entrada-saída, quais as respostas que devem ser fornecidas pelo sistema para determinadas entradas ou impulsos externos. Quem terá esse conhecimento será uma espécie de

supervisor, o qual através da diferença entre os valores esperados e os valores obtidos, será capaz de saber qual o erro que está sendo produzido, realizando então os ajustes dos parâmetros da rede. O aprendizado estará completo quando o erro não mais existir, ou assumir valores satisfatoriamente pequenos. A partir desse momento, pode-se dizer que o sistema adquiriu o conhecimento passado pelo supervisor, estando então treinado para o problema apresentado. Esse processo pode ser visto com mais detalhes em [36].

Não Supervisionada: Esse modelo de aprendizado não requer saídas desejadas e por isso é conhecido pelo fato de não precisar de "professores" para o seu treinamento. O treinamento da rede utiliza apenas os valores de entrada. A rede trabalha essas entradas e as organiza em categorias, usando para isso, os seus próprios critérios. Para uma entrada aplicada à rede, será fornecida uma resposta indicando a classe a qual a entrada pertence. Se o padrão de entrada não corresponde às classes existentes, uma nova classe é gerada.

Aprendizagem por Reforço: Neste tipo de aprendizado o usuário possui apenas indicações imprecisas sobre o comportamento final desejado. Para sermos mais exatos, neste tipo de aprendizado nós dispomos apenas de uma avaliação qualitativa do comportamento do sistema, sem no entanto poder medir quantitativamente o erro (desvio do comportamento em relação ao comportamento de referência desejado). O aprendizado por reforço é um método por tentativa e erro, baseando suas ações somente em um índice de desempenho, chamado de "sinal de reforço", que é utilizado para otimização.

3.6.3 Outras Áreas de Aplicações

A IA apresenta um vasto campo de subdomínios, dentre eles temos: Processamento de Conversa e Linguagem Natural, Jogos, Robótica, Prova de Teoremas e Raciocínio Automático, Problemas de Escalonamento e Combinação, Percepção, Planejamento, entre outros.

3.7 Inteligência Artificial Distribuída (IAD)

Inteligência Artificial Distribuída (IAD) é um campo da Inteligência Artificial que trabalha com vários sistemas ao mesmo tempo que se comunicam entre si e dividem tarefas (interação de agentes inteligentes). Cada sistema pode assumir uma função específica e todo o sistema torna-se responsável pelo gerenciamento das interações para alcançar um objetivo comum [22].

Enquanto a IA utiliza sistemas de concepções centralizadas onde os agentes atuam sozinhos, a IAD utiliza sistemas de concepções descentralizadas ou distribuídas onde vários componentes trabalham conjuntamente dividindo tarefas. Permitindo com que vários processos autônomos sejam representados em um só sistema que coordena o conhecimento de vários especialistas, possibilitando a resolução de problemas complexos e atos que requisitam uma inteligência global, por meio de processamentos locais e comunicações interprocessos.

A IAD permite realizar tarefas melhor do que um especialista humano. Trabalha com diferentes módulos que podem manipular características de alto nível como intencionalidade, racionalidade, etc. A resolução de problemas abordados pela IAD encontra-se dividida em dois grupos principais:

Resolução Distribuída de Problemas (RDP): se preocupa com a solução de problemas específicos usando um número de módulos cooperando e compartilhando conhecimentos entre si. Sendo assim, o planejamento das ações é o resultado da decomposição do problema em vários sub-problemas que são distribuídos aos diversos agentes envolvidos. Como resultado desta partilha, os agentes cooperam apenas na divisão do esforço e na partilha de conhecimentos e resultados;

Sistemas Multi-Agente (SMA): se preocupa com a coordenação dos comportamentos (conhecimentos, objetivos, habilidades e planos) de diversos agentes inteligentes autônomos para atingir um ou mais objetivos. Assim, os agentes podem trabalhar em direção a um único objetivo global ou rumo a objetivos individuais separados que podem interagir, sendo então a autonomia dos agentes relacionada com a existência de cada agente independentemente da existência dos demais [2].

Os sistemas de IAD apresentam uma série de vantagens, que são:

Modularidade: é mais fácil trabalhar em módulos independentes, pois desta forma permitimos que um problema possa ser dividido em problemas menores e a cada um atribuímos um agente para solucioná-lo;

Eficiência: cada agente terá um grau de autonomia diferenciado e devido a modularidade este tipo de sistema é mais rápido;

Aumento da confiabilidade e segurança: a solução será distribuída em uma série de agentes diferentes e os problemas poderão ser resolvidos mesmo que um dos agentes apresente falhas;

Interação de múltiplos agentes: usufruindo dos ambientes distribuídos, podemos determinar vários agentes paralelos para trabalhar de forma cooperativa;

Maior leque de opções (criatividade): a probabilidade de mais de um agente autônomo encontrar a solução do problema ou decompor problemas complexos em problemas de menor escala, cada um com suas características distintas para solucioná-los; etc.

3.8 Agentes Inteligentes

Um agente pode ser definido de diferentes formas, de acordo com o enfoque e o objetivo que se quer atingir. De uma forma geral, agente é um novo paradigma para o desenvolvimento de aplicações de software. É uma entidade que percebe o ambiente através de sensores e age neste ambiente através de atuadores, ou seja, é capaz de interagir com o meio em questão, tomando decisões que irão auxiliar

ou até mesmo substituir o trabalho do agente humano (ver Figura 3.2). Então, o objetivo de um agente é interagir com o ambiente para atingir seus objetivos devendo ter capacidade de reunir informações do seu ambiente, tomar decisões baseadas nestas informações e iniciar uma ação especifica baseada nas decisões.

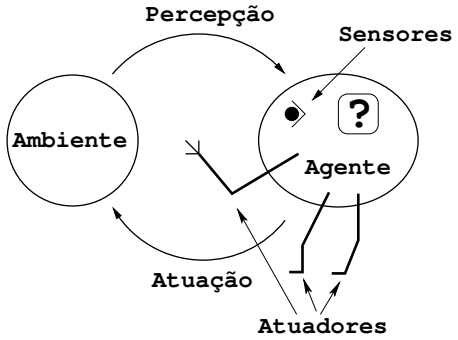

Figura 3.2: Representação da ação de um agente

Os agentes devem apresentar algumas características desejáveis, como por exemplo, **Autonomia:** utilizado como sinônimo de independência; **Adaptação:** devem ser capazes de se adaptar à novos conhecimentos, às mudanças do meio e às interações estabelecidas; **Confiabilidade**: demonstra veracidade e benevolência nas informações e ações realizadas; **Iniciativa:** devem ser guiados por suas intenções (expressão dos objetivos e dos meios); **Racionalidade:** capacidade de justificar sua decisões, saber escolher o melhor caminho para atingir seus objetivos; **Flexibilidade:** escolher a dinâmica das ações e da seqüência de execuções das mesmas, em resposta a um estado do ambiente; **Inteligência:** conjunto de recursos, atributos e características que habilitam o agente a decidir que ações executar; etc.

Segundo Lima [**?**], os agentes podem ser classificados como: **deliberativos ou cognitivos:** que são agentes sofisticados que mantém um histórico das suas ações, apresentam um pequeno número de agentes, possuem uma representação explicita do seu ambiente, possuem um modelo social de organização e tem a função delegar tarefas aos demais agentes; e reativos ou não deliberativos: que não possuem um histórico de suas ações, apresentam um grande número de agentes, não possuem uma representação explícita do seu ambiente, possuem um modo biológico de organização e possuem o modo de funcionamento baseado no processo de estímulo-resposta.

A partir dessas definições, um agente é considerado inteligente se é um ser cognitivo, racional, intencional e adaptável.

Para projetarmos agentes devemos criar uma visão geral para os mesmos, como: percepções, ações, objetivos a serem alcançados e o ambiente. O ambiente pode ser classificado de diversas formas:

acessível $\times$ **inacessível** - onde acessível é quando os sensores do agente conseguem perceber o estado completo do ambiente.

estático $\times$ **dinâmico** - o ambiente estático não muda enquanto o agente está escolhendo a ação a realizar.

determinista $\times$ **não-determinista** - determinista é quando o próximo estado do ambiente pode ser completamente determinado pelo estado atual e as ações selecionadas pelo agente.

discreto $\times$ **contínuo** - discreto é aquele que quando existe um número distinto e claramente definido de percepções e ações em cada turno e continuo é quando as percepções e ações mudam em um espectro contínuo de valores.

episódico $\times$ **não-episódico** - episódico é quando a experiência do agente é dividida em episódios. Cada episódio consiste em o agente perceber e então agir. Cada episódio não depende das ações que ocorreram em episódios prévios.

Continuando a projetar o agente, nós devemos também especificar o tipo de conhecimento a ser adotado pelo agente, a arquitetura e o método de resolução do problema (Sistemas Multi-Agentes ou Resolução de Problemas Distribuídos).

3.9 Considerações Finais

Neste capítulo, foi visto com mais detalhes o que é a Inteligência Artificial, partindo do ponto de vista focado sob quatro enfoques, dois centrados nas semelhanças com os humanos (sistemas que pensam como humanos; sistemas que agem como humanos) e dois baseadas no conceito de racionalidade (sistemas que pensam racionalmente; sistemas que agem racionalmente). Abordou-se também um breve histórico da IA, além de mostrar suas subáreas que vão desde a robótica até ao processamento de linguagem natural, passando pela aprendizagem automática, entre outras. Falou-se também sobre os sistemas especialistas, que são sistemas que se utilizam de conhecimento especializado para as resoluções dos seus problemas e o aprendizado de máquina, que é uma área da IA que estuda métodos computacionais para adquirir novos conhecimentos bem como meios de organizar os conhecimentos já existentes. Falou-se da Inteligência Artificial Distribuída que é um campo da IA que se utiliza de sistemas de concepções descentralizadas ou distribuídas onde vários componentes trabalham conjuntamente dividindo tarefas com o fim de alcançar seus objetivos e por último falou-se dos agentes inteligentes que são capazes de aprender e de se adaptar melhor ao ambiente e com maior capacidade de sucesso, relativamente ao objetivo para o qual foi designado.

O próximo capítulo apresentará uma introdução à Aprendizagem por Reforço para que o leitor fique a par da técnica utilizada no objeto de estudo dessa dissertação.

Capítulo 4

Aprendizagem por Reforço (AR)

4.1 Introdução

Nas abordagens tradicionais de aprendizagem automática, geralmente os sistemas aprendem através de exemplos de pares de entrada e saída, que fornecem indicativos do comportamento esperado do sistema, tendo como tarefa aprender uma determinada função que poderia ter gerado tais pares. Estes métodos são apropriados quando existe alguma espécie de "professor" fornecendo os valores corretos ou quando a saída da função representa uma predição sobre o futuro que pode ser verificada pelas percepções do agente no próximo passo de iteração [38].

Mas, quando se deseja que o agente tenha uma total autonomia, este terá que ser capaz de aprender com base em outras informações, como por exemplo, recompensas ou reforços fornecidos por um "crítico" ou pelo próprio ambiente. Em certos casos é possível que o próprio agente determine as suas recompensas através da observação das transições de estado que realiza no ambiente, passando este a "experimentar" autonomamente o ambiente no qual está inserido [7].

Segundo Sutton e Barto em [41], *Aprendizagem por Reforço* é uma abordagem da Inteligência Artificial que permite a um indivíduo aprender a partir da sua interação com o ambiente onde ele se encontra, através do conhecimento sobre o estado do indivíduo no ambiente, das ações efetuadas no ambiente e das mudanças de estado que aconteceram depois de efetuadas as ações, que é um conceito básico na área de *Aprendizado de Máquina*.

Aprendizagem por Reforço é antes de tudo indicado quando se deseja obter a política ótima (representa o comportamento que o agente segue para alcançar o objetivo) nos casos em que não se conhece *a priori* a função que modela esta política. O agente deve interagir com seu ambiente diretamente para obter informações, que serão processadas através de um algoritmo apropriado, afim de executar a ação que maximize a satisfação dos seus objetivos nos estados do ambiente.

Assim, AR consiste no aprendizado do mapeamento de estados em ações de modo que um valor numérico de retorno seja maximizado. A princípio, o sistema não precisa conhecer as ações que deve tomar, mas deve descobrir quais ações o levam a obter maiores valores de retorno. Estes valores de

retorno podem ser vistos como imediatos (locais) ou como retornos a longo prazo (globais), que neste último caso, permitem orientar o indivíduo a alcançar um dado objetivo.

Neste capítulo abordaremos os conteúdos voltados à AR de forma mais ampla, apresentando suas características, seus problemas, sua formulação matemática e seus métodos de resolução, dando uma maior ênfase ao método conhecido como Diferença Temporal (DT), mais precisamente à resolução baseada no algoritmo *Q-learning*, que foi o algoritmo adotado nos experimentos do estudo de caso deste documento.

4.2 Características da Aprendizagem por Reforço

Aprendizado pela Interação: essa é a característica principal que define um problema de Aprendizagem por Reforço. Onde um agente AR age no ambiente e aguarda pelo valor de reforço que o ambiente deve informar como resposta perante a ação tomada, assimilando através do aprendizado o valor de reforço obtido para tomar decisões posteriores.

Retorno Atrasado: um máximo valor de reforço que o ambiente envia para o agente não quer dizer necessariamente que a ação tomada pelo agente foi a melhor. Uma ação é produto de uma decisão local no ambiente, sendo seu efeito imediato de natureza local, enquanto, em um sistema de Aprendizagem por Reforço, busca-se alcançar objetivos globais no ambiente. Assim as ações tomadas devem levar a maximizar o retorno total, isto é, a qualidade das ações tomadas é vista pelas soluções encontradas à longo prazo.

Orientado ao Objeto: em Aprendizagem por Reforço, o problema tratado é considerado como um ambiente que dá respostas perante ações efetuadas, não sendo necessário conhecer detalhes da modelagem desse ambiente. Simplesmente, existe um agente que age dentro do ambiente desconhecido tentando alcançar um objetivo. O objetivo é, geralmente, otimizar algum comportamento dentro do ambiente.

Investigação x Exploração: Em aprendizagem por reforço os agentes vivem um dilema conhecido na literatura como "*the Exploration* $\times$ *Exploitation dilemma*", que consiste em decidir quando se deve aprender e quando não se deve aprender sobre o ambiente, mas usar a informação já obtida até o momento. Para que um sistema seja realmente autônomo, esta decisão deve ser tomada pelo próprio sistema.

A decisão é fundamentalmente uma escolha entre agir baseado na melhor informação de que o agente dispõe no momento ou agir para obter novas informações sobre o ambiente que possam permitir níveis de desempenho maiores no futuro. Isto significa que o agente deve aprender quais ações maximizam os valores dos ganhos obtidos no tempo, mas também, deve agir de forma a atingir esta maximização, explorando ações ainda não executadas ou regiões pouco visitadas do espaço de estados. Como ambas as formas trazem, em momentos específicos, benefícios à solução dos problemas, uma boa estratégia é mesclar as formas.

Este é um problema crucial no contexto da aprendizagem por reforço, pois agir para obter informação pode aumentar o desempenho à longo prazo, embora faça com que o desempenho a

curto prazo diminua. Tomando-se estes cuidados, quanto mais tempo o agente estiver atuando no ambiente, mais corretas serão suas ações no decorrer de sua tarefa.

4.3 O Problema de Aprendizagem por Reforço

Um sistema típico de Aprendizagem por Reforço constitui-se basicamente de um agente interagindo em um ambiente via percepção e ação. Ou seja, o agente percebe as situações dadas no ambiente, pelo menos parcialmente, e baseado nessas medições, seleciona uma ação a tomar no ambiente. A ação tomada muda de alguma forma o ambiente, afetando o estado na tentativa de alcançar o objetivo relacionado, e as mudanças são comunicadas ao agente através de um sinal de reforço. Como pode ser visto na Figura 4.1 (traduzida a partir da original, obtida no livro de Sutton e Barto [41]).

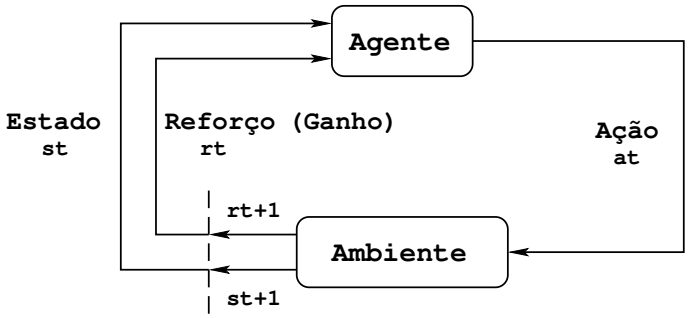

Figura 4.1: A interação agente-ambiente na Aprendizagem por Reforço

Os efeitos das ações não podem ser perfeitamente antecipados, com isso, o agente deve monitorar o ambiente freqüentemente e reagir apropriadamente. Em um sistema de AR, o estado do ambiente é representado por: 1) um conjunto de variáveis de estado percebidas pelo agente, onde o conjunto das combinações de valores dessas variáveis forma o conjunto de estados discretos do agente (S); 2) um conjunto de ações discretas, que escolhidas por um agente muda o estado do ambiente ($A(s)$) e 3) valor das transições de estados, que é passado ao agente através de um sinal de reforço, denominado ganho (valores tipicamente entre $[0,1]$).

O objetivo do método é levar o agente a escolher a seqüência de ações que tendem a aumentar a soma de valores de reforço, ou seja, é encontrar a política π, definida como o mapeamento de estados em ações que maximize as medidas do reforço acumuladas no tempo.

O Problema de Aprendizagem por Reforço apresenta cinco partes fundamentais, que são:

1. **O Ambiente:** Todo sistema de AR aprende um mapeamento de situações e ações por experimentação em um ambiente dinâmico. O ambiente no qual está inserido o sistema, deve ser pelo menos parcialmente observável através de sensores, descrições simbólicas, ou situações mentais. Também é possível, entretanto, que toda informação relevante do ambiente esteja perfeitamente disponível. Neste caso, o agente poderá escolher ações baseadas em estados reais do ambiente.

2. **A Política:** Uma política expressa pelo termo π, representa o comportamento que o sistema AR segue para alcançar o objetivo. Em outras palavras, uma política π é um mapeamento de estados s e ações a em um valor $\pi(s,a)$. Assim, se um agente AR muda a sua política, então as probabilidades de seleção de ações sofrem mudanças e conseqüentemente, o comportamento do sistema apresenta variações à medida que o agente vai acumulando experiência a partir das interações com o ambiente. Portanto, o processo de aprendizado no sistema AR pode ser expresso em termos da convergência até uma política ótima ($\pi^*(s,a)$) que conduza à solução do problema de forma ótima.

3. **Reforço e Retorno:** O Reforço é um sinal do tipo escalar (r_{t+1}), que é devolvido pelo ambiente ao agente assim que uma ação tenha sido efetuada e uma transição de estado ($s_t \rightarrow s_{t+1}$) tenha ocorrida. Existem diferentes formas de definir o reforço para cada transição no ambiente, gerando-se funções de reforço que, intrinsecamente, expressam o objetivo que o sistema AR deve alcançar. O agente deve maximizar a quantidade total de reforços recebidos chamado de retorno, que nem sempre significa maximizar o reforço imediato a receber, mas o reforço acumulado durante a execução total.

   De modo geral, o sistema AR busca maximizar o valor esperado de retorno, com isso, o retorno pode ser definido como uma função da seqüência de valores de reforço até um tempo T final. No caso mais simples é um somatório como aparece na equação seguinte:

$$R_T = r_{t+1} + r_{t+2} + r_{t+3} + \ldots + r_T \tag{4.1}$$

   Em muitos casos a interação entre agente e ambiente não termina naturalmente em um episódio (seqüência de estados que chegam até o estado final), mas continua sem limite, como por exemplo em tarefas de controle contínuo. Para essas tarefas a formulação do retorno é um problema, pois $T = \infty$ e o retorno que se deseja também tenderá ao infinito ($R_T = \infty$).

   Para estes problemas foi criada a taxa de amortização (γ), a qual determina o grau de influência que têm os valores futuros sobre o reforço total. Assim, a expressão do retorno aplicando taxa de amortização é expressa pela seguinte equação:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{4.2}$$

   onde, $0 \leq \gamma \leq 1$, se $\gamma \rightarrow 0$, o agente tem uma visão míope dos reforços, maximizando apenas os reforços imediatos, e se $\gamma \rightarrow 1$, a visão do reforço abrange todos os estados futuros dando maior importância ao estado final, desde que a seqüência R_T seja limitada.

   Um sistema de AR faz um mapeamento de estados em ações baseado nos reforços recebidos. Assim, o objetivo do AR é definido usando-se o conceito de função de reforço, a qual é uma função dos reforços futuros que o agente procura maximizar. Ao maximizar essa função, o objetivo será alcançado de forma ótima. A função de reforço define quais são os bons e maus eventos para os agentes.

4. **Função de Reforço:** As funções de reforço podem ser bastante complicadas, porém existem

pelo menos três classes de problemas freqüentemente usadas para criar funções adequadas a cada tipo de problema:

- *Reforço só no estado final:* Nesta classe de funções, as recompensas são todas zero, exceto no estado final, em que o agente recebe uma recompensa real (ex: $+1$) ou uma penalidade (ex: -1). Como o objetivo é maximizar o reforço, o agente irá aprender que os estados correspondentes a uma recompensa são bons e os que levaram a uma penalidade devem ser evitados.
- *Tempo mínimo ao objetivo:* Funções de reforço nesta classe fazem com que o agente realize ações que produzam o caminho ou trajetória mais curta para um estado objetivo. Toda ação tem penalidade (-1), sendo que o estado final é (0). Como o agente tenta maximizar valores de reforço, ele aprende a escolher ações que minimizam o tempo que leva a alcançar o estado final.
- *Minimizar reforços:* Nem sempre o agente precisa ou deve tentar maximizar a função de reforço, podendo também aprender a minimizá-las. Isto é útil quando o reforço é uma função para recursos limitados e o agente deve aprender a conservá-los ao mesmo tempo em que alcança o objetivo.

5. **Função Valor-Estado:** Define-se uma função valor-estado como o mapeamento do estado, ou par estado-ação em um valor que é obtido a partir do reforço atual e dos reforços futuros.

   Se a função valor-estado considera só o estado s é denotada como $V(s)$, enquanto se é considerado o par estado-ação (s,a), então a função valor-estado é denotada como função valor-ação $Q(s,a)$.

   (a) *Função valor-estado:* Uma vez que os reforços futuros mantêm dependências das ações futuras, as funções valor dependem também da política π que o AR segue. Em um *Processo de Decisão Markoviano* (ver subseção 4.4.2) se define uma função valor-estado $V^{\pi}(s)$ dependente da política π como a equação:

   $$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t = s\} = E_{\pi}\{\sum_{k=o}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\} \tag{4.3}$$

   onde a função $V^{\pi}(s)$ é o valor esperado do retorno para o estado $s_t = s$. Isto é, o somatório dos reforços aplicando a taxa de amortização γ.

   (b) *Função valor-ação:* Se consideramos o par estado-ação, a equação para a função valor-estado $Q^{\pi}(s,a)$ será a seguinte:

   $$Q^{\pi}(s,a) = E_{\pi}\{R_t \mid s_t = s, a_t = a\} = E_{\pi}\{\sum_{k=o}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a\} \tag{4.4}$$

   que é semelhante à equação (4.3), só que considerando o reforço esperado para um estado $s_t = s$ e uma ação $a_t = a$.

As equações (4.3) e (4.4) apresentam as funções valor-estado e valor-ação respectivamente, que dependem exatamente dos valores de reforço, o qual implica o conhecimento completo da dinâmica do ambiente como um *Processo de Decisão Markoviano*.

4.4 Fundamentos Matemáticos

Existem dois conceitos que devem ser conhecidos para facilitar a modelagem de um problema como um sistema de Aprendizagem por Reforço. A seguir, apresentamos uma breve descrição destes conceitos.

4.4.1 Propriedade de Markov

Quando a probabilidade de transição de um estado s para um estado s' depende apenas do estado s e da ação a adotada em s, isso significa que o estado corrente fornece informação suficiente para o sistema de aprendizado decidir que ação deve ser tomada. Quando o sistema possui essa característica, diz-se que ele satisfaz a *Propriedade de Markov* [1].

No caso mais geral, se a resposta em $t+1$ para uma ação efetuada em t depende de todo o histórico de ações até o momento atual, a dinâmica do ambiente é definida pela especificação completa da distribuição de probabilidades, como mostra a equação abaixo:

$$Pr\{s_{t+1} = s', r_{t+1} = r \mid s_t,\ a_t,\ r_t,\ s_{t-1},\ a_{t-1}, \ldots,\ r_1,\ s_0,\ a_0\}$$

onde a probabilidade (P_r) do estado s_{t+1} ser o estado s' e o reforço r_{t+1} ser igual a r é uma função que depende de todos os estados, ações e reforços passados.

Se a resposta do ambiente em $t+1$ depende apenas dos estados e reforços em t, então, a probabilidade da transição para o estado s' é dada pela expressão da equação abaixo.

$$P^a_{s,s'} = Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\}$$

A probabilidade de transição satisfaz às seguintes condições: 1) $P^a_{s,s'} \geq 0$, $\forall s, s' \in S$, $\forall a \in A(s)$; e 2) $\sum_{s' \in S} P^a_{s,s'} = 1$, $\forall s \in S$, $\forall a \in A(s)$

A *Propriedade de Markov* é de fundamental importância na AR, uma vez que tanto as decisões como os valores são funções apenas do estado atual, abrindo a possibilidade de métodos de soluções incrementais, onde pode-se obter soluções a partir do estado atual e para cada um dos estados futuros, como é feito no método de *Programação Dinâmica* (ver subseção 4.5.1).

4.4.2 Processos de Decisão Markoviano (PDM)

Segundo Bellman em [1], um *Processo de Decisão Markoviano* é definido como um conjunto de estados S, $\forall s \in S$, um conjunto de ações $A(s)$, um conjunto de transições entre estados associadas com as ações e um conjunto de probabilidades P sobre o conjunto S que representa uma modelagem das transições entre os estados. Assim, dado um par de estado e ação, a probabilidade do estado s passar a um estado s' é:

$$P^a_{s,s'} = Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\}$$

onde P_r é o operador de probabilidade; neste caso representa-se a probabilidade do estado s_{t+1} ser s', sempre que o estado s_t for igual a s e a ação a_t for igual a a. Desta forma, a dependência que o estado seguinte s_{t+1} seja o estado s' está relacionada a tomar a ação a no instante t.

De forma análoga, dados um estado e ação atuais e um estado seguinte s', o valor esperado do retorno é:

$$R^a_{s,s'} = E\{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\}$$

onde E é o valor esperado do retorno r_{t+1}, sempre que o estado s_t no instante t passe a ser o estado s' no instante $t+1$.

Os valores de probabilidade $P^a_{ss'}$ e retorno esperado $R^a_{ss'}$ determinam os aspectos mais importantes da dinâmica de um **PDM finito**. Podemos caracteriza-lo como: 1) um ambiente evolui probabilisticamente baseado num conjunto finito e discreto de estados; 2) para cada estado do ambiente, existe um conjunto finito de ações possíveis; 3) cada passo que o sistema de aprendizado executar uma ação, é verificado um custo positivo ou negativo para o ambiente em relação à ação; e 4) estados são observados, ações são executadas e reforços são relacionados.

Assim para quase todos os problemas de *Aprendizagem por Reforço* é suposto que o ambiente tenha a forma de um *Processo de Decisão Markoviano*, desde que seja satisfeita a *Propriedade de Markov* no ambiente. Nem todos os algoritmos de AR necessitam uma modelagem PDM inteira do ambiente, mas é necessário ter-se pelo menos a visão do ambiente como um conjunto de estados e ações [41].

Exemplo:

Podemos demonstrar os conceitos apresentados até o momento através de um exemplo simples de um robô reciclador, não particularmente realístico, apresentado na seção 3.6 do livro do Sutton e Barto [41].

O robô funciona a bateria e tem como objetivo coletar o maior número de latas possíveis, gastando o mínimo de energia. Suas ações são baseadas por um agente AR, onde este decide se o robô irá: 1) procurar ativamente por latas por um determinado período de tempo, 2) permanecer parado a espera que alguém lhe traga as latas, ou 3) voltar para a base para recarregar as suas baterias. Essa decisão tem que ser tomada periodicamente ou sempre que determinados eventos ocorram, como encontrar uma lata vazia.

Na Figura 4.2, podemos ver caracterizado o problema de aprendizagem, onde temos um sinal que representa as escolhas feitas pelo agente (ação), um sinal que indica o estado do ambiente (estado) e outro que define as metas a serem alcançadas (ganhos).

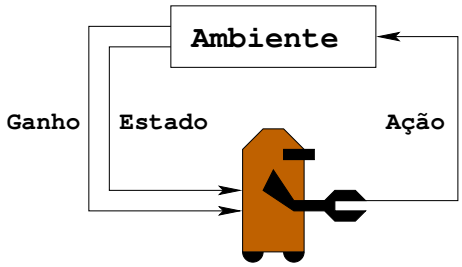

Figura 4.2: Modelo padrão de aprendizagem do robô reciclador

O agente toma suas decisões com base no nível de energia da bateria, onde podemos distinguir dois níveis, representados no livro [41] por *(high, low)*, de modo que o espaço de estados é dado por $S = \{high,\ low\}$, e suas possíveis ações são conhecidas como *(search, wait, recharge)*. Sabendo que o agente se baseia no nível de energia para tomar suas ações, quando este encontrar-se no nível *high*, tomar a ação *recharge* não seria sensata, assim não incluímos essa ação para este estado e o conjunto de ações do agente então passa a ser: $A(high) = \{search,\ wait\}$ e $A(low) = \{search,\ wait,\ recharge\}$.

A cada lata coletada é adicionado uma recompensa $+1$ e caso ele fique sem carga o mesmo leva uma punição (-3). Como queremos que o robô colete o maior número possível de latas adotou-se um retorno $R^{search} \geq R^{wait}$, visto que terá um melhor resultado, e adotou-se que nenhuma lata poderá ser coletada durante os períodos de recarga e quando a bateria estiver esgotada. Por este ser um sistema PDM finito, podemos apresentar as probabilidades de transição e os ganhos previstos, como na Tabela 5.2 (retirada do livro [41]).

$s = s_t$	$s' = s_{t+1}$	$a = a_t$	$P^a_{s,s'}$	$R^a_{s,s'}$
high	high	search	α	R^{search}
high	low	search	$1-\alpha$	R^{search}
low	high	search	$1-\beta$	-3
low	low	search	β	R^{search}
high	high	wait	1	R^{wait}
high	low	wait	0	R^{wait}
low	high	wait	0	R^{wait}
low	low	wait	1	R^{wait}
low	high	recharge	1	0
low	low	recharge	0	0

Tabela 4.1: Tabela das probabilidades das transições e retornos previstas para o PDM finito

Estando a bateria no nível *high* e executando a ação *search* tem-se duas possibilidades: uma com a probabilidade (α) se a bateria continuar no mesmo nível, e outra com uma probabilidade $(1-\alpha)$ se mudar para o nível *low*. Caso esteja no nível *low* e execute a ação *search* tem-se duas possibilidades: uma com probabilidade (β) de continuar no mesmo nível, e uma probabilidade $(1-\beta)$ de esgotar a bateria, neste caso o robô terá que ser salvo para recarregar a sua bateria. Pelo objetivo proposto o robô não deve ficar sem energia e caso isso ocorra ele é severamente punido. Quando se escolhe a

opção *wait* não há gasto de energia, ficando o robô no mesmo estado, desta forma aquelas opções em que há mudanças de estado têm probabilidade 0 de ocorrer. No caso de escolha da ação *recharge* o próximo estado será de bateria *high*, não havendo outra possibilidade.

Outra forma de representar a dinâmica de um PDM finito é através de um diagrama de transição de estados visto na Figura 4.3 (retirada do livro [41]), onde este possui um nó para cada estado possível (grande círculo marcando o nome do estado), e um nó para cada par estado-ação (círculo pequeno cheio), as suas setas representam as probabilidades ($P^a_{ss'}$) de executar a transição do estado s para s' se o agente tomar a decisão a e o retorno ($R^a_{ss'}$) que é obtido se executar a transição de s para s' após tomar a ação a.

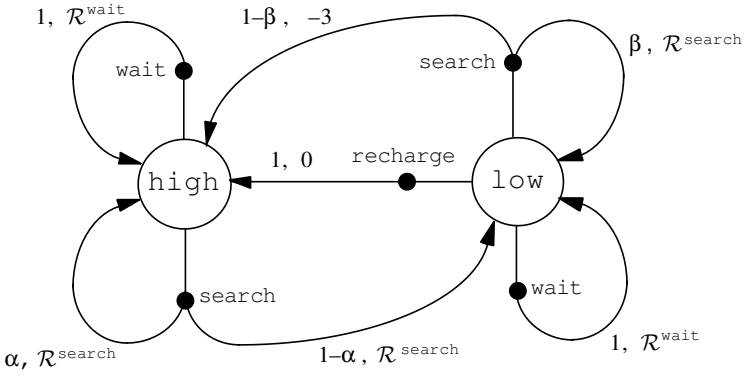

Figura 4.3: Gráfico das transições do robô reciclador

4.5 Métodos de Solução

Para solucionar o problema de Aprendizagem por Reforço, que são: 1) avaliação de política; e 2) encontrar a política ótima, existem três classes de métodos fundamentais, que apresentaremos e analisaremos nas subseções a seguir. Também ilustraremos estes métodos com o exemplo do robô reciclador, visto anteriormente.

4.5.1 Programação Dinâmica (PD)

Segundo Bellman em [1], a *Programação Dinâmica* tem a vantagem de ser matematicamente bem fundamentada, mas exige uma modelagem bem precisa do ambiente como um *Processo de Decisão Markoviano*.

Programação Dinâmica é uma coleção de algoritmos que podem obter políticas ótimas sempre que exista uma modelagem perfeita do ambiente como um PDM, isto é, como um conjunto de estados, ações, retornos e probabilidades da transição em todos os estados. Os algoritmos clássicos de PD são usados de forma limitada, uma vez que a modelagem perfeita do ambiente como PDM exige um grande custo computacional, porém, fornece um bom fundamento para o conhecimento dos outros métodos usados na solução do problema de AR e um padrão de comparação.

A dinâmica do sistema é dada por um conjunto de probabilidades de transição de estado, $P^a_{ss'} = Pr\{s_{t+1} = s',\, s_t = s,\, a_{t+1} = a\}$, e por um conjunto de reforços imediatos esperados, $R^a_{ss'} = E\{r_{t+1} \mid a_t = a,\, s_t = s,\, s_{t+1} = s'\}$, para todo $s, s' \in S$, $a \in A(s)$.

Avaliação da Política

Segundo Monteiro em [26], as escolhas das ações do agente são feitas a partir de uma função do estado, chamada política ($\pi : S \rightarrow A$). O valor de utilidade de um estado, dada uma política, é o reforço esperado partindo do estado e seguindo a política apresentada na equação 4.3. E paralelamente a essa função valor-estado, existe uma função valor-ação para a política π, que é definida pela equação 4.4.

As funções valores V^π e Q^π podem ser estimadas por experiências. Por exemplo, se um agente seguir uma política π e mantiver uma média, para cada estado encontrado, dos atuais retornos que tem seguido aquele estado, então a média convergirá ao valor-estado V^π. Se médias diferentes forem mantidas para cada ação feita em um estado, então estas médias convergirão similarmente aos valores da ação Q^π.

Uma propriedade fundamental das funções de valor usadas durante a *Aprendizagem por Reforço* e *Programação Dinâmica* é que elas satisfazem particularidades recursivas:

$$\begin{aligned} V^\pi &= E_\pi\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\} \\ &= E_\pi\{r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s\} \\ &= \sum_{a \in A(s)} \pi(s,a) \sum_{s' \in S} P^a_{s,s'} \left[R^a_{s,s'} + \gamma E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_{t+1} = s' \right\} \right] \\ &= \sum_{a \in A(s)} \pi(s,a) \sum_{s' \in S} P^a_{s,s'} \left[R^a_{s,s'} + \gamma V^\pi(s') \right], \quad \forall s \in S \end{aligned} \tag{4.5}$$

A equação 4.5 é a equação de Bellman para V^π. Expressa um relacionamento entre o valor de um estado e os valores de seus estados sucessores. Calcula todas possíveis médias excessivamente, ponderando cada um por sua probabilidade de ocorrer. Indica que o valor do estado s inicial deve igualar ao valor amortizado do estado seguinte previsto, mais a recompensa esperada ao longo da caminho. O algoritmo para resolver a avaliação de política pode ser visto abaixo.

Avaliação da Politica Iterativa

```
Entrar π, a politica a ser avaliada
Inicializar V(s) = 0 para todo s ∈ S
Repete
  Δ ← 0
  Para cada s ∈ S:
    v ← V(s)
```

$V(s) \leftarrow \sum_{a \in A(s)} \pi(s,a) \sum_{s' \in S} P^a_{ss'} [R^a_{ss'} + \gamma V(s')]$
$\Delta \leftarrow max(\Delta, \mid v - V(s) \mid)$
ate $\Delta < \theta$ (um pequeno numero positivo)
Sair $V \approx V^\pi$

Como demonstração, resolveremos o exemplo do robô reciclador. Para facilitar o entendimento adotamos abreviações nos estados (ex: *high* = h) e nas ações (ex: *search* = s), e adotamos os valores abaixo na implementação do algoritmo no Matlab, fazendo uma avaliação iterativa da política.

$$\begin{array}{rclcrclcrclcrcl} R^w & = & 1 & , & \alpha & = & 0.8 & , & \pi(l,r) & = & \frac{1}{3} & , & \pi(h,w) & = & \frac{1}{2} \\ R^s & = & 10 & , & \beta & = & 0.2 & , & \pi(l,w) & = & \frac{1}{3} & , & \pi(h,s) & = & \frac{1}{2} \\ R^r & = & 0 & , & \gamma & = & 0.9 & , & \pi(l,s) & = & \frac{1}{3} & , & & & \\ R^{f,r} & = & -3 & , & & & & & & & & & & & \end{array}$$

Resolvendo a equação de Bellman, temos:

$$\begin{aligned} V^\pi(l) &= \sum_{a \in A(l)} \pi(l,a) \sum_{s' \in S} P^a_{l,s'} [R^a_{l,s'} + \gamma V^\pi(s')] \\ &= \pi(l,r) \left\{ P^r_{l,h} [R^r_{l,h} + \gamma V^\pi(h)] + P^r_{l,l} [R^r_{l,l} + \gamma V^\pi(l)] \right\} + \\ &\quad \pi(l,w) \left\{ P^w_{l,h} [R^w_{l,h} + \gamma V^\pi(h)] + P^w_{l,l} [R^w_{l,l} + \gamma V^\pi(l)] \right\} + \\ &\quad \pi(l,s) \left\{ P^s_{l,h} [R^s_{l,h} + \gamma V^\pi(h)] + P^s_{l,l} [R^s_{l,l} + \gamma V^\pi(l)] \right\} \\ &= \frac{1}{3} \{1[0 + 0.9V^\pi(h)] + 0[0 + 0.9V^\pi(l)]\} + \\ &\quad \frac{1}{3} \{0[0 + 0.9V^\pi(h)] + 1[1 + 0.9V^\pi(l)]\} + \\ &\quad \frac{1}{3} \{(1-\beta)[-3 + 0.9V^\pi(h)] + \beta[10 + 0.9V^\pi(l)]\} \\ &= 0.47 + 0.54V^\pi(h) + 0.36V^\pi(l) \end{aligned}$$

$$\begin{aligned} V^\pi(h) &= \sum_{a \in A(h)} \pi(h,a) \sum_{s' \in S} P^a_{h,s'} [R^a_{h,s'} + \gamma V^\pi(s')] \\ &= \pi(h,w) \left\{ P^w_{h,h} [R^w_{h,h} + \gamma V^\pi(h)] + P^w_{h,l} [R^w_{h,l} + \gamma V^\pi(l)] \right\} + \\ &\quad \pi(h,s) \left\{ P^s_{h,h} [R^s_{h,h} + \gamma V^\pi(h)] + P^s_{h,l} [R^s_{h,l} + \gamma V^\pi(l)] \right\} \\ &= \frac{1}{2} \{1[1 + 0.9V^\pi(h)] + 0[0 + 0.9V^\pi(l)]\} + \\ &\quad \frac{1}{2} \{\alpha[10 + 0.9V^\pi(h)] + (1-\alpha)[10 + 0.9V^\pi(l)]\} \\ &= 5.5 + 0.81V^\pi(h) + 0.09V^\pi(l) \end{aligned}$$

Resolvendo as equações lineares acima no Matlab, obtivemos os seguintes resultados: $V^{\pi}(l) = 45.19$ e $V^{\pi}(h) = 50.35$, que indicam a solução encontrada usando a equação de Bellman através do cálculo iterativo para as seqüências de $V^{\pi}(s)$.

Política Ótima

A *Programação Dinâmica* organiza e estrutura a busca de boas políticas a partir das funções de valor. Deste modo, políticas ótimas são obtidas sempre que funções de valor ótimas são obtidas. Usualmente, as funções valor-estado ótimas são denotadas por $V^*(s)$ e valor-ação $Q^*(s,a)$ as quais satisfazem as equações de otimização de Bellman, como é expresso nas equações abaixo, respectivamente:

$$\begin{aligned} V^*(s) &= \max_{a \in A(s)} Q^{\pi^*}(s,a) \\ &= \max_a E_{\pi^*}\{R_t \mid s_t = s,\, a_t = a\} \\ &= \max_a E_{\pi^*}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s,\, a_t = a\right\} \\ &= \max_a E_{\pi^*}\left\{r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s,\, a_t = a\right\} \\ &= \max_a E\{r_{t+1} + \gamma V^*(s_t+1) \mid s_t = s,\, a_t = a\} \\ &= \max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^*(s')] \end{aligned} \tag{4.6}$$

$$\begin{aligned} Q^*(s,a) &= E\left\{r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') \mid s_t = s,\, a_t = a\right\} \\ &= \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma \max_{a'} Q^*(s', a')] \end{aligned} \tag{4.7}$$

Na Equação (4.6) a função de valor ótimo $V^*(s)$ é encontrada como o máximo das funções de valor esperadas segundo a ação selecionada. E a partir de Q^*, pode-se determinar uma política ótima simplesmente como $\pi^*(s) = arg\max_{a \in A(s)} Q^*(s,a)$.

Para atualizar as funções valor com a finalidade de melhorar a política, utiliza-se a iteração de valores. A função valor $V_{k+1}(s)$ do estado s para o passo $k+1$ de avaliação é encontrada como na equação:

$$\begin{aligned} V_{k+1}(s) &= \max_a E\{r_{t+1} + \gamma V_k(s_{t+1}) \mid s_t = s,\, a_t = a\} \\ &= \max_a \sum_{s'} P^a_{s,s'}[R^a_{s,s'} + \gamma V_k(s')], \forall s \in S. \end{aligned} \tag{4.8}$$

onde o valor atualizado $V_{k+1}(s)$ é encontrado a partir dos valores armazenados no passo k da seqüência de iterações, aplicando a equação de otimalidade de Bellman (4.6). Esta seqüência de iterações deve alcançar no ponto final a política ótima $V^*(s)$.

O método de procura da política ótima da *Programação Dinâmica* exige a varredura de todos os estados no espaço de estados do modelo PDM, fazendo com que exista um grande custo computacional para modelagens complexas, resultando em uma desvantagem do método.

O termo gulosa (*greedy*) é usado para descrever um procedimento de busca ou de decisão, que seleciona alternativas baseadas somente em considerações locais e/ou imediatas, sem considerar a possibilidade que tal seleção poder encontrar no futuro alternativas melhores. Conseqüentemente, descreve as políticas que as ações baseiam somente em conseqüências à curto prazo.

Valor de Iteração

Inicializar V de forma arbitraria, por exemplo $V(s) = 0$, para todo $s \in S$

Repete
  $\Delta \leftarrow 0$
  Para cada $s \in S$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_{a \in A(s)} \sum_{s' \in S} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
    $\Delta \leftarrow max(\Delta, \mid v - V(s) \mid)$
  ate $\Delta < \theta$ (um pequeno numero positivo)

Sair uma politica deterministica, π, tal que
  $\pi(s) = arg \max_{a \in A(s)} \sum_{s' \in S} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$

Continuando com o exemplo do robô reciclador, resolveremos as equações de Otimalidade de Bellman, assim temos para a função valor-estado:

$$
\begin{aligned}
V^*(l) &= \max_{a \in A(s)} \sum_{s' \in S} P^a_{s,s'}[R^a_{s,s'} + \gamma V^*(s')] \\
&= \max \left\{ \begin{array}{l} P^r_{l,h}[R^r_{l,h} + \gamma V^*(h)] + P^r_{l,l}[R^r_{l,l} + \gamma V^*(l)], \\ P^w_{l,h}[R^w_{l,h} + \gamma V^*(h)] + P^w_{l,l}[R^w_{l,l} + \gamma V^*(l)], \\ P^s_{l,h}[R^s_{l,h} + \gamma V^*(h)] + P^s_{l,l}[R^s_{l,l} + \gamma V^*(l)] \end{array} \right\} \\
&= \max \left\{ \begin{array}{c} 0.9V^*(h), \\ 1 + 0.9V^*(l), \\ -0.4 + 0.72V^*(h) + 0.18V^*(l) \end{array} \right\} \\
V^*(h) &= \max_{a \in A(s)} \sum_{s' \in S} P^a_{s,s'}[R^a_{s,s'} + \gamma V^*(s')] \\
&= \max \left\{ \begin{array}{l} P^w_{h,h}[R^w_{h,h} + \gamma V^*(h)] + P^w_{h,l}[R^w_{h,l} + \gamma V^*(l)]\}, \\ P^s_{h,h}[R^s_{h,h} + \gamma V^*(h)] + P^s_{h,l}[R^s_{h,l} + \gamma V^*(l)]\} \end{array} \right\} \\
&= \max \left\{ \begin{array}{c} 1 + 0.9V^*(h), \\ 10 + 0.72V^*(h) + 0.18V^*(l) \end{array} \right\}
\end{aligned}
$$

e, para a função valor-ação:

$$\begin{aligned} Q^*(l,r) &= \sum_{s' \in S} P^r_{l,s'}[R^r_{l,s'} + \gamma \max_{a' \in A(s)} Q^*(s',a')] \\ &= P^r_{l,h}[R^r_{l,h} + \gamma \max\{Q^*(h,w), Q^*(h,s)\}] \\ &= 0.9 \max\{Q^*(h,w), Q^*(h,s)\} \end{aligned}$$

$$\begin{aligned} Q^*(l,w) &= \sum_{s' \in S} P^w_{l,s'}[R^w_{l,s'} + \gamma \max_{a' \in A(s)} Q^*(s',a')] \\ &= P^w_{l,l}[R^w_{l,l} + \gamma \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\}] \\ &= 1 + 0.9 \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\} \end{aligned}$$

$$\begin{aligned} Q^*(l,s) &= \sum_{s' \in S} P^s_{l,s'}[R^s_{l,s'} + \gamma \max_{a' \in A(s)} Q^*(s',a')] \\ &= P^s_{l,l}[R^s_{l,l} + \gamma \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\}] + P^s_{l,h}[R^s_{l,h} + \gamma \max\{Q^*(h,w), Q^*(h,s)\}] \\ &= -0.4 + 0.18 \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\}] + 0.72 \max\{Q^*(h,w), Q^*(h,s)\}] \end{aligned}$$

$$\begin{aligned} Q^*(h,w) &= \sum_{s' \in S} P^w_{h,s'}[R^w_{s,s'} + \gamma \max_{a' \in A(s)} Q^*(s',a')] \\ &= P^w_{h,h}[R^w_{h,h} + \gamma \max\{Q^*(h,w), Q^*(h,s)\}] \\ &= 1 + 0.9 \max\{Q^*(h,w), Q^*(h,s)\} \end{aligned}$$

$$\begin{aligned} Q^*(h,s) &= \sum_{s' \in S} P^s_{h,s'}[R^s_{s,s'} + \gamma \max_{a' \in A(s)} Q^*(s',a')] \\ &= P^s_{h,l}[R^s_{h,l} + \gamma \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\}] + P^s_{h,h}[R^s_{h,h} + \gamma \max\{Q^*(h,w), Q^*(h,s)\}] \\ &= 10 + 0.18 \max\{Q^*(l,w), Q^*(l,s), Q^*(l,r)\}] + 0.72 \max\{Q^*(h,w), Q^*(h,s)\}] \end{aligned}$$

Onde obtivemos no Matlab os seguintes resultados: $V^*(l) = 91.98$, $V^*(h) = 88.41$, $Q^*(l,r) = 79.57$, $Q^*(l,w) = 74.78$, $Q^*(l,s) = 81.98$, $Q^*(h,w) = 80.57$ e $Q^*(h,s) = 88.41$. Observe que os valores $V^*(l)$ e $V^*(h)$ são bem superiores aos obtidos com a política equiprovável π, $V^\pi(l) = 45.19$ e $V^\pi(h) = 50.35$, conforme resolvidos acima. A política π^* obtida a partir de Q^* é $Q^*(h,s) = 88.41$.

4.5.2 Monte Carlo (MC)

O método de *Monte Carlo* [37] não precisa da modelagem do ambiente e se apresenta de forma simples em termos conceituais, baseia-se no cálculo da média de retornos obtidos em seqüências. Para assegurar-se que exista um valor de retorno bem definido, o método de *Monte Carlo* é utilizado apenas para tarefas episódicas, isto é, a experiência é dividida em episódios que de algum modo alcançam o estado final sem importar as ações que foram selecionadas (exemplo: jogo de xadrez). Desta forma, somente depois da conclusão de um episódio o valor de retorno é obtido e as políticas

são atualizadas. Entretanto, não são viáveis quando a solução do problema é possível apenas de forma incremental, porque para se atualizar, o método de *Monte Carlo* exige que seja alcançado o estado final no processo e com isso o mesmo pode se apresentar lento.

Uma vantagem do método de *Monte Carlo* é que, diferente do método de *Programação Dinâmica*, não necessita de informação completa do ambiente, apenas necessita das amostras da experiência como seqüências de dados, ações e reforços a partir de uma interação real ou simulada com o ambiente.

O aprendizado a partir de experiência real é notável, uma vez que não exige o conhecimento *a priori* das dinâmicas do ambiente, e ainda, pode levar a um comportamento ótimo. Embora seja requerida uma modelagem, esta deve apenas gerar transições de estados, sem precisar todo o conjunto de distribuições de probabilidade para todas as possíveis transições, como é exigido pela *Programação Dinâmica*.

Avaliação da Política

Supondo-se que o método de *Monte Carlo* é considerado para obter a função valor sob uma dada política, que é representada pelo retorno esperado, isto é, a acumulação amortizada dos futuros reforços desde o estado s até o estado desejado. Uma forma de se aproximar do valor de retorno esperado a partir da experiência é calcular a média dos retornos observados após visitar esse estado. Na medida em que mais retornos são observados, a média deve se aproximar do valor real esperado, sendo esta conseqüência o princípio básico do método de *Monte Carlo*.

Seja $V^{\pi}(s)$ a função valor-estado sob a política π. Dados um conjunto de episódios obtidos sob a mesma política passando pelo estado s (cada ocorrência de s em um episódio é chamada de visita a s), existem duas variantes do método de *Monte Carlo*: A primeira, chamada de *Every-Visit MC Method*, estima a função de valor como a média dos retornos após todas as visitas ao estado s, enquanto a segunda, chamada de *First-Visit MC Method*, estima a função de valor como a média dos retornos após a primeira visita ao estado s.

De qualquer forma, se o número de visitas for infinito, ambas as variantes do método de *Monte Carlo*, convergem ao valor $V^{\pi}(s)$. Podemos ver através do algoritmo abaixo.

Primeira visita ao metodo MC para estimar V^{π}

```
Inicializar:
  π ← politica a ser avaliada
  V ← uma função valor-estado arbitraria
  Retornos(s) ← uma lista vazia, para todo s ∈ S

Repetir sempre:
  (a) Gerar um episodio usando π
```

(b) Para cada estado s que aparece no episodio:
$R \leftarrow$ quantidade de retorno apos a primeira ocorrência de s
Adiciona R a lista *Retornos(s)*
$V(s) \leftarrow$ media de (*Retornos(s)*)

Resolvendo o algoritmo para o exemplo do robô reciclador, onde definiu-se como episódio uma seqüência de 2000 passos de iterações, obtivemos um comportamento que pode ser observado na Figura 4.4, com resultados bem próximos ao obtido na PD, $V^{\pi}(l) = 45.43$ e $V^{\pi}(h) = 50.32$.

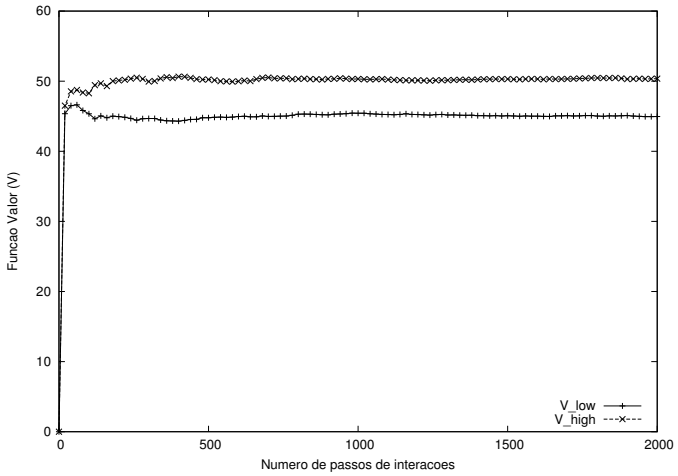

Figura 4.4: Comportamento do algoritmo MC para a avaliação de política

Política Ótima

A fim de melhorar a política é necessário fazer com que esta seja mais gulosa para a função valor-estado $V^{\pi}(s)$ atual. Neste caso é conveniente assumir como valor de retorno a função valor-ação $Q^{\pi}(s,a)$. Assim, uma política gulosa para uma função valor-ação $Q(s,a)$ é aquela que para um estado s toma a ação que maximiza o valor Q como na equação abaixo.

$$\pi(s) = arg \max_{a \in A(s)} Q(s,a) \tag{4.9}$$

Desta forma, uma melhora na política pode ser obtida fazendo a política π_{k+1} ser gulosa em respeito à função valor-ação Q^{π_k}, logo após a avaliação da função valor-ação Q^{π_k} de π_k, podemos gerar uma seqüência de avaliação da função e melhora da política, abaixo:

$$\pi_0 \xrightarrow{A} Q^{\pi_0} \xrightarrow{M} \pi_1 \xrightarrow{A} Q^{\pi_1} \ldots \xrightarrow{A} Q^{\pi_k} \xrightarrow{M} \pi_{k+1} \ldots \xrightarrow{M} \pi^* \xrightarrow{A} Q^*$$

Onde A indica avaliação de política e M indica processo guloso de melhora de política. Segundo este processo se o número de episódios é muito grande, a função valor se aproximará à função valor-ação ótima Q^*.

MC com exploração no inicio

Inicializa, para todo $s \in S$, $a \in A(s)$:
  $Q(s,a) \leftarrow$ de forma arbitraria
  $\pi(s) \leftarrow$ de forma arbitraria
  $Retorno(s,a) \leftarrow$ lista vazia para $s \in S$

Repetir infinitamente:
  (a) Gerar um episodio usando π explorando novos estados
  (b) Para cada par (s,a) gerado no episodio:
    $R \leftarrow$ retorno apos a primeira ocorrência de (s,a)
    Adiciona R a lista *Retorno(s, a)*
    $Q(s,a) \leftarrow$ media de (*Retorno(s,a)*)
  (c) Para cada s do episodio:
    $\pi(s) \leftarrow arg\max_a Q(s,a)$

4.5.3 Diferença Temporal (DT)

Os métodos de *Diferenças Temporais* não exigem um modelo exato do sistema e permitem ser incrementais, da mesma forma que os métodos de Monte Carlo.

Os métodos de *Diferenças Temporais* são uma combinação de características dos métodos de *Monte Carlo* com as idéias da *Programação Dinâmica*, que buscam estimar valores de utilidade para cada estado no ambiente [41]. Em outras palavras, quanto mais próximo da convergência do método, mais o agente tem certeza de qual ação tomar em cada estado.

O aprendizado é feito diretamente a partir da experiência, sem a necessidade de uma modelagem completa do ambiente, como característico do método de *Monte Carlo*, mas leva vantagem em cima deste por atualizar as estimativas da função valor a partir de outras estimativas já aprendidas em estados sucessivos (*bootstrap*), sem a necessidade de alcançar o estado final de um episódio antes da atualização. Neste caso a avaliação de uma política é abordada como um problema de predição, isto é, estimar a função valor V^{π} sob a política π.

Avaliação da Política - Predição DT

Tanto DT como MC utilizam a experiência para resolver o problema da predição. Dada certa experiência sob a política π, se é visitado um estado intermediário s_t , ambos os métodos atualizam suas estimativas $V^{\pi}(s_t)$ baseando-se no acontecido depois de visitado o estado. Sendo que o método

de *Monte Carlo* espera até que o retorno total seja conhecido e usa esse retorno como objetivo para a atualização de $V^{\pi}(s_t)$, como aparece na equação abaixo.

$$V^{\pi}(s_t) \leftarrow V^{\pi}(s_t) + \alpha[R_t - V^{\pi}(s_t)] \tag{4.10}$$

onde R_t representa o retorno atual no instante t, e o símbolo α é uma constante de atualização (taxa de aprendizagem), $\alpha \in [0,1]$.

Os métodos de *Diferenças Temporais* não necessitam alcançar o estado final de um episódio, e sim o estado seguinte no instante $t+1$. Em DT são utilizados, o valor de reforço imediato r_{t+1} e a função de valor estimada $V^{\pi}(s_{t+1})$ para o próximo estado ao invés do valor real de retorno R_t como no método de *Monte Carlo*, executando a atualização imediatamente após cada passo. Com estas condições, nos métodos de *Diferenças Temporais* a Equação 4.10 converte-se na Equação abaixo.

$$V^{\pi}s_t \leftarrow V^{\pi}s_t + \alpha[r_{t+1} + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t)] \tag{4.11}$$

onde o objetivo para atualização é o valor $r_{t+1} + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t)$ que precisamente define a diferença no tempo t e $t+1$, característica esta que neste método recebe o nome de *Diferenças Temporais*. Como a atualização é feita a partir do estado seguinte, os métodos DT são conhecidos como métodos *single-step*.

Predição DT para estimar V^{π}

Inicializar $V(s)$ de forma arbitraria, e π (politica a ser avaliada)
Repete (para cada episodio):
  Inicializar s
  Repete (para cada passo do episodio):
    $a \leftarrow$ ação dada por π para s
    Tomar a ação a, observar retorno r e proximo estado s'
    $V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$
    $s \leftarrow s'$
  ate s ser o estado final

Para o exemplo do robô reciclador, implementamos o algoritmo acima, definindo-se como episódio uma seqüência de 2000 passos de iterações, e o comportamento obtido pode ser acompanhado na Figura 4.5, onde obtivemos os seguintes resultados finais: $V(l) = 45.86$ e $V(h) = 50.55$, que são bem próximos aos valores obtidos na *Programação Dinâmica* e no *método de Monte Carlo*.

Vantagens dos Métodos de Predição DT

A vantagem mais notável do método DT é a relacionada com o método de *Programação Dinâmica*, onde esta não necessita da modelagem PDM do ambiente, de seus reforços e das distribuições de probabilidade das transições dos seus estados.

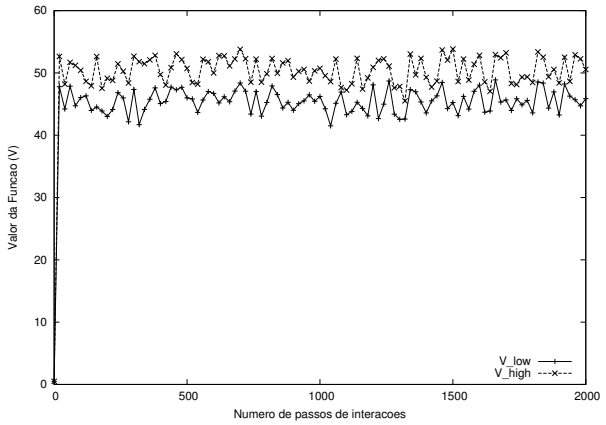

Figura 4.5: Comportamento do algoritmo Predição DT para a avaliação de política

A vantagem seguinte diz respeito ao método de *Monte Carlo*, visto que DT pode ser implementado de forma totalmente incremental para aplicações *On-Line*; o método de *Monte Carlo* deve aguardar até o final de um episódio para obter o retorno verdadeiro, enquanto DT só necessita aguardar até o estado seguinte. Em aplicações em que os ambientes são definidos como sendo contínuo, o conceito de episódio não é aplicável com facilidade.

Embora as atualizações das funções valor não sejam feitas a partir de reforços reais, mas de valores supostos, é garantida a convergência até a resposta correta. Tanto em DT como em MC a convergência às predições corretas tem forma assintótica. Dentre os dois métodos, algum deles devem convergir mais rápido; a resposta ainda não é dada formalmente, uma vez que até este momento não existe uma demonstração matemática de qual dos métodos é o mais rápido. Mesmo assim, é mostrado experimentalmente que os métodos DT são mais rápidos para tarefas estocásticas [44].

4.6 *Q-learning*

Um dos maiores avanços na área de *AR* foi o desenvolvimento de um algoritmo baseado em *Diferenças Temporais* que dispensa a política, *(off-policy methods)* conhecido como *Q-learning*. A versão mais simples, *One-step Q-learning* [48], é definida pela seguinte expressão:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \ max_a Q(s_{t+1},a) - Q(s_t,a_t)] \tag{4.12}$$

onde a função de valor-ação $Q(s_t,a_t)$ é atualizada a partir do seu valor atual, o reforço imediato r_{t+1}, e a diferença entre a máxima função valor no estado seguinte (encontrando e selecionando a ação do estado seguinte que a maximize), menos o valor da função valor-ação no tempo atual. O fato de selecionar a ação que maximize a função valor no estado seguinte permite achar de uma forma simples a função valor-ação estimada.

Uma característica do *Q-learning* é que a função valor-ação Q aprendida, aproxima-se diretamente da função valor-ação ótima Q^* sem depender da política que está sendo utilizada. Este fato simplifica bastante a análise do algoritmo e permite fazer testes iniciais da convergência. A política

ainda mantém um efeito ao determinar quais pares estado-ação devem ser visitados e atualizados, porém, para que a convergência seja garantida, é necessário que todos os pares estado-ação sejam visitados continuamente e atualizados, por isso *Q-learning* é um método *off-policy* [44].

Algoritmo Q-learning

Inicializar $Q(s,a)$ de forma arbitraria
Repete (para cada episodio):
  Inicializar s
  Repete (para cada passo do episodio):
    Escolher a para s usando politica obtida dado Q (e.g, ε-gulosa)
    Tomar a ação a, observar o proximo estado s', e o retorno r
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a' \in A(s')} Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
  ate s ser o estado final

A política ε-gulosa é definida no algoritmo pela escolha da ação que possui o maior valor esperado, com a probabilidade definida por $(1-\varepsilon)$, e de ação aleatória, com probabilidade ε. Este processo permite que o algoritmo explore o espaço de estados e esta é uma das condições necessárias para garantir que algoritmos RL encontrem a ação ótima.

Para o exemplo do robô reciclador, implementamos o algoritmo *Q-learning* onde cada episódio foi definido como uma seqüência de 5000 passos de iterações, e o comportamento das 1500 primeiras iterações pode ser observado nas Figuras 4.6 e 4.7, onde obtivemos os seguintes resultados finais: $Q(l,w) = 74.37$, $Q(l,s) = 81.46$, $Q(l,r) = 79.70$, $Q(h,w) = 79.96$ e $Q(h,s) = 88.24$ que são bem próximos aos valores obtidos na *Programação Dinâmica*. E com isso constatar que a política ótima π^* obtida a partir de Q^* é $Q(h,s) = 88.24$.

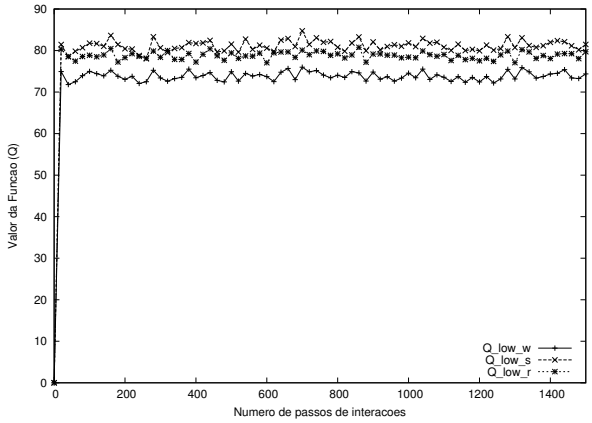

Figura 4.6: Comportamento do algoritmo *Q-learning* para os valores $Q(l)$

Q-learning foi o primeiro método AR a possuir fortes provas de convergência. É uma técnica muito simples que calcula diretamente as ações sem avaliações intermediárias e sem uso de modelo.

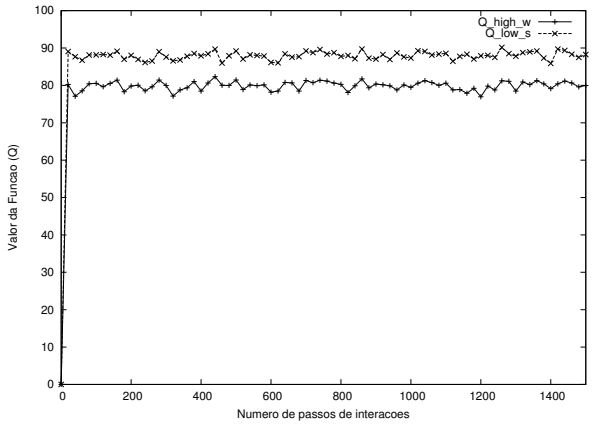

Figura 4.7: Comportamento do algoritmo *Q-learning* para os valores $Q(h)$

Dados os valores Q, existe uma política definida pela execução da ação a, quando o agente esta em um estado s, que maximiza o valor $Q(s,a)$. Em [48] é mostrado que se cada par estado-ação for visitado um número suficientemente grande de vezes e α decrescer apropriadamente, as funções valor-ação Q irão convergir com probabilidade um para Q^* e, conseqüentemente, a política irá convergir para uma política ótima.

A convergência do algoritmo *Q-learning* não depende do método de exploração usado. Um agente pode explorar suas ações a qualquer momento, não existem requisitos para a execução de ações estimadas como as melhores. No entanto, para melhorar o desempenho do sistema é necessária, durante o aprendizado, a busca das ações que maximizam o retorno [21].

Resumidamente, pode-se enumerar os mais importantes aspectos do algoritmo *Q-learning*:

- O objetivo do uso do algoritmo *Q-learning é achar uma regra de controle que maximize cada ciclo de controle*;
- O uso do reforço imediato é indicado sempre que possível e necessário, desde que ele contenha informação suficiente que ajude o algoritmo a achar a melhor solução;
- *Q-learning* é adotado quando o número de estados e ações a serem selecionados é finito e pequeno.

4.7 Considerações Finais

Neste capítulo foi visto que os problemas em Aprendizagem por Reforço são caracterizados por um agente que deve aprender comportamentos através de interações de tentativa e erro em um ambiente dinâmico, ou seja, se uma ação desse é seguida de estados satisfatórios, ou por uma melhoria no estado, então a tendência para produzir esta ação é reforçada. Foi visto também que AR não é definido como um conjunto de algoritmo de aprendizagem, mas como uma classe de problemas de aprendizagem e que todo algoritmo que resolver bem esse problema será considerado um algoritmo de AR [18].

Apresentou-se a seus fundamentos matemáticos, através da *Propriedade de Markov* e do *Processo de Decisão Markoviano* que é quando uma tarefa de AR satisfaz as *Propriedades de Markov*. Foi visto que a AR dispõe de vários métodos de aprendizagem, apresentou-se as características desses métodos, e através de um exemplo estudou-se seus métodos de resolução, avaliando e comparando suas eficiências entre si, com o objetivo de salientar similaridades e diferenças entre estas teorias.

Foi apresentado o algoritmo *Q-learning*, algoritmo adotado no estudo dessa dissertação, por ter uma facilidade de aproximação da função ótima sem depender da política que está sendo utilizada, simplificando a analise e permitindo testes iniciais de convergência.

O capítulo seguinte irá tratar do caso de estudo, onde apresentaremos a rede sob estudo, a implementação de algoritmos, dentre eles o *Q-learning*, apresentaremos os resultados obtidos, as comparações e avaliação entre os mesmos.

Capítulo 5

Aprendizagem por Reforço Aplicada ao Controle de Tráfego Urbano

5.1 Introdução

Como foi visto no Capítulo 2, vários são os problemas relacionados ao sistema de tráfego urbano, e também muitas são as pesquisas em produtos que visam, pelo menos, amenizar os efeitos desses problemas. Hoje há uma tendência dessas pesquisas direcionarem aos sistemas que trabalham com controle de tráfego em tempo real, que atuam rapidamente em resposta aos padrões de fluxo. Segundo Hunt (*apud* [9]), estudos comprovam que estes sistemas apresentam benefícios da ordem de 10% a 15%, reduzindo o tempo de espera em semáforos. Mas sabemos que a adoção desses sistemas esbarra em um alto custo de implementação, além de muitos deles serem antigos, e por isso apresentam recursos computacionais limitados, problemas esses devido a morosidade com que novas tecnologias de controle e gerenciamento de tráfego vem sendo transferidas para as redes de tráfego.

Com o propósito de desenvolver soluções a baixo custo, acredita-se que técnicas de *aprendizagem de máquina* possam desempenhar um papel bastante significativo no aumento da eficácia das tecnologias de controle de tráfego quando empregadas. São muitas as oportunidades para o seu desenvolvimento, variando desde a concepção de modelos inteligentes de predição de fluxo de tráfego até o controle inteligente em tempo real. Em particular, a intensa concentração de pesquisa em *aprendizagem por reforço* e as recentes aplicações com sucesso, principalmente em robótica e controle inteligente, indicam a possibilidade de se aplicar tais técnicas ao controle de tráfego urbano [10].

Como exemplo, temos: Thorpe e Anderson [42] aplicaram o algoritmo SARSA no controle de semáforos em uma pequena rede de tráfego urbano, e verificou-se que o tempo de espera nos semáforos foi reduzido em até 87% se comparado a uma estratégia aleatória de controle. Mais recentemente, Wiering [49], desenvolveu um modelo baseado em algoritmos de *aprendizagem por reforço* com o intuito de controlar semáforos, isto é, um modelo foi usado para estimar a dinâmica do tráfego e o método aplicado foi o de *programação dinâmica*, obtendo também resultados expressivos em comparação a outras estratégias de controle.

Desta forma, esse trabalho vem a ser um passo na direção de se estender as técnicas de aprendizagem por reforço ao controle de redes veiculares. Onde, neste capítulo moldamos o problema da rede do tráfego como um jogo estocástico distribuído entre os múltiplos agentes de controle espalhados sob uma rede de tráfego, fornecendo desse modo uma nova visão do problema e um meio de compreender o jogo entre os agentes. Em foco o desenvolvimento de uma estratégia de controle responsivo usando um algoritmo de aprendizagem livre de modelo (*model-free*), conhecido como *Q-learning*. E por fim a evidência experimental de comparação entre as técnicas de controle convencionais (aleatória uniforme e melhor esforço) e a técnica de controle do algoritmo em estudo, através de uma simulação em uma rede representativa do tráfego de Florianópolis.

5.2 Simulador

As pesquisas na técnica de simulação digital estão evoluindo constantemente. Os motivos são os avanços tecnológicos e as solicitações de usuários de diversas áreas do conhecimento, que necessitam de simulações de sistemas cada vez mais complexos. Existe hoje no mercado uma gama enorme de pacotes de software para simulação de tráfego veicular, dentre eles podemos destacar, o Transyt [46], o SITRA-B+ [11], o SATURN [47], o Green Light District [49], entre outros.

Do ponto de vista prático, a simulação digital constitui-se no projeto e construção de modelos computadorizados de sistemas reais e protótipos, visando compreender seus comportamentos em um conjunto de situações específicas, ou seja, constitui a base do projeto em si, onde são efetuados os vários casos de estudo que nos permitirão investigar e tirar conclusões. Elas permitem analisar os mais distintos cenários sem a necessidade de interferir no sistema real.

Portanto, podemos fazer uso de um simulador, modelando uma rede de tráfego veicular como um problema de aprendizagem por reforço, o qual naturalmente incorpora a natureza estocástica e a dinâmica do fluxo de veículos [9]. Sendo promissoras as vantagens que essa abordagem pode apresentar, como a sua habilidade de aprender sem conhecimento prévio, ou seja, a tarefa seria especificada em termos do que é considerado um resultado correto em vez de se definir exatamente as ações que induzem desempenho ótimo. Outra vantagem seria a sua capacidade de se adaptar a novas situações, isto é, o sistema aprenderia continuamente a partir da experiência, adaptando-se desta forma aos eventos não antecipados. A rede seria caracterizada por:

- S**:** um conjunto discreto dos estados da rede, compreendendo o número de veículos em cada via;
- A**:** um conjunto discreto de ações que o agente controlador pode tomar, o qual contém os comandos factíveis para os semáforos e outros dispositivos de controle;
- R**:** um conjunto de sinais de reforço, correspondendo às penalidades devidas ao comportamento não desejável (ganho obtido), ex: aumento do número de veículos em espera nas intersecções;

- T**:** um modelo da dinâmica do fluxo veicular, expressando como que a rede evolui no tempo de um estado para outro em resposta às ações tomadas a cada passo de tempo; e poderia ser um *Processo de Decisão Markoviano*, dado por um conjunto de probabilidades de transições $\{T(s,a,s') : s, s' \in S \text{ e } a \in A\}$.

O problema é encontrar uma política π, um mapeamento estocástico de estados para ações, que maximize uma função dos sinais de reforço a longo prazo, tal como a esperança matemática do ganho amortizado $E = [\sum_{t=0}^{\infty} \gamma^t r_t]$, onde $0 \leq \gamma \leq 1$ é o valor de amortização e r_t é o sinal de reforço durante o t-ésimo passo. Existe um número de questões e sutilezas ainda não tratadas para esse modelo, mas concentraremos a atenção em fatores que se acredita serem limitantes à aplicação de técnicas de *aprendizagem por reforço* em redes de tráfego, tais como: a explosão combinatória do espaço de estados e a impossibilidade de controle centralizado. Em um cenário mais prático, a operação da rede pode ser entendida como um jogo estocástico distribuído:

- N**:** número de agentes distribuídos ou dispositivos de controle;
- S**:** um conjunto discreto dos possíveis estados da rede, consistindo no número de veículos nas vias;
- $\Theta = \{\Theta_n\}$**:** um conjunto de funções que modela a visão parcial na rede, onde $\Theta_n(s)$ é a fração do estado $s \in S$ percebido pelo agente n. No caso de semáforos, Θ_n proveria informações relativas apenas à vizinhança do semáforo sob controle do agente n;
- $A = A_1 \times \ldots \times A_N$**:** um conjunto de ações, ou seja, dos sinais do controle de tráfego, onde A_n é o subconjunto de ações delegadas ao agente n;
- $T : S \times A \times S \rightarrow [0,1]$**:** uma função de transição do estado que simula a dinâmica do fluxo de tráfego, onde $T(s,a,s')$ é a probabilidade de se alcançar o estado s' partindo do estado s, se a ação de controle comum $a \in A$ for feita em um determinado instante, o que implica $\sum_{s' \in S} T(s,a,s') = 1, \forall s \in S \text{ e } \forall a \in A$, freqüentemente, a função de transição do estado é aproximada pelo simulador.
- $R = \{R_n\}$**:** um conjunto de sinais de reforço, onde $R_n : S \times A \times S \rightarrow \mathbb{R}$, os quais devolvem aos agentes os seus respectivos sinais de reforço correspondente à transição de estado s_t para s_{t+1} se a ação comum a for executada. R_n define claramente o comportamento desejado para um agente n, que pode ser ajustado para se reduzir atrasos nas intersecções ou aumentar o fluxo de tráfego.

Não diferentemente do problema padrão, cada agente n procura uma política π_n que maximize alguma medida do retorno obtido a longo prazo, tal como o seu retorno amortizado $E = [\sum_{t=0}^{\infty} \gamma^t r_{n,t}]$, onde $r_{n,t}$ é o sinal de reforço recebido pelo agente n durante a t-ésima transição de estado. De forma mais compacta, um jogo estocástico distribuído pode ser definido como uma sêxtupla $\Gamma = (N, S, \Theta, A, T, R)$ que agrega os elementos listados acima. Pode-se pensar sobre jogos estocásticos

como generalizações de *Processos de Decisão Markoviano*, no sentido que os múltiplos jogadores tomam decisões independentes e recebem recompensas que dependem das decisões conjuntas e das transições de estado; e modelos da teoria de jogos, no sentido que os jogos estocásticos podem ser vistos como uma série dos jogos que evoluem através do espaço de estados.

5.2.1 A Rede de Interesse

O objetivo deste trabalho é investigar o potencial da modelagem de redes de tráfego como jogos estocásticos distribuídos e medir o desempenho obtido com agentes distribuídos, para isso, uma malha da rede de tráfego de Florianópolis foi escolhida como modelo para simular os experimentos. A rede em questão pode ser vista na área demarcada da Figura 5.1, que compreende boa parte do centro da cidade, apresentando uma taxa de fluxo bem expressiva. Os limites da rede estão apresentados através de letras como segue: A) Norte, que representa a Av. Jornalista Rubens de Arruda Ramos (Beira-Mar); B) Leste, que representa a Av. Professor Othon Gama D'Eça; C) Oeste, que representa a Rua Desembargador Arno Hoeschl; e D) Sul, que representa a Av. Rio Branco.

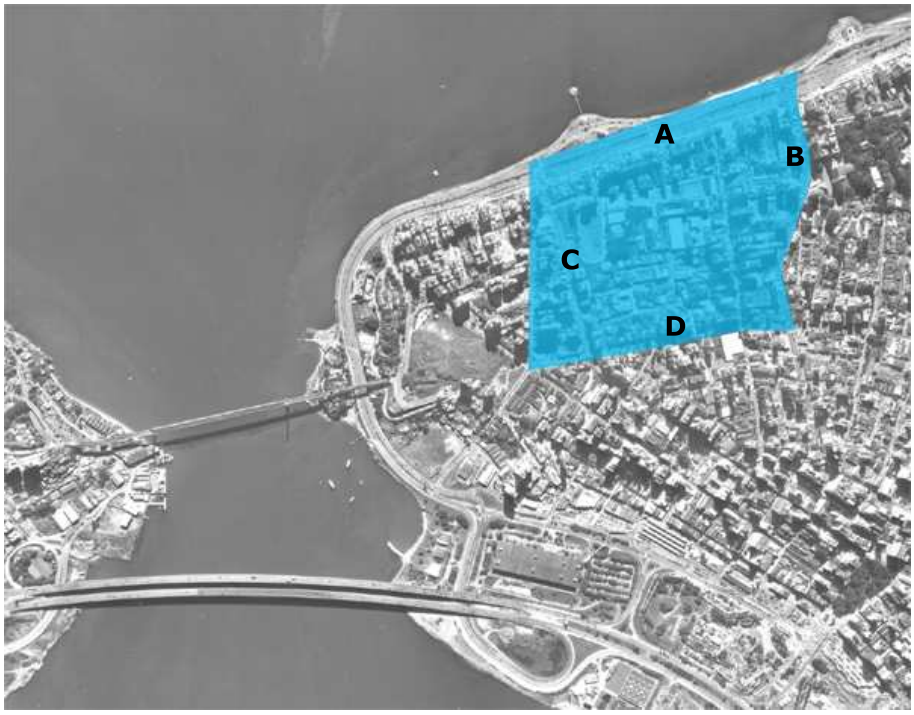

Figura 5.1: Rede de interesse

A rede de interesse pode ser vista com mais detalhes através da Figura 5.2 (figura adaptada a partir da original, obtida no site: `http://www.ipuf.sc.gov.br`), onde podemos ver a representação das intersecções semaforizadas representadas por círculos hachurados e indicadas pela letra *i*, acrescidas de uma numeração que representa o número correspondente do controlador, numeração essa adotada pelo Instituto de Planejamento Urbano de Florianópolis (IPUF), órgão responsável dentre outras atribuições pelo gerenciamento e controle do tráfego urbano da cidade, como forma de facilitar a localização e manutenção. Podemos ver também as intersecções não semaforizadas, representadas

pela letra j, acrescidas de uma numeração que representa o número da intersecção, números esses adotados por nós como forma de uma melhor representação na simulação.

Figura 5.2: As Intersecções na rede de interesse

A rede de interesse, é representada por 16 intersecções, entre semaforizadas ($i_{30}, \ldots, i_{66}$) e não semaforizadas (j_1, j_2 e j_3), e 11 pontos nos quais o tráfego veicular pode entrar e/ou deixar a sub-rede (i_{30}, i_{31}, i_{32}, i_{36}, i_{37}, i_{40}, i_{41}, i_{42}, i_{43}, i_{44} e i_{66}). Os segmentos de via apresentam limites para o número de veículos, os quais entram na sub-rede de acordo com as distribuições de probabilidade que descrevem o comportamento do tráfego. As intersecções são operadas de forma semelhante, ou seja, se uma intersecção i_n libera o fluxo proveniente de uma de suas vias de acesso, então o fluxo proveniente das vias conflitantes são bloqueados, enquanto o fluxo habilitado pode se dirigir para seu destino, obedecendo às distribuições de probabilidades do tráfego. Mas estas possuem diferenças, quanto à quantidade de movimentos, enquanto algumas possuem 3 movimentos possíveis, outras possuem apenas 2 movimentos possíveis.

5.2.2 Simulador em Grafos

Como vimos na Seção 5.2, existe muitos simuladores hoje no mercado, mas esses tendem a apresentar interfaces complexas e embutir uma dinâmica bastante detalhada. No sentido de facilitar as

investigações experimentais, ou seja, a evolução do estado da rede de tráfego em resposta à dinâmica do fluxo e aos sinais de controle, foi implementado um simulador protótipo feito para simular jogos estocásticos. Este simulador foi projetado especificamente para o domínio de redes de tráfego veicular. O simulador foi implementado em linguagem C/C++ ANSI para incentivar seu uso em computadores baseado em UNIX e, igualmente importante, para facilitar a integração com algoritmos de controle semafórico. O simulador adota as seguintes especificações:

- Um grafo direcionado $G = (V, E)$ descrevendo a topologia da rede de tráfego, isto é, seus vértices modelam as intersecções e cruzamentos, enquanto que os arcos modelam as vias de rodagem, como podemos ver na Figura 5.3. Detalhes de como o grafo foi modelado podem ser encontrados no Apêndice (A.1);

- Um conjunto de movimentos excludentes permitidos a cada nó de G, sendo estes especificados por pares de arcos adjacentes, bem como as probabilidades correspondentes à função de transição de estados T. Por exemplo, na intersecção i_{38} existe 2 movimentos possíveis. 1º Movimento: que apresenta um fluxo proveniente da intersecção i_{36} e quando este chega em i_{38}, lhe é proporcionado 2 possibilidades de destino, uma de seguir para i_{39} e outra de seguir para i_{40} (ver Figura 5.3), que representamos da seguinte forma: $[(i_{36}, i_{38}), (i_{38}, i_{39}), P(i_{36}, i_{39})]$, $[(i_{36}, i_{38}), (i_{38}, i_{40}), P(i_{36}, i_{40})]$; e no 2º Movimento: temos um fluxo proveniente de i_{40} que quando chega em i_{38}, também lhe dá 2 possibilidades de destinos, i_{36} ou i_{39}, que representamos da forma como segue: $[(i_{40}, i_{38}), (i_{38}, i_{36}), P(i_{40}, i_{36})]$, $[(i_{40}, i_{38}), (i_{38}, i_{39}), P(i_{40}, i_{39})]$, onde $P(i_n, i_m)$ é a probabilidade do tráfego proveniente de i_n se dirigir para i_m. Como vimos anteriormente, a rede possui intersecções que não apresentam semáforos, mas estas são representadas da mesma forma, como por exemplo, em j_1: 1º Movimento: $[(i_{41}, j_1), (j_1, i_{39}), P(i_{41}, i_{39})]$; e no 2º Movimento: $[(j_2, j_1), (j_1, i_{39}), P(j_2, i_{39})]$. Podemos ver todos movimentos da rede no Apêndice (A.2);

- Parâmetros para cada via, que estabelecem o número máximo de veículos em cada via da rede e o número de células (idênticas subdivisões da via) que cada via comporta. Cada célula acomoda 30 veículos que se movem desta para uma próxima célula em um passo de simulação, caso esta possua espaço para acomodar esses veículos, ou seja, o movimento é baseado nas regras de transição de Autômatos Celulares. Detalhes de como os parâmetros são representados no simulador podem ser vistos no Apêndice (A.3);

- O estado inicial da rede (S_0), que consiste no número de veículos em cada célula de acordo com o parâmetro de densidade do tráfego, ou seja, descreve a ocupação de uma célula qualquer de uma via no instante inicial, variando de uma rede vazia (0%) a uma rede cheia (100%) e com isso modelando a capacidade de fluxo da via.

O grafo G e todos os outros elementos da instância Γ são definidos em arquivos tipo texto (ver apêndice A), que são posteriormente carregados pelo simulador. O simulador procede de maneira a traçar a trajetória de estados da rede, de forma iterativa e obedecendo às probabilidades da função de transição de estados. Onde são analisados os possíveis estados das células vizinhas (Especificação

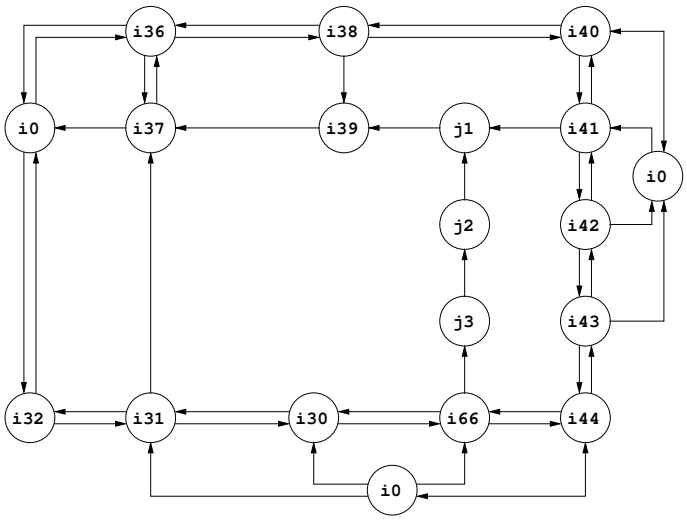

Figura 5.3: Rede de interesse baseada em grafos

Direta), caso estas possuam espaços, poderão receber veículos seguindo uma probabilidade de movimentos possíveis em cada intersecção (Regras Probabilísticas). No estágio corrente, a política de controle de um agente deve ser compilada juntamente com o simulador, entretanto a interface entre estes é realizada através de estruturas de dados simples.

5.2.2.1 Como é Codificado o Sistema Real no Simulador?

Vimos nas Seções 5.2.1 e 5.2.2, as apresentações da rede de interesse e do simulador em grafos responsável pelas implementações computacionais respectivamente. Nesta seção apresentaremos como o sistema real é codificado no simulador. A rede de interesse representada por Ω, foi modelada como um jogo estocástico distribuído, e definida como uma sêxtupla $\Gamma = (N, S, \Theta, A, T, R)$, cujos elementos são:

- $N = 13$ é o número de agentes controladores, indicados na figura da rede de interesse pelas intersecções semaforizadas: i_{30}, i_{31}, i_{32}, i_{36}, i_{37}, i_{38}, i_{39}, i_{40}, i_{41}, i_{42}, i_{43}, i_{44} e i_{66} (ver Figura 5.3);

- S é o conjunto de estados obtido ao agregar-se o número de veículos parados em cada uma das vias que conduzem a uma das intersecções i_n, mas este número é aproximado com valores $0, 4, 8, \ldots, 64$ para manter o uso da memória sob limites;

- Θ_n mapeia $s \in S$ para o sub-conjunto de variáveis que correspondem ao número e a posição dos veículos nas vias adjacentes à intersecção i_n, ou seja, Θ_n retorna ao agente n somente o estado das vias que conduzem à intersecção i_n;

- A_n modela o conjunto de ações de controle nos semáforos, indicando os movimentos factíveis do tráfego que podem ser executados simultaneamente numa intersecção i_n;

- T é a função de transição de estados, a qual segue os padrões de tráfego delineados acima, tendo sido obtida a partir das estatísticas geradas no fluxo de tráfego veicular nas intersecções, estatísticas essas fornecidas pelo Instituto de Planejamento Urbano de Florianópolis (IPUF); e

- R é um conjunto de funções sinal-reforço, onde R_n retorna a cada passo de simulação o valor negativo do número de veículos aguardando passagem pela intersecção i_n e que podem ser observados pelos agentes n.

A taxa na qual os veículos entram na rede varia entre experimentos, sendo esta modelada por uma probabilidade σ (densidade de tráfego) de um veículo entrar na rede a cada passo de simulação. O fluxo de veículos nas vias obedece a uma probabilidade deste seguir para este ou aquele destino, obedecendo a uma prioridade de liberação da mesma. Podemos ver esses possíveis movimentos com detalhes no Apêndice A.2

5.2.3 Técnicas de Controle

Com a finalidade de comparação, um conjunto de cenários experimentais de simulação foi sintetizado através da aplicação de uma das seguintes políticas de controle estacionárias ou método de aprendizagem abaixo para controlar os sinais semafóricos:

Política aleatória uniforme: atribui uma probabilidade igual a cada conjunto de movimentos simultaneamente realizáveis, isto é, $\pi_n(s_n, a_n) = \frac{1}{|A_n|}$ para cada agente $n \in \{1, \ldots, 13\}$, estado $s_n \in S_n$, e ação de controle $a_n \in A_n$. Esta estratégia, talvez seja a mais simples de todas as estratégias sendo insensível às condições do tráfego possibilitando até formação de congestionamento em algumas vias;

Política de controle de melhor-esforço: atribui a cada conjunto de movimentos simultaneamente realizados, isto é, a cada elemento de A_n, uma probabilidade proporcional ao número dos veículos parados que podem progredir se a_n for executado. Contrária à estratégia anterior, esta é sensível às condições do tráfego vigente, já que tende a favorecer o fluxo de tráfego das filas mais longas, ou seja, ela libera o fluxo de tráfego para a via que apresentar um maior número de veículos em espera. Mas, apresenta também possibilidade de formação de congestionamento;

Método *Q-learning* implementado pelo agente n: o agente n aplica o algoritmo *Q-learning* para controlar os sinais de tráfego na sua respectiva intersecção, enquanto que os outros agentes seguem a política aleatória uniforme, e com isso o agente n otimiza a intersecção i_n onde o mesmo atua; e

Política *Q-learning* distribuída: os agentes de controle buscam um conjunto de políticas de controle π interagindo com o modelo da rede, procurando iterativamente por funções valor-ação ótimas. Desse modo o conjunto de políticas de controle π sintetizada pelos agentes tende a se aproximar da política centralizada ótima π^*, a qual maximiza a soma dos ganhos de todos os agentes a longo prazo, o que se entende como objetivo principal da operação da rede de tráfego veicular.

Nos experimentos computacionais variou-se o número de agentes *Q-learning* e a política aleatória uniforme foi implementada nas intersecções restantes. A taxa de amortização foi fixada em $\gamma_n = 0.9$ e a taxa de aprendizado foi $\alpha_n = 0.1$ para cada agente n.

5.2.4 Algoritmo *Q-learning* Distribuído

A extensão do problema de aprendizagem por reforço de um único agente para jogos estocásticos tem como obstáculo à dificuldade em se encontrar um conjunto de políticas $\pi = (\pi_1, \ldots, \pi_N)$ tal que cada política π_n seja ótima do ponto de vista do agente n. Uma política π_n é uma fração $\pi_n : S_n \times A_n \rightarrow [0,1]$ tal que $\pi_n(s_n, a_n)$ corresponde à probabilidade do agente n tomar a ação a_n quando este se encontrar no estado s_n.

A noção de otimalidade tem que ser melhor elaborada, porque cada agente faz o melhor para si próprio. Segundo Basar (apud [10]), a política do agente só pode ser ótima se o conjunto de todas as políticas induz um equilíbrio, isto é, um ponto de equilíbrio Nash. O conjunto de políticas π produz um equilíbrio Nash, no sentido estocástico, se cada agente n não tiver qualquer incentivo para divergir de sua política de decisão, enquanto os outros agentes se mantém estáveis às suas políticas atuais. Ao contrário de Processos de Decisão Markoviano, nem a existência e nem a convergência para pontos de equilíbrio Nash podem ser garantidos em jogos estocásticos.

A distinção principal entre jogos estocásticos distribuídos e sua forma padrão, encontra-se na visão limitada do estado por parte dos agentes distribuídos, isto é, cada agente n detecta o valor de apenas uma fração das variáveis, a saber $\theta_n(s)$.

Propomos a aplicação de modificações de algoritmo para aprendizagem por reforço de um único agente ao problema de encontrar um conjunto de políticas de controle ótimas π. Os métodos de Diferença Temporal destacam-se como a mais promissora classe de algoritmos de aprendizagem por reforço, pois as evidências experimentais obtidas demonstram que elas podem ser eficazes, onde a predição e a aprendizagem têm sido muito prospera em uma variedade de aplicações, incluindo problemas de otimização de controle. Enquanto que os algoritmos de Programação Dinâmica exigem informação precisa da função de probabilidade de transição de estado (T), e a taxa de convergência do método de Monte Carlo é excessivamente lenta.

Para o propósito das análises aqui realizadas, assume-se que a tarefa é dividida em episódios, tais como dias da semana. Seja π_n uma política seguida por um agente n e seja $\lambda_n = \pi - \{\pi_n\}$ o conjunto das políticas implementadas pelos demais agentes. Assumindo um conjunto de políticas estacionárias λ_n, a função valor-ação induzida pela política π_n do agente n, para todo estado $s_n \in S_n$, onde $S_n = \{s_n : s_n = \theta_n(s)$ para algum $s \in S\}$, e a ação $a_n \in A_n$, é definida como:

$$Q_n^{\pi_n,\lambda_n}(s_n, a_n) \quad = \quad E_{\pi_n,\lambda_n}[\sum_{k=0}^{L} \gamma^k r_{n,t+k+1} : s_{n,t} = s_n, a_{n,t} = a_n] \tag{5.1}$$

onde L é a duração do episódio, $0 \leq \gamma \leq 1$ é o fator de amortização e $r_{n,t+1}$ é o ganho recebido pelo agente n ao executar a ação a_n no instante t e causar a transição do estado $s_{n,t} = s_n$ para $s_{n,t+1}$. Em

outras palavras, $Q_n(s_n, a_n)$ é o ganho amortizado esperado que o agente n recebe, se este começa no estado s_n, toma a ação a_n, e depois segue a política π_n enquanto os agentes restantes se comportam de acordo com λ_n.

A fim de acumular seu ganho amortizado máximo, o agente n busca aprender uma função valor-ação ótima dada por:

$$Q_n^{\pi_n^*,\lambda_n}(s_n,a_n) \quad = \quad \underset{\pi_n}{\text{Maximize}} \quad Q_n^{\pi_n,\lambda_n}(s_n,a_n) \quad \text{enquanto } \lambda_n \text{ é invariante} \tag{5.2}$$

para cada $s_n \in S_n$ e $a_n \in A_n$. Do ponto de vista do agente n, a política ótima é uma função das políticas dos outros agentes: $\pi_n^* = \Pi_n^*(\lambda_n)$. Isto dá origem à questão da existência de um conjunto de políticas π^* que sejam simultaneamente ótimas para cada agente n, isto é, $\pi_n^* = \Pi_n^*(\lambda_n^*)$ para cada n onde $\lambda_n^* = \pi^* - \{\pi_n^*\}$. Tal conjunto π^* é conhecido como conjunto de políticas de equilíbrio Nash: nenhum agente racional n divergirá de π_n^* porque caso contrário este incorreria perdas a si próprio. As políticas de equilíbrio Nash induzem funções valor-ação ótimas a todos os agentes, as quais podem ser expressas como:

$$\begin{cases} Q_n^{\pi_n^*,\lambda_n^*}(s_n,a_n) \quad = \quad \underset{\pi_n}{\text{Maximize}} \quad Q_n^{\pi_n,\lambda_n^*}(s_n,a_n) & \text{para } n = 1,\ldots,N \\ \qquad\qquad\qquad\qquad\quad \text{enquanto } \lambda_n^* \text{ é invariante} \end{cases} \tag{5.3}$$

A existência de políticas de equilíbrio Nash e a convergência para tais políticas são temas recorrentes no campo dos jogos estocásticos, para as quais não existem respostas definitivas. Freqüentemente, estas questões são analisadas caso-a-caso por meio de experimentações numéricas. Para aproximar políticas Nash, propomos a busca iterativa por um conjunto ótimo de funções valor-ação, como descrito em (5.3), pela qual a operação da rede sob a política dos agentes é iterativamente simulada e melhorada ao fim de cada episódio. Esta estratégia de busca consiste na aplicação de um algoritmo *Q-learning* modificado para cada agente, daqui por diante denominado algoritmo *Q-learning* distribuído, cujo pseudo-código é dado abaixo.

Algoritmo *Q-Learning* Distribuido

```
Cada agente n inicializa Q_n(s_n,a_n) arbitrariamente
Repita (para cada episodio)
   Inicializa s
   Repita para cada passo do episodio
      Para cada agente n faça
        Escolha uma ação a_n com base em Q_n(s_n), s_n = θ_n(s),
            usando uma politica derivada de Q_n tal como ε-guloso
        Implemente a ação a_n
      Fim-para

      Aguarde a rede reagir as ações e evoluir do estado s para s'

      Para cada agente n faça
        Observe r_n e s'_n = θ_n(s'), onde s' e o proximo estado
```

$Q_n(s_n,a_n) \leftarrow Q_n(s_n,a_n) + \alpha_n[r_n + \gamma_n \max_{a'_n} Q_n(s'_n,a'_n) - Q_n(s_n,a_n)]$,
onde $\alpha_n \in [0,1]$ e a taxa de aprendizagem e
$\gamma_n \in [0,1]$ e a taxa de amortização do agente n
Fim-para
Fim-repita
Fim-repita

5.2.5 Comparação das Técnicas

Esta seção avalia o potencial do algoritmo *Q-learning* distribuído para síntese de políticas de controle responsivo de tráfego aplicada na rede de interesse (Figura 5.1). Com finalidade de comparação, as políticas apresentadas na subseção 5.2.3 e frisadas abaixo, foram adotadas como forma de controle dos sinais semafóricos.

Política aleatória uniforme: onde se utiliza a mesma probabilidade para todas as ações disponíveis a um agente n, isto é, $\pi_n(s_n,a_n) = \frac{1}{|A_n|}, \forall s_n \in \{\Theta_n(s) : s \in S\}$ e $\forall a_n \in A_n$;

Política de melhor esforço: esta política libera o fluxo de tráfego para a via que apresentar um maior número de veículos em espera;

Método *Q-learning* implementado pelo agente n: o agente n aplica o algoritmo *Q-learning* para controlar os sinais de tráfego na sua respectiva intersecção, enquanto que os demais agentes seguem a política aleatória uniforme. Exemplo: O agente 1, aplica o algoritmo *Q-learning* na intersecção i_{40}, enquanto que os demais agentes seguem a política aleatória uniforme.

Método *Q-learning* distribuído: dois ou mais agentes implementam o algoritmo *Q-learning* ao mesmo tempo, enquanto os demais agentes executam a política aleatória uniforme até completar o total de 13 agentes implementando o método *Q-learning* ao mesmo tempo. Adotou-se uma hierarquia na implementação deste método pelos agentes, onde foi dada prioridade aos agentes que apresentavam um maior número médio de veículos em espera nas intersecções, ou seja, aquele que apresentava um maior número médio de veículos em espera na sua intersecção passaria a adotar o algoritmo *Q-learning* no próximo experimento de simulação.

Um conjunto de cenários experimentais de simulação foi sintetizado através da variação da densidade de tráfego σ (parâmetro este que descreve o nível de ocupação da rede de tráfego Ω), que teve uma variação de ocupação de $0.05 \leq \sigma \leq 0.75$, ou seja, a ocupação das vias variou de 5% a 75% de sua capacidade máxima (saturação), possibilitando simular diversas condições de tráfego.

Cada cenário experimental consistia em um método de controle e um parâmetro de densidade de tráfego. Para os métodos *Q-learning*, adotou-se os seguintes parâmetros de aprendizagem: a taxa de amortização foi fixada em $\gamma = 0.95$; a taxa de aprendizagem foi definida como $\alpha = 0.1$; e o parâmetro para geração de políticas ε-guloso foi reduzido gradualmente com o número de episódios de acordo com a regra $\varepsilon_k = \max\{0.1, 0.75 * 0.995^k\}$, onde k é o número do episódio.

Os resultados numéricos das experiências mencionadas acima são relatados em tabelas que veremos a seguir.

A Tabela 5.1 mostra a quantidade média do número de veículos em espera nas intersecções segundo a densidade de tráfego e segundo a variação da quantidade de agentes que usam o algoritmo *Q-learning* distribuído. Podemos observar que na primeira coluna (mais a esquerda) está relacionado o parâmetro da densidade do tráfego (σ), que representa as condições do fluxo de tráfego. Nas outras colunas é mostrado a média do número de veículos em espera nas intersecções segundo a quantidade de agentes usando o algoritmo *Q-learning*, ou seja, é feita uma média aritmética do número de veículos aguardando nas filas pelo número de intersecções da rede. Tais médias foram obtidas sobre 10 simulações (número de vezes que a experiência foi corrida), com o objetivo de efetuar uma média das várias simulações e obter resultados mais precisos, cada uma dessas simulações consistia de 600 episódios cada um com 2000 passos discretos de iterações (parâmetro deve ser suficientemente grande para que as políticas de escolha de ações dos agentes convirjam, isto é, o desempenho estabilize através da aprendizagem obtida ao longo dos episódios).

Nos experimentos envolvendo o método *Q-learning* distribuído, foi adotada uma hierarquia no sequenciamento dos agentes distribuídos. Segundo este sequenciamento, o agente com maior número de veículos em espera passava a utilizar o algoritmo *Q-learning* no experimento seguinte. Em uma primeira simulação, feita com todos os agentes implementando a política aleatória uniforme, observou-se que a intersecção i_{44}, foi a que apresentou o maior número de veículos em espera, e com isso o agente que a representa passou a ser a primeiro a adotar o algoritmo *Q-learning*, bem como a intersecção i_{42} foi a que apresentou o menor número de veículos em espera, e foi o último a adotar o algoritmo. A seqüência das intersecções a adotar o método foi: $i_{44}, i_{66}, i_{30}, i_{36}, i_{40}, i_{31}, i_{41}, i_{37}, i_{32}, i_{38}, i_{39}, i_{43}, i_{42}$. Com isso, pode ser observado na Tabela 5.1, um ganho maior nas primeiras colunas e um ganho menor nas últimas colunas.

Na última linha da Tabela 5.1, podemos observar a média total da quantidade de veículos em espera segundo a quantidade de agentes. Onde podemos observar com mais detalhes a diferença dos ganhos obtidos nas primeiras colunas em comparação com as últimas colunas.

A Figura 5.4, mostra o desempenho induzido por diferentes quantidades de agentes usando o método *Q-learning*. Podemos indicar uma redução média de 10.1% na comparação entre 1 e 3 agentes usando o algoritmo, ou 19.9% entre 1 e 5 agentes até alcançarmos uma redução média de 34.4% entre 1 e 13 agentes usando o método *Q-learning* distribuído.

A Tabela 5.2 mostra a quantidade média do número de veículos em espera nas intersecções segundo a densidade de tráfego e segundo a política de controle adotada. Como na Tabela 5.1, a primeira coluna relaciona o parâmetro da densidade do tráfego (σ), que simula as condições do fluxo de tráfego. As outras colunas mostram a média do número de veículos em espera nas intersecções induzidas pela política aleatória uniforme, pela política de melhor-esforço, e pela política de controle *Q-learning* distribuída adotada por 13 agentes. As médias foram obtidas da mesma forma, como na Tabela 5.1, ou seja, foram obtidas sobre 10 simulações, cada uma consistindo de 600 episódios de 2000 passos iterações.

Tabela 5.1: Número médio de veículos em espera segundo a densidade de tráfego e segundo a quantidade de agentes usando o algoritmo *Q-learning* distribuído

σ	Número de agentes *Q*-learning												
	1	2	3	4	5	6	7	8	9	10	11	12	13
0.05	54.63	46.43	39.50	32.05	26.72	20.86	17.50	13.84	11.35	8.72	7.19	6.71	4.59
0.10	108.61	92.93	79.88	67.75	55.64	43.36	36.88	29.72	24.35	18.99	15.94	15.93	10.87
0.15	179.20	151.80	136.31	117.79	107.55	88.61	82.71	71.01	65.57	60.27	59.25	43.20	38.83
0.20	297.37	242.48	234.84	206.53	173.14	171.77	161.88	150.21	142.97	132.16	130.71	133.18	125.89
0.25	396.76	445.92	396.76	376.20	363.59	327.47	313.84	316.01	293.41	289.11	277.63	261.13	249.15
0.30	562.15	627.08	562.15	531.64	512.89	452.58	447.82	445.80	401.04	414.79	380.15	340.93	337.52
0.35	738.56	755.79	681.51	646.24	636.46	555.35	551.40	477.37	510.46	521.27	508.63	466.31	452.29
0.40	1072.33	976.56	886.35	794.15	778.16	708.15	614.00	714.07	670.72	654.81	665.34	621.73	612.25
0.45	1151.33	1058.22	956.33	898.89	894.26	787.98	779.50	755.92	755.89	789.29	782.21	691.67	668.12
0.50	1350.97	1249.17	1148.69	1042.54	988.10	935.48	936.92	873.46	914.03	862.03	950.47	863.18	841.22
0.55	1506.14	1432.13	1340.17	1249.38	1155.74	1072.99	1071.93	1048.24	1118.26	1023.17	1013.21	1011.92	992.83
0.60	1865.86	1776.56	1680.75	1509.69	1396.46	1421.95	1345.64	1242.47	1403.04	1339.25	1393.67	1211.69	1245.49
0.65	2220.68	2160.62	2017.59	1855.77	1750.77	1688.94	1673.42	1747.05	1757.55	1821.55	1731.23	1620.98	1455.51
0.70	2527.36	2504.14	2351.43	2218.41	2168.64	2245.10	2194.35	2128.22	2139.76	2122.92	2080.78	1893.83	1900.03
0.75	2891.52	2800.97	2700.39	2531.15	2542.16	2509.55	2542.86	2229.26	2292.87	2289.89	2233.52	2205.41	2161.41
Média	1128.23	1081.45	1014.17	938.54	903.35	868.68	851.38	816.18	833.45	823.21	815.32	759.18	739.73

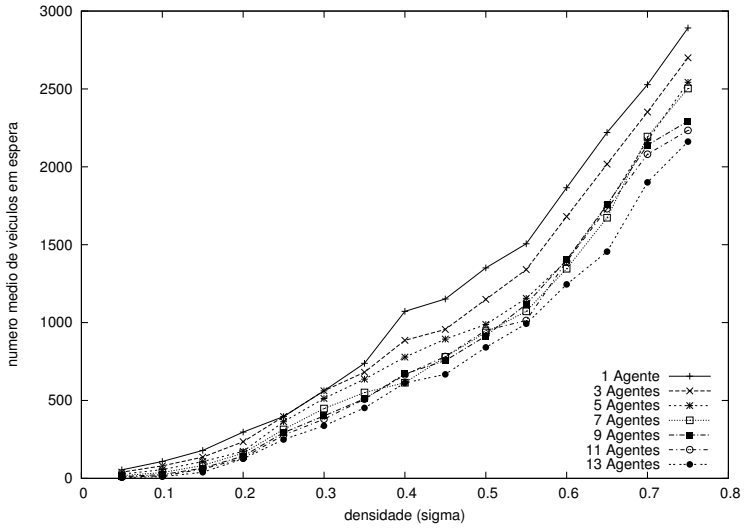

Figura 5.4: Comportamento das médias de veículos em espera obtidas pela quantidade de agentes que utilizam o algoritmo *Q-learning* distribuído

Nas duas últimas colunas da Tabela 5.2, são mostradas as porcentagens de ganho obtido pelo desempenho da política de controle *Q-learning* distribuída adotada por 13 agentes em comparação com as políticas de controle aleatório uniforme e de melhor-esforço. Segundo os dados obtidos, podemos observar um ganho maior que 92% em comparação com a política aleatória uniforme e 64% em relação à política de melhor-esforço. Na última linha da Tabela 5.2 podemos observar a média geral da quantidade de veículos em espera segundo as políticas adotadas e uma média geral dos ganhos que indicam uma redução média de 41% no tempo de espera em relação a política aleatória, e uma redução média de 29% em relação a política de melhor esforço. A evolução dos ganhos pode ser observada na Figura 5.5.

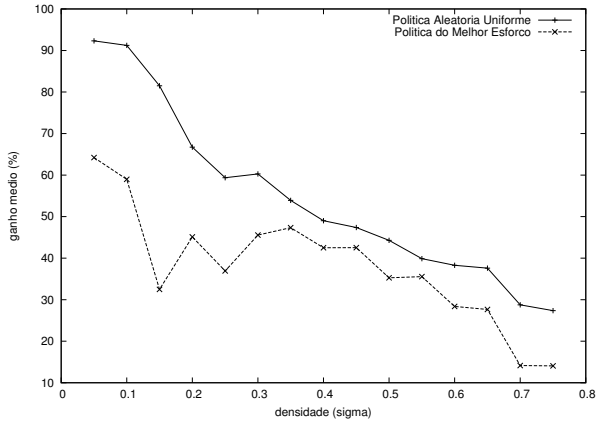

Figura 5.5: Ganho médio do método *Q-learning* para 13 agentes em relação aos métodos das políticas aleatória uniforme e de melhor esforço

Foi conduzida uma série de experimentos para se aproximar ainda mais do comportamento real de uma via de tráfego urbano, e medir o desempenho relativo das políticas de controle sob o tráfego, foi feita uma série de experimentos em que houve variações da densidade de fluxo durante cada

σ	Número médio de veículos em espera			Porcentagem de ganho	
	Política Aleatória	Melhor Esforço	13 agentes	Rand	Melhor
0.05	59.84	12.83	4.59	92.32	64.22
0.10	123.88	26.51	10.87	91.22	58.99
0.15	210.09	57.51	38.83	81.51	32.48
0.20	378.43	229.35	125.89	66.73	45.11
0.25	613.28	395.00	249.15	59.37	36.92
0.30	849.66	620.04	337.52	60.27	45.56
0.35	981.18	858.88	452.29	53.90	47.33
0.40	1201.18	1065.05	612.25	49.02	42.51
0.45	1269.70	1162.32	668.12	47.37	42.52
0.50	1510.06	1299.54	841.22	44.29	35.26
0.55	1651.67	1540.60	992.83	39.88	35.55
0.60	2018.43	1739.25	1245.49	38.29	28.38
0.65	2332.67	2011.49	1455.51	37.60	27.64
0.70	2667.03	2213.37	1900.03	28.75	14.15
0.75	2975.95	2515.14	2161.41	27.37	14.06
Média	1256.20	1049.79	739.73	41.11	29.53

Tabela 5.2: Número médio de veículos em espera segundo a densidade de tráfego e política de controle

episódio, ou seja, a simulação foi feita com σ variando da seguinte forma: $\{0.25, 0.40, 0.55, 0.70\}$, simulando assim o comportamento normal de uma via ao longo do dia. Esses episódios duraram 5000 passos de tempo e as trocas de densidade de fluxo aconteciam a cada 1250 passos de tempo. Os resultados obtidos são descritos na Tabela 5.3, onde esta relata o número médio de veículos em espera induzidos pelas políticas de controle aleatório uniforme, de melhor-esforço, e *Q-learning* distribuído para números variados de agentes (2, 4, 6, 8, 10 e 13).

σ	Número médio de veículos em espera								Porcentagem de ganho	
	Política Aleatória	Melhor Esforço	Número de agentes *Q-learning*							
			2	4	6	8	10	13	Rand	Best
Ciclo	1181.76	995.80	1066.33	895.84	773.71	740.73	725.61	738.45	30.75	25.85

Tabela 5.3: Número médio de veículos em espera segundo a densidade e tráfego e política de controle

As evidências numéricas obtidas a partir destas experiências computacionais confirmam a hipótese de que as técnicas de aprendizagem de máquina podem levar a ganhos substanciais na operação de redes de tráfego. Vide as porcentagem de ganhos obtidas nas duas últimas colunas da Tabela 5.3, 30.75% em relação à política aleatória e 25.85% em relação à política de melhor esforço, médias essas calculadas para o caso de 13 agentes *Q-learning*.

Na Figura 5.6, podemos observar o comportamento das porcentagens de ganhos em relação às quantidades de agentes que usam o algoritmo *Q-learning* distribuído em relação às políticas aleatória uniforme e de melhor-esforço. Como por exemplo, para uma quantidade de 6 agentes utilizando o algoritmo *Q-learning*, temos um ganho médio de 34.5% em relação à política aleatória e um ganho

médio de 22.3% em relação à política de melhor esforço e assim por diante.

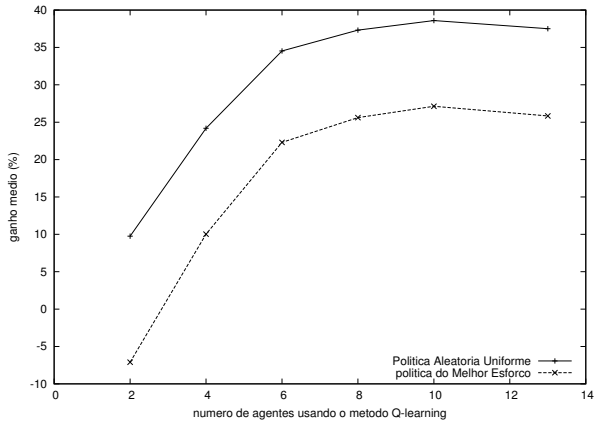

Figura 5.6: Ganho médio dos agentes que usam o método *Q-learning* em relação aos métodos das políticas aleatória uniforme e de melhor esforço

5.2.6 Considerações Finais

Neste capítulo, foi visto a implementação de agentes inteligentes distribuídos como uma forma de melhorar o desempenho das redes de tráfego veicular urbano, através de uma malha viária que compreende boa parte do centro da cidade de Florianópolis, modelando esta como um jogo estocástico distribuído, dando origem a um conjunto de problemas de aprendizagem por reforço, um para cada agente controlador. Apresentou-se também, algumas técnicas de controle convencionais e realizou-se experimentos computacionais como forma de comparação com a técnica utilizada no caso de estudo, algoritmo *Q-learning* distribuído, onde através de evidências númericas a partir de experimentos computacionais, o algoritmo se mostrou bastante eficaz, atingindo desempenho superior ao induzido pelas outras técnicas de controle (aleatória uniforme e de melhor esforço).

No capítulo a seguir serão apresentadas as conclusões do trabalho realizado e uma perspectiva de desenvolvimento e aplicações futuras.

Capítulo 6

Conclusões e Perspectivas Futuras

6.1 Conclusões

Este trabalho representa uma parte da pesquisa do projeto SINCMobil - Sistema de Informação e Controle para Mobilidade Urbana, que esta sendo desenvolvido pelo Departamento de Automação e Sistemas (DAS) da Universidade Federal de Santa Catarina (UFSC), que tem como objetivo principal desenvolver a implantar uma central provedora de informações para mobilidade urbana e uma central de controle de tráfego em tempo real. O trabalho dessa dissertação enfoca apenas uma parte relacionada a sistemas de transportes e otimização de sistemas, que abrange algoritmos de otimização baseados em aprendizagem por reforço aplicados ao controle de tráfego.

Numa primeira etapa deste trabalho, para uma melhor compreensão do problema a ser abordado, foram estudadas algumas teorias relacionadas à Engenharia de Tráfego, abrangendo problemas e dificuldades que são enfrentados pelos grandes centros e cidades de médio porte do país no gerenciamento e controle do tráfego. Também teorias sobre Inteligência Artificial foram apresentadas, pois o trabalho enfoca estudos com agentes inteligentes distribuídos, que são capazes de aprender e de se adaptar melhor a ambientes desconhecidos trabalhando conjuntamente, dividindo tarefas com o fim de obter uma maior capacidade de sucesso. Outro ponto importante estudado nesta etapa foi a teoria de Aprendizagem por Reforço, onde através da utilização de suas técnicas tenta-se mostrar que o uso do algoritmo *Q-learning* vem a ser capaz de maximizar o desempenho da malha viária em estudo.

A partir do conhecimento adquirido nesta primeira etapa, uma análise do caso de estudo passou a ser feita, onde este tinha como objetivo principal investigar a utilização de técnicas de aprendizagem por reforço, mas especificamente o uso do algoritmo *Q-learning* distribuído, como um método de controle eficaz na maximização do desempenho de uma malha viária da cidade de Florianópolis. Então, modelou-se essa malha viária em um simulador protótipo para jogos dinâmicos distribuídos, projetado especificamente para o domínio de redes de tráfego e utilizando-se da estratégia de decisão aplicada ao tamanho da fila em espera, ou seja, o número médio de veículos em espera nas intersecções da malha em estudo. Foram feitas análises de simulações entre os diferentes tipos de controle.

Nessas simulações buscou-se avaliar o desempenho das diferentes políticas de controle (política aleatória uniforme, política do melhor esforço, algoritmo *Q-learning* implementado pelo agente-*n* e algoritmo *Q-learning* distribuído), que tinham como objetivo minimizar o tamanho das filas apresentadas nas intersecções. Essas políticas de controle foram submetidas a diferentes condições de tráfego, ou seja, diferentes densidades de fluxo que variaram de 5% à 75% da capacidade das vias. Onde no uso do algoritmo *Q-learning* distribuído se observou uma diminuição no tamanho das filas de até 92% em relação à política aleatória uniforme e uma diminuição de até 64% em relação à política de melhor esforço. E em uma situação mais real de simulação, ou seja, uma variação da densidade de fluxo numa mesma simulação, simulando o comportamento diário de uma via observou-se ganhos de até 30% e 25% respectivamente.

Esses resultados comprovam que o algoritmo em estudo apresentou um desempenho bem superior frente ao induzido por outras políticas de controle e com isso comprovam que as técnicas de aprendizagem por reforço podem se estender ao controle de redes veiculares.

6.2 Perspectivas Futuras

Visando motivar futuras aplicações, direcionadas à gestão e controle do tráfego urbano da cidade de Florianópolis, algumas sugestões serão apontadas a seguir:

1. Aplicar o algoritmo em estudo a uma rede de tráfego mais acurada, ou até mesmo sobre toda a malha viária da cidade, procurando assim uma otimização global e uma sincronização do fluxo de tráfego. Pois, como vimos na Seção 2.3.1, as sincronizações de sinais de trânsito são na maioria das vezes, realizadas sem qualquer estudo prévio, ou quando são realizados tais estudos, apenas alguns trechos são considerados. Conseqüentemente, o problema é solucionado somente em uma região e eventualmente comprometendo a qualidade do trânsito em outras regiões. Mas para que isso ocorra, deve ser feito o desenvolvimento de aproximadores de função, que irão tratar do problema de explosão combinatória;

2. Aplicar simulações extensivas utilizando simuladores melhor adaptados para as mais diversas situações reais do comportamento de tráfego, como por exemplo, o SITRA-B+, um simulador francês dedicado aos problemas do tráfego veicular urbano;

3. O modelo de fila empregado nessa dissertação foi o baseado em filas horizontais, utilizando sistemas de Autômatos Celulares, onde suas regras de transição são baseadas no estado atual da célula e de suas vizinhas. Na dissertação as vias da malha em estudo foram divididas em células cada uma com a capacidade de 30 veículos, e a transição entre elas é feita com a observação da célula posterior, se esta possui espaço então é feita a transição. Mas esta não consegue observar o estado da célula anterior e se ela estiver há muito tempo sem ser atendida ela pode se encontrar com sua capacidade esgotada, ocasionando congestionamento na via. Então como uma sugestão para um trabalho futuro, que uma via possa observar o comportamento da outra a fim de se evitar congestionamentos;

4. Adotar outras estratégias de controle como por exemplo: tempo médio de espera nas filas, onde aquelas vias com pouco fluxo de veículos não sejam prejudicadas, por esperar longos tempos o seu direito de passagem; ou quantidade de paradas na rede, onde tentaríamos reduzir ao máximo o número de paradas nas intersecções das vias, reduzindo assim o tempo de viagem;

5. Outra possibilidade é de observar comportamentos emergentes nas simulações realizadas, fenômeno complexo e global, inesperado, não previamente estipulado, mas surgido. Um exemplo de comportamento emergente seria, observar um movimento de onda verde nas intersecções da malha em questão (sincronização semafórica), etc.

Apêndice A

Configuração do Simulador

A.1 Topologia da Rede de Tráfego

Nesta seção apresentamos a topologia da rede de tráfego no simulador, onde seus vértices modelam as intersecções e cruzamentos e seus arcos modelam as vias. Os números 19 e 47 significam que existem 19 intersecções na rede de interesse e 47 arcos (pistas) respectivamente.

As intersecções são representadas da seguinte forma: $1 = i_{30}$, onde esta nomenclatura indica que o número 1 representa a intersecção i_{30}. O arco é representado da seguinte forma: 1 13 $i_{30}\_i_{66}$, onde esta indica que existe um arco ligando a intersecção $1 = (i_{30})$ com direção e sentido à intersecção $13 = (i_{66})$.

A intersecção $14 = (i_0)$, representa o ponto de entrada e saída da rede de interesse, ou seja, é ponto de abastecimento e escape da via.

19 47

1 = i_{30} , 2 = i_{31} , 3 = i_{32} , 4 = i_{36}
5 = i_{37} , 6 = i_{38} , 7 = i_{39} , 8 = i_{40}
9 = i_{41} , 10 = i_{42} , 11 = i_{43} , 12 = i_{44}
13 = i_{66} , 14 = i_0 , 17 = j_1 , 18 = j_2
19 = j_3

1 13 $i_{30}\_i_{66}$, 13 1 $i_{66}\_i_{30}$, 1 2 $i_{30}\_i_{31}$, 2 1 $i_{31}\_i_{30}$

2 3 $i_{31}\_i_{32}$, 3 2 $i_{32}\_i_{31}$, 2 5 $i_{31}\_i_{37}$

4 5 $i_{36}\_i_{37}$, 5 4 $i_{37}\_i_{36}$, 4 6 $i_{36}\_i_{38}$, 6 4 $i_{38}\_i_{36}$

6 7 $i_{38}\_i_{39}$, 6 8 $i_{38}\_i_{40}$, 8 6 $i_{40}\_i_{38}$, 7 5 $i_{39}\_i_{37}$

8	9	$i_{40}\_i_{41}$	,	9	8	$i_{41}\_i_{40}$	,	9	17	$i_{41}\_j_1$							
9	10	$i_{41}\_i_{42}$	,	10	9	$i_{42}\_i_{41}$	,	10	11	$i_{42}\_i_{43}$	,	11	10	$i_{43}\_i_{42}$			
10	18	$i_{42}\_j_2$	,	11	12	$i_{43}\_i_{44}$	,	12	11	$i_{44}\_i_{43}$	,	11	19	$i_{43}\_j_3$			
12	13	$i_{44}\_i_{66}$	,	13	12	$i_{66}\_i_{44}$	,	13	19	$i_{66}\_j_3$							
14	1	$i_0\_i_{30}$	,	14	2	$i_0\_i_{31}$	,	14	3	$i_0\_i_{32}$	,	3	14	$i_{32}\_i_0$			
14	4	$i_0\_i_{36}$	,	4	14	$i_{36}\_i_0$	,	5	14	$i_{37}\_i_0$	,	14	8	$i_0\_i_{40}$			
8	14	$i_{40}\_i_0$	,	11	14	$i_{43}\_i_0$	,	12	14	$i_{44}\_i_0$	,	14	9	$i_0\_i_{41}$			
10	14	$i_{42}\_i_0$	,	14	13	$i_0\_i_{66}$	,	14	12	$i0\_i_{44}$							
17	7	$j_1\_i_{39}$	,	18	17	$j_2\_j_1$	,	19	18	$j_3\_j_2$							

A.2 Movimentos Factíveis e Probabilidade das Conversões

Aqui apresentaremos o conjunto de movimentos excludentes permitidos a cada nó do grafo G, sendo estes especificados por pares de arcos adjacentes, bem como a probabilidade das conversões. Por exemplo:

2	3			
14	2	2	1	0.3
14	2	2	3	0.65
14	2	2	5	0.05

Onde, o número $2(i_{31})$ representa a intersecção em estudo, o número 3 indica que existem 3 movimentos excludentes possíveis quando o fluxo é proveniente da intersecção $14(i_0)$, que são: 1^o) com 30% das conversões com destino à intersecção 1; 2^o) com 65% das conversões com destino à intersecção 3; e 3^o) com 5% das conversões com destino à intersecção 5, ou seja, 14 2 2 1 0.3, indica que 30% dos carros que se aproximarem da intersecção 2 seguirão para a intersecção 1, passando de 14 para 2 e de 2 para 1.

O número 47, representa a quantidade de vias existente na rede de interesse.

47

1	1					,	1	1					,	1	1				
2	1	1	13	1		,	13	1	1	2	1		,	14	1	1	13	1	

```
2  3                          ,  2  2                          ,  2  1
14 2  2  1  0.3               ,  1  2  2  3  0.7               ,  3  2  2  1  1
14 2  2  3  0.65              ,  1  2  2  5  0.3
14 2  2  5  0.05

3  1                          ,  3  1
2  3  3  14 1                 ,  14 3  3  2  1

4  2                          ,  4  2                          ,  4  1
5  4  4  6  0.46              ,  14 4  4  5  0.05              ,  6  4  4  14 1
5  4  4  14 0.54              ,  14 4  4  6  0.95

5  2                          ,  5  1                          ,  5  1
7  5  5  14 0.58              ,  2  5  5  4  1                 ,  4  5  5  14 1
7  5  5  4  0.42

6  2                          ,  6  2
4  6  6  8  0.96              ,  8  6  6  4  0.94
4  6  6  7  0.04              ,  8  6  6  7  0.06

7  1                          ,  7  1
17 7  7  5  1                 ,  6  7  7  5  1

8  2                          ,  8  2                          ,  8  1
6  8  8  9  0.10              ,  9  8  8  14 0.41              ,  14 8  8  6  1
6  8  8  14 0.90              ,  9  8  8  6  0.59

9  3                          ,  9  2                          ,  9  1
14 9  9  17 0.23              ,  8  9  9  10 0.83              ,  10 9  9  8  1
14 9  9  8  0.17              ,  8  9  9  17 0.17
14 9  9  10 0.60

10 2                          ,  10 2
9  10 10 11 0.91              ,  11 10 10 9  0.96
9  10 10 18 0.09              ,  11 10 10 14 0.04

11 2                          ,  11 2
10 11 11 12 0.92              ,  12 11 11 10 0.9
10 11 11 19 0.08              ,  12 11 11 14 0.1

12 2                          ,  12 2                          ,  12 1
11 12 12 13 0.19              ,  14 12 12 11 0.5               ,  13 12 12 14 1
11 12 12 14 0.81              ,  14 12 12 13 0.5

13 3                          ,  13 2                          ,  13 1
14 13 13 1  0.3               ,  12 13 13 1  0.8               ,  1  13 13 12 1
14 13 13 12 0.15              ,  12 13 13 19 0.2
```

```
14  13  13  19  0.55

14  8
 3  14  14  1  0.125  ,  3  14  14  2  0.125  ,  3  14  14  3  0.125
 3  14  14  4  0.125  ,  3  14  14  8  0.125  ,  3  14  14  9  0.125
 3  14  14  12 0.125  ,  3  14  14  13 0.125

14  8
 4  14  14  1  0.125  ,  4  14  14  2  0.125  ,  4  14  14  3  0.125
 4  14  14  4  0.125  ,  4  14  14  8  0.125  ,  4  14  14  9  0.125
 4  14  14  12 0.125  ,  4  14  14  13 0.125

14  8
 5  14  14  1  0.125  ,  5  14  14  2  0.125  ,  5  14  14  3  0.125
 5  14  14  4  0.125  ,  5  14  14  8  0.125  ,  5  14  14  9  0.125
 5  14  14  12 0.125  ,  5  14  14  13 0.125

14  8
 8  14  14  1  0.125  ,  8  14  14  2  0.125  ,  8  14  14  3  0.125
 8  14  14  4  0.125  ,  8  14  14  8  0.125  ,  8  14  14  9  0.125
 8  14  14  12 0.125  ,  8  14  14  13 0.125

14  8
10  14  14  1  0.125  ,  10  14  14  2  0.125  ,  10  14  14  3  0.125
10  14  14  4  0.125  ,  10  14  14  8  0.125  ,  10  14  14  9  0.125
10  14  14  12 0.125  ,  10  14  14  13 0.125

14  8
11  14  14  1  0.125  ,  11  14  14  2  0.125  ,  11  14  14  3  0.125
11  14  14  4  0.125  ,  11  14  14  8  0.125  ,  11  14  14  9  0.125
11  14  14  12 0.125  ,  11  14  14  13 0.125

14  8
12  14  14  1  0.125  ,  12  14  14  2  0.125  ,  12  14  14  3  0.125
12  14  14  4  0.125  ,  12  14  14  8  0.125  ,  12  14  14  9  0.125
12  14  14  12 0.125  ,  12  14  14  13 0.125

17  1                  ,  17  1
 9  17  17  7   1     ,  18  17  17  7   1

18  1                  ,  18  1
10  18  18  17  1     ,  19  18  18  17  1

19  1                  ,  19  1
11  19  19  18  1     ,  13  19  19  18  1
```

A.3 Parâmetros da Via

Nesta seção apresentaremos os parâmetros da via, que são representados pela quantidade de vias (indicada na parte superior pelo número 47), pela quantidade de veículos que a via comporta, bem como o número de células em que é dividida cada via e o número máximo de veículos em cada célula. Sendo que os veículos se movem baseados em Autômatos Celulares de uma célula para outra da sua vizinhança em um passo de simulação. por exemplo:

$i_{30}\_i_{66}$	240	8	30	30	30	30	30	30	30	30

Onde, $i_{30}\_i_{66}$ indica que há uma via ligando as intersecções no sentido de i_{30} para i_{66}, o número 240 indica a quantidade de veículos que percorre essa via em 1 hora, o número 8 indica a quantidade de células existente na via e os números 30 indicam o número máximo de veículos em cada célula.

47

$i_{30}\_i_{66}$	240	8	30	30	30	30	30	30	30	30		
$i_{66}\_i_{30}$	600	20	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{30}\_i_{31}$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30					
$i_{31}\_i_{30}$	210	7	30	30	30	30	30	30	30			
$i_{31}\_i_{32}$	570	19	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	
$i_{32}\_i_{31}$	210	7	30	30	30	30	30	30	30			
$i_{31}\_i_{37}$	210	7	30	30	30	30	30	30	30			
$i_{36}\_i_{37}$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30					
$i_{37}\_i_{36}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{36}\_i_{38}$	1500	50	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{38}\_i_{36}$	1500	50	30	30	30	30	30	30	30	30	30	30

			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{38}\_i_{39}$	210	7	30	30	30	30	30	30	30			
$i_{38}\_i_{40}$	1500	50	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{40}\_i_{38}$	1500	50	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{39}\_i_{37}$	600	20	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_{40}\_i_{41}$	180	6	30	30	30	30	30	30				
$i_{41}\_i_{40}$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30					
$i_{41}\_j_1$	300	10	30	30	30	30	30	30	30	30	30	30
$j_1\_i_{39}$	510	17	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30			
$i_{41}\_i_{42}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{42}\_i_{41}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{42}\_i_{43}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{43}\_i_{42}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{42}\_j_2$	120	4	30	30	30	30						
$i_{43}\_i_{44}$	300	10	30	30	30	30	30	30	30	30	30	30

$i_{44}\_i_{43}$	300	10	30	30	30	30	30	30	30	30	30	30
$i_{43}\_j_3$	120	4	30	30	30	30						
$i_{44}\_i_{66}$	210	7	30	30	30	30	30	30	30			
$i_{66}\_i_{44}$	300	10	30	30	30	30	30	30	30	30	30	30
$i_{66}\_j_3$	120	4	30	30	30	30						
$j_3\_j_2$	210	7	30	30	30	30	30	30	30			
$j_2\_j_1$	300	10	30	30	30	30	30	30	30	30	30	30
$i_0\_i_{30}$	600	20	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_0\_i_{31}$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
$i_0\_i_{32}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_{32}\_i_0$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30					
$i_0\_i_{36}$	810	27	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30			
$i_{36}\_i_0$	810	27	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30			
$i_{37}\_i_0$	360	12	30	30	30	30	30	30	30	30	30	30
			30	30								
$i_0\_i_{40}$	810	27	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30			
$i_{40}\_i_0$	870	29	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30	30	30	30	30	
$i_0\_i_{41}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							

$i_{42}\_i_0$	300	10	30	30	30	30	30	30	30	30	30	30
$i_{43}\_i_0$	270	9	30	30	30	30	30	30	30	30	30	
$i_{44}\_i_0$	450	15	30	30	30	30	30	30	30	30	30	30
			30	30	30	30	30					
$i_0\_i_{44}$	390	13	30	30	30	30	30	30	30	30	30	30
			30	30	30							
$i_0\_i_{66}$	300	10	30	30	30	30	30	30	30	30	30	30

Referências Bibliográficas

[1] R.E. Bellman. *Dynamic Programming*. Princeton Univ. Press, Princeton, New Jersey, 1957.

[2] R.A.C. Bianchi. *Uma Arquitetura de Controle Distribuída para um Sistema de Visão Computacional Propositada*. Dissertação de mestrado, Escola Politécnica da USP, São Paulo, 1998.

[3] M. Bielli e P. Reverberi. *New Operations Research and Artificial Intelligence Approaches to Traffic Engineering Problems*. *European Journal of Operational Research*, No.92: pp.550–572, 1996.

[4] G. Bittencourt. *Inteligência Artificial - Ferramentas e Teorias*. Editora da UFSC, Florianópolis, 2ª edição, 2001.

[5] W. Bonetti Jr. e H. Pietrantonio. *Utilização de Semáforos Atuados pelo Tráfego*. Artigo técnico, EMDEC, Campinas-SP, Set. 2001.

[6] C.M. Caeiro et al. *Estudo sobre Inteligência Artificial*. Trabalho apresentado para disciplina Educação e Tecnologia, da Universidade Nova de Lisboa.

[7] D.A. Callegari. *Aplicando Aprendizagem por Reforço a uma Arquitetura Multiagente para Suporte ao Ensino de Eduacação Ambiental*. Dissertação de mestrado, PUC-RS, 2000.

[8] K.F. Campana. *Programação Linear Aplicada ao Controle do Congestionamento de Tráfego*. Dissertação de mestrado, UFSC, Florianópolis, Jul. 2000.

[9] E. Camponogara, W. Kraus Jr. e M. Serra. *Aprendizagem por Reforço Aplicada ao Controle de Tráfego Veicular Urbano: Um Estudo Preliminar*. *VI Simpósio Brasileiro de Automação Inteligente*, Set. 2003.

[10] E. Camponogara e M. Serra. *Distributed Control Agents: A Case Study of Traffic Networks*. Artigo técnico, UFSC, Florianópolis-SC, Março 2004.

[11] ONERA Cert. *Manuel d'utilisation de SITRA-B+ - Versão 1.1*. S.O.D.I.T - Société pour le développement de l'innovation dans les transports, Toulouse - France, 042/92 edition, Jan. 1992.

[12] G.C. Chávez e Z. Liang. *Sistema Celular Evolutivo para Reconhecimento de Padrão Invariante*. *Anais do IV Workshop em Tratamento de Imagens, UFMG*, pp.62–70, Jul. 2003.

[13] P.C.M da Silva. *Teoria do Fluxo de Tráfego*. Apostila, UNB, Brasília-DF, 2001.

[14] M.S. de Luna. *Sobre o Fluxo de Saturação: Conceituação, Aplicação, Determinação e Variação*. Dissertação de mestrado, UFC, Fortaleza, Set. 2003.

[15] L.L. de Morais. *Modelo de Framework para Integração de Dispositivos de Controle em Sistemas Inteligentes de Tráfego Urbano*. Artigo acadêmico, UFRGS, Porto Alegre, 2000.

[16] G.M.B. de Oliveira. *Autômatos Celulares: Aspectos Dinâmicos e Computacionais*. *Anais do XXIII Congresso da Sociedade Brasileira de Computação*, Ago. 2003.

[17] Conselho Nacional de Trânsito. *Serviços de Engenharia: Manual de Semáforos*. Departamento Nacional de Trânsito, Brasília, 1984.

[18] G. Faria e R.F. Romero. *Navegação de Robôs Móveis Utilizando Aprendizagem por Reforço e Lógica Fuzzi*. *Revista Controle & Automação*, Vol.13, No.3, Set. 2002.

[19] R.M. Garbacz. *Adaptive Signal Control: What to Expect*. *12th Ann. Meeting of the Intelligent Transportation Soc. of America, Whashington, DC, U.S.A*, 2002.

[20] S.J. Hanson, W. Remmele e R.L. Rivest. *Machine Learning: From Theory to Applications - Cooperative Research at Siemens and MIT*, Vol.661 of *Lecture Notes in Computer Science*. Springer, Berlin-DE, 1993.

[21] L.P. Kaelbling, M.L. Littman e A.W. Moore. *Reinforcement Learning: A Survey*. *Journal of Artificial Intelligence Research*, Vol.4: pp.237–285, May. 1996.

[22] C. M. Lima e S. Labidi. Introdução à inteligência artificial. Artigo técnico, UFMA, São Luís, 2001.

[23] H.B. Meneses. *Interface Lógica em Ambiente SIG para Bases de Dados de Sistemas Centralizados de Controle do Tráfego Urbano em Tempo Real*. Dissertação de mestrado, UFC, Fortaleza, Abr. 2003.

[24] S.H. Ming. *Uma Breve Descrição do Sistema SCOOT*. Nota Técnica NT.201, CET - Companhia de Engenharia de Tráfego, São Paulo, Mai. 1997.

[25] S.H. Ming. *Recursos do SCOOT para congestionamentos*. Nota Técnica NT.203, CET - Companhia de Engenharia de Tráfego, São Paulo, 1998.

[26] S.T. Monteiro e C.H.C. Ribeiro. *Desempenho de Algoritmos de Aprendizagem por Reforço sob Condições de Ambiguidade Sensorial em Robótica Móvel*. *Revista Brasileira de Controle e Automação (SBA)*, Set. 2003.

[27] E.A.M. Munhoz. *Aspectos do Projeto SEMCO Relativos a Estratégia e Controle*. Nota Técnica NT.14, CET - Companhia de Engenharia de Tráfego, São Paulo, 1978.

[28] S.C. Navega. *Inteligência Artificial: Presente, Passado e Futuro*. *Publicado nos Anais do INFOIMAGEM, Cenadem*, Out. 2001.

[29] J.C. Neto. *O Controle Semafórico Centralizado e a Operação de Campo*. Artigo técnico, Mackenzie, São Paulo, Set. 2001.

[30] A. Olivato. *Percepção e Avaliação da Conduta de Motoristas e Pedestres no Trânsito: Um Estudo Sobre Espaço Público e Civilidade na Metrópole Paulista*. Dissertação de mestrado, USP, São Paulo, 2002.

[31] R.S. Oliveira et al. *Controle Ótimo de um Cruzamento Automatizado de Tráfego Urbano*. *XIV Congresso Brasileiro de Automática - CBA*, Vol.1: pp.1501–1506, Set. 2002.

[32] F. Osório. *Redes Neurais Artificiais: do Aprendizado Natural ao Aprendizado Artificial*. I fórum de inteligência artificial, Ulbra, Canoas-RS, Ago. 1999.

[33] L.A.M. Palazzo. *Inteligência Artificial*. Notas de aula da disciplina de código: 053273, Ago. 2003.

[34] L.M. Pereira. *Inteligência Artificial: Mito ou Ciência*. *Revista Colóquio/Ciências*, No.3: pp.1–13, Out. 1998.

[35] P.R. Rodegheri. *Arquiteturas para Estratégias de Learning Machines*. Technical report, UFRGS, Porto Alegre, 1999.

[36] D.S. Rodrigues, E.Y. Uwaide e O. Tonon. *Aprendizagem de Máquina*. Technical report, UNESP, Rio Claro-SP, 1999.

[37] R.Y. Rubinstein. *Simulation and the Monte Carlo Method*. John Wiley & Sons, Inc., New York, USA, 1st edition, 1981.

[38] S. Russel e P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, 1st edition, 1995.

[39] M. Schmitz e J.F. Hübner. *Uso de SMA para Avaliar Estratégias de Decisão no Controle de Tráfego Urbano*. *Seminário de Computação em Blumenau-SC*, pp.243–254, 2002.

[40] M.R.G. Serra. *Projeto do Controlador Programável de Semáforo (CPS II) com Tempo Variável*. Monografia de conclusão da graduação, UFMA, São Luis, Abr. 2001.

[41] R.S. Sutton e A.G. Barto. *Reinforcement Learning: An Introduction*. A Bradford book, Cambridge, Massachusetts, 1998.

[42] T.L. Thorpe e C.W. Anderson. *Traffic Light Control Using SARSA with Three State Representations*. Technical report, IBM Corporation, Boulder, CO, U.S.A, 1996.

[43] H.H. Trindade Filho. *Análise Comparativa do Potencial de Sistemas Centralizados para Controle de Tráfego no Brasil e Exterior, Considerando os Investimentos, os Benefícios e as Possíveis Adequações às Diferentes Realidades*. Dissertação de mestrado profissionalizante, UFRGS, Porto Alegre, Set. 2002.

[44] J.N. Tsitsiklis. *Asynchronous Stochastic Approximation and Q-Learning*. *Machine Learning*, Vol.16, No.3: pp.185–202, 1994.

[45] A.M. Turing. *Computing Machinery and Intelligence*. *Mind*, Vol.59, No.236: pp.433–460, Out. 1950.

[46] R.A. Vincent, A.I. Mitchell e D.I. Robert. *User Guide to TRANSYT Version 8*. Trrl lab report 888, Transport and Road Research Laboratory, Crowthorne, 1980.

[47] D.V. Vliet. *SATURN - A Modern Assignment Model*. *In: Traffic Engineering and Control*, Vol.23: pp.578–581, Dec. 1982.

[48] C. Watkins e P. Dayan. *Q-Learning*. *Machine Learning*, Vol.8: pp.279–292, 1992.

[49] M. Wiering. *Multi-agent Reinforcement Learning for Traffic Light Control*. *In Proc. 17th Int'l Conf. on Machine Learning*, pp.1151–1158, 2000.