UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DENISE DE OLIVEIRA

Aprendizado em Sistemas Multiagente Através de Coordenação Oportunista

Tese apresentada como requisito parcial
para a obtenção do grau de
Doutor em Ciência da Computação

Profa. Dra. Ana Lúcia Cetertich Bazzan
Orientador

Porto Alegre, dezembro de 2009

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Oliveira, Denise de

Aprendizado em Sistemas Multiagente Através de Coordenação Oportunista / Denise de Oliveira. – Porto Alegre: PPGC da UFRGS, 2009.

106 p.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2009. Orientador: Ana Lúcia Cetertich Bazzan.

1. Sistemas multiagente. 2. Aprendizado por reforço. 3. Coordenação. I. Bazzan, Ana Lúcia Cetertich. II. Título.

“If history repeats itself,
and the unexpected always happens,
how incapable must Man be of learning from experience!”
— GEORGE BERNARD SHAW, MAXIMS FOR REVOLUTIONISTS

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

AC	Autômatos Celulares
CV	Coeficiente de Variação
DT	Diferença Temporal
EPTC	Empresa Pública de Transporte e Circulação
FIPA-ACL	*Foundation for Intelligent Physical Agents–Agent Communication Language*
GT	*Game Theory*
IA	Inteligência Artificial
IL	*Independent Learner*
ITSUMO	*Intelligent Transportation System for Urban Mobility*
JC	Jogo Cooperativo
JAL	*Joint Action Learner*
KQML	*Knowledge Query and Manipulation Language*
MARL	*Multiagent Reinforcement Learning*
MASPA	*Multi-Agent Supervisory Policy Adaptation*
SMA	Sistema Multiagente
OPPORTUNE	*Opportunistic Coordination Learning*
MDP	*Markov Decision Process*
SCOOT	*Split Cycle and Offset Optimization Technique*
SCATS	*Sydney Coordinated Adaptive Traffic System*
TRANSYT	*Traffic Network Study Tool*
TUC	*Traffic-responsive Urban Traffic Control*
CTVU	Controle de Tráfego Veicular Urbano

WPL *Weighted Policy Learner*

WoLF *Win or Learn Fast*

LISTA DE SÍMBOLOS

Notação geral

$\cdot$	marcador genérico para argumento ou lista de argumentos
$\lvert \cdot \rvert$	cardinalidade de um conjunto ou vetor
$\cdot \leftarrow$	atribuição de valor
$\langle \cdot, \cdot \rangle$	tupla
$P(\cdot)$	probabilidade
$P(\cdot \lvert \cdot)$	probabilidade condicional
t	é um passo de tempo discreto
x'	próximo valor
x^*	valor ótimo
x^t	valor no tempo t
$\mathbf{x}$	vetor
i, j, k	índices que representam agentes

Aprendizado por Reforço Multiagente

γ	fator de desconto ($0 \leq \gamma < 1$)
α	taxa de aprendizado ($0 \leq \alpha < 1$)
π	política
$\mathcal{S}$	um conjunto finito e discreto de estados
$\mathcal{A}$	um conjunto finito e discreto de ações
$P(s' \lvert a, s)$	probabilidade de transição para o estado s' dado que estava no estado s e realizou a ação a
Ω	conjunto finito de observações
O	probabilidades de observações, onde $O(s, a, s', o)$ é a probabilidade do agente observar o, realizando a ação a dado o estado atual s e o próximo estado s'
K	um número racional representando o valor de limiar

$U(s)$	utilidade do estado s
a	variável de ação de um agente
$V(s)$	valor do estado s
$Q(s, a)$	valor da execução da ação a no estado s
R	é a função de recompensa recebida estando no estado s e realizando a ação a ($R(s, a)$)
$\mathcal{T}$	uma função de transição de estados
O	são as probabilidades de observações onde $O(s, a, s', o)$ é a probabilidade do agente observar o, realizando a ação a, dado o estado atual s e o próximo estado s'
T	é o horizonte de tempo, finito ou infinito
ε	probabilidade de uma ação randômica em uma política ε-*greedy*
λ	tamanho do traço de elegibilidade
δ	erro de diferença temporal
$\mathbf{e}$	vetor de elegibilidade
θ	vetor de parâmetros
ϕ	conjunto de características

Aprendizado multiagente com coordenação oportunista

$\mathcal{N}_i$	conjunto de agentes na vizinhança do agente i
$\mathbb{S}_i$	conjunto que representa um estado composto do agente i
$\mathbb{A}_i$	conjunto que representa uma ação composta do agente i
$E(\mathbb{S}_i, \mathbb{A}_i)$	erro do par $(\mathbb{S}_i, \mathbb{A}_i)$
E_{max}	valor do erro máximo
$\mathbb{R}$	conjunto com histórico de recompensas
CV	coeficiente de variação
s_i	estado do agente i
a_i	ação do agente i

Controle de Tráfego Veicular Urbano

F	fluxo de saturação
L	largura
v	velocidade

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

RESUMO

O tamanho da representação de ações e estados conjuntos é um fator chave que limita o uso de algoritmos de aprendizado por reforço multiagente em problemas complexos. Este trabalho propõe o *Opportunistic Coordination Learning* (OPPORTUNE), um método de aprendizado por reforço multiagente para lidar com grandes cenários. Visto que uma solução centralizada não é praticável em grandes espaços de estado-ação, um modo de reduzir a complexidade do problema é decompô-lo em subproblemas utilizando cooperação entre agentes independentes em algumas partes do ambiente. No método proposto, agentes independentes utilizam comunicação e um mecanismo de cooperação que permite que haja a expansão de suas percepções sobre o ambiente e para que executem ações cooperativas apenas quado é melhor que agir de modo individual. O OPPORTUNE foi testado e comparado em dois cenários; jogo de perseguição e controle de tráfego urbano.

Palavras-chave: Sistemas multiagente, aprendizado por reforço, coordenação.

Towards Joint Learning in Multiagent Systems Through Opportunistic Coordination

ABSTRACT

The size of the representation of joint states and actions is a key factor that limits the use of standard multiagent reinforcement learning algorithms in complex problems. This work proposes *Opportunistic Coordination Learning* (OPPORTUNE), a multiagent reinforcement learning method to cope with large scenarios. Because a centralized solution becomes impractical in large state-action spaces, one way of reducing the complexity is to decompose the problem into sub-problems using cooperation between independent agents in some parts of the environment. In the proposed method, independent agents use communication and cooperation mechanism allowing them to extended their perception of the environment and to perform cooperative actions only when this is better than acting individually. OPPORTUNE was tested and compared in two scenarios: pursuit game and urban traffic control.

Keywords: multiagent systems,reinforcement learning,coordination.

1 INTRODUÇÃO

Em diversos problemas de decisão, agentes precisam interagir para executar tarefas em cenários complexos e dinâmicos. Usualmente, problemas deste tipo são caracterizados pela comunicação incompleta entre os agentes, limitação na percepção, ações correlacionadas, etc. Ao longo deste texto, serão mostrados cenários e situações onde há a interação entre agentes em um ambiente distribuído e na maioria dos casos uma solução centralizada não é possível, seja pela falta de um meio de comunicação entre os agentes ou por restrições em tempo de processamento.

A pesquisa em aprendizado por reforço multiagente, ou *Multiagent Reinforcement Learning*, está concentrada em problemas envolvendo poucos agentes e, normalmente, assumindo ambientes plenamente cooperativos. Em (PANAIT; LUKE, 2005), são mostradas direções de pesquisa que ainda não foram devidamente exploradas:

- Grande número de agente, de 10 a mil ou mais;
- Times heterogêneos, a maior parte da literatura presume que os agentes são idênticos tanto no comportamento quanto nas suas habilidades;
- Agentes com estados internos (WOOLDRIDGE, 2002, p.35) mais complexos;
- Mudança dinâmica de times e cenários.

Dentro destas direções, este trabalho está relacionado com a questão do grande número de agentes e com a mudança dinâmica de times e cenários. A principal questão abordada por este trabalho é:

> **Como tornar o aprendizado multiagente uma solução aplicável em sistemas complexos?**

Assume-se que:

- Os agentes cooperam quando é vantajoso, buscando realizar ações com a melhor recompensa.
- O problema deve ser solucionado de modo distribuído ou parcialmente distribuído, utilizando comunicação entre os agentes.
- Deve haver no mínimo uma tarefa para a qual o agente se beneficie, ou seja, obtenha uma recompensa maior, caso realize-a recebendo informações de outro agente ou agindo de maneira cooperativa.

- O ambiente deve possuir as seguintes características: dinâmico, não-determinístico, discreto, parcialmente observável e cooperativo ou parcialmente cooperativo. Estas características serão apresentadas com mais detalhes no Capítulo 3.

Outro ponto importante de ser ressaltado é que a pesquisa em *Multiagent Reinforcement Learning* (MARL), normalmente, é realizada sob o ponto de vista de teoria de jogos (ou *Game Theory* (GT)). A GT é um ramo da matemática que estuda interações e estratégias entre indivíduos em um jogo. Mais sobre GT será visto no Capítulo 4. A aplicação de MARL em jogos tem como objetivo encontrar a melhor estratégia quando um agente tem um ou mais adversários, e normalmente costuma buscar o equilíbrio. Um ponto de equilíbrio estável é um resultado de um jogo em que nenhuma mudança nas estratégias pode melhorar os resultados. Ele é considerado estável porque, se um jogador escolhe unilateralmente uma estratégia diferente da atual receberá um retorno (ganho) menor ou no máximo igual ao retorno no estado de equilíbrio. Uma discussão sobre aprendizado multiagente e do seu futuro como pesquisa foi aberta com o artigo intitulado "*If multi-agent learning is the answer, what is the question?*" (SHOHAM; POWERS; GRENAGER, 2007). A discussão foi centrada basicamente nos aspectos teóricos do aprendizado em jogos e levantou alguns pontos importantes. Os autores ressalvam que há uma necessidade de ser claro sobre o problema a ser abordado e os respectivos critérios de avaliação. Para isso, foram identificadas cinco linhas distintas para a pesquisa em aprendizado multiagente:

Computacional: visualiza o algoritmo de aprendizado como uma maneira iterativa de computar as propriedades do jogo;

Descritiva: o objetivo é investigar métodos formais de aprendizado que sejam similares ao comportamento humano ou com o comportamento de outros agentes;

Normativa: foca em determinar se conjuntos de regras de aprendizado estão em equilíbrio entre si;

Normativa cooperativa e não cooperativa a questão aqui é como os agentes devem aprender em jogos de ganho comum (cooperativos) repetitivos ou em jogos de ganho independente (não cooperativos) repetitivos.

Além dessas linhas de pesquisa, os autores afirmam que o campo de pesquisa não pode avançar enquanto estratégias de aprendizagem arbitrárias são criadas e analisadas considerando-se apenas quando as dinâmicas resultantes convergem em determinados casos.

Em resposta ao artigo de Shoham e colegas, vários artigos foram publicados (SANDHOLM, 2007; GORDON, 2007; YOUNG, 2007; STONE, 2007). Sandholm propôs um refinamento nas agendas de Shoham et al., acrescentando uma hierarquia e renomeando a agenda normativa para "algoritmos de aprendizagem em equilíbrio". Esta taxonomia proposta pode ser vista na Figura 1.1. Além destas agendas, Gordon propôs mais duas agendas: modelagem e projeto. O objetivo dessas agendas é cobrir os problemas que precisam ser considerados antes de que os agentes comecem o processo de aprendizagem.

Peter Stone reforça que a aprendizagem multiagente é um meio-termo entre teoria de jogos e IA e há diversas obras na intersecção dessas duas áreas. Interações entre agentes podem ser caracterizadas como jogos, mas em vários casos, não é apenas que a convergência para um equilíbrio não é um objetivo, mas que a própria formulação como

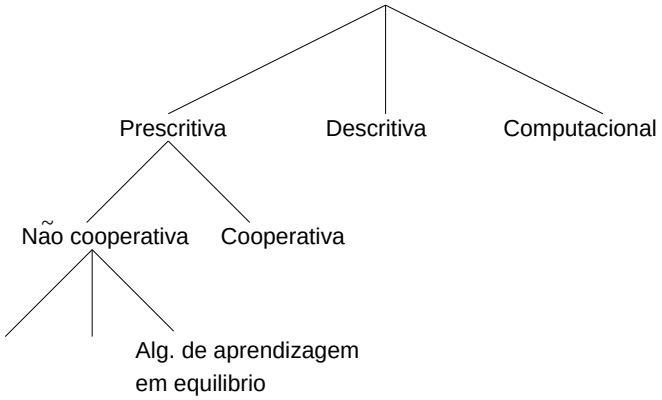

Figura 1.1: Agendas de pesquisa, proposta em (SANDHOLM, 2007).

uma forma normal ou forma extensiva de jogo pode trazer pouco ou nenhum progresso no sentido de encontrar uma solução para o problema. A partir de uma perspectiva de IA, aprendizagem multiagente deve ser considerada uma área mais ampla do que a teoria dos jogos pode endereçar.

Esta tese se enquadra na agenda *descritiva* de Shoham, uma vez que o objetivo é encontrar um método melhor para o agente aprender em ambientes complexos.

Existem dois extremos a serem considerados no aprendizado multiagente. Em um extremo, existem os *Independent Learner* (IL)s, (CLAUS; BOUTILIER, 1998), abordagens que consideram outros agentes como parte do ambiente. Na direção oposta, há os *Joint Action Learner* (JAL)s que consideram todas as observações e as ações de cada agente no ambiente. Estes JALs têm uma enorme representação de estados, uma vez que todos os estados de todos os agentes estão representados de modo conjunto.

O tamanho do espaço de busca dos estados é um fator chave para se estabelecer os limites da utilização de algoritmos de aprendizado por reforço em problemas complexos. Abstração de estados, ou agregação de estados, tem sido amplamente estudada. Abstração pode ser considerada como o mapeamento da descrição original de um problema para uma representação mais compacta, permitindo ao agente distinguir a informação relevante das informações irrelevantes.

De uma perspectiva computacional, a abstração de estados é uma técnica para tornar possível a aplicação de algoritmos de aprendizagem em grandes problemas, porém normalmente é um método centralizado. (LI; WALSH; LITTMAN, 2006) apresenta uma revisão desses métodos, dividindo os diferentes enfoques de acordo com dois critérios: se o algoritmo precisa ter pleno conhecimento do MDP e da exatidão do método. No estudo de (LI; WALSH; LITTMAN, 2006) é mostrado que existe um meio-termo entre maximização da redução do espaço de estados e a minimização da perda de informações pela seleção de abstrações. Outra questão é que o espaço de estados pode ficar quase tão grande quanto o espaço base em domínios complexos. Em resumo, existem diversas técnicas para lidar com MARL em várias situações, mas, até agora, não existe método de aprendizagem totalmente distribuído que utilize uma modificação adaptativa da representação do conhecimento dos agentes utilizando negociação.

Este trabalho parte da ideia de que mesmo em ambiente complexos é possível aplicar um método de aprendizado multiagente distribuído, cuja organização dos agentes emerge através da utilização de cooperação e da troca de informações.

Este trabalho propõe uma nova abordagem de aprendizado por reforço chamada *Opportunistic Coordination Learning* (OPPORTUNE), na qual o agente começa com uma simples representação individual e expande-a para uma representação mais complexa e completa do ambiente, permitindo que o agente aumente a sua percepção do ambiente e obtenha um melhor desempenho, se comparado com um agente independente no mesmo ambiente. O agente age como um IL em estados onde apenas a sua própria informação é utilizada, age como um IL com percepções partilhadas em estados onde ele usa as informações recebidas por parte de parceiros (agentes em um grupo) e tenta atuar como um JAL em situações onde ele avalia que é necessário cooperar para obter maiores ganhos. Este trabalho busca formar uma solução de compromisso apresentando características úteis de ambos os extremos: aprendizado monoagente e multiagente; de modo a ter um mecanismo de aprendizagem capaz de lidar com os diferentes aspectos de ambientes complexos.

Este trabalho propõe uma abordagem de aprendizado multiagente, onde é utilizado o aprendizado independente em conjunto com o aprendizado cooperativo, aumentando de modo gradual e seletivo o espaço de estados e de ações conjuntas.

Para isso, cada agente inicia seu aprendizado sem nenhum conhecimento prévio sobre os demais agentes, mas com possibilidade de percepção e de comunicação com demais. O agente é capaz de interagir com os outros agentes dentro de uma área de observação e de comunicação, podendo esta área estar definida de modo estático ou dinâmico. A abordagem proposta utiliza comunicação e negociação entre os agentes para que eles obtenham uma visão melhor do ambiente e para que possam avaliar quais ações (conjuntas ou independentes) podem ser mais promissoras, ou seja, com maior possibilidade de ganho de recompensa.

Na abordagem proposta, todos os mecanismos de coordenação surgem a partir das interações entre os agentes, e não há uma predefinição de onde e quando os agentes devem agir de modo conjunto nem as situações onde informações sobre o ambiente devem ser compartilhadas. O objetivo de utilizar a coordenação oportunista é duplo:

- reduz o espaço de busca aumentando a probabilidade de convergência;
- representação inicial do problema se torna mais simples para o projetista, já que não é necessário informar previamente as interações entre os agentes.

Este trabalho contribui para o avanço do estado-da-arte no estudo e desenvolvimento de aprendizagem em Sistema Multiagente (SMA) e na modelagem e aplicação de técnicas de aprendizado por reforço em problemas reais, onde deve-se destacar:

- A abordagem apresentada mostra-se como uma solução de compromisso para ambientes de aprendizado multiagente, obtendo ganhos tanto no tempo de aprendizado (tempo para atingir a convergência) quanto em uma representação mais enxuta e funcional do ambiente percebido.

- Uma grande vantagem da abordagem proposta é a adaptação dos agentes ao ambiente, tornando o aprendizado menos sensível às mudanças no ambiente e facilitando a fase de modelagem do ambiente pelo projetista.

Todos os mecanismos do OPPORTUNE foram experimentados e validados. Os resultados empíricos destes experimentos mostraram que a abordagem proposta, baseada em modelos de aprendizado por reforço padrão, pode ser utilizada com sucesso em problemas de aprendizado multiagente.

Esta tese esta organizada do seguinte modo:

Capítulo 2 Neste capítulo são abordados alguns fundamentos necessários para a compreensão deste trabalho, apresentando uma revisão sobre *Markov Decision Process* (MDP), algoritmos e abordagens de aprendizado monoagente e multiagente. Além disso, serão apresentados os trabalhos relacionados à abordagem proposta;

Capítulo 3 Introduz o OPPORTUNE, analisando o método, o algoritmo, discutindo suas necessidades e aplicações;

Capítulo 4 Apresenta o problema de Jogo Cooperativo (JC)s, suas características e o cenário de jogo de captura cooperativo. O principal objetivo deste capítulo é apresentar o OPPORTUNE aplicado neste problema clássico de aprendizado e os experimentos realizados para a sua validação;

Capítulo 5 Apresenta o domínio do Controle de Tráfego Veicular Urbano (CTVU) e trabalhos relacionados. É apresentado o uso do OPPORTUNE neste domínio bem como os experimentos realizados para sua validação;

Capítulo 6 Neste capítulo serão apresentadas as conclusões e os aspectos não cobertos por esse trabalho que podem levar a trabalhos futuros.

2 APRENDIZADO POR REFORÇO MULTIAGENTE

O método padrão de aprendizado por reforço consiste na interação repetida do agente com o ambiente em intervalos discretos (KAELBLING; LITTMAN; MOORE, 1996). A cada passo de tempo, o agente percebe o estado atual do ambiente e seleciona uma ação. O ambiente responde com um sinal de recompensa e uma nova percepção que formará um novo estado. Os métodos de aprendizado por reforço são baseados nas teorias do behaviorismo clássico, onde um comportamento é sempre uma resposta a um estímulo específico, e também na teoria do condicionamento operante, onde todo e qualquer ato que produza satisfação cria associações na mente do animal de forma que sua probabilidade de repetição torna-se maior.

De acordo com Sutton e Barto, (SUTTON; BARTO, 1998), as duas propriedades mais importantes do aprendizado por reforço são o método de tentativa e erro e a recompensa atrasada. Este tipo de aprendizado implica a determinação de um mapeamento de situações para comportamentos de forma a maximizar o total de recompensas futuras quanto possível. As dificuldades para esse tipo de aprendizado se devem principalmente a três razões:

1. O aprendiz não recebe instruções explícitas de como atingir seus objetivos, e portanto precisa determinar, por tentativa-e-erro, quais ações são mais vantajosas;

2. O ambiente pode ser altamente estocástico e imprevisível e, no caso geral, não se pode assumir que o agente possui quaisquer modelos de predição;

3. As ações tomadas não necessariamente geram recompensas no exato instante em que são tomadas.

A terceira razão esta diretamente ligada a casos em que uma ação instantaneamente desvantajosa pode ser essencial para levar o aprendiz às áreas do espaço de estados capazes de fornecer bons sinais de reforço futuro (recompensa atrasada). Esse fator na recompensa faz com que o agente precise ser capaz de determinar o quanto cada uma de suas ações passadas contribuiu para a obtenção de uma determinada recompensa acumulada. Na IA, essa dificuldade é conhecida como o problema da atribuição de crédito. De certa forma, todas abordagens baseadas em reforço são tentativas de resolver este problema.

Neste capítulo vamos apresentar uma revisão sobre os processo de decisão de Markov, do inglês *Markov Decision Process* (MDP), já que este é o formalismo utilizado nos métodos de aprendizado por reforço. Após a revisão inicial, serão apresentados alguns métodos de aprendizado por reforço por diferença temporal: atualização por Diferença Temporal (DT), Aprendizagem-Q (WATKINS; DAYAN, 1992) e *Watkins's Q(λ)*. Na Seção 2.3 mostraremos uma visão geral sobre o aprendizado multiagente, caracterizando

seus tipos, domínios e possíveis aplicações. Apresentaremos na Seção 2.4, algumas abordagens de aprendizado multiagente que estão mais relacionadas com este trabalho. Finalizaremos este capítulo com uma conclusão geral sobre o aprendizado por reforço multiagente e suas perspectivas.

2.1 Revisão sobre processos de decisão de Markov

Um *Markov Decision Process* (MDP), consiste em cinco elementos: episódios de decisão, estados, ações, probabilidades de transição entre estados e recompensas. Formalmente, um MDP é uma tupla $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R \rangle$, onde:

$\mathcal{S}$ é um conjunto discreto de estados;

$\mathcal{A}$ é um conjunto discreto de ações;

$\mathcal{T}$ é uma função de transição de estados $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow P$, onde $P(S)$ é uma distribuição de probabilidade sobre $\mathcal{S}$; sendo que $\mathcal{T}(s, a, s')$ é a probabilidade de ocorrer a transição de s para s', dada a ação tomada;

R é uma função de recompensa $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathfrak{R}$.

Ele modela um agente atuando em um ambiente estocástico onde o objetivo é maximizar a recompensa ou ganho a longo prazo. Também serve como ferramenta de resolução de diversos problemas.

Um MDP consiste em um processo estocástico caracterizado por um conjunto de estados, ações e matrizes de probabilidade de transição entre os estados. A transição entre os estados ocorre de maneira discreta no tempo e depende de uma ação. Além disso, a cada par estado-ação) é associada uma função de recompensa, a qual define o ganho instantâneo que o agente recebe.

Os MDP em geral assumem que a *Hipótese de Markov* é válida. Esta hipótese diz que um sistema possui a *propriedade de Markov*, ou seja, é *Markoviano*, se o estado atual depende apenas do estado imediatamente anterior. Em um jogo de xadrez, por exemplo, o estado poderia ser representado diretamente pelas posições das peças no tabuleiro. Em sistemas deste tipo, a melhor política a ser seguida a partir de determinado estado é independente da trajetória específica que levou o sistema até aquele estado.

Caso o sistema possua a *propriedade de Markov*, considerando $P(a|b)$ como a notação para probabilidade condicional da ocorrência de a dado o evento b, para todo estado e toda ação tem-se:

$$P(s^{t+1} = s', r^{t+1} = r | s^t, a^t, r^t, s^{t-1}, a^{t-1}, ..., r^1, s^0, a^0) = P(s^{t+1}, r^{t+1} | s^t, a^t)$$

Em sistemas *Markovianos*, um modelo de predição de um passo é suficiente para determinar a probabilidade de cada próximo estado e da próxima recompensa, sabendo-se apenas o último estado e ação.

A resolução de um MDP, no contexto do aprendizado por reforço, consiste em encontrar uma sequência de ações que garanta o maior ganho esperado para o sistema. Denotamos π^* como a política ótima para o MDP, ou seja, o mapeamento de estados para ações que gera o maior ganho futuro esperado.

2.2 Aprendizado por reforço por Diferença Temporal (DT)

Esta seção apresenta três métodos de aprendizado por reforço por DT. Os métodos de programação dinâmica exigem um modelo completo do mundo, e fazem atualizações (correções) gerando estimativas de valor de estado a partir de outras estimativas. A abordagem da programação dinâmica não funciona em cenários de aprendizado em tempo real. Métodos Monte Carlo têm a vantagem de não exigirem modelos completos do ambiente e podem funcionar em tempo real, porém, precisam esperar até o final de um episódio para obterem melhorias na política, não podendo então utilizar de estimativas parciais. Os métodos que utilizam diferença temporal podem ser considerados um meio termo entre as abordagens de programação dinâmica e Monte Carlo, já que podem tanto aprender com base em experimentação direta, sem modelo, quanto fazer uma inicialização de valor de estado a partir de outras estimativas a fim de acelerar a obtenção de políticas.

2.2.1 Atualização por diferença temporal

A atualização de valores por Diferença Temporal (DT) busca propagar atualizações nos valores coletando recompensas recebidas após a primeira visita a cada estado, ou seja, é possível atualizar V^{π} após a observação de cada recompensa individual. Assim, novas estimativas do valor de um estado vão sendo criadas corrigindo-se as estimativas antigas em direção a um valor alvo, que em geral equivale à soma da amostra de recompensa recém recebida, e do valor descontado para a estimativa do estado seguinte. A ideia principal é de que a soma $r + \gamma V(s')$ equivale a uma amostra do valor real de $V(s)$. Por exemplo, considere que um agente, em um estado s escolhe determinada ação no tempo t, observa uma transição para o estado s' e recebe um sinal de recompensa r no tempo $t + 1$. A partir dessas informações ele é capaz de corrigir o valor do estado predecessor, s. O método mais simples para fazer esse tipo de atualização de valor se chama $TD(0)$, definida pela Equação 2.1.

$$V(s) \leftarrow V(s) + \alpha\Big(r + \gamma V(s') - V(s)\Big) \tag{2.1}$$

Nessa equação, o alvo, dado um passo de correção α, é $r + \gamma V(s')$, ou seja, é em direção a esse valor que corrigimos a estimativa atual $V(s)$. Essa equação possui tanto características Monte Carlo, pois contém uma amostra real de recompensa, quanto de programação dinâmica, pois faz a inicialização (*bootstrap*) com o valor estimado do estado seguinte, s'. Ao contrário da programação dinâmica, os métodos de diferença temporal não fazem salvamentos completos (*full backup*) de todos os passos sucessores, pois suas estimativas não se baseiam no valor de todos possíveis estados sucessores, e sim apenas em uma amostra de estado sucessor. O fato de poderem fazer correções incrementais, ao contrário dos métodos Monte Carlo, que precisam esperar até o final de um episódio, é importante em tarefas com episódios muito longos, onde, mesmo já tendo recebido informações úteis, o agente precisa seguir um longo período de tempo com uma política que já poderia ter sido melhorada de forma incremental. Além disso, muitas tarefas não podem ser divididas naturalmente em episódios, e para essas o uso de Monte Carlo se torna inviável, já que este depende de episódios.

Caso atualizemos o valor de V^{π} de acordo com a equação (2.1), dada uma política e um ciclo de experimentações do agente (observações de estado-recompensa, tomada de decisão, e assim por diante), garante-se que há convergência para o valor correto de V, desde que o passo de correção α decaia com o tempo e que o número de atualizações tenda ao infinito, ou seja, um numero suficientemente grande.

2.2.2 *Q-Learning*

O método *Q-Learning* pode ser considerado como uma adaptação do aprendizado por diferença temporal para o caso onde não há uma política de ação fixa. Este método é um dos métodos mais conhecidos de aprendizado por reforço. O método estima valores de estado–ação, chamados valores Q, que são estimadores numéricos da *qualidade* para um dado par estado–ação. Mais precisamente, um valor $Q(s, a)$ representa a soma máxima descontada das recompensas futuras que um agente pode esperar receber se inicia no estado s, escolhe a ação a e então continua seguindo uma política ótima.

Algoritmo 2.1: Q-Learning

```
1 Inicializa Q(s, a) arbitrariamente ;
2 para cada episódio faça
3     Inicializa s;
4     repita para cada passo do episódio
5         Escolhe a ação a para o estado s utilizando uma política derivada de Q;
6         Executa a ação a, observa o próximo estado s' e a recompensa r;
7         Q(s, a) ← Q(s, a) + α (r + γ max_a' Q(s', a') − Q(s, a)) ;
8         s ← s' ;
9     até s é terminal;
```

O algoritmo aproxima os valores de $Q(s, a)$, à medida que o agente age em um ambiente. Os valores são atualizados de acordo com a regra de atualização, mostrada na linha 7 do Algoritmo 2.1, para cada tupla de experiência $\langle s, a, s', r\rangle$, onde α é a taxa de aprendizado e γ é a taxa de desconto para recompensas futuras. Quando os valores $Q(s, a)$ estão próximos da convergência aos valores ótimos, para qualquer a ou s, a política mais adequada ao agente é a gulosa, ou seja, escolher a ação com o maior valor $Q(s, a)$. Como pode ser visto na regra de atualização, a estimativa dos valores Q não dependem do modelo de transições nem de recompensas.

Uma grande vantagem de se armazenar informações de utilidade na forma de valores Q é que o aprendizado passa a não depender da existência de um modelo de mundo. Por essa razão, costuma-se dizer que a aprendizagem Q é livre de modelo. No entanto, o custo de não haver um modelo explícito de probabilidades de transição é uma maior demora para a convergência, (DOYA et al., 2002).

2.2.3 *Watkins's Q(λ)*

O algoritmo *Q-Learning* pode demorar para convergir porque a cada passo, o valor de um estado é transferido apenas para o estado imediatamente anterior. Uma possível solução para este problema é o uso dos chamados traços de elegibilidade (SUTTON; BARTO, 1998, p.163). Esses traços são basicamente indicadores de quão suscetível cada estado estará para ser corrigido pelo erro de diferença temporal; quanto mais recentemente o estado tiver sido visitado, maior sua elegibilidade. Na prática, os traços de elegibilidade criam uma marcação dos estados visitados em uma trajetória, de forma que a intensidade da marcação é tanto maior quanto mais recentemente o estado tiver sido visitado. O número de estados que esta marcação exerce influência é regulado pelo parâmetro λ. Este parâmetro possui intervalo $[0, 1]$ e quanto mais próximo de 1, maior é o número de estados que pertencem a este traço de atualizações. Numericamente, os traços podem ser vistos como *registros temporais* das ocorrências de um estado, ou par estado-ação.

Os traços de elegibilidade resolvem um dos motivos que levam às baixas velocidades de convergência de algoritmos como o *Q-Learning*, buscando solucionar a limitação de atualizações de passo único. Com o uso de traços de elegibilidade todos os estados da trajetória são "marcados" de acordo com algum grau de intensidade (nível de elegibilidade para atualização). A cada passo de atualização, o erro de diferença temporal é usado para corrigir todos estados anteriores, sendo essa correção tanto maior quanto for a elegibilidade do estado em questão. Se os traços não sofrerem diminuição de intensidade após a última visita do estado, então cada passo de iteração corresponderia a atualizar igualmente todos os estados passados no passado.

O *Watkins's Q(λ)* consiste em uma adaptação do *Q-Learning* para o uso de traços de elegibilidade ($e(s,a)$). O Algoritmo 2.2 mostra o pseudo-código do *Watkins's Q(λ)*.

Algoritmo 2.2: Versão tabular do algoritmo Watkins's Q(λ)

para cada *episódio* **faça**
  Inicializa s e a;
  repita para cada passo do episódio
    Realizar ação a, observar recompensa r e próximo estado s';
    Escolhe a ação a' para o estado s' utilizando uma política derivada de Q;
    $a^* \leftarrow \underset{b}{\text{argmax}} Q(s', b)$;
    $\delta \leftarrow r + \gamma Q(s', a^*) - Q(s, a)$;
    $e(s, a) \leftarrow e(s, a) + 1$;
    para todo *s,a* **faça**
      $Q(s, a) \leftarrow Q(s, a) + \alpha\delta e(s, a)$;
      se $a' = a^*$ **então**
        $e(s, a) \leftarrow \gamma\lambda e(s, a)$;
      senão
        $e(s, a) \leftarrow 0$;
    $s \leftarrow s'$;
    $a \leftarrow a'$;
  até *que* s *seja terminal*;

Este algoritmo também pode ser utilizado como método de controle com aproximação de funções. Este método estima a função de valor de forma parametrizada através de um vetor de parâmetros $\vec{\theta}$. A utilização do método *Watkins's Q(λ)* para representação de estados e substituindo traços de elegibilidade conduz ao fato de que, com o uso de aproximação de funções, não existe somente um traço para cada estado, mas um traço para cada componente de $\vec{\theta}$, que corresponde a muitos estados. O cálculo da características presentes, ϕ_a, correspondem ao estado atual e todas as possíveis ações, a. O $\vec{e}$ é um vetor coluna de traços de elegibilidade, um para cada componente de $\vec{\theta}$. Se a função de valor para cada ação é uma função linearmente separável, então os índices de ϕ_a para cada ação são essencialmente os mesmos, simplificando o processo computacional.

2.3 Aprendizado Multiagente

Há diversas definições para Sistema Multiagente (SMA), no entanto, neste texto é seguida a definição mais simples, encontrada em (SHOHAM; LEYTON-BROWN, 2009,

p.xiii): “Sistema Multiagente são sistemas que incluem múltiplas entidades autônomas (agentes) com informações divergentes, com interesses divergentes ou ambos.” Dentro deste contexto, um SMA cooperativo é formado por um conjunto de agentes autônomos interagindo de modo cooperativo com o objetivo de resolver alguma tarefa, ou um conjunto de tarefas, em conjunto. Por outro lado, em um SMA competitivo os agentes possuem objetivos independentes e buscam maximizar seus ganhos individuais. Mesmo havendo essa distinção clara entre os dois tipos de SMA, de acordo com (HOEN et al., 2006), muitas vezes é difícil definir se os agentes em um SMA são cooperativos ou competitivos, já que na prática os agentes podem ter comportamentos que tanto podem ser classificados como cooperativos quanto competitivos. Por exemplo, um agente cooperativo pode apresentar um comportamento competitivo em relação a algum recurso do sistema e um agente competitivo pode apresentar um comportamento cooperativo em busca de um ganho maior através da cooperação.

Há grandes restrições na aplicação de técnicas tradicionais de aprendizado em SMA, principalmente porque normalmente os agentes não possuem uma visão global do sistema, restrições impedem que cada agente perceba o ambiente de forma completa. Técnicas tradicionais de aprendizado normalmente não podem ser aplicadas diretamente na maioria dos problemas de aprendizado multiagente visto que o espaço de busca de políticas que maximizam a utilidade conjunta é muito grande para o aprendizado conjunto. Outro problema é que agentes independentes aprendendo de modo concorrente fazem com que o ambiente apresente um comportamento emergente imprevisível.

O aprendizado multiagente cooperativo pode ser dividido em duas categorias: aprendizado de times e aprendizado concorrente. No aprendizado de times, um único aprendiz descobre qual deve ser o comportamento de cada um dos n agente, no caso concorrente vários agentes aprendem simultânea e independentemente comportamentos que maximizam a utilidade conjunta. A seguir vamos introduzir esses dois tipos de aprendizado.

2.3.1 Aprendizado de Times

Neste tipo de aprendizado, um agente “aprendiz” busca no espaço conjunto de ações e estados os comportamentos de todos os agentes do time. Este tipo de aprendizado pode utilizar técnicas tradicionais de aprendizado por reforço, já que um agente controla todos os demais, porém este deve conhecer os estados internos de todos os demais. Há problemas de escalabilidade devido ao espaço de ações conjuntas ser muito grande. Por exemplo, suponha um cenário com 2 agentes, onde cada possui 100 estados possíveis, o espaço total de busca do agente “aprendiz” seria 10.000 estados. Outro problema é que todos os dados precisam estar centralizados ou com acesso irrestrito, o que é inviável em cenários intrinsecamente distribuídos.

Existem três tipos de aprendizado de times: homogêneo, heterogêneo e híbrido. Nos times homogêneos, o mesmo comportamento é utilizado para controlar todos os agentes, porém não funciona caso a tarefa exija agentes especializados em subtarefas diferentes, por exemplo: ambiente de busca e resgate onde há agentes com tipos diferentes (bombeiros, policiais e ambulância) que devem cooperar entre si afim de resgatar o maior número possível de civis. No aprendizado de times heterogêneos cada agente pode apresentar um comportamento diferente, porém o número de possibilidades que têm que ser buscadas pelo agente aprendiz se torna mais complexo, sendo que se torna possível apenas para ambientes com poucos agentes. E por fim, no aprendizado de times híbridos, os agentes podem ser divididos em subgrupos de agente, onde cada subgrupo é homogêneo e com um aprendiz.

2.3.2 Aprendizado Concorrente

Neste modelo, há múltiplos agentes que aprendem simultaneamente, isso reduz o espaço conjunto, projetando-o em N espaços separados, onde N é o número de agentes. Este tipo de aprendizado é interessante quando o problema pode ser decomposto em subproblemas razoavelmente independentes. No entanto, ele se torna difícil para o agente porque o aprendizado e adaptação acontece em um ambiente onde há outros agentes também aprendendo, inclusive, em função do resultado do aprendizado dos demais agentes.

Os agentes com aprendizado concorrente podem ser classificados de acordo com informação disponível e compartilhada, (MERKE; RIEDMILLER, 2001):

- Agente Caixa Preta: O agente só toma conhecimento dos outros agentes a partir da dinâmica do ambiente;
- Agente Caixa Branca: O agente possui todo o conhecimento dos demais agentes, e das ações tomadas por eles (o mesmo que JAL);
- Agente Caixa Cinza: são agentes do tipo "caixa preta" mas que podem (ou não) comunicar suas ações ou intenções de ações.

Dentro do aprendizado concorrente, destacamos o problema da atribuição de crédito para os agentes participantes de uma tarefa e a modelagem dos companheiros de equipe. A seguir vamos descrever brevemente estas duas questões.

2.3.2.1 Atribuição de Crédito

Um dos problemas no caso de aprendizado cooperativo, é o de atribuição de credito: se os agentes obtiverem em conjunto determinada recompensa, como distribuí-la entre os agentes? Nem sempre é fácil calcular uma recompensa global e nem sempre essa recompensa global traz informação suficiente para que cada aprendiz consiga se adaptar de modo adequado. Por exemplo, em alguns casos de cooperação, em que um agente efetua uma ação especifica, que pode não ser a melhor ação individual, para permitir que outro efetue alguma ação que depende dela. Neste caso, seria melhor fazer médias das recompensas do que recompensa descontadas, senão o primeiro agente receberia muito pouco em comparação ao agente que "aproveitou" a situação possibilitada por ele.

No caso das recompensas individuais, nem sempre é bom dar uma recompensa local, pois isso faz com que o agente não necessariamente tenha incentivo para cooperar, sendo difícil de ser usada diretamente em problemas de aprendizado cooperativo. O que garante que melhorias locais necessariamente levem a uma melhoria global? O tipo de recompensa depende muito do problema, por exemplo: em problemas como futebol de Robôs a recompensa global se mostra eficaz, já em problemas de busca em mapas o local funciona melhor que o global. Também é possível um tipo de recompensa localizada, sendo um meio-termo entre o global e o local.

2.3.2.2 A Dinâmica do Aprendizado

Em cenários totalmente cooperativos existem métodos que convergem para o equilíbrio ótimo global de Nash, mas para o caso de jogos gerais de soma-zero, isso é mais difícil. Um equilíbrio de Nash é uma estratégia conjunta (um estratégia para cada agente) onde nenhum agente possui um incentivo racional (em termos de recompensa) de mudar a sua estratégia, saindo do equilíbrio. No entanto existem diversas abordagens para resolver este problema, dentre elas:

- aprender não apenas uma tabela Q para si mas estima também tabelas Q para os outros jogadores;

- *Friend-or-Foe Q-Learning* (LITTMAN, 2001): encontra um equilíbrio de coordenação, para agentes cooperativos, ou um equilibro contra adversário, para agentes competitivos;

- *Win or Learn Fast* (WoLF) (BOWLING; VELOSO, 2002): regula a taxa de aprendizado. Se ganhando, aprende pouco; se perdendo, aprende agressivamente

Quando há mais de um equilíbrio de Nash, abordagens que estimulam os agentes a alternarem entre qual equilíbrio vão ficar podem fazer com que o sistema não fique preso em um equilíbrio de baixo ganho para todos. Em alguns casos de coadaptação competitiva, as políticas dos agentes podem resultar em comportamentos cíclicos não esperados pelo o projetista.

2.3.2.3 Modelagem de Companheiros de Equipe

A modelagem de companheiros de equipe (*teammate modeling*) é uma área de pesquisa do aprendizado concorrente que busca aprender sobre os outros agentes do ambiente para fazer estimativas sobre o comportamento esperado e assim agir de acordo. Isto pode ser utilizado para cooperar de modo mais eficiente. Há diversos métodos para este fim, mas normalmente, como não há como modelar exatamente o comportamento dos companheiros de equipe, há uma probabilidade do modelo estar correto.

Há uma questão da recursão infinita (pensamento recursivo) que deve ser tratada, do tipo " O agente A está fazendo X, porque pensa que o agente B pensa que o agente A pensa que o agente B pensa que...", e que precisa ser resolvida em um tempo finito. Em (VIDAL; DURFEE, 1997) há uma classificação dos agentes de acordo com a complexidade que eles percebem os seus companheiros de equipe. Nesta classificação um agente de nível 0 percebe os demais agentes como parte do ambiente. No nível 1 reconhecem que há outros agentes no ambiente mas não tem nenhum outro conhecimento sobre eles. Os agentes de nível 2 reconhecem outros agentes e também tem acesso a algumas informações sobre suas decisões e observações passadas.

Em alguns casos, o desempenho final é muito dependente das crenças iniciais sobre os demais agentes e a realização de suposições sobre as crenças dos demais agentes pode impedir a convergência para uma solução ótima. É mais apropriado minimizar as suposições sobre a política dos demais agentes. Pode ser feito um modelo de cooperação explícito, onde se coopera esperando receber alguma vantagem no futuro. A vantagem pode ser direta (retribuição do favor) ou indireta (não necessariamente o agente favorecido vai me devolver o favor).

2.3.3 Aprendizado e Comunicação

A comunicação pode ser utilizada para que agentes solicitem que outros executem subtarefas ou tarefas dependentes. Caso exista um um modelo das capacidades dos demais agentes, pode-se reduzir a comunicação evitando o envio para todos (*broadcast*), pois o envio fica restrito a um agente ou grupo com uma determinada característica. A comunicação pode ser utilizada para melhorar a coordenação tanto para distribuir modelos mais precisos do ambiente como para aprender subtarefas com outros agentes. Devemos ressaltar que a suposição que há possibilidade de comunicação ilimitada entre todos os

agentes e sem custos é equivalente a reduzir um SMA a um sistema centralizado com um único agente.

Os SMA reais, ou que simulam problemas reais, apresentam restrições quanto à comunicação, por exemplo: custo de envio de mensagem, latência, alcance limitado, etc. O recebimento de mensagens pode aumentar o espaço de estados em que um agente pode se encontrar e a possibilidade de enviar mensagens pode aumentar o espaço de ações do agente (se a ação de enviar mensagem também deve ser aprendida). É preciso pesar o ganho da comunicação e o custo envolvido, Panait e Luke (2005) afirmam que mais pesquisas precisam ser realizadas sobre a utilização de comunicação seletiva, para que a comunicação seja utilizada somente quando necessário.

Os agentes podem usar comunicação para trocar diferentes tipos de informação, por exemplo: leitura de sensores, experiências, políticas, etc. A comunicação pode ser realizada através de comunicação direta (por troca de mensagens) ou indireta, por exemplo: feromônios ou movimentação do corpo ("dança" de recrutamento das abelhas).

2.3.4 Domínios e Aplicações

Os SMAs têm diversos problemas onde pode ser aplicado o aprendizado por reforço multiagente. Esta seção mostra apenas alguns destes domínios. Os domínios podem ser divididos em três classes principais: agentes situados, teoria dos jogos e aplicações reais.

Alguns exemplos de problemas com agentes situados:

- Jogo de perseguição cooperativo: presa-predador, podem ser vários predadores cooperando para pegar uma presa em movimento;
- Busca e Resgate: exemplo RoboCup Rescue, onde agentes com capacidades diferentes devem cooperar para resgatar o maior número de civis o possível;
- Navegação cooperativa: grupo de agentes que se movimentam em um espaço, com o menor tempo possível, sem colidir com obstáculos ou com outros agentes (tipo problema de evacuação de salas em emergência)

Diversos SMA podem ser vistos como jogos, por exemplo, como jogos de estratégia com matrizes de *payoff* baseadas nas ações conjuntas. Por exemplo:

- Jogos de coordenação: jogos com pontos ótimos e vales com altas penalidades (ou pena de atuar com um comportamento sub-ótimo) caso os comportamentos não sejam coordenados
- Dilemas sociais:
  - Dilema do prisioneiro: confessar ou não?
  - Paradoxo de Braess: como escolher o recurso menos utilizado? Se todos escolherem ao mesmo tempo, não é mais o menos o utilizado
  - Tragédia dos comuns: um reservatório finito de recursos. Se todos usarem, acaba e todo mundo morre. Como usar os recursos?
  - *El Farol Bar* (ou *Santa Fe Bar*): agentes têm que decidir se vão ou não a um bar. Se todos (ou poucos) decidirem, ninguém aproveita. O que decidir?

Dentre as aplicações reais, podemos destacar:

- Monitoramento distribuído de veículos;
- Controle de tráfego veicular urbano: os semáforos podem ser controlados por agentes ou agentes controlando partes específicas de uma rede;
- Controle de trafego aéreo: cálculo do percurso de aviões por regiões pré-definidas do espaço aéreo, utilizando rotas mínimas, satisfazendo as restrições (número de aeronaves por região no período) e em tempo real;
- Gerenciamento de redes e roteamento de dados: distribuição de pacotes, estratégias para lidar com falhas, balanceamento de carga, etc;
- Distribuição de energia elétrica: planejamento de instalação de postes de forma a atender todos os consumidores, minimizando custos;
- Cadeias de produção *supply chains*: gerenciar os vários passos de produção de algum produto, respeitando restrições de custo, tempo e requisitos do produto;
- Planejamento de rotas: o problema de transporte de cargas pode ser visto como uma coordenação entre diversas empresas de transporte, entre vários veículos de transporte, ou mesmo como o planejamento conjunto da rota de cada veículo.

2.4 Trabalhos Relacionados

Nesta seção apresentaremos alguns trabalhos de aprendizado multiagente que estão relacionados ao método proposto por este trabalho. A seção 2.4.1 mostra um dos primeiros trabalhos de análise de agentes cooperativos e o impacto da troca de informação nesse tipo de agente. Na Seção 2.4.2, é apresentado o conceito de aprendizado coordenado, esse conceito está diretamente relacionado ao trabalho apresentado na seção 2.4.3. A última seção mostra uma abordagem de aprendizado onde a comunicação é essencial para o aprendizado e há um sistema de hierarquia entre os agentes. Esta última seção é mais extensa, sendo que o método é apresentado com mais detalhes por ser um trabalho relacionado que usa ativamente a comunicação no aprendizado dos agentes, característica fundamental do método OPPORTUNE.

2.4.1 Agentes Cooperativos vs. Agentes Independentes

Em (TAN, 1993), são apresentadas três maneiras de cooperação entre agentes: informação, experiência e conhecimento aprendido. A tese principal do artigo é:

> *"Se a cooperação é feita de modo inteligente, então cada agente pode se beneficiar das informações instantâneas de outros agentes, das experiências episódicas e do conhecimento aprendido."*

A avaliação desta questão é feita em um cenário de presa-predador, onde os agentes buscam capturar uma presa que se move aleatoriamente em mapa do tipo grade 10x10, como mostrado na Figura 2.1. Em cada passo de tempo, cada agente tem quatro ações possíveis: mover para cima, para baixo, esquerda ou direita. Mais do que um agente pode ocupar a mesma célula. A presa é capturada quando ela ocupa a mesma célula que um ou dois caçadores ou os caçadores ocupam células próximas à presa. Após a captura da presa, os predadores envolvidos recebem 1 de recompensa. Os caçadores recebem $-0,1$ de recompensa por cada passo quando não capturam uma presa. Cada caçador possui um campo visual limitado, representado na Figura 2.2.

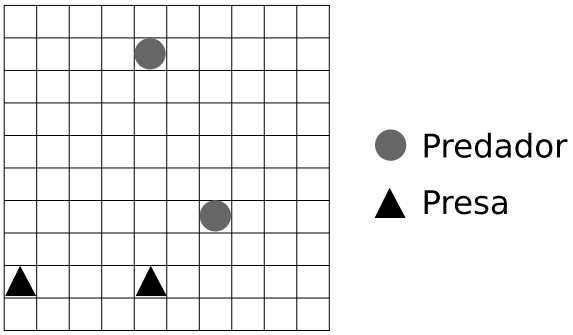

Figura 2.1: Cenário grade 10x10, adaptado de (TAN, 1993).

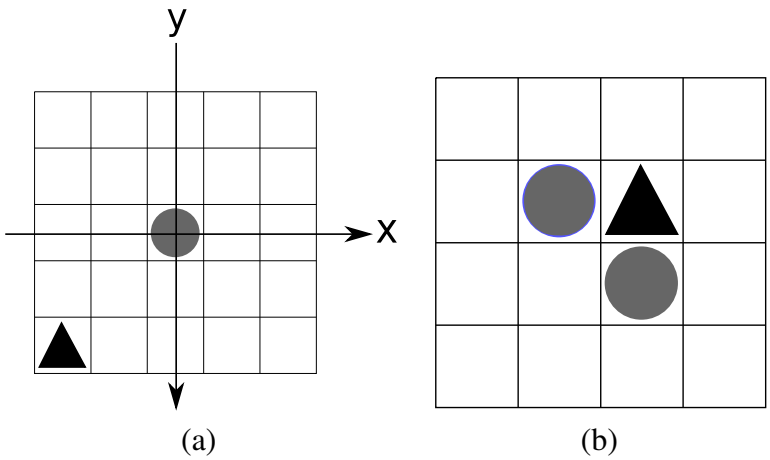

Figura 2.2: Percepção do agente (a) e exemplo de posição de captura (b).

2.4.1.1 Caso 1: Compartilhamento de sensações

Primeiro, o efeito de percepção de outro agente é estudado. Para isolar a percepção do aprendizado, foi realizado em uma tarefa com um caçador para uma presa e um agente observador que não pode capturar a presa. O agente observador realiza movimentos aleatórios e suas observações estão diretamente ligadas ao agente caçador. Em cada etapa, o observador envia a sua ação e percepção de volta para o caçador. As entradas de percepção do agente observador são utilizadas pelo agente caçador somente se ele não detectar a presa.

Após a verificação de que a utilização das informações do observador ajudam no desempenho do caçador, foi introduzido o conceito onde há dois caçadores, e ambos atuam como observador e caçador. Os resultados de quando há somente um caçador e um observador passivo, mostraram uma melhora de 11% na inclusão de observadores e caçadores com o mesmo campo de visão e uma melhoria em torno de 20% quando o observador tem o dobro da percepção do caçador. O desempenho foi avaliado sempre em relação ao tempo em passos para a captura da presa em um cenário sem observador. No caso da observação mútua, o desempenho com uma visão dobrada para ambos os caçadores/observadores ficou 13% melhor do que agentes independentes com o mesmo campo de visão. Quando o campo de visão de ambos é limitado a troca de informações prejudicou o tempo de treinamento dos agentes, embora o resultado em tempo de captura não tenha sido alterado. Esse fato sugere que a troca de informações pode prejudicar o aprendizado, caso seja insuficiente.

2.4.1.2 *Caso 2: Compartilhamento de políticas ou de episódios*

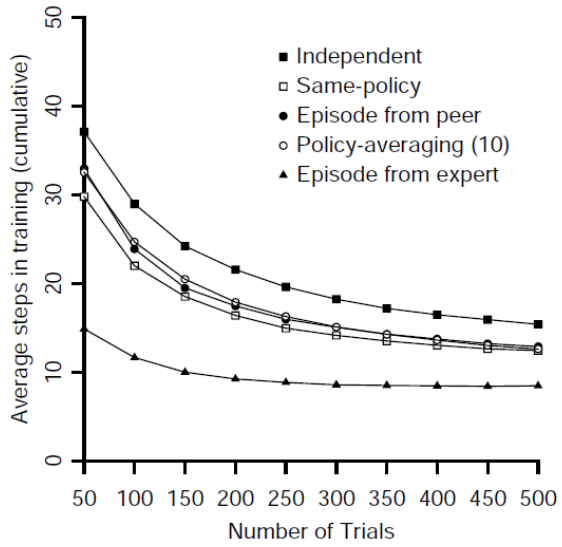

Figura 2.3: Resumo dos resultados, de (TAN, 1993).

Neste caso assume-se que os agentes não compartilham observações. A questão neste caso é: "Se o agente pode completar a tarefa sozinho, a cooperação ainda é útil?" Para responder esta questão, foram estudadas a troca de políticas e a troca de episódios. No caso, um episódio é considerado como uma sequência de observações, ações e recompensas experimentadas pelo agente.

Na troca de políticas, dois ou mais agentes agem com a mesma política, sendo que um agente calcula a política, ou seja, o cálculo é centralizado; ou então eles fazem uma média das políticas recebidas a cada intervalo de tempo pré definido, por exemplo uma média das políticas a cada 10, 50 ou 200 passos. Todos convergem mais rapidamente que os agentes independentes

O compartilhamento de episódios se dá de duas maneiras: um agente envia o episódio para o outro agente assim que atinge o objetivo (captura a presa), o agente receptor interpreta e esse episódio e atualiza sua política; o outro modo seria um agente especialista enviar os seus episódios para agentes inexperientes. Os resultados dos experimentos apresentados na Figura 2.3 mostram que a troca com um agente especialista se mostra mais eficaz se comparando com as demais.

2.4.1.3 *Caso 3: Tarefas Conjuntas*

O caçador pode capturar somente com o outro caçador. A cooperação pode se dar pela observação passiva ou pela troca ativa de sensações e de localização. Os resultados mostram que nesse caso, agentes independentes tendem a se aproximar da presa diretamente, ignorando o outro. No caso da observação passiva (com a extensão da visão para a inclusão da localização do agente companheiro), os agentes mostram-se muito mais eficazes, embora tenham um maior espaço de estados. A cooperação mútua gera um aumento no espaço de estados sem um aumento na dimensão da representação dos estados, isso fez com que o aprendizado fosse mais lento que o aprendizado nos agentes independentes, porém o seu desempenho foi superior, como pode ser visto na Figura 2.4.

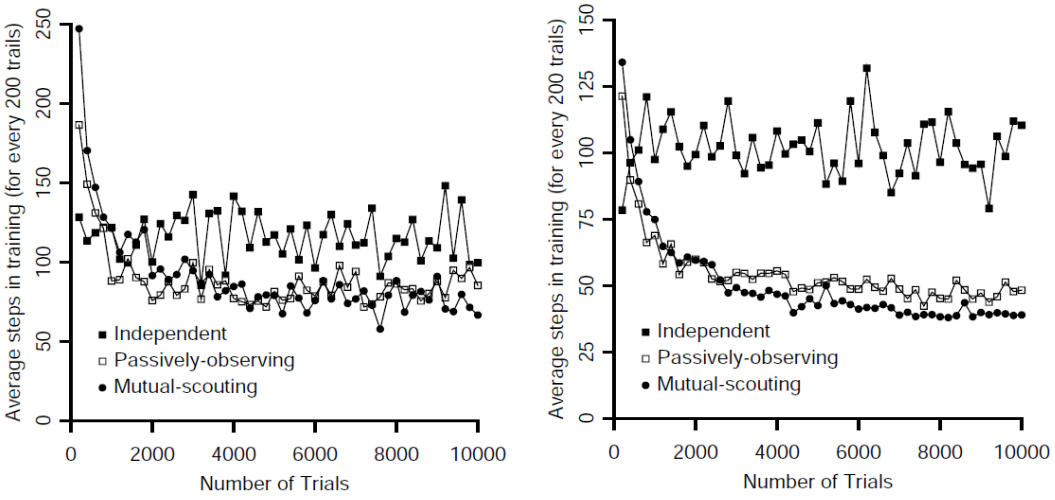

Figura 2.4: Resultados para 2-caçadores/1-presa (a) e 2-presas/2-caçadores (b) com ações conjuntas, de (TAN, 1993).

2.4.1.4 Conclusões

O artigo de (TAN, 1993) foi um dos primeiros a avaliar a troca de informações e seu impacto no aprendizado. Também mostra as diferentes formas de cooperação entre agentes e como essas cooperações podem influenciar o desempenho dos mesmos. Além disso, a troca de conhecimento gera um custo de comunicação e é preciso um espaço de estados grande para o aprendizado de um comportamento cooperativo para tarefas conjuntas.

O artigo também enumera alguns problemas no aprendizado multiagente que ainda se mostram atuais, sendo que alguns pontos relevantes são:

- Como os agente podem aprender a se comunicar?
- E se a presa também aprendesse?
- A troca de informações gera um custo de comunicação que deve ser avaliado;
- A troca de informações deve ser seletiva devido ao crescimento no tamanho do espaço de estados.

2.4.2 Aprendizado por Reforço Coordenado

Nesta seção é feita uma revisão do método de aprendizado por reforço coordenado proposto em (GUESTRIN; LAGOUDAKIS; PARR, 2002), chamado *Coordinated Reinforcement Learning*. Na apresentação do aprendizado coordenado são descritos três abordagens de aprendizado que buscam tirar proveito da estrutura do problema em questão, são eles: uma variação do *Q-Learning* , iteração de política e uma busca direta de política. Nesta seção vamos somente discutir a variante do método *Q-Learning*, já que é a variante mais diretamente relacionada com este trabalho.

A ideia básica do método é decompor a função Q global em uma combinação linear de funções Q locais, dependente dos agentes, como definido na Equação 2.2.

$$Q(s,a) = \sum_{i=1}^{n} Q_i(s_i, a_i) \tag{2.2}$$

Cada função local Q_i para o agente i é baseado no estado (s_i) e na ação (a_i) do agente i, que representam um subconjunto de ações e estados relevantes para o agente i. Utilizando esta representação, cada agente precisa apenas observar as variáveis de estados que são parte da sua função local Q_i. Tais dependências são estabelecidas *a priori* de acordo com o problema tratado e o grafo de coordenação correspondente é construído adicionando-se arestas entre os agentes que tenham suas ações diretamente relacionadas para otimizar um Q_i em particular.

O problema pode ser descrito do seguinte modo: cada agente i seleciona uma ação individual a_i de seu conjunto de ações $\mathcal{A}_i$ e a ação conjunta resultante $\mathbf{a} = (a_i, \ldots, a_n)$ gera uma recompensa $R(\mathbf{a})$ para o time. O problema de coordenação é encontrar a tarefa conjunta ótima $\mathbf{a}^*$ que maximiza $R(\mathbf{a})$ para o time, ou seja, $\mathbf{a}^* = \underset{a}{\operatorname{argmax}} R(\mathbf{a})$. Assumindo a existência de um controlador central, um método de resolver o problema da coordenação seria enumerar todas as ações conjuntas possíveis e selecionar a que maximiza $R(\mathbf{a})$, no entanto esta abordagem se torna impraticável a medida que o tamanho do espaço de estados conjunto aumenta de modo exponencial com número de agentes.

A Figura 2.5 mostra um exemplo de grafo de coordenação, onde devemos selecionar a ação conjunta A que maximiza $\sum_j \max\ Q_j(A)$.

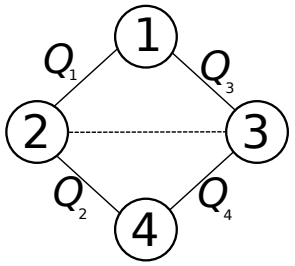

Figura 2.5: Grafo de Coordenação

Temos que:

$$Q = Q_1(a_1, a_2) + Q_2(a_2, a_4) + Q_3(a_1, a_3) + Q_4(a_3, a_4)$$

Ou seja, queremos computar:

$$\max_{a_1,a_2,a_3,a_4} = Q_1(a_1, a_2) + Q_2(a_2, a_4) + Q_3(a_1, a_3) + Q_4(a_3, a_4)$$

Para realizar a maximização, não somamos diretamente todas as variáveis, mas fazemos uma maximização aos pares. Seguindo o exemplo, vamos realizar a otimização do agente 4. Para este agente as funções Q_1 e Q_3 são consideradas irrelevantes, porque ele deve considerar apenas onde ele está efetivamente envolvido, assim:

$$max_{a_1,a_2,a_3}\left[Q_1(a_1, a_2) + Q_3(a_1, a_3) + max_{a_4}\left[Q_2(a_2, a_4) + Q_4(a_3, a_4)\right]\right]$$

$$max_{a_4} \rightarrow a_2 \wedge a_3$$

O agente 4 pode realizar essa maximização no sistema através de uma nova função: $f_4(a_2, a_3)$, onde o valor desejado é o máximo local. Para os demais agentes temos:

Agente 3:
$max_{a_1,a_2}\left[Q_1(a_1, a_2) + f_3(a_1, a_2)\right]$
$f_3(a_1, a_2) = max_{a_3}\left[Q_3(a_1, a_3) + f_4(a_2, a_3)\right]$

Agente 2 toma a decisão:
$f_2(a_1) = max_{a_2}\left[Q_1(a_1, a_2) + f_3(a_1, a_2)\right]$

Agente 1 simplesmente escolhe a ação a_1 que maximiza:
$f_1 = max_{a_1} f_2(a_1)$

O resultado neste ponto é um número que indica $\max_{a_1,a_2,a_3,a_4}$. A maximização da ação conjunta A é obtida através de processo reverso:

1. Maximizando f_1 é selecionada a ação a_1^* para o agente 1;
2. O agente 2 escolhe a ação a_2^* que maximiza $f_2(a_1^*)$;
3. Os agentes 3 e 4 escolhem suas ações ótimas.

Os grafos de coordenação podem ser considerados uma abordagem de aprendizado cooperativo mais eficiente se comparado com a abordagem tradicional e permite uma aproximação do ótimo global. Um problema neste método é que o grafo de coordenação é estático e se for dinâmico o domínio das funções Q locais têm que incluir informação sobre todos os agentes. Com *Q-Learning*, só pode ser utilizado com aproximação de função e assim não há garantias formais de convergência. O método descrito a seguir visa resolver a questão dos grafos de coordenação estáticos.

2.4.3 Aprendizagem-Q esparsa, cooperativa e específica ao contexto

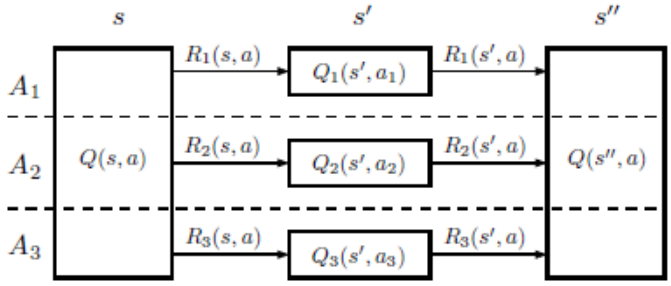

Figura 2.6: Tabelas Q onde o estado s' não é coordenado.

Aprendizagem-Q esparsa, cooperativa e específica ao contexto, ou *Sparse and cooperative multiagent Q-learning*, é uma técnica de aprendizado por reforço que modela requisitos específicos ao contexto dos estados (KOK; VLASSIS, 2004, 2006; KOK, 2006). A ideia geral do algoritmo é permitir grupos de agentes a aprenderem a resolver uma tarefa de maneira cooperativa quando os requisitos globais de coordenação do sistema são conhecidos previamente, sendo que nos estados não coordenados os agentes aprendam de modo independente.

Inicialmente é estudada uma forma compacta de representação do espaço de estados, onde os agentes precisam coordenar explicitamente as suas ações em um conjunto pré definido de estados. Na segunda parte é usada uma abordagem de grafos de coordenação, descrita anteriormente, no qual os valores Q são representados por regras de valor que especificam as dependências de coordenação dos agentes em estados particulares.

A abordagem foi testada em um cenário "presa-predador", de grade toroidal de tamanho 10x10, com uma única presa e dois caçadores, similar ao visto na Seção 2.4.1. Na Tabela 2.1 podemos ver a média das quatro abordagens. A política manual foi feita de

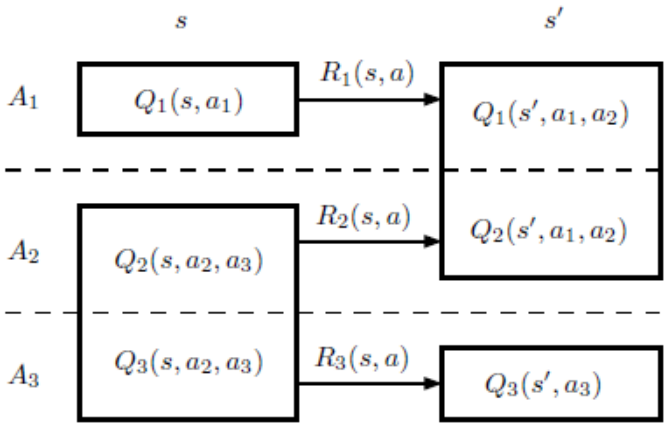

Figura 2.7: Representação tabelas Q de 3 agentes para uma transição do estado s ao estado s', de (KOK; VLASSIS, 2004).

Tabela 2.1: Comparativo entre os métodos

Método	Tempo médio de captura	Número de valores Q
Independente	16.86	97020
Política Manual	14.43	-
Esparso e Cooperativo	13.14	32190
MDP centralizado	12.87	242550

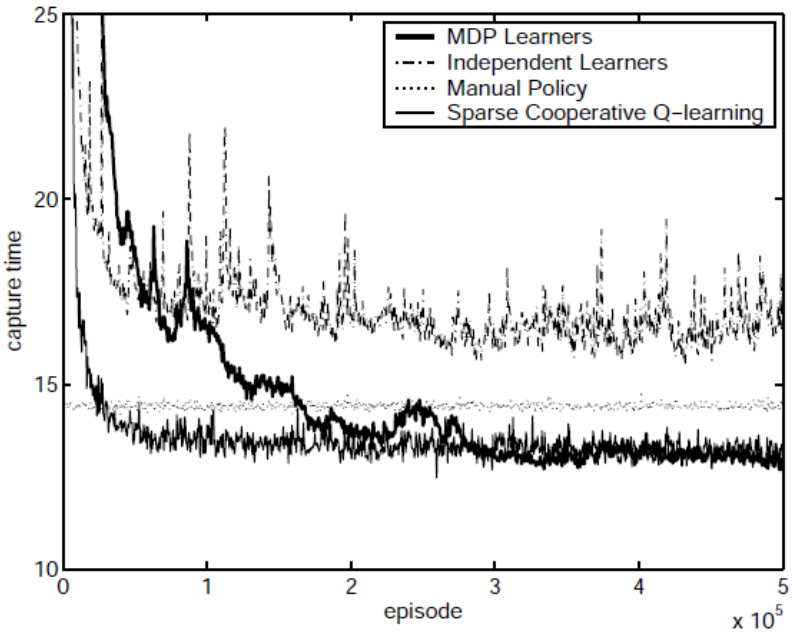

Figura 2.8: Comparativo dos 4 métodos no tempo para captura da presa, durante os primeiros 500 mil episódios (KOK; VLASSIS, 2004).

modo que ambos os predadores buscam minimizar a distância à presa e depois esperar até que ambos estejam situados próximo à presa. Quando ambos os predadores estão situados próximo à presa, um dos dois predadores desloca-se para a posição da presa. Os resultados mostrados na Figura 2.8 indicam que a abordagem Esparsa e Cooperativa converge para um nível ligeiramente superior ao tempo médio de captura se comparado

a um MDP centralizado. A explicação dada pelos autores para essa diferença é o fato de que nem todos os requisitos de coordenação necessários foram adicionados inicialmente como regras de coordenação.

2.4.4 MASPA

O *Multi-Agent Supervisory Policy Adaptation* (MASPA), proposto em (ZHANG; ABDALLAH; LESSER, 2009), é uma arquitetura organizacional de controle com o objetivo de acelerar a convergência de algoritmos de MARL em uma rede de agentes. Este método vai ser descrito com mais detalhes, por ser um método com características mais relevantes à esta tese: estrutura organizacional e comunicação entre os agentes utilizada para a troca de informações.

No MASPA é definida uma estrutura organizacional multicamadas pra uma supervisão automatizada e um protocolo de comunicação para a troca de informação entre os agentes de baixo nível e os agentes supervisores da camada de mais alto nível. É composto de 3 componentes: uma organização de supervisão, um protocolo de comunicação e um mecanismo de adaptação de política que integra o controle organizacional dentro dos algoritmos de aprendizado para guiar o processo exploratório de cada agente. A ideia básica do MASPA é que cada nível da organização de supervisão é uma rede de sobreposição (*Overlay network*) em si mesma. Os agentes supervisores podem gerar uma visão mais ampla do estado da rede utilizando a informação sobre os estados dos agentes do baixo nível bem como a informação dos supervisores vizinhos. Esta visão mais ampla resulta na criação de informação de supervisão que é passada para baixo na hierarquia. Essa informação guia o aprendizado dos agentes de modo a explorar o espaço de estados de modo mais efetivo e consequentemente a uma convergência mais rápida. Para que a informação de supervisão seja atualizada o processo é periodicamente repetido.

O MASPA é a primeira arquitetura que coordena MARL com um controle organizacional. Como outras abordagens de melhoria de algoritmos de MARL, o MASPA requer *conhecimentos adicionais*. O conhecimento é utilizado para decidir qual estrutura organizacional precisa ser criada, quando a informação de abstração de estados é útil e como converter essa informação em informação de supervisão. No entanto, o MASPA é uma arquitetura geral que guia dinamicamente o aprendizado dos agentes. MASPA assume que os agentes vão compartilhar voluntariamente as suas informações de estado. Também assume, de modo implícito, que o SMA pode ser decomposto em uma hierarquia de pelo menos um nível. Essa suposição implica que os comportamentos de agentes distantes espacialmente estão também distantes causalmente, ou seja, agentes próximos afetam o comportamento do seu grupo. Isso funciona em sistemas que podem ser decompostos por proximidade.

Para focar na essência do MASPA, os autores utilizaram estruturas predefinidas de supervisão. As organizações de supervisão podem ser formadas de modo dinâmico durante o aprendizado utilizando uma abordagem auto-organizada *bottom-up*.

2.4.4.1 Supervisão organizacional

A utilização de um sistema de supervisão foi inspirado nos comportamentos organizacionais humanos. Para adicionar o mecanismo de supervisão ao SMA, MASPA utiliza uma estrutura de agrupamentos multinível. Agentes na rede de superposição original, chamados operários, são agrupados de acordo com alguma medida (pode ser distância espacial). Cada agrupamento é supervisionado por um agente supervisor e os agentes membros são chamados de subordinados. O papel de supervisor pode ser realizado por

um agente dedicado ou um operário. Se o número de supervisores é grande, um grupo de supervisores de maior nível pode ser adicionado, assim sucessivamente, para montar uma camada multinível. No artigo, a discussão é focada em uma situação onde cada agente pertence a somente um agrupamento. Dois supervisores de um mesmo nível são considerados adjacentes se pelo menos um de seus subordinados é adjacente a um subordinado do outro supervisor. Existem canais de comunicação entre agentes operários e agentes supervisores adjacentes, subordinados e supervisores. Estes canais podem ser físicos ou lógicos.

2.4.4.2 Protocolo de comunicação

O MASPA considera que o agente pode demonstrar tanto uma dinâmica rápida quanto lenta em relação a como seus atributos (*features*) mudam. O estado abstrato é considerado como de dinâmica lenta e é definido como um vetor de atributos que pode ser projetado a partir de atributos de dinâmica rápida utilizando técnicas como: utilização de componentes parciais, aplicação de métodos estatísticos sobre uma escala temporal ou espacial ou substituindo-se o atributo de mudança rápida por seus parâmetros de distribuição caso estes parâmetros sigam alguma distribuição estatística. O estado abstrato de um supervisor é a projeção da abstração de estados de seus subordinados ou diretamente da dinâmica destes.

O mecanismo usa três tipos de mensagem de comunicação: informação, sugestão e regra. As mensagens de informação são usadas quando o agente subordinado passa informações de estado para o agente superior e quando os supervisores trocam informações de estado. Com base nas informações os supervisores enviam mensagens do tipo regras e sugestões para seus subordinados:

- Uma regra é definida por uma condição e um conjunto de ações proibidas para os estados definidos pela condição;
- Uma sugestão é formada por uma condição, um conjunto de ações e o grau de sugestão.

O grau de sugestão pode ser negativo ou positivo, sendo que o negativo indica quais ações não devem ser tomadas e o positivo indica as ações que devem ser tomadas. Quanto maior o grau de sugestão mais forte é o impacto da sugestão no agente supervisionado. Uma suposição implícita é que as suposições dos supervisores serão corretas na maior parte do tempo assim a penalidade de sugestões ruins é compensada pelas sugestões boas, no entanto as sugestões não são adotadas de modo rígido pelos subordinados e as regras possuem uma prioridade maior que as sugestões. O mecanismo considera que os supervisores não geram regras e sugestões em conflito, no entanto, em uma estrutura de múltiplos níveis, uma decisão local do supervisor pode entrar em conflito com seu superior. Dependendo da aplicação, pode ser definido um sistema de resolução de conflitos. A estratégia aplicada no artigo é a de escolher a regra com mais restrições e combinar sugestões somando os graus da sugestão mais fortemente negativa com a sugestão mais fortemente positiva.

2.4.4.3 Adaptação da política de supervisão

Cada agente melhora sua política conforme suas interações com outros agentes e o ambiente. Uma política pura é definida como uma política que escolhe de modo determinístico uma ação para cada estado. Uma política mista ou estocástica especifica uma

distribuição de probabilidade sobre as possíveis ações em cada estado. O MASPA direciona diretamente a seleção da ação através da exploração sem mudanças no processo de atualização de política. A adaptação de política integra as regras e sugestões recebidas dentro da política aprendida por um algoritmo de aprendizado não supervisionado, gerando uma política adaptada: $\pi(s,a) = \pi(s,a)+$adaptação. A adaptação é influenciada pelo grau de receptividade do agente, esse grau pode variar ao longo do tempo, à medida que o agente visita mais vezes o mesmo estado. Para integrar as sugestões dentro do aprendizado é usado uma estratégia que quanto mais baixa for a probabilidade de um par estado-ação, maior será o efeito de uma sugestão positiva no par e menor será o efeito de uma sugestão negativa.

2.4.4.4 Resultados

O MASPA foi testado em dois domínios diferentes: problema de alocação de tarefas distribuído e roteamento de redes. Os agentes foram agrupados de modo manual utilizando a distância Manhattan. O agente mais central de cada agrupamento é o supervisor que também tem o papel de operário. Os supervisores utilizam heurísticas para gerar as regras e as sugestões, de acordo com o problema a ser tratado.

O algoritmo de aprendizado utilizado por cada agente é o *Weighted Policy Learner* (WPL), (ABDALLAH; LESSER, 2006). Neste problema o estado abstrato do operário é projetado de seus estados e definido pela carga média de trabalho por um período de tempo (no caso igual a 500). O estado abstrato dos supervisores é definido pela carga média em seu agrupamento, que pode ser computada pelos estados abstratos de seus subordinados. Um subordinado envia o estado abstrato para o supervisor a cada período de tempo. Com os estados abstratos, o supervisor gera uma regra que define que para todos os estados onde a carga excede a carga média do agrupamento o subordinado não adiciona uma nova tarefa em sua fila. O grau de sugestão para cada subordinado depende da diferença entre a carga média entre dois agrupamentos, o número de agentes nas bordas e a distância do subordinado à borda. As sugestões são usadas para balancear a carga entre os agrupamentos.

Foram utilizadas três medidas de desempenho: tempo médio de serviço, numero médio de mensagens e tempo de convergência. O tempo médio de serviço indica o desempenho geral do sistema. Para o cálculo do tempo de convergência foi feita uma média sobre um número de tempos de serviço e quando o desvio estava abaixo de um limiar de 0.025 era considerada a convergência. O tempo de convergência então é o primeiro tempo do intervalo que o desvio cai dentro do limiar de 0.025. Foram realizados experimentos em 3 tamanhos de rede do tipo grade: 6x6, 10x10 e 27x27, sendo que todos obtiveram resultados similares.

No artigo (ZHANG; ABDALLAH; LESSER, 2009) foi mostrado somente o desempenho na rede 27x27. Foram testados três padrões de taxa de chegada de tarefas: carga central desbalanceada, onde os agentes centrais recebem tarefas com uma frequência mais alta; com carga de canto e carga de borda. Em cada simulação, o tempo de médio de serviço e o numero médio de mensagens foram computados utilizando uma medida sobre 500 passos de tempo. os resultados são apresentados mostrando a média em 10 simulações. Foram também comparadas quatro tipos de estrutura de supervisão: sem supervisão, supervisão local (onde o agente é o seu próprio supervisor), um nível de supervisão e dois níveis. Na supervisão local o agente tem os seus vizinhos também dentro de sua visão (como se fosse o agrupamento). Segundo os autores, sistemas de supervisão com 3 ou mais níveis não mostraram melhoria se comparado ao sistema com dois níveis.

Como esperado, o MASPA acelerou a convergência do aprendizado e quanto maior a visão dos supervisores, maior é o ganho em desempenho. Tanto que o sistema com carga central e de canto ambos não atingiram a convergência sem o MASPA. As simulações acabaram com os recursos computacionais antes de mostrarem sinais de convergência. Durante os experimentos também foi observado que informação dos supervisores correspondente a um controle de maior granularidade tende a ser de maior ajuda que o correspondente com menor granularidade, aumentando o desempenho do sistema. Sendo que granularidade baixa pode até prejudicar o desempenho do sistema. A maior granularidade consiste em avaliar o agrupamento inteiro como uma única entidade enquanto a granularidade fina opera em membros individuais do agrupamento. Foi também verificado que o tamanho dos agrupamentos e o tempo de intervalo de envio de informações para os supervisores afeta o desempenho. Assim, essas medidas devem ser avaliadas em cada ambiente.

2.5 Conclusão

Este capítulo apresentou uma revisão sobre o aprendizado multiagente e artigos mais relacionados com a abordagem apresentada nesta tese. O objetivo foi apresentar algumas abordagens atuais e as questões em aberto. Analisando-se os trabalhos na área verificamos que existem diversas limitações no modo que o aprendizado multiagente é utilizado. A maioria dos artigos que existem considera poucos agentes (normalmente 2 agentes). Não é considerada a heterogeneidade dos agentes ou abordagens híbridas, ou seja, quando são utilizados mais agentes, todos são idênticos em termos de capacidades, percepções e políticas. Agentes com estados internos ou complexidades internas também não são utilizados no aprendizado multiagente. Podemos resumir os problemas em aberto:

Escalabilidade: dimensionalidade do espaço de estados cresce com o número e complexidade das interações entre eles;

Dinâmicas adaptativas e equilíbrio: como aprender em um ambiente onde os objetivos estão constantemente e adaptativamente sendo alterados? Nem sempre atinge-se a convergência;

Decomposição do problema: decompor tarefas complicadas em subproblemas mais simples que podem ser aprendidos de forma independente.

3 APRENDIZADO MULTIAGENTE COM COORDENAÇÃO OPORTUNISTA

Este capítulo apresenta a abordagem proposta neste trabalho, *Opportunistic Coordination Learning* (OPPORTUNE), e está organizado da seguinte forma: nas primeiras seções a abordagem será contextualizada; na Seção 3.3 será apresentado o método desenvolvido; na Seção 3.4 é apresentado o mecanismo de comunicação e como pode ser aplicada a negociação ao modelo e por fim serão introduzidos os problemas que servirão de cenários para a validação.

3.1 Introdução

O objetivo da abordagem proposta é a utilização do aprendizado por reforço onde os agentes são livres para escolher de modo oportuno quando buscar mais informações sobre o ambiente, bem como quando agir de maneira conjunta com os demais agentes do ambiente. O grande diferencial da abordagem está na negociação entre os agentes e na auto-organização, na medida que não há uma estrutura hierárquica pré-definida.

Para isso, cada agente inicia seu aprendizado sem nenhum conhecimento prévio sobre os demais agentes, mas com possibilidade de percepção e de comunicação com os demais. O agente é capaz de interagir com outros agentes dentro de uma área de observação e de comunicação, podendo esta área estar definida de modo estático ou dinâmico. Dentro da classificação dos agentes de aprendizado concorrente de acordo com o acesso à informação (vista no Capítulo 2, Seção 2.3), os agentes são do tipo "Caixa Cinza", ou seja, possuem informações que podem ou não ser compartilhadas entre os demais agentes. A abordagem proposta utiliza comunicação e negociação entre os agentes para que eles obtenham uma melhor visão do ambiente e para que possam avaliar quais ações (conjuntas ou independentes) podem ser mais promissoras, ou seja, com maior possibilidade de ganho de recompensa.

O ambiente deve possuir as seguintes características, adaptadas da classificação de ambientes de (RUSSELL; NORVIG, 2004, p.41):

- **Dinâmico**: o ambiente está além das capacidades de ação dos agentes por possuir processos externos ditando seu funcionamento, o ambiente está sempre fora do controle do agente. Do ponto de vista do agente, ambientes dinâmicos têm pelo menos duas propriedades importantes: a primeira é que, mesmo se um agente não realiza nenhuma ação entre t_{n-1} e t_n ele não pode assumir que o ambiente em t_{n-1} será idêntico ao ambiente em t_n. Isto significa que, para que o agente possa selecionar uma ação adequada a ser executada, é necessário obter informações para determinar o estado atual do ambiente. A segunda propriedade é que outros processos no

ambiente podem interferir com as ações que o agente pretende executar.

- **Não-determinístico**: cada ação independente pode ter mais de um efeito possível, visto que o efeito é resultado de sua própria ação, das ações dos demais agentes e de fatores externos.

- **Discreto**: a definição de discreto pode ser para *ações*, *percepções* e ao modo de como o *tempo* é tratado. Cada agente possui um número finito de ações e estados observáveis.

- **Parcialmente observável**: cada agente possui uma percepção limitada do estado corrente do ambiente.

- **Cooperativo ou Parcialmente cooperativo**: em um ambiente cooperativo todos os agentes buscam um objetivo em comum, em um parcialmente cooperativo eles buscam um objetivo comum mas há uma competição por algumas questões locais, por exemplo: em um sistema com 3 predadores e uma presa, onde é necessário somente dois predadores para a captura da presa e o agente que fica de fora da captura não recebe recompensa, um dos predadores está competindo para participar da captura da presa, sendo que essa captura deve ser feita de modo cooperativo.

Como exemplo, considere um cenário em que o agente deve mover caixas e é capaz de agir sozinho em algumas situações, ou seja, ele tem a capacidade de mover sozinho alguns tipos de caixa. O agente inicia no sistema tendo o conhecimento das suas possibilidades e com a percepção do ambiente que permite que ele reconheça caixas diferentes. À medida que o agente vai executando ações no ambiente ele percebe que em algumas vezes a recompensa é muito diferente executando a mesma ação no mesmo estado e chegando-se ao mesmo estado objetivo, e que além disso o estado seguinte é o mesmo. Quando o agente não é capaz de mover uma nova caixa, vai tentar obter uma informação sobre o ambiente a fim de perceber o que mudou. Neste caso, o agente teria que combinar a sua observação com a observação dos outros agentes e criar um novo estado, que represente essa situação. Se a caixa ainda não se mover talvez o problema seja a necessidade da existência de mais de um agente para mover a caixa. Neste caso, o agente deveria incluir individualmente as ações dos agentes do grupo até que haja combinação de estado e de ação que representa o que é necessário para mover esse tipo de caixa.

3.2 Ideias gerais

O OPPORTUNE se baseia nos seguintes conceitos gerais, os quais foram, na maior parte, vistos no Capítulo 2:

- Aprendizado por diferença temporal livre de modelo;

- Aprendizado concorrente;

- Troca de informações para gerar uma melhor visão local;

- Coordenação emergente entre agentes pela negociação;

- Representação esparsa do conhecimento;

- Identificação de pontos com diferenças de recompensa.

O OPPORTUNE utiliza o *Q-Learning* como base de seu método. O *Q-Learning* foi escolhido por ser um método de aprendizado simples, por diferença temporal, livre de modelo e amplamente estudado. Além disso ele possui garantias comprovadas de convergência (WATKINS; DAYAN, 1992). O *Q-Learning* também serviu como método de aprendizado em abordagens que motivaram este trabalho: agentes cooperativos com troca de informações, aprendizado cooperativo e aprendizagem-Q esparsa, apresentadas no Capítulo 2, na Seção 2.4.

A estratégia de escolha de ação utilizada no trabalho é $\varepsilon-$gulosa, que é uma estratégia de escolha de ação onde o agente escolhe a ação com o maior valor de qualidade (valor Q) associado com uma probabilidade ε ele irá executar uma ação aleatória (exploratória). Ela foi escolhida por ser uma estratégia que combina de modo simples a escolha dos melhores valores com a curiosidade exploratória. Entretanto, pode ser utilizada qualquer estratégia de escolha de ação que o projetista achar mais adequada, como por exemplo "*softmax*". Em (SUTTON; BARTO, 1998, p.29) há discussão sobre os valores de ε e quando esse valor pode ser decaído ao longo do tempo e também sobre a estratégia "*softmax*".

É utilizado aprendizado é concorrente, ou seja, os agentes aprendem de modo independente e podem ser classificados como "Caixa Cinza", como mencionado anteriormente. Chamamos esse método de aprendizado de oportunista já que o agente escolhe cooperar de acordo com o que lhe é mais oportuno no momento. Não há uma recompensa global a ser partilhada (mitigando o problema da atribuição de recompensas) e as cooperações visam melhorar as recompensas locais.

No OPPORTUNE, uma grande diferença em relação a estrutura de dados do *Q-Learning* tradicional está na representação da tabela Q, sendo utilizada uma representação esparsa dos estados inspirada em uma abordagem apresentada no Capítulo 2, Seção 2.4.3. Porém a abordagem de Aprendizagem-Q esparsa e de contextos específicos considera a hipótese de que os agentes possuem *a priori* informações sobre a sua coordenação e dependências em todo o ambiente. Na abordagem proposta, todos os mecanismos de coordenação surgem a partir das interações entre os agentes, e não há uma predefinição de onde e quando os agentes devem agir de modo conjunto nem as situações onde informações sobre o ambiente devem ser compartilhadas.

Na tabela Q dos agentes OPPORTUNE, as entradas Q armazenam valores relativos aos estados e ações tanto conjuntos quando individuais. Além disso, os agentes não possuem um acesso direto as tabelas dos demais agentes e nem há uma tabela central. Toda a informação é trocada por meio de mensagens e o conteúdo das mensagens possui tamanho limitado. Para isso assume-se que os agentes possuem capacidade de comunicação com os agentes na sua área ou grupo. A comunicação é utilizada para a cooperação.

O tipo de cooperação que utilizamos é inspirado no caso apresentado no Capítulo 2 na Subseção 2.4.1.1. Nesse caso, um agente pode obter informações sobre o ambiente a partir de outro agente (agente observador ou *scout*), a fim de obter uma melhor representação do ambiente e também poder atuar individualmente utilizando essa informação. No OPPORTUNE, não é necessário definir um número fixo de possíveis parceiros para a troca de percepções, porém é necessário que haja uma área de comunicação mesmo que possa variar ao longo da simulação.

Os agentes devem buscar as informações na sua área de comunicação através de um protocolo de comunicação. A comunicação como método de troca de informação é essencial para que tenhamos sistemas distribuídos. No caso do MASPA, dentro dos trabalhos relacionados, visto no Capítulo 2,Seção 2.4.4, a comunicação era utilizada para a sugestão de ações e há uma hierarquia pré-definida, que pode ser composta de vários níveis. No

caso do OPPORTUNE, não há uma hierarquia e todos os agentes são capazes de enviar e receber propostas de ação dentro de sua área de comunicação. Isto facilita o trabalho do projetista do sistema e deixa mais descentralizado, já que os agentes mais no topo da hierarquia possuem uma visão bem extensa do ambiente, gerando pontos com grande fluxo de comunicação.

Para a avaliação das diferenças de recompensa, é utilizado um conceito de erro no par estado–ação, calculado como sendo o Coeficiente de Variação (CV) das recompensas recebidas no par. O CV é calculado com o desvio padrão dividido pela média do conjunto de recompensas $\mathbb{R}$, associado à uma entrada estado–ação. Quanto menor o CV mais homogêneo é o conjunto de dados, deste modo, quanto menor for o CV de um histórico de recompensas mais estável é a recompensa, então menor é a variação da recompensa na entrada. Outra qualidade deste método de cálculo é que o CV é adimensional, isto é, um número puro, que será positivo se a média for positiva, e zero quando não houver variabilidade entre os dados. Isto é muito útil no caso de recompensas com ordens de grandeza diversas que devem ser comparadas.

Na seção seguinte apresentamos o método proposto.

3.3 Método

Algoritmo 3.1: OPPORTUNE

Inicia os valores de tabela Q de modo arbitrário;
$i \leftarrow$ identificador do agente ;
$\mathcal{N}_i \leftarrow$ identificador dos agentes da vizinhança inicial para i;
$\mathbb{S}_i \leftarrow \{ s_i^0 \}$;
repita para cada passo t
    se $|\mathbb{S}_i| > 1$ **então**
        Requisita informações para os agentes participantes de $\mathbb{S}_i$;
        Verifica estado $\mathbb{S}_i$ com base nas informações recebidas ;
    $\mathbb{A}_i \leftarrow$ ação escolhida de $\mathbb{S}_i$, utilizando uma política derivada de Q (ex: ε-gulosa);
    se $|\mathbb{A}_i| > 1$ **então**
        Envia proposta para os agentes participantes de $\mathbb{A}_i$;
        Recebe propostas, podendo mudar $\mathbb{A}_i$;
        se *negociação falhou* **então**
            Escolhe uma ação individual ($|\mathbb{A}_i| = 1$) de $\mathbb{S}_i$, utilizando uma política derivada de Q (ex: ε-gulosa);
    Executa a ação a_i^t de $\mathbb{A}_i$;
    Observe o próximo estado individual s_i^{t+1} e recompensa r_i ;
    `AvaliaPassoAnterior`$(\mathbb{S}_i, \mathbb{A}_i)$;
    $t \leftarrow t + 1$;
    $\mathbb{S}'_i \leftarrow \underset{s_i^t \in \mathbb{S}_i}{\text{argmax}} Q(\mathbb{S}_i, \mathbb{A}_i)$;
    `AtualizaTabelaQ`$(\mathbb{S}_i, \mathbb{A}_i, \mathbb{S}'_i, r_i)$;
    $\mathbb{S}_i \leftarrow \mathbb{S}'_i$;
até s_i^t *é terminal*;

Nesta seção, será apresentado e detalhado o funcionamento do método proposto. O método pode ser dividido em 6 etapas básicas:

I Escolha de uma ação;

II Negociação da ação;

III Execução da ação;

IV Avaliação do passo anterior;

V Observação e avaliação do próximo passo;

VI Atualização da tabela Q e informação de precisão.

O Algoritmo 3.1 mostra o pseudocódigo do OPPORTUNE, sendo que as partes mais extensas foram colocadas como procedimentos externos. A seguir serão apresentados os símbolos e a nomenclatura do método.

3.3.1 Nomenclatura e símbolos do Algoritmo

A notação geral e símbolos pode ser vista na Lista de Símbolos, no início deste texto (página 9). Para facilitar o entendimento, os principais símbolos utilizados no Algoritmo 3.1 são:

- $\mathcal{N}_i$ conjunto de agentes na vizinhança do agente i
- s_i estado do agente i
- a_i ação do agente i
- $\mathbb{S}_i$ conjunto que representa um estado composto do agente i
- $\mathbb{A}_i$ conjunto que representa uma ação composta do agente i
- CV coeficiente de variação
- $\mathbb{R}$ conjunto com histórico de recompensas
- $E(\mathbb{S}_i, \mathbb{A}_i)$ erro do par $(\mathbb{S}_i\mathbb{A}_i)$
- E_{max} valor do erro máximo

Cada agente i possui uma vizinhança $\mathcal{N}_i$ composta pelos identificadores dos agentes que estão dentro de uma área de alcance do agente (linha 3, do Algoritmo 3.1).

No método, é utilizado sempre o estado composto. O estado composto ($\mathbb{S}_i$) é um conjunto de estados relacionados aos agentes (s_i), onde i indica o agente observador do estado particular s. Assim, o agente i pode ter um estado composto (i) com o agente j, ou seja, $\mathbb{S}_i = \{s_i, s_j\}$. Sempre o estado $\mathbb{S}_i$ deve conter um s_i que indica o estado do agente i.

De modo similar à definição do estado, cada ação ($\mathbb{A}_i$) é um conjunto composto por ações relacionados aos agentes a_i, onde i indica o agente responsável pela ação particular. Assim se a ação composta do agente i é composta por mais um agente j, $\mathbb{A}_i = \{a_i, a_j\}$, onde a_i e a_j são as ações individuais dos agentes i e j respectivamente.

3.3.2 Escolha de uma ação

Inicialmente, o agente possui apenas informações sobre o ambiente percebido (estados individuais) e sobre o conjunto de ações individuais possíveis neste ambiente.

A cada passo o agente verifica se o seu estado $\mathbb{S}_i$ é conjunto ou não (linha 6 através da cardinalidade de $\mathbb{S}_i$, no Algoritmo 3.1). Se for conjunto, na verdade ele possui um estado possível, já que i possui apenas suas informações locais. Para verificar se $\mathbb{S}_i$ é o estado corrente, i deve requisitar informação (linha 7) para os agentes que participam de $\mathbb{S}_i$, ou seja, os agentes que observam as informações constantes no estado $\mathbb{S}_i$.

A partir das informações recebidas pelos demais agentes, o agente i constrói o estado atual $\mathbb{S}_i$ (linha 8). Por exemplo: o agente i supõe que o estado atual seja $\mathbb{S}_i = \{(s_i = 1), (s_j = 2), (s_k = 2)\}$, ou seja $|\mathbb{S}_i| = 3$, e sua observação local s_i é "1". Para verificar se o estado suposto é o estado atual, i precisa requisitar aos agentes j e k as suas informações correntes. O agente j responde que s_j é "2", confirmando a suposição, e o agente k envia que s_k é "1". Com base nas informações recebidas, o agente i verifica que estado atual é $\mathbb{S}_i = \{(s_i = 1), (s_j = 2), (s_k = 1)\}$.

Com base neste estado observado e verificado, é realizada a busca da ação a ser executada (linha 9, no Algoritmo3.1). A escolha da ação é realizada utilizando uma estratégia de escolha de ações ($\varepsilon-$gulosa).

3.3.3 Negociação e Execução de uma ação

A próxima etapa é onde pode ser utilizada a *negociação* com os demais agentes, linhas 10 a 14. A estratégia, no passo anterior, retorna uma ação $\mathbb{A}_i$ que pode conter somente a ação do agente ou mais ações (composta). Se for composta por mais agentes, o agente que escolheu a ação deve enviar mensagens para os agentes que estejam envolvidos nessa ação de modo a indicar um valor de recompensa possível caso façam novamente essa ação. Neste mesmo momento o agente pode receber propostas dos outros agentes com ações que podem divergir ou não de sua ação escolhida $\mathbb{A}_i$.

O protocolo de comunicação e negociação será explicado em mais profundidade na seção seguinte. Cada mensagem de proposta contém um valor associado. Com base no valor da mensagem, o agente compara o que é mais vantajoso: manter a intenção de ação atual ou mudar para uma ação que foi indicada com um valor mais alto.

Dependendo dos valores envolvidos, o agente aceita ou rejeita as ofertas escolhendo uma ação a_i a ser executada. O agente i sempre envia uma mensagem de aceitação ou rejeição para todas as ofertas recebidas requisitando a execução da ação individual escolhida a_i. Por exemplo: dois agentes j e k enviaram mensagens propondo que o agente i execute as ações "2" e "3", com valores associados de 0.9 e 0.7, respectivamente. Sendo que o valor indicado pelo agente j é maior que o indicado pelo agente k, o agente i escolhe realizar a ação "2". O agente i envia para o agente j a aceitação da ação "2" e envia para o agente k uma mensagem de rejeição do proposta, sem indicar a ação que ele irá efetivamente executar. Mais detalhes sobre a negociação serão vistos na seção seguinte.

Tendo escolhido a ação, o agente i executa-a no ambiente (linha 15).

3.3.4 Avaliação do passo anterior

Esta etapa está descrita no Procedimento `AvaliaPassoAnterior`. Após a execução o agente i avalia a qualidade do último par estado-ação (linha 17). O agente verifica se o erro do par $E(\mathbb{S}_i, \mathbb{A}_i)$ está acima de um limiar pré definido denominado E_{max} (linha 1). A definição do erro foi introduzida na Seção 3.2, e a Equação 3.1 mostra o cálculo. Se

Procedimento `AvaliaPassoAnterior`$(\mathbb{S}_i, \mathbb{A}_i)$

Entrada: $\mathbb{S}_i, \mathbb{A}_i$

1 **se** $E(\mathbb{S}_i, \mathbb{A}_i) > E_{max}$ **então**
2 $\mathbb{S}_{aux} \leftarrow \mathbb{S}_i$;
3 **se** $|\mathbb{S}_{aux}| \neq |\mathcal{N}_i| + 1$ **então**
4 **enquanto** $(|\mathbb{S}_{aux}| == |\mathbb{S}_i|)$*ou*$(|\mathbb{S}_{aux}| < |\mathcal{N}_i| + 1)$ **faça**
5 Escolhe próximo $j \in \mathcal{N}_i$ que não esteja participando de $\mathbb{S}_{aux}$;
6 Envia mensagem para j requisita seu estado anterior;
7 **se** *recebe* s_j^t *de* j **então**
8 Adiciona s_j^{t-1} ao conjunto $\mathbb{S}_{aux}$;
9 **se** $Q(\mathbb{S}_{aux}, \mathbb{A}'_i) \notin$ *tabela* Q **então**
10 Adiciona $Q(\mathbb{S}_{aux}$,$\mathbb{A}'_i)$ à tabela Q;
11 **senão**
12 $\mathbb{A}_{aux} \leftarrow \mathbb{A}_i$;
13 **enquanto** $(|\mathbb{A}_{aux}| == |\mathbb{A}_i|)$*ou*$(|\mathbb{A}_{aux}| < |\mathcal{N}_i| + 1)$ **faça**
14 Escolhe próximo $j \in \mathcal{N}_i$ mais próximo e que não esteja participando de $\mathbb{A}_{aux}$;
15 Envia mensagem para j perguntando sua última ação executada;
16 **se** *recebe* a_j^t **então**
17 Adiciona a_j^t ao conjunto $\mathbb{A}_{aux}$;
18 **se** $Q(\mathbb{S}'_i, \mathbb{A}_n) \notin$ *tabela* Q **então**
19 Adiciona $Q(\mathbb{S}'_i$,$\mathbb{A}_{aux})$ à tabela Q;
20 $\mathbb{S}_i \leftarrow \mathbb{S}_{aux}$;
21 $\mathbb{A}_i \leftarrow \mathbb{A}_{aux}$;

Saída: $\mathbb{S}_i, \mathbb{A}_i$

estiver, é necessário que i se comunique a fim de obter mais informações sobre o ambiente (linha 4).

$$E(\mathbb{S}_i, \mathbb{A}_i) = CV(\mathbb{R}(\mathbb{S}_i, \mathbb{A}_i)) \tag{3.1}$$

Para o cálculo desse valor, cada entrada na tabela Q possui uma lista de recompensas associadas, ou seja $\mathbb{R}(\mathbb{S}_i, \mathbb{A}_i)$. Que é atualizada juntamente com a tabela Q, este procedimento de atualização será visto na Subseção 3.3.6.

O conjunto $\mathbb{S}_i$ tem a sua cardinalidade aumentada sempre em, no máximo, uma unidade. O agente i busca informações do vizinho j mais próximo em $\mathcal{N}_i$, que não esteja ainda contribuindo em $\mathbb{S}_i$. A proximidade pode ser definida como a distância euclidiana entre o agente e o seu vizinho ou algum outro tipo de medida de distância. O agente i envia um pedido de informação a j e caso receba a informação, adiciona a informação recebida ao estado composto $\mathbb{S}_i$ criando um novo estado composto $\mathbb{S}'_i$. O par $Q(\mathbb{S}'_i$,$\mathbb{A}_i)$ é adicionando a tabela Q do agente i.

Por exemplo, se o estado $\mathbb{S}_i$ já tem o tamanho igual ao número de vizinhos de sua vizinhança atual mais um, ou seja, ele não tem como adicionar mais informação ao estado (linha 11, do Procedimento `AvaliaPassoAnterior`), então o agente i busca obter informações sobre as ações que os seus agentes vizinhos executaram. De modo análogo à adição de informação aos estados, a ação $\mathbb{A}_i$ também só é aumentada em uma unidade

por vez e o agente i requisita informação ao vizinho mais próximo em $\mathcal{N}_i$, que não esteja ainda contribuindo em $\mathbb{A}_i$. O agente i envia um pedido de informação de ação executada previamente ao agente j, e caso receba, adiciona a informação a_j recebida ao conjunto ação $\mathbb{A}_i$, adicionando o novo par $Q(\mathbb{S}_i,\mathbb{A}'_i)$ à sua tabela Q, se necessário.

tabela Q no tempo $t-1$	tabela Q em t	tabela Q em $t+1$
$Q(\{s_i\},\{a_i\})$	$Q(\{s_i\},\{a_i\})$	$Q(\{s_i\},\{a_i\})$
$Q(\{s_i\},\{a'_i\})$	$Q(\{s_i\},\{a'_i\})$	$Q(\{s_i\},\{a'_i\})$
$Q(\{s'_i\},\{a_i\})$	$Q(\{s'_i\},\{a_i\})$	$Q(\{s'_i\},\{a_i\})$
$Q(\{s'_i\},\{a'_i\})$	$Q(\{s'_i\},\{a'_i\})$	$Q(\{s'_i\},\{a'_i\})$
	$Q(\{s'_i,s_j\},\{a'_i\})$	$Q(\{s'_i,s_j\},\{a'_i\})$
		$Q(\{s'_i,s_j\},\{a'_i,a'_j\})$

Figura 3.1: Evolução da tabela Q do agente i

Para ilustrar o funcionamento desta etapa, vamos utilizar a Figura 3.1. A Figura mostra uma tabela Q de um agente i com duas ações possíveis a_i e a'_i, dois estados s_i e s'_i e vizinhança $\mathcal{N}_i$= $\{j\}$. No tempo $t-1$ o agente possui apenas informações sobre ações locais em sua tabela Q, ou seja, ainda não está utilizando informação de vizinhos. Se no tempo t o agente i verifica que $E(\{s'_i\},\{a'_i\})$ é maior que o máximo tolerável (E_{max}), e ainda há a possibilidade de adicionar conhecimento à $\mathbb{S}_i$, já que $|\{s'_i\}| = 1$. o agente i busca o agente j (único em $\mathcal{N}_i$) e envia uma mensagem para j requisitando informação de seu estado anterior. O agente j responde à requisição com a informação s_j.

A partir da informação recebida de j, o agente i cria uma nova entrada $Q(\{s'_i,s_j\},\{a'_i\})$ na sua tabela Q (linha 10, do Procedimento `AvaliaPassoAnterior`). Se tempo $t+1$, i observa erro no par $(\{s'_i,s_j\},\{a'_i\})$, já não é mais possível adicionar mais informação de estado ao conjunto $\mathbb{S}_i$, visto que o estado já contem informação dos dois agentes i e j. Uma vez que o estado já está no seu tamanho máximo atual, o agente i busca adicionar informação de ação (linha 11) de modo análogo à busca por informação de estado. Após receber a mensagem a'_j do agente j, o agente i adiciona mais uma nova entrada $Q(\{s'_i,s_j\},\{a'_i,a'_j\})$ em sua tabela Q.

3.3.5 Observação e avaliação do próximo passo

A quinta etapa é a da observação e avaliação do próximo estado (linha 19 do Algoritmo 3.1). Neste ponto, o agente i busca encontrar o estado $\mathbb{S}_i$ que contenha o estado observado individualmente, s_i^t, e que tenha a maior qualidade associada. Se o estado com maior qualidade associada é composto ele busca obter informações sobre o estado atual dos agentes participantes de $\mathbb{S}_i$ para verificar se este é o estado corrente ou não. Se o estado for simples, ou seja, contém somente a sua observação local, ele não requisita informações. Se for composto ele monta um novo estado $\mathbb{S}_i$ com base nas informações recebidas. Se este estado construído, $\mathbb{S}_i$, não está na tabela Q do agente i, ele usa somente o estado que seja um subconjunto de $\mathbb{S}_i$ que esteja na tabela Q, para a busca da ação a executar.

Por exemplo, temos um agente i que percebe que está no estado s_i^t="1" e o estado composto para s_i^t com a melhor qualidade associada (maior valor Q) é um estado conjunto com o agente j, onde ambos estavam no estado "1". O agente i envia uma mensagem para j perguntando qual o seu estado atual, e j responde que s_j^t="4", então o estado conjunto atual é na verdade $\mathbb{S}_i = \{(s_i = 1),(s_j = 4)\}$.

Caso o agente i não tenha recebido informações ou todos rejeitaram suas propostas

(linha 13) consideramos que a negociação falhou. Desta forma o agente busca uma ação individual com base na mesma política. O agente sempre possui uma ação individual possível porque sua tabela Q é iniciada com todos os valores individuais e esses valores individuais são mantidos sempre atualizados. A seguir será mostrado o funcionamento desta atualização.

3.3.6 Atualização da tabela Q

Procedimento `AtualizaTabelaQ` $(\mathbb{S}_i, \mathbb{A}_i, \mathbb{S}'_i, r_i)$

Entrada: $\mathbb{S}_i, \mathbb{A}_i, \mathbb{S}'_i, r_i$

1 **para cada** $s_j \in \mathbb{S}_i$ **faça**
2 Adiciona s_j a $\mathbb{S}_{aux}$;
3 **para cada** $a_j \in \mathbb{A}_i$ **faça**
4 Adiciona a_j a $\mathbb{A}_{aux}$;
5 **se** $Q(\mathbb{S}_{aux}, \mathbb{A}_{aux}) \in$ *tabela* Q **então**
6 $Q(\mathbb{S}_{aux},\mathbb{A}_{aux}) \leftarrow Q(\mathbb{S}_{aux},$ $\mathbb{A}_{aux}) + \alpha(r_i + \gamma \max_{\mathbb{A}'_i} Q(\mathbb{S}'_i, \mathbb{A}'_i) - Q(\mathbb{S}_{aux},\mathbb{A}_{aux}))$;
7 $\mathbb{R}(\mathbb{S}_{aux},\mathbb{A}_{aux}) \leftarrow r_i$;

O Procedimento `AtualizaTabelaQ` é referente ao passo final do método (linha 20). Há uma atualização dos valores da tabela Q e de recompensa de todos pares de estado-ação que sejam subconjuntos do estado e ação executados e que contenham o agente responsável pela tabela Q. Por exemplo, se o agente i precisa fazer a atualização de sua tabela Q_i, tendo os seguintes valores de estado e ação: $\mathbb{S}_i = \{s_i, s_j, s_k\}$, $\mathbb{A}_i = \{a_i\}$, $\mathbb{S}'_i = \{s_i, s_j\}$, ele vai atualizar as entradas: $\mathbb{S}_i = \{s_i\}$, $\{s_i, s_j\}$, $\{s_i, s_k\}$ e $\{s_i, s_j, s_k\}$. De modo similar os subconjuntos de ações também são construídos e atualizados.

O erro calculado como o maior valor do Coeficiente de Variação (CV) das recompensas ganhas pelo par $(\mathbb{S}_{aux}, \mathbb{A}_{aux})$, por isso o valor r_i é adicionado ao conjunto $R(\mathbb{S}_{aux}, \mathbb{A}_{aux})$. O cálculo do CV foi mostrado anteriormente na Seção 3.2. Para o cálculo desse valor, cada entrada na tabela Q possui uma lista de recompensas associadas, ou seja $\mathbb{R}(\mathbb{S}_i, \mathbb{A}_i)$. A lista de recompensas tem um tamanho máximo pré estabelecido, de modo a guardar as últimas recompensas recebidas tendo realizado a ação $\mathbb{A}_i$ no estado $\mathbb{S}_i$.

Na seção seguinte será abordado o protocolo de comunicação e negociação que os agentes utilizam no método proposto.

3.4 Protocolo de Comunicação e Negociação

Como vimos na seção anterior, no OPPORTUNE os agentes devem utilizar comunicação direta para obter informações sobre estados e ações de outros agentes. Em alguns outros modelos cooperativos a comunicação é deixada de fora do processo de aprendizado e as informações são consideradas sempre disponíveis aos agentes, já na abordagem proposta ela é parte importante porque os agentes sem a comunicação ficariam restritos a ações individuais. O mecanismo de troca de informações busca ser suficientemente simples para que possa ser aplicável em sistemas de tempo real, onde o tempo de troca de mensagem pode ser crítico. O protocolo é simples e foi baseado em modelos conhecidos de linguagem de agentes como o *Knowledge Query and Manipulation*

Language (KQML) e *Foundation for Intelligent Physical Agents–Agent Communication Language* (FIPA-ACL). Não foi utilizado um protocolo padrão mas não há uma restrição teórica que impeça a extensão para o uso destes protocolos futuramente. Para que os agentes se compreendam, é necessário que eles compartilhem a mesma linguagem e ontologia.

Cada mensagem é composta pela identificação do agente que envia, a identificação do agente destinatário, informação, performativa e um valor da mensagem. Sendo assim, mensagem é definida como um tupla <*para, de, performativa, conteúdo, valor*>, onde:

- *para* é o identificador do agente para o qual a mensagem é endereçada;
- *de* é o identificador do agente que enviou a mensagem;
- *performativa* é uma das 13 performativas apresentadas na Tabela 3.1;
- *conteúdo* é conteúdo da mensagem que pode ser um identificador de ação ou de estado individual;
- *valor* é valor da mensagem, que pode ser vazio $\emptyset$ (valor *nulo*), caso seja uma informação sem valor agregado.

Foram definidas 13 performativas, e seus significados, para a troca de mensagens. As mensagem servem para todas as comunicações possíveis entre os agentes: requisições, propostas, envio de informações e aviso de mensagem inválida. As mensagens não são do tipo "broadcast", ou seja, são enviadas para agentes específicos. Essa característica evita o uso desnecessário de banda de comunicação.

As mensagens tem *conteúdo* e *valor* associados. O *conteúdo* é a informação que o agente deseja enviar para outro agente; o conteúdo pode ser vazio caso o agente não esteja enviando uma mensagem com conteúdo. Por exemplo, uma mensagem com a performativa `REQUEST_LAST_ACTION_INFO` é um pedido de informação bem específico que não precisa ter um conteúdo associado. O *valor* também é um campo que pode ser nulo ($\emptyset$), porque quando o agente informa algum valor ele não esta ofertando nenhum valor para o agente que está recebendo a mensagem.

O mecanismo de negociação pode ser escolhido de acordo com o problema ou escolha do projetista. Nesta tese foi utilizado sempre o mecanismo do tipo leilão de primeiro-preço fechado. Foi escolhido devido a sua simplicidade, de modo a avaliar melhor o método de aprendizado em si e não o de negociação. Neste tipo de leilão há apenas um turno, onde os apostadores submetem suas ofertas (*bids*) para os leiloeiros (no caso os leiloeiros podem ser apostadores). Não há rodadas subsequentes e o leiloeiro escolhe a maior oferta. Os valores Q são utilizados pelos agentes como moeda de negociação, mas aqui também poderíamos adicionar outro tipo de moeda. O valor Q foi escolhido devido a ser uma aproximação do valor que o agente espera caso uma ação seja executada em um estado específico.

Como exemplo, considere que o agente i está enviando uma mensagem para o agente j propondo a execução de uma ação x, com um valor de proposta 0.5. A tupla desta mensagem seria $< i, j,$ `PROPOSE_JOINT_ACTION` $, x, 0.5 >$. Caso o agente j tenha recebido outra oferta, com valor maior que 0.5, ou a sua ação previamente escolhida possui valor associado maior que o recebido na oferta, ele envia uma mensagem de rejeição da proposta para o agente i. A mensagem de rejeição seria algo do tipo $< j, i,$ `REFUSE_JOINT_ACTION` $, x, \emptyset >$

Tabela 3.1: Tipos de mensagens trocadas entre os agentes.

Performativa	**Descrição**
REQUEST_LAST_STATE_INFO	Requisita a informação do estado anterior individual do agente.
INFORM_LAST_STATE_INFO	Envia a informação do estado anterior individual do agente.
REQUEST_CURRENT_STATE_INFO	Requisita a informação atual do estado individual do agente.
INFORM_CURRENT_STATE_INFO	Envia a informação atual do estado individual do agente.
REFUSE_STATE_INFO	Recusa o pedido de envio de informação do estado individual.
REQUEST_LAST_ACTION_INFO	Requisita a informação da ação anterior individual tomada pelo agente.
INFORM_LAST_ACTION_INFO	Envia a informação da ação anterior individual tomada pelo agente.
REQUEST_CURRENT_ACTION_INFO	Requisita a informação atual da ação individual tomada pelo agente.
INFORM_CURRENT_ACTION_INFO	Envia a informação da ação individual a ser tomada pelo agente no passo atual.
REFUSE_ACTION_INFO	Recusa o pedido de envio de informação do estado individual.
PROPOSE_JOINT_ACTION	Propõe a execução de uma determinada ação.
REFUSE_JOINT_ACTION	Recusa a execução de uma determinada ação.
UNKNOWN	Acusa o recebimento de uma mensagem inválida.

3.5 Aplicações

O método é facilmente aplicável em SMA onde seja possível a comunicação e onde os agentes sejam livres para interagir e cooperar quando conveniente. O escopo de testes foi reduzido a dois cenários: jogo de perseguição cooperativo e Controle de Tráfego Veicular Urbano (CTVU). O primeiro é um cenário simples, com poucos agentes, porém como vimos no Capítulo 2, é muito utilizado para validação de métodos de aprendizado. Além disso no próximo Capítulo falaremos também um pouco mais sobre a **TJ!** (**TJ!**) e a possibilidade da aplicação do método em jogos cooperativos maiores.

O segundo cenário apresenta grande complexidade real e pode ser modelado como um SMA. Neste cenário, inclusive a modelagem dos estados e recompensas é crítica para o desempenho do aprendizado. Desta forma poderemos explorar como utilizar o método em um cenário prático. Investigando também as propriedades relacionadas às limitações físicas e operacionais existentes em sistemas reais de controle de tráfego. Neste cenário também serão efetuados testes comparativos com outros métodos de aprendizado.

A Tabela 3.2 mostra uma comparação entre os cenários utilizados na validação do método proposto. A primeira característica é no número de agentes. O cenário do jogo

Tabela 3.2: Comparativo entre os cenários de validação.

Característica	**Jogo de Perseguição**	**Controle de Tráfego**
Número de agentes	2 (normalmente menos de 10)	1 a muitos (dezenas)
Agentes no time	Homogêneo	Hétero-/Homogêneo
Colaboração emergente	Importante	Importante
Tempo-Real	Passo	Segundos/Minutos
Acesso a Informação	Ruim/Boa	Muito Ruim/Boa
Ambiente	Estático/Dinâmico	Dinâmico
Controle	Distribuído	Distribuído/Semi-Centralizado/Central
Cooperação	Irrestrita	Restrita

de perseguição cooperativo foi escolhido já que este é um cenário padrão de testes de métodos de aprendizagem por reforço multiagente, embora possua um número pequeno de agentes envolvidos. O controle de tráfego normalmente possui muitos agentes envolvidos, sendo que pode ser simulado considerando-se somente um agente controlador (por exemplo, um cruzamento isolado). Existem diversos trabalhos para o controle de intersecções isoladas, porém este tipo de controle não é o foco deste trabalho.

Em relação aos agentes em um time, ambos os domínios podem ser considerados homogêneos, embora no controle de tráfego nem todos os semáforos possuam o mesmo número de ações ou de percepções, se estas forem modeladas de acordo com os planos semafóricos ou número de vias controladas. A colaboração para o caso do jogo de perseguição cooperativo é essencial já que a presa somente é capturada quando há dois agentes próximos a ela. No caso do controle semafórico a cooperação é essencial para que haja um bom desempenho geral da rede e não somente intersecções isoladas.

Em ambos os cenários o acesso à informação é restrito, sendo que no caso do controle semafórico pode ser muito ruim quando não há sensores suficientes instalados.

Quanto ao ambiente, no caso do jogo de perseguição cooperativo, podemos ter um ambiente estático onde a presa permanece parada na mesma posição. No caso do controle semafórico o ambiente normalmente é considerado dinâmico.

O controle no jogo de perseguição cooperativo normalmente é considerado distribuído, embora se os agentes compartilham todas as informações e as ações são conjuntas não há como diferenciar de um controle centralizado. No CTVU, podemos ter um sistema de controle que pode ser totalmente distribuído (um controlador por semáforo) até completamente centralizado (um controle central para toda a rede). Neste trabalho, utilizaremos o controle semafórico distribuído com a abordagem proposta e centralizado somente como método de comparação.

Outra questão importante nestes cenários é a diferença do tipo de cooperação. No cenário da presa-predador a cooperação é irrestrita porque temos dois predadores que precisam capturar uma presa, não havendo conflito entre os objetivos dos predadores. No cenário de tráfego a cooperação é restrita porque, normalmente, o agente pode estar agindo de maneira cooperativa apenas com um número finito de agentes vizinhos. O motivo deste tipo de cooperação restrita será mostrado em detalhes no Capítulo 5.

Nos capítulos seguintes serão mostradas as particularidades de cada um destes cenários e como o método proposto foi aplicado.

4 JOGO DE PERSEGUIÇÃO COOPERATIVO

Neste capítulo será apresentado o jogo de perseguição cooperativo, também chamado de Presa-Predador (*predator-prey* ou *hunter-prey*), como cenário de experimentação e avaliação do método proposto. Este cenário foi escolhido por ser de simples modelagem e de fácil compreensão, para que pudesse ser avaliado o OPPORTUNE de modo mais separado do domínio o possível. Os trabalhos relacionados à este cenário foram apresentados no Capítulo 2.

4.1 Definição do Problema

Este cenário é composto por agentes "predadores" cujo objetivo é capturar um agente "presa" em uma grade infinita. Em cada unidade discreta de tempo, cada agente (predador ou presa) possui cinco ações possíveis: se movimentar para cima, para a direita, para baixo, para a esquerda ou permanecer parado. O jogo é realizado em passos, onde a cada passo cada jogador pode mover-se para esquerda, direita, acima, abaixo ou permanecer no mesmo lugar. Há diversas variações para este tipo de jogo, em (DENZINGER; FUCHS, 1996), são elas: formato do mapa (tamanho, limitações ou obstáculos), tipos de agentes (movimentação, tamanho, velocidade, capacidades perceptivas e memória), tipo de cooperação (número de predadores necessários para capturar a presa, etc.), objetivo dos caçadores (capturar ou "matar", onde matar é considerado quando o predador ocupa a mesma célula da presa).

Neste trabalho vamos utilizar a variação do problema apresentada em (TAN, 1993). Nesta variação, a grade tem tamanho 10x10 e não há limitações (paredes) no deslocamento e percepção dos agentes, sendo de movimentação infinita (toroidal). O estado terminal de cada episódio (captura da presa) é alcançado somente se ambos predadores estiverem em células adjacentes a presa (Figura 4.1(b)). A cada passo, caso a presa não seja capturada, os predadores são penalizados com -1 pontos. A presa se desloca pelo ambiente escolhendo uma das 4 ações disponíveis aleatoriamente. Caso o estado objetivo seja alcançado (captura da presa), a recompensa recebida por ambos é 10. Não há restrições quanto aos agentes ocuparem a mesma célula.

A percepção de um predador é composta por um número de células em torno da célula na qual o predador se encontra. Nos experimentos realizados esse raio é de duas células por predador, ou seja, cada agente percebe 25 células. O estado perceptivo do predador é representado por coordenadas cartesianas (x, y) e a coordenada relativa à posição da presa. Por exemplo, na Figura 4.1(a) está exemplificado a posição $(2, 2)$ representando que a presa se encontra na célula superior direita em relação ao estado do predador. Caso não haja nenhuma presa dentro do campo perceptivo de um predador, a posição da presa é considerada nula $(\emptyset, \emptyset)$.

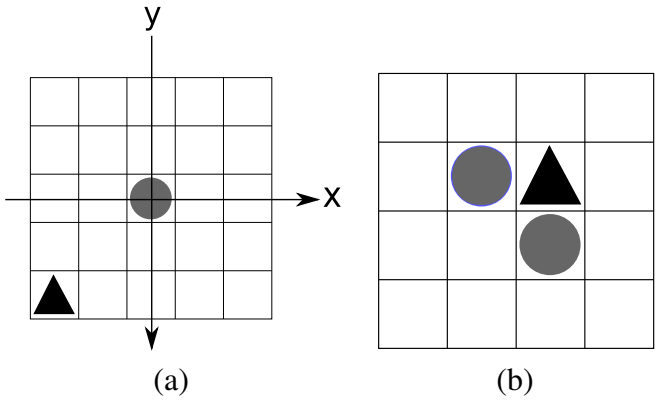

Figura 4.1: (a): estado perceptivo representado por (2,2). (b): possível posição de captura da presa pelos predadores

4.2 Modelagem e análise do MDP

Cada estado neste cenário representa as coordenadas cartesianas (x, y) relativas entre os predadores e a presa. Os atributos que identificam cada par estado-ação são as posições relativas entre os agentes. Por exemplo, no caso de ambos predadores estarem situados na mesma célula que a presa, as coordenadas relativas entre os 3 agentes é $(0, 0)$, pois todos se encontram na mesma célula. Neste caso, o estado conjunto é identificado como que seria $(0, 0), (0, 0), (0, 0), (0, 0)$. Sendo o primeiro e segundo pares de coordenadas referentes aos primeiro e segundo predadores, respectivamente.

Neste cenário percebe-se que o processo de coordenação entre dois agentes exige uma estrutura de dados com dimensões muito maiores do que uma estrutura necessária para aprendizado independente. Para um único agente, a dimensão da tabela Q é de 26 estados (25 células +1 estado nulo). Considerando a combinação de estados entre os dois predadores, a dimensão passa a ser de 676 (26^2) estados por agente, considerando que a posição da presa também deva ser considerada e do outro predador, totalizando 456.976 (26^4) estados. Sendo que cada agente pode realizar 4 ações, a combinação de ações conjuntas é 4^2. Desta forma, a tabela conjunta teria 7.311.616 entradas $(26^4 \times 4^2)$.

Neste cenário, os requisitos formais para a convergência da Aprendizagem Q dificilmente acontecem, pois o espaço de estados e ações é grande demais para permitir uma representação independente do valor de cada estado e ação. Mesmo conseguindo a representação de todos os estados e ações possíveis, o emprego desta técnica leva a um número de passos muito grande por episódio, sendo inviável utilizar esta técnica em aplicações de tempo real.

4.3 Jogo de Perseguição Cooperativo com OPPORTUNE

No cenário de Presa-Predador, cada agente OPPORTUNE é um predador capaz de se comunicar com o outro predador se este está dentro do seu campo de visão. O estado percebido pelo agente é a sua localização e a localização relativa da presa. Cada predador tem 4 ações de movimento possíveis: cima, baixo, direita, e esquerda. Quando a presa tem a capacidade de se mover, ela também possui estas 4 possibilidades de ação.

A vizinhança ($\mathcal{N}_i$) de cada um dos predadores é composta pelo outro predador (somente o identificador) que esteja no seu campo de visão, ou seja, temos as seguintes vizinhanças possíveis: $\mathcal{N}_1 = \{2\}$ (vizinhança do predador 1) e $\mathcal{N}_2 = \{1\}$ (vizinhança do

predador 2). Os parâmetros de aprendizado utilizados no OPPORTUNE para os experimentos neste cenário foram $\alpha = 0.5$, $\gamma = 0.9$, $\varepsilon = 0.05$ para a seleção da ação (ε-gulosa) e $E_{max} = 0.001$.

4.4 Experimentos e Resultados

Nesta seção serão apresentados os experimentos e resultados da aplicação do OPPORTUNE no cenário de Presa-Predador.

Tabela 4.1: Experimentos no cenário Presa–Predador.

Experimento	Política da Presa	Predadores	Episódios
I	fixa (parada)	2	10^4
II	movimento aleatório	2	10^4

Como mostra a Tabela 4.1 foram realizados dois experimentos neste cenário, sendo que em ambos foram utilizados os algoritmos *Q-Learning* e sua variação $Q(\lambda)$ para comparação.

Foi observado que neste ambiente foi necessário a colocação de uma tolerância E_{max} próxima de zero, mais precisamente 0.001. Isto porque o agente executa muitas ações recebendo uma recompensa igual (-1) e assim o desvio é muito próximo a zero e com valores de E_{max} maiores o OPPORTUNE tende a comportar-se como *Q-Learning*, porque não há a busca por informações e a cooperação.

Um episódio acaba quando a presa é capturada, e em todos os experimentos, a cada novo episódio a posição dos agentes é iniciada conforme mostrado na Figura 4.2. Foram realizadas 10 repetições de cada experimento.

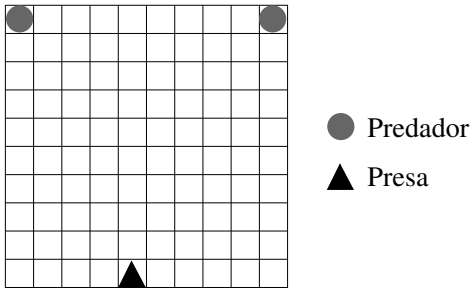

Figura 4.2: Ambiente grade 10x10 em seu estado inicial.

O Experimento I é um ambiente estático, onde a presa permanece na mesma posição durante toda a simulação e os predadores são recolocados na posição inicial a cada novo episódio (Figura 4.2).

A Figura 4.3 mostra o desempenho comparativo no Experimento I. É possível observar que todos o três algoritmos atingiram a convergência, sendo que cada um em um episódio diferente. O OPPORTUNE demora mais a atingir o número de passos para a captura igual aos demais, convergindo apenas depois de aproximadamente 5 mil episódios. Isto se deve ao tempo de adaptação da tabela Q. Após a convergência todos tiveram um resultado muito próximo, conforme mostra a Tabela 4.2. Para a confirmação de que os resultados são estatisticamente equivalentes, foi realizado o teste t-*student* para uma amostra de 200

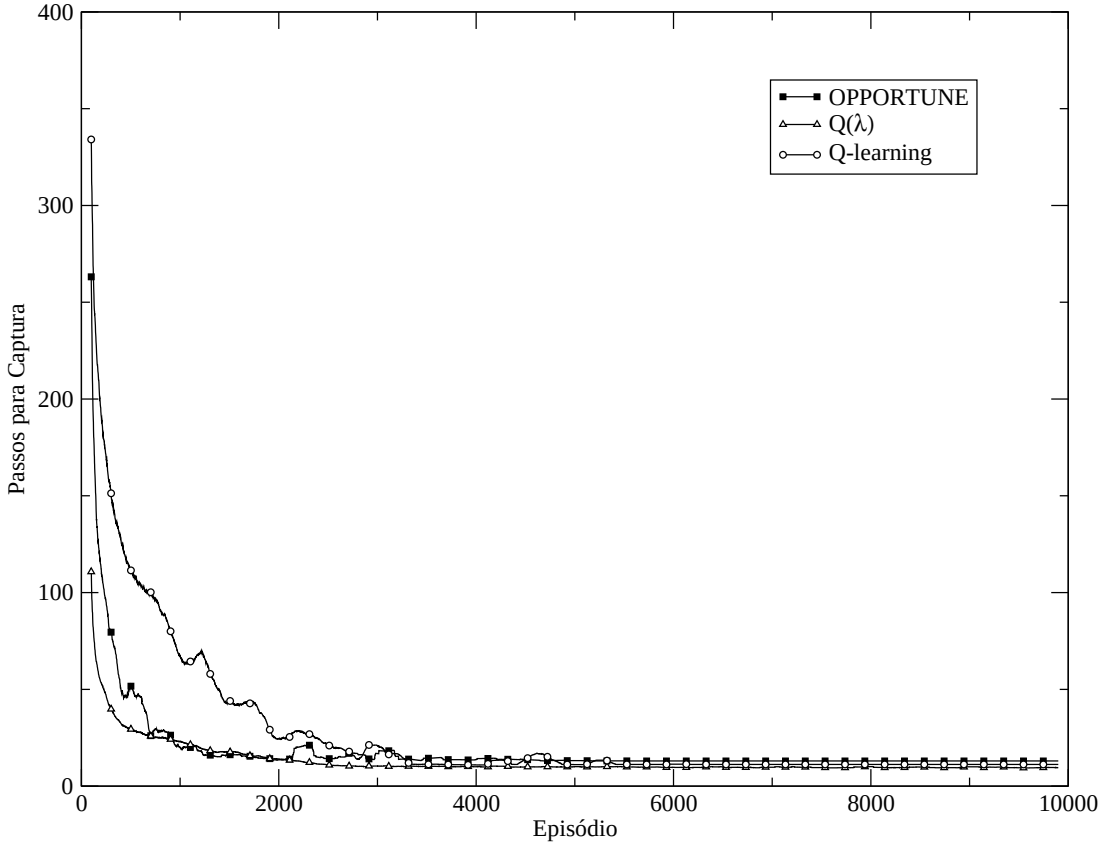

Figura 4.3: Comparação de desempenho no Experimento I

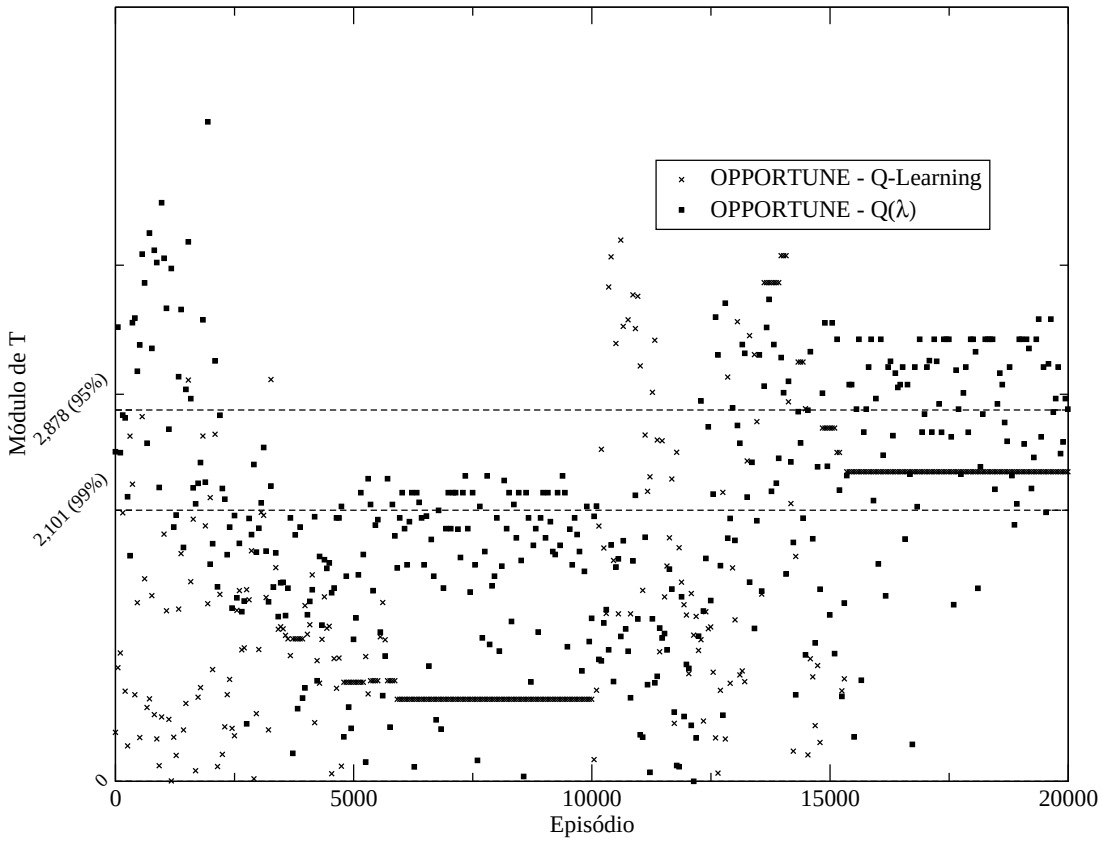

Figura 4.4: Teste t-*student* dos resultados do Experimento I

Tabela 4.2: Comparação dos resultados após a convergência no Experimento I

Método	Média	Desvio
OPPORTUNE	13	1,51
$Q(\lambda)$	9	3,13
Q-Learning	11,2	1,72

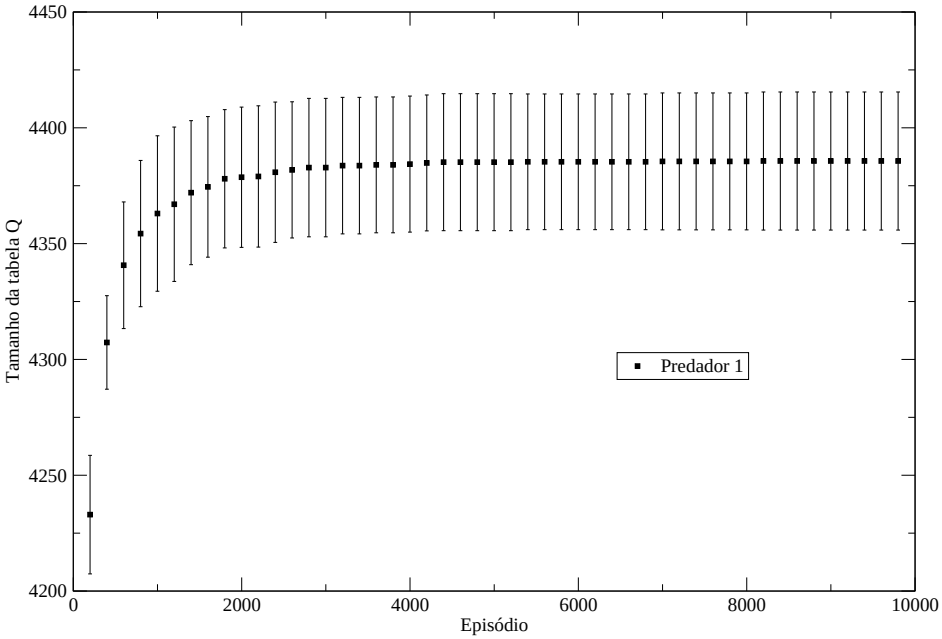

Figura 4.5: Evolução da tabela Q do Predador 1 no Experimento I

episódios de cada um do métodos. A Figura 4.4 mostra o valor do módulo de T dos pontos e os limites dos intervalos de confiança.

As Figuras 4.5 e 4.6 mostram a evolução (média) das Tabelas Q dos predadores 1 e 2, respectivamente, neste experimento. As Figuras apresentam os dados de amostras a cada 200 episódios com seus respectivos desvios. Observando-se estes gráfico, é possível observar que os agentes estabilizaram o número de entradas de suas tabelas Q por volta do episódio 4000. Observa-se também que crescimento das tabelas ocorre por inclusões pequenas por episódio, sendo que os episódios iniciais tem muitos passos, por isso podem apresentar um aumento maior.

Os resultados do OPPORTUNE neste ambiente indicam que a abordagem apresentada tem a capacidade de convergência do algoritmo *Q-Learning* preservada quando há um número suficientemente grande de observações e ações realizadas.

No Experimento II a presa apresenta movimento aleatório. A comparação apresentada na Figura 4.7 mostra que o desempenho do $Q(\lambda)$ foi superior aos demais métodos. Isto porque este método é centralizado e utiliza aproximação de funções para a redução do tamanho da área de busca da solução. O OPPORTUNE apresentou desempenho inferior a todos no início da simulação, mas atingiu um nível de desempenho próximo ao *Q-Learning* após o período de adaptação. A Figura 4.8 mostra o teste T para estes experimentos, nota-se que o OPPORTUNE e o $Q(\lambda)$ novamente são mais distintos que o OPPORTUNE e *Q-Learning*.

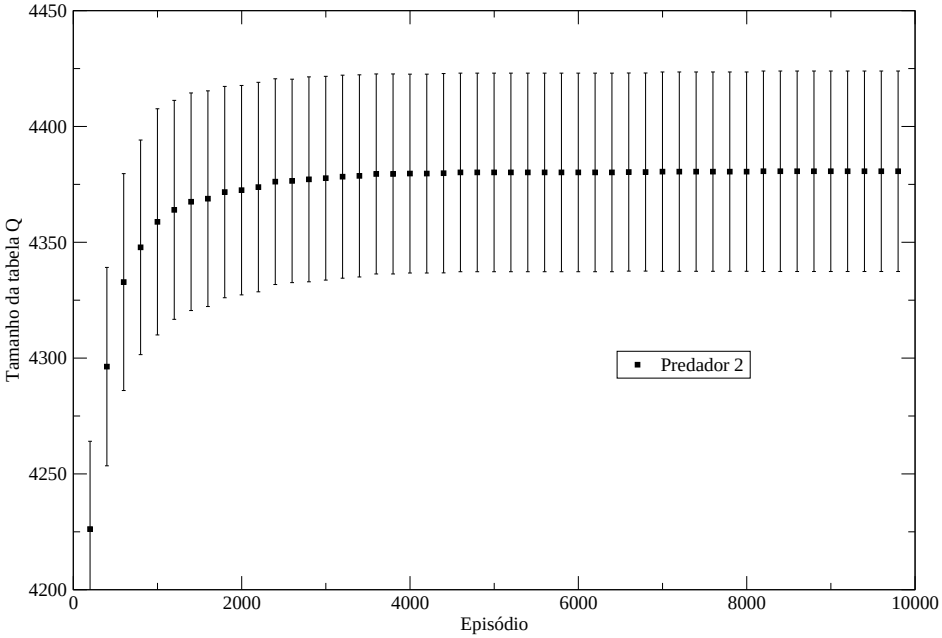

Figura 4.6: Evolução da tabela Q do Predador 2 no Experimento I

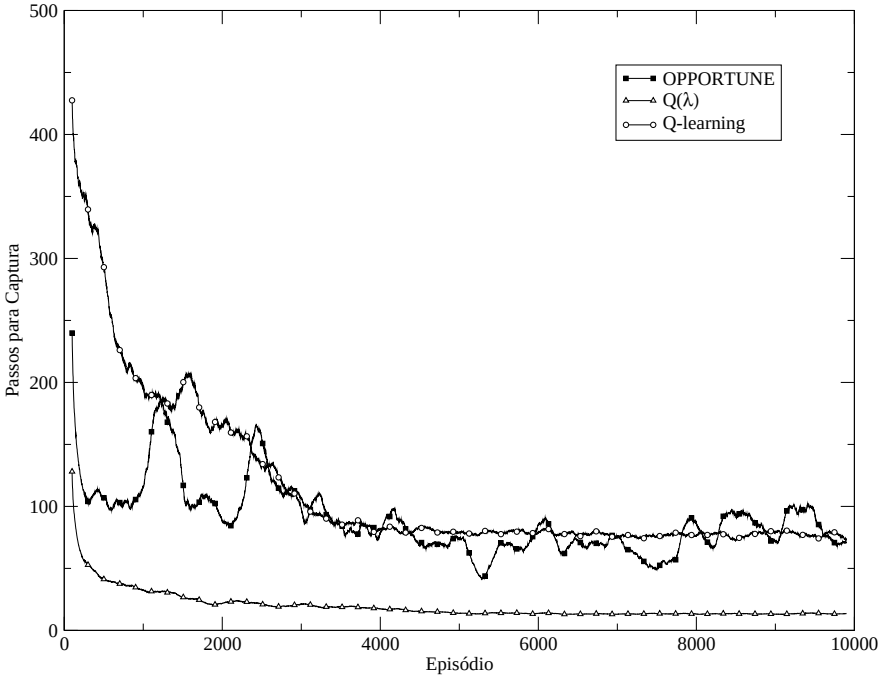

Figura 4.7: Comparativo de desempenho no Experimento II

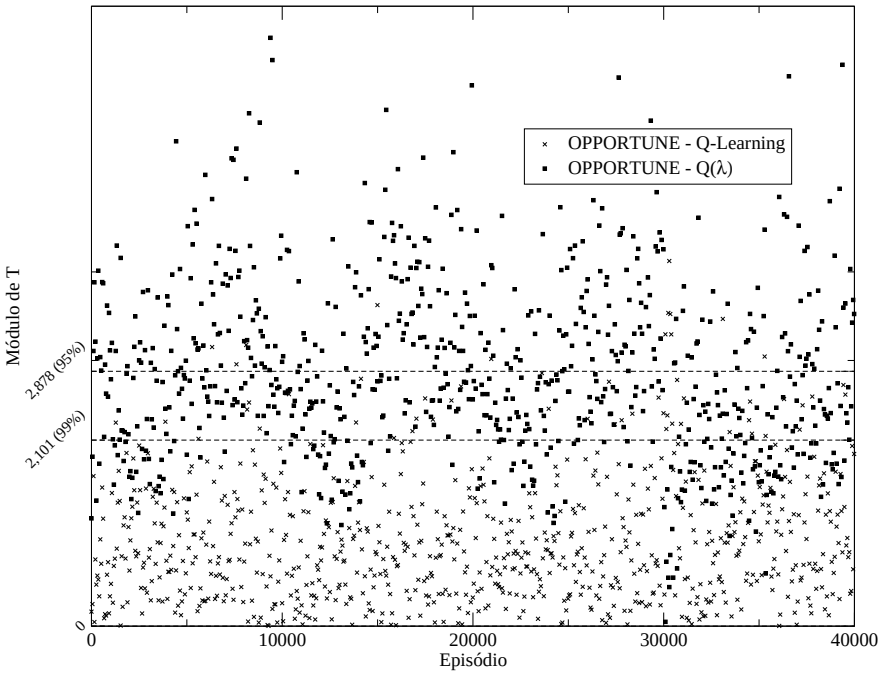

Figura 4.8: Teste t-*student* dos resultados do Experimento II

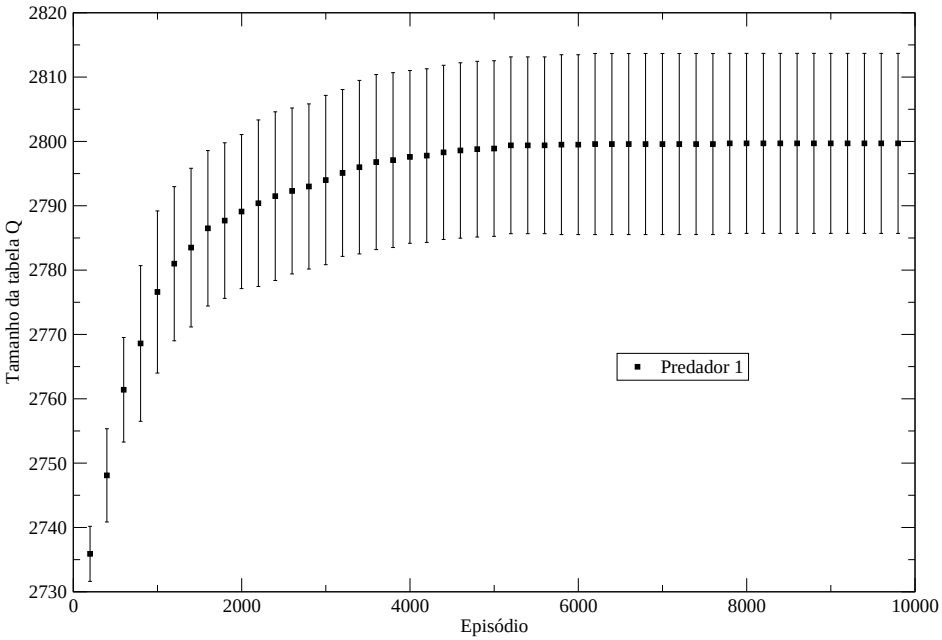

Figura 4.9: Evolução das tabelas Q no Experimento II

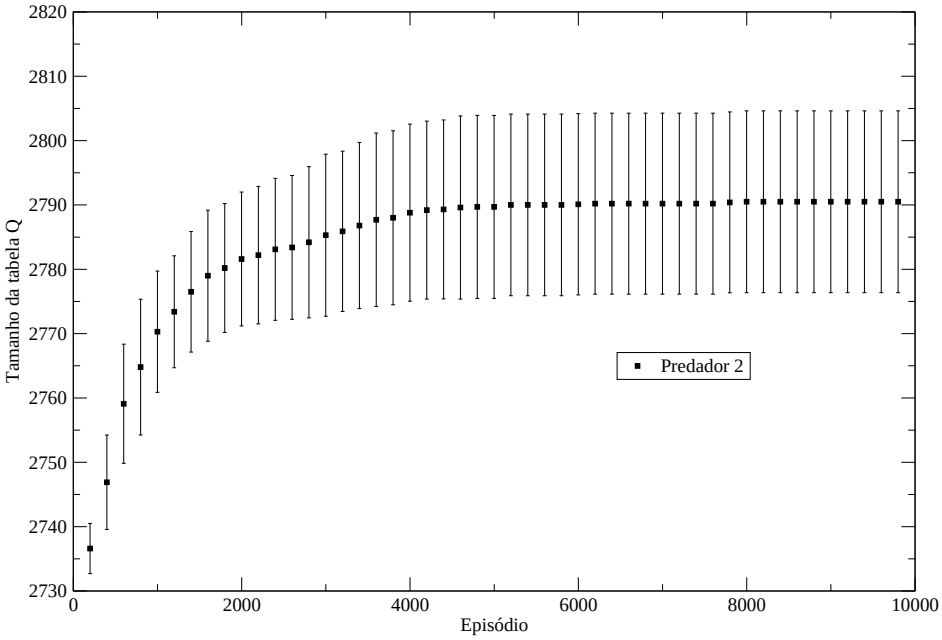

Figura 4.10: Evolução das tabelas Q no Experimento II

Tabela 4.3: Comparação dos resultados no passo 10mil no Experimento II

Método	Média	Desvio
OPPORTUNE	46.14	31.3
$Q(\lambda)$	12.6	7.7
Q-Learning	72,7	60,85

As Figuras 4.9 e 4.10 mostram a evolução da média das tabelas Q de ambos os predadores para o Experimento II, amostra a cada 200 episódios da média e desvio padrão dos valores. Estes resultados mostram que mesmo em um ambiente dinâmico, as tabelas tiveram um comportamento de crescimento similar ao comportamento do ambiente estático.

A Tabela 4.3 mostra o resultado no episódio final (10mil) no Experimento II. Não é possível dizer que houve uma convergência dos métodos OPPORTUNE e *Q-Learning* neste passo visto que o desvio padrão do resultados ainda é alto. No entanto, observando-se novamente a evolução das tabelas Q dos agentes OPPORTUNE nota-se que o crescimento da tabela já esta estável, conclui-se portanto que somente é necessário mais tempo para a convergência dos valores Q em si e não mais de mudança nas tabelas individuais (com a inclusão de novos valores).

4.5 Conclusão

Este Capítulo apresentou a abordagem proposta aplicada a um problema clássico de aprendizado por reforço multiagente. O OPPORTUNE foi testado e obteve bons resultados neste cenário padrão, sendo que a abordagem se mostrou suficientemente seletiva no crescimento de sua base de conhecimento (tabela Q). Estes experimentos servem como teste e comprovação empírica de que o método mantêm as características de convergência

do *Q-Learning*, pelo menos em ambientes estacionários, como mostrado no Experimento I. No entanto, não há uma vantagem em utilizar um tipo de abordagem mais complexa em um cenário tão simples, visto que o *Q-Learning* tradicional, com agentes independentes, obteve um desempenho tão bom quando o OPPORTUNE e $Q(\lambda)$. O *Q-Learning* individual com agentes do tipo "Caixa Preta" (visto no Capítulo 2, Seção 2.3) que percebem os demais como parte do ambiente, funciona bem neste tipo de ambiente porque observar e considerar o outro predador como parte do ambiente é suficiente para os predadores construírem políticas com base somente na posição do outro predador.

O Próximo Capítulo apresentará o problema de controle de tráfego veicular urbano e a aplicação do OPPORTUNE em cenários deste domínio.

5 CONTROLE DE TRÁFEGO VEICULAR URBANO

O Controle de Tráfego Veicular Urbano (CTVU) possui várias características importantes a serem consideradas no contexto de aprendizado, são elas: um grande número de estados possíveis, comunicação limitada, observação limitada, frequência de ação e informação de recompensa atrasada.

Primeiramente será apresentada a definição do problema de CTVU. Dentro deste domínio apresentaremos seus conceitos básicos, alguns sistemas de controle, abordagens que utilizam agentes e o um simulador microscópico que foi utilizado para os experimentos. Também será apresentada uma análise do MDP deste domínio com o objetivo de demonstrar que o mesmo possui um grande espaço de estados, uma das características principais que motivam o uso de uma nova abordagem de aprendizado. Após as definições iniciais, será mostrado como o método proposto OPPORTUNE foi utilizado neste domínio. A seguir, serão apresentados os experimentos em cenários regulares, em um cenário real e seus resultados. Por fim, são apresentadas algumas conclusões e considerações finais sobre este cenário e a aplicação do método.

5.1 Definição do problema

As redes de tráfego urbano estão cada dia mais saturadas e os congestionamentos são um problema em praticamente todas as áreas urbanas densamente populosas. A expansão da malha viária com a criação de rotas alternativas não é mais uma solução possível na maioria das cidades. Sistemas que controlam o fluxo de veículos estão tendo que lidar com cenários cada vez maiores e mais complexos. O controle deve levar em conta diversos fatores, tais como: tempos de percurso, velocidades médias, segurança de motoristas, etc. Além disso, muitas vezes acidentes ou eventos climáticos alteram significativamente o comportamento do tráfego.

A partir da década de 60 do século XX, sistemas de controle de tráfego urbano vêm sendo desenvolvidos tendo como principais objetivos:

- maximizar a capacidade de fluxo das redes urbanas;
- aumentar a segurança do tráfego;
- diminuir os tempos de percurso;
- diminuir os impactos negativos do tráfego no ambiente e no consumo de energia.

A maneira mais comum de controlar o tráfego é a instalação de semáforos em cruzamentos. A inserção de controle semafórico em um cruzamento pode resolver problemas

de priorização, aumentando a segurança do cruzamento. Porém, em muitos casos, a instalação de um semáforo diminui o fluxo e aumenta os tempos de percurso. Para minimizar os efeitos negativos da instalação de semáforos, é essencial uma boa configuração dos planos semafóricos, que são compostos por estágios, tempos de ciclo, e, no caso de semáforos sincronizados, defasagem.

Em redes de tráfego onde não há um padrão bem definido de fluxo, por exemplo picos da manhã e fim de tarde, um sistema de controle de tráfego não adaptativo pode não ser eficaz. Nas grandes cidades os sistemas de tráfego que consideram fluxos bem definidos não podem lidar com fluxos variáveis onde não há uma concentração somente na zona central da cidade, devido à dispersão dos centros de negócios ao longo da rede. Além disso, em algumas cidades, algumas vias consideradas secundárias tornaram-se também vias importantes devido à saturação das vias arteriais. Normalmente, as abordagens visam a otimização de rotas (geralmente rotas principais), com isso nem sempre é obtida uma melhora significativa na qualidade do tráfego na rede como um todo.

Na próxima Seção serão apresentados conceitos que são essenciais para o entendimento sobre funcionamento dos semáforos. Uma discussão detalhada sobre CTVU e sistemas inteligentes de controle de tráfego é apresentada em (BAZZAN; KLÜGL, 2007).

5.1.1 Conceitos Básicos

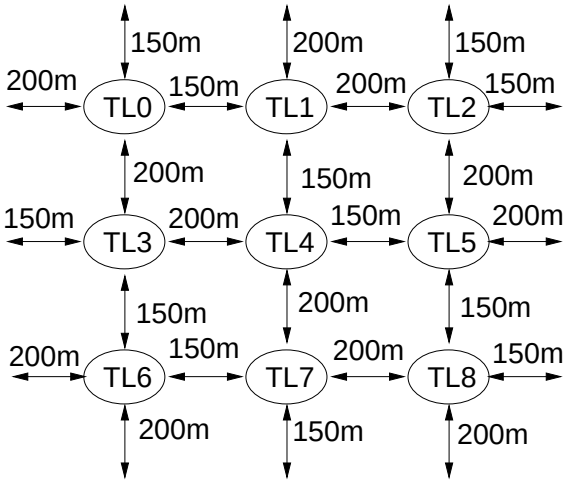

Figura 5.1: Cenário exemplo.

A seguir serão definidos alguns conceitos básicos que devem ser apresentados para a melhor compreensão do funcionamento dos semáforos:

Semáforo: pode ser classificado, (LEITE, 1980), como sendo toda a instalação de controle existente em um cruzamento, incluindo os sinais luminosos, os fios, os instrumentos de controle, etc;

Tempo de ciclo: é o número de segundos para uma sequência completa das indicações do sinal, sendo que pode ter um tempo variável ou fixo;

Estágio: movimentos de tráfego permitidos;

Tempo de fase (ou *split*): uma porção do tempo de ciclo (tempo relativo de verde) reservado para qualquer estágio, (PAPAGEORGIOU et al., 2003);

Plano semafórico: um plano é um conjunto único de fases;

Defasagem (ou *offset*): tempo necessário para um veículo percorrer o espaço entre dois semáforos adjacentes a uma velocidade média pré-determinada.

Outro conceito importante é o de *fluxo de saturação*. De acordo com (ROESS; PRASSAS; MCSHANE, 2004, p.473), o fluxo de saturação é o volume de tráfego (veículos/hora) que pode cruzar um cruzamento sinalizado considerando que o sinal esteja sempre verde. Este valor depende de diversos fatores (DENATRAN, 1979, p.64), dentre os quais: largura da interseção, número de veículos que fazem conversão, declividade da via, estacionamento e a presença de veículos comerciais (ônibus e caminhões). A fórmula padrão deste cálculo é Equação 5.1.

$$F = 525L \tag{5.1}$$

Onde F é o fluxo de saturação em unidades de veículos por hora de tempo de verde e L é a largura (em metros) da aproximação.

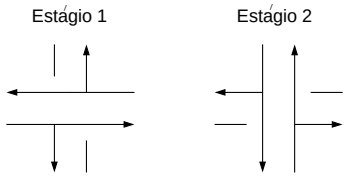

Figura 5.2: Exemplo de movimentos de dois estágios.

Analisando o cenário mostrado na Figura 5.1 é possível exemplificar algumas características do problema. A especificação dos estágios é o que determina as movimentações do tráfego em cada parte do ciclo. A Figura 5.2 mostra um exemplo de especificação de dois estágios para um cruzamento. Cada um destes estágios deve ter uma duração relativa de verde, que é uma porção do tempo de ciclo do semáforo, esta fatia de tempo é chamada de tempo de verde ou *split*.

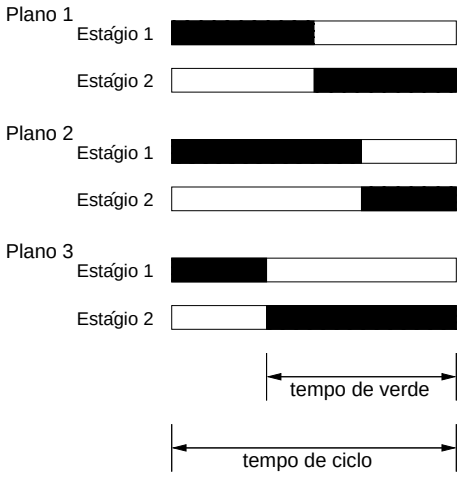

Figura 5.3: Especificação básica de planos semafóricos

A Figura 5.3 mostra três diferentes especificações de planos semafóricos: no "Plano 1" ambos os estágios têm uma porção igual do tempo de ciclo, no "Plano 2", o estágio 1

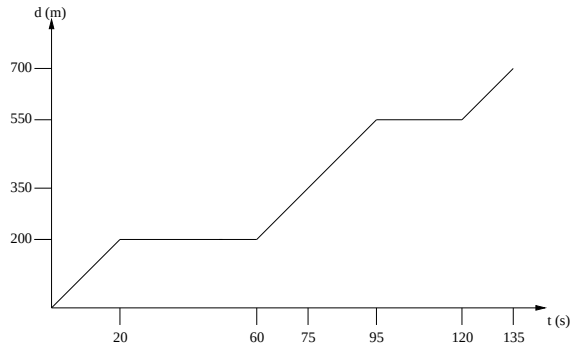

Figura 5.4: Distância x Tempo

tem o dobro do tempo de verde do estágio 2, e no “Plano 3” o estágio 2 tem o dobro do tempo do estágio 1.

Como exemplo, considere um veículo que entra no cenário visto na Figura 5.1 pelo Norte em direção ao Sul, que deve cruzar os seguintes semáforos: “TL1", “TL4"e “TL7", mantendo uma velocidade média de 10m/s. Tal veículo levaria 70 segundos ($\frac{700m}{10m/s}$) para cruzar todos os semáforos sem paradas e sair do cenário. Agora, considerando que todos os semáforos do cenário possuem a mesma especificação de estágios, vista na Figura 5.2, e todos estão utilizando o “Plano 3” com um ciclo de 60 segundos de duração. Nestas condições, o veículo levaria 20s para chegar ao primeiro semáforo “TL1” e finalizaria o percurso em 135 segundos. O gráfico da Figura 5.4 mostra este percurso. Este exemplo mostra que a configuração dos tempos dos semáforos pode gerar um atraso considerável para uma ou mais direções em uma via.

De acordo com Diakaki e colegas, (DIAKAKI; PAPAGEORGIOU; ABOUDOLAS, 2002), existem quatro possibilidades de influenciar as condições do tráfego utilizando semáforos:

- Especificação do estágio: para cruzamentos complexos, a especificação do numero ótimo de estágios pode ter impacto na eficiência do cruzamento;
- Tempo de fase: o tempo de verde de cada estágio deve ser dimensionado de acordo com a demanda das faixas envolvidas;
- Tempo de ciclo: tempos de ciclo maiores geralmente aumentam a capacidade do cruzamento, por outro lado, tempos de ciclo menores aumentam os tempos de espera em cruzamentos subsaturados;
- Defasagem: a especificação da defasagem ótima deve levar em conta a existência de possíveis filas e a velocidade média dos veículos.

A defasagem (*offset*) é utilizada para a sincronização entre semáforos. Esta sincronização é atingida quando dois ou mais semáforos estão executando planos semafóricos que permitem que um veículo passe por ambos sem paradas. Por exemplo, a Figura 5.5 mostra um sistema de sincronização onde a velocidade média é de 45Km/h (12,5 m/s) e o tempo de banda é de 30 segundos. Considerando que do semáforo “S1” ao semáforo “S2” há uma distância de 50 metros, a defasagem do semáforo “S2” em relação ao “S1” seria de 4s (tempo necessário para percorrer 50 metros). Há uma propagação da defasagem até

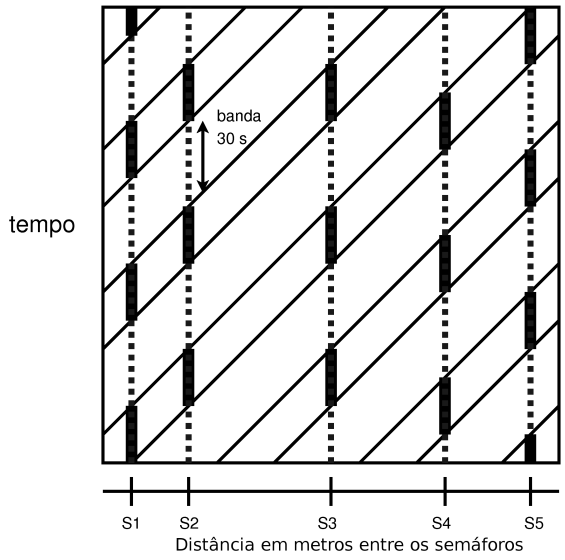

Figura 5.5: Diagrama tempo-espaço de sincronização.

o último semáforo da onda verde (“S5”), que está a 350 metros de “S1”, fazendo com que um veículo saindo de “S1” e mantendo uma velocidade constante de 45Km/h atravesse todos os semáforos sem paradas.

O problema clássico na sincronização de semáforos é encontrar o tamanho de banda maior (ótimo) para diferentes tempos de ciclo e velocidades. Bons planos semafóricos coordenados são muito trabalhosos de serem criados. De acordo com (ROBERTSON; BRETHERTON, 1991), uma rede com 30 a 40 semáforos demanda um ano de trabalho-homem.

5.1.2 Sistemas de Controle de Tráfego

Existem basicamente dois tipos de temporização em sistemas de controle de tráfego urbano: fixo ou adaptativo. A sincronização com temporização fixa é a mais amplamente utilizada e a mais simples, já que não requer a instalação de detectores de tráfego. Os detectores são circuitos eletrônicos digitais que quando conectados a uma bobina, detector de laço ou *loop*, compõem um conjunto capaz de acusar (detectar), eletricamente, a presença de uma massa metálica (veículo) no campo de influência do detector. A temporização que percebe o fluxo de tráfego e adapta-se a ele, exige a instalação de detectores e de um sistema de controle computadorizado mais complexo. O meio-termo entre esses dois tipos de sistema também é utilizado.

O controle fixo utiliza dados históricos de tráfego e normalmente consideram alguns padrões de tráfego distintos de acordo com a hora do dia. O problema com este tipo de controle está nas mudanças que podem ocorrer no fluxo por diversas razões como: adaptação do tráfego (pelos motoristas) aos tempos fixados, acidentes, eventos naturais, obras na via, entre outras. Serão apresentados brevemente alguns dos sistemas mais utilizados comercialmente, sendo que a ferramenta de temporização mais utilizada até hoje é o *Traffic Network Study Tool* (TRANSYT).

5.1.2.1 TRANSYT

O *Traffic Network Study Tool* (TRANSYT) é um dos programas de temporização de tempos de fase mais amplamente utilizados e mais antigos. É executado de maneira *offline* para determinar o tempo fixo ótimo de semáforos coordenados em qualquer rede de vias na qual o fluxo médio de veículos seja conhecido, utilizando modelos macroscópicos e determinísticos de simulação. Embora o TRANSYT seja geralmente utilizado como ferramenta de otimização *offline*, pode ser utilizado de um modo *on-line* fazendo-se a atualização dos dados da rede em pequenos intervalos de tempo e realimentando a rede com os resultados obtidos no simulador. O modelo a ser simulado deve ter entradas de veículos e probabilidades de mudança de direção constantes ao longo de todo o período de simulação.

Os critérios de otimização utilizados são: tamanho da fila, maximização do tamanho da "banda" da onda verde e quantidade de paradas. O programa otimiza as fases e as defasagens relativas dado um conjunto de tempos de ciclo realizando diversas interações entre o módulo de simulação de tráfego e o módulo de otimização de semáforos. Os resultados são calculados em relação a uma rota específica na rede definida pelo usuário.

5.1.2.2 SCOOT e SCATS

O *Split Cycle and Offset Optimization Technique* (SCOOT) (ROBERTSON; BRETHERTON, 1991) é um modelo centralizado de controle de tráfego desenvolvido pelo *Transportation Road Research Laboratory* no Reino Unido. O SCOOT usa detectores instalados nas vias para medir perfis do fluxo de tráfego em tempo real e, juntamente com tempos de percurso e graus de saturação (ocupação relativa à capacidade nominal da via) pré-determinados, prediz filas em intersecções. O sistema de previsão de filas se baseia no uso de Padrões Cíclicos de Fluxo (PCFs). Um PCF é o fluxo de veículos médio em uma direção em qualquer ponto da via durante o tempo de verde, sendo então um platô de veículos como passos sucessivos dentro do ciclo do semáforo. De acordo com (ROBERTSON; BRETHERTON, 1991), o sistema utiliza os mesmos critérios de otimização utilizados pelo TRANSYT.

Quando um veículo passa pelo detector, o SCOOT converte a informação em um tipo híbrido composto por informações de fluxo e ocupação. A partir das informações coletadas, o sistema central envia instruções para os equipamentos localizados nos semáforos. Essas instruções são interpretadas e verificadas pelo semáforo. Caso as instruções sejam realizáveis, ele envia uma mensagem de aceitação, caso contrário, envia uma mensagem indicando uma falha na instrução.

O sistema possui três procedimentos de otimização para sincronizar os semáforos: o de tempos de fase, o de defasagem e o do tempo de ciclo. Cada procedimento de otimização estima o efeito de uma mudança incremental pequena dos tempos no desempenho total da rede de tráfego da região. Um índice de desempenho é calculado baseado em predições sobre paradas e velocidade dos veículos.

Os ajustes do tempo de ciclo, tempos de fase e as defasagens são feitos em conjunto para operar num grau de saturação pré-definido. Os testes mostraram que SCOOT é mais eficaz quando a demanda se aproxima da capacidade (saturação altíssima, em torno de 90%), onde a demanda é imprevisível, e quando as distâncias entre intersecções são curtas (ROBERTSON; BRETHERTON, 1991).

Outra ferramenta similar é o *Sydney Coordinated Adaptive Traffic System* (SCATS), (LOWRIE, 1982). Ele foi inicialmente desenvolvido na Austrália para aplicação em Syd-

ney e em outras cidades australianas. É um sistema dinâmico de controle de semáforos com uma arquitetura parcialmente descentralizada. A otimização do sistema se dá através de mudanças no tamanho do ciclo da fase e tempo de defasagem, além disso, permite também que algumas fases não sejam executadas. O sistema possui uma biblioteca de planos que podem ser selecionados.

5.1.2.3 Traffic-responsive Urban Traffic Control (TUC)

Muitas metodologias buscam a otimização do tráfego mas a maioria apresenta baixo desempenho em condições de saturação, como é o caso do SCOOT e SCATS. Por outro lado, ferramentas mais sofisticadas como OPAC, PRODYN e RODHES aplicam algoritmos com complexidade exponencial. O mecanismo TUC(DIAKAKI; PAPAGEORGIOU; ABOUDOLAS, 2002) foi desenvolvido para prover um método de sincronização sensível ao tráfego em redes urbanas de grande escala, mesmo em casos de saturação. A estratégia consiste em quatro partes principais: controle de tempos de fase, controle de ciclo, controle de offset e priorização de transporte público. Cada um dos quatro controles pode ser utilizado separadamente.

As medidas do número de veículos dentro dos links (ruas) da rede a cada ciclo são enviadas pelo controlador local. Essas medidas podem ser obtidas apenas com o uso de câmeras de vídeo, caso a rede não possua o monitoramento por vídeo, as medidas podem ser estimadas a partir dos dados coletados por detectores de laço induzido, utilizando-se funções não lineares apropriadas. Além disso, o controle utiliza informações sobre o transporte público.

Para o controle dos tempos de fase, formula-se o problema e resolve-se usando controle ótimo linear quadrático, descrito em (DIAKAKI; PAPAGEORGIOU; ABOUDOLAS, 2002). Um tempo de ciclo único é calculado para toda a rede, a fim de permitir a coordenação de cruzamentos através das defasagens. O processo de definição do tempo de ciclo consiste em 3 passos:

1. uma porcentagem “p” pré-definida de ruas com a carga máxima corrente;
2. o tempo de ciclo calculado com controle de retroalimentação;
3. Se o valor resultante para o tempo de ciclo, as vias com um grau de saturação com o nível abaixo de um limiar σ_t pré-definido, então esses cruzamentos não saturados usam o tempo de ciclo igual a metade do tempo de ciclo calculado.

O controle de defasagens considera as seguintes afirmativas:

- Inicialmente calcula-se a defasagem em vias arteriais que não se interceptam;
- Em caso de uma via ser bidirecional, uma defasagem é calculada para cada direção e depois utiliza-se uma média ponderada ou utiliza-se a defasagem que prioriza a direção com maior carga (escolha em tempo real);
- Caso vias arteriais se interceptem, utiliza-se uma ordem de prioridades pré-definida. Cada defasagem é calculada modo sequencial, da maior para a menor prioridade, que deve ser definida pelo engenheiro de tráfego.

Há também critérios pré-definidos que devem garantir que será dada a prioridade para o transporte público apenas quando o tráfego na área solicitada estiver em saturação, ou

acima de um limiar aceitável. Caso necessário, o TUC quebra a coordenação em determinado ponto a fim de responder a um pedido de prioridade. Quando ocorre a quebra, o TUC volta para o estado de coordenação no próximo ciclo.

No artigo, (DIAKAKI et al., 2003), o TUC foi testado em dois cenários reais: nas cidades de Tel Aviv e Jerusalém. Em ambas as redes as decisões do controle de duração de fase são aplicadas uma vez a cada ciclo de semáforo, sendo que o controle do tempo de ciclo e defasagem é ativado em intervalos constantes de 7,5 minutos. Em ambos os cenários, o TUC foi simulado por um período de 4 horas, equivalente ao período das 6h as 10h da manhã, utilizando dados de picos previstos. O desempenho foi comparado ao controle fixo (incluído defasagens pré-calculada), desenvolvidos por engenheiros de trafego utilizando ferramentas específicas.

5.1.2.4 Conclusões

Abordagens centralizadas de controle de tráfego não podem lidar com a complexidade crescente das redes de tráfego urbano. Os sistemas adaptativos geralmente não permitem mudanças na topologia da coordenação dos semáforos, por exemplo, a troca de direção de coordenação muitas vezes não é possível. Já no TUC, o cálculo depende em muito das prioridades definidas pelo engenheiro de tráfego. Além disso o processamento ocorre de maneira centralizada, sendo que uma falha no controle central ou na difusão de dados pode levar a um mal funcionamento de todo o sistema. Nos cenários apresentados, as redes utilizadas para os experimentos são relativamente pequenas e as prioridades são previamente definidas em todos os pontos onde há diferentes possibilidades de coordenação, não havendo uma necessidade do algoritmo detectar pontos críticos e resolver conflitos.

Embora existam diversas estratégias de controle de tráfego, a maioria das cidades ainda operam com estratégias de controle fixo que são de difícil manutenção. De acordo com (PAPAGEORGIOU et al., 2003), mesmo quando sistemas modernos de controle são instalados, as estratégias empregadas são ingênuas ou fracamente testadas. Outro problema é que normalmente estes sistemas modernos são instalados em apenas algumas vias arteriais da rede e isto pode gerar problemas para o resto da rede, como por exemplo, aumentar o grau de saturação em vias com controle fixo.

As questões apresentadas mostram que uma otimização, mesmo com atualização em tempo real, que vise apenas à sincronização de uma rota não pode lidar com mudanças nos padrões de tráfego de toda a rede. Isso acontece já que o movimento de veículos é um processo altamente dinâmico onde o plano para o estado corrente, raramente pode ser determinado previamente. Abordagens flexíveis e robustas, embora apresentem elevado custo inicial da instalação de equipamentos de sensoriamento, mostram-se necessárias.

5.1.3 Abordagens baseadas em IA

Em (BAZZAN, 2005) uma abordagem baseada em IA é descrita, onde cada semáforo é modelado como um agente. Cada agente possui planos pré-definidos para coordenação com agentes adjacentes. Planos diferentes podem ser escolhidos para haver coordenação em diferentes direções de acordo com a hora do dia. Os principais benefícios dessa abordagem são que os agentes podem criar subgrupos de sincronização para melhor atender às necessidades do fluxo em alguma direção, não há um controle central e não há comunicação nem negociação direta entre os agentes. No entanto, são necessárias matrizes de recompensa (*payoff*) e essas matrizes devem ser explicitamente definidas pelo projetista do sistema. Isto faz com que a abordagem consuma tempo quando diferentes opções de coordenação são possíveis ou se há um grande número de vias e cruzamentos a serem

controladas na rede.

Em (OLIVEIRA et al., 2004), foram utilizados conceitos de inteligência de enxames de modo que cada semáforo seja um agente com comportamento de inseto social. Cada um de seus planos semafóricos é visto como uma tarefa a ser executada e tem um estímulo associado que varia de acordo com as mudanças no meio. O estímulo de cada plano semafórico é relacionado com o rastro de feromônio deixado pelos veículos que estão trafegando nas ruas controladas. O agente percebe os rastros de feromônio e assim identifica o estímulo relacionado com cada plano. Por agirem como insetos sociais, os agentes possuem limiares associados a cada tarefa que indicam a propensão do agente em executá-la dado um certo estímulo. A abordagem inspirada em insetos sociais mostrou-se adequada para o controle de semáforos. A liberação de feromônio pelos veículos é uma metáfora factível com os detectores reais de movimentação de veículos. A comunicação por stigmergia (comunicação pelo ambiente) mostrou-se vantajosa pelo fato de não haver a necessidade de trocas de mensagens diretas. O grande problema desta abordagem é a aleatoriedade do sistema e a falta de garantias, além do que o modelo visa muito o ótimo local e não o global. Em (OLIVEIRA; BAZZAN, 2006, 2007) o modelo foi avaliado sob o aspecto da criação de grupos dinâmicos de coordenação.

O problema de coordenação de semáforos foi abordado como um meio-termo entre uma coordenação completamente autônoma com comunicação implícita (como o modelo anterior) e uma solução centralizada clássica em (OLIVEIRA; BAZZAN; LESSER, 2005). Neste modelo, a coordenação de semáforos é abordada como um problema de otimização em tempo real (*on-line*) para a utilização do algoritmo OptApo, (MAILLER; LESSER, 2004). Usando o algoritmo OptApo, a mediação cooperativa é executada pelo agente com mais informações sobre o subsistema (determinado pelo grafo de relacionamentos). Dentro desse subsistema, o mediador busca as mudanças que minimizarão o custo. No cenário do tráfego, a mediação além de reduzir os custos tem que lidar com as mudanças no padrão de tráfego. O processo de mediação demora alguns ciclos, especialmente nas primeiras mediações, já que a visão dos agentes vai expandindo-se ao longo da execução. O grande problema deste método está na grande quantidade de mensagens trocadas e na demora do processo de otimização como um todo.

Em (BAZZAN, 2009) é apresentada uma visão geral de controle de tráfego e direções de pesquisa nesta área, tais como: quando considerar o problema de tráfego como cooperativo; a dimensionalidade do problema associado ao aprendizado multiagente com diversos agentes; coevolução e o papel dos motoristas em uma rede de tráfego veicular. Uma das características do OPPORTUNE é que os agentes são capazes de decidir quando cooperar, esta capacidade está diretamente relacionada com a questão de quando considerar o cenário de controle de tráfego como cooperativo. Outra característica do OPPORTUNE está relacionada com a dimensionalidade do problema de aprendizado multiagente, já que o número de ações estados conjuntos é inferior ao das abordagens de aprendizado cooperativas tradicionais.

5.1.4 Modelagem e análise do MDP

Modelar o problema do controle de tráfego urbano como um MDP é essencial para que possam ser aplicados mecanismos de aprendizado por reforço. O mapeamento dos estados pode ser feito considerando-se diferentes níveis de detalhamento.

Considerando-se um nível de detalhamento de estados definido pela ocupação relativa de vias no cruzamento, podemos considerar os estados uma relação entre as filas e as direções. Por exemplo, se um cruzamento possui 4 vias de entrada, uma em cada direção,

poderíamos considerar que o semáforo possui 5 estados possíveis: um estado onde todas as vias estão com aproximadamente o mesmo tamanho de fila média e 1 estado para cada via predominante (estado 1, via Norte com maior fila; estado 2, via Sul com maior fila; estado 3, via Leste com maior fila; estado 4, via Oeste com maior fila).

O número de estados conjuntos possíveis ($|\mathcal{S}|$) é relativo ao número de cruzamentos controlados multiplicado pelo número de estados locais por cruzamento.

Considerando que cada agente possui o mesmo numero de ações possíveis e que o número de ações de um cruzamento é o número de planos do mesmo, o número de ações conjuntas possíveis ($|\mathcal{S}|$) é o número de ações por cruzamento elevado ao número de cruzamentos controlados.

Utilizando Q-Learning de modo centralizado, onde cada estado-ação conjunto seria uma entrada na tabela Q, teríamos uma tabela de tamanho $|\mathcal{S}| \times |\mathcal{A}|$, o número de estados possíveis multiplicado pelo número de ações. É fácil perceber que mesmo em um cenário pequeno o número de entradas na tabela é enorme. Por exemplo, considere um cenário com 9 semáforos onde cada semáforo observa 5 estados locais possíveis. Neste caso teríamos 1.953.125 ($|\mathcal{S}| = 5^9$) estados conjuntos possíveis e o número de ações seria 19.683 ($\mathcal{A} = 3^9$), se considerarmos que cada semáforo com 3 ações possíveis. A tabela Q teria 38.443.359.375 ($19.683 \times 1.953.125$) entradas.

Com esta breve análise mostramos que solução centralizada que use aprendizado por reforço em qualquer cenário com um número razoável de agentes é intratável sem a utilização de métodos aproximados.

5.1.5 Simulação Microscópica com ITSUMO

Existem duas abordagens básica de simulação de tráfego veicular: macroscópica e microscópica. A abordagem microscópica permite uma descrição detalhada de cada veículo da via. De modo geral, o modelo microscópico descreve o ato de dirigir em uma via, para o modelo de movimentação do veículo, o ITSUMO(SILVA et al., 2006) utiliza o modelo Nagel-Schreckenberg (NAGEL; SCHRECKENBERG, 1992), baseado em Autômatos Celulares (AC). O modelo Nagel-Schreckenberg representa um modelo mínimo que reproduz características básicas do tráfego real. Nesse modelo, cada via é dividida em células de tamanho fixo, que podem conter um veículo. Cada veículo possui uma velocidade que é representada pelo número de células que ele pode percorrer por ciclo de simulação. O comportamento do veículo é representado por algumas regras de aceleração, desaceleração e movimentação:

Regra I Aceleração: Se a velocidade v do veículo é inferior à $v_{máxima}$ e a distância até o próximo veículo é superior à $v + 1$, então ele deve acelerar, aumentando sua velocidade $v \leftarrow v + 1$;

Regra II Desaceleração: Caso a distância até o próximo veículo seja inferior ou igual a sua velocidade, isto é, $v \geq gap$ (*gap* é o termo empregado para designar a distância inter-veicular, isto é, a distância entre o veículo que se está analisando até o veículo mais próximo, neste caso, o imediatamente a frente) então o veículo deve reduzir sua velocidade: $v \leftarrow gap$;

Regra III Aleatoriedade: cada veículo pode, com um probabilidade $p_{desaceleracão}$, reduzir sua velocidade em uma unidade, isto é, $v \leftarrow v - 1$;

Regra IV Movimentação: cada veículo avança um número células igual a velocidade v.

Uma grande vantagem da utilização ITSUMO é que ele é um sistema livre onde a parte de controle semafórico pode ser efetuada por um programa externo ao simulador. Há uma interface entre o controlador do semáforo (agente) e o simulador. Essa interface faz uso de *sockets* para estabelecer o canal de comunicação que permite que o estado da simulação seja passado aos agentes controladores e que por sua vez também possam se comunicar com o simulador, de modo a realizar o controle dos semáforos.

Na simulação, cada agente pode ser executado em um processo ou programa independente, onde é possível definir que um agente controla um semáforo ou um conjunto de semáforos da malha viária.

Abaixo, um resumo do protocolo de comunicação entre os agentes controladores e o simulador:

1. O simulador inicia a execução;

2. Os agentes controladores se conectam ao simulador através do envio de uma mensagem de conexão;

3. Após o ITSUMO receber a mensagem de conexão, ele retorna uma mensagem resposta confirmando a conexão, composta pelos dados dos semáforos para os quais o agente solicitou controle;

4. Com os passos 2 e 3 a conexão é estabelecida;

5. A cada novo estado, o simulador envia uma mensagem a cada um dos agentes informando o estado atual da simulação no seu ponto controlado;

6. A cada intervalo pré-definido o simulador solicita uma ação de controle dos agentes;

7. Quando o agente recebe o pedido de ação, deve ser enviada uma mensagem com as mudanças (ou ações) desejadas.

Tabela 5.1: Fluxos de saturação no ITSUMO

Velocidade		Fluxo de saturação
Células	Km/h	veículos/hora
1	18	1800
2	36	2400
3	54	2700
4	72	2820
5	90	2940
6	108	3000

A Tabela 5.1 mostra os fluxos de saturação referentes às velocidades possíveis no ITSUMO considerando as células de tamanho $5m$. Diferente da fórmula 5.1, apresentada na Subseção 5.1.1, os fluxos de saturação foram calculados através de contagens obtidas com o simulador. Esses fluxos serão utilizados como referência nas simulações realizadas na seção de experimentos. Em todas as simulações foi utilizada a velocidade 3 (54km/h), por se tratarem de cenários urbanos.

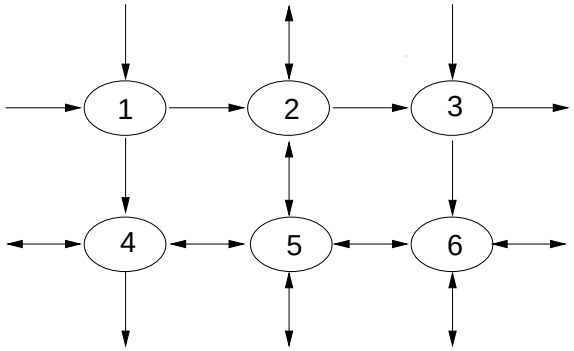

Figura 5.6: Exemplo de vizinhança de um semáforo.

5.2 Controle de Tráfego com o OPPORTUNE

No cenário de controle de tráfego, cada agente OPPORTUNE controla um semáforo e é capaz de se comunicar com semáforos adjacentes, desde que estes também sejam controlados por agentes capazes de compreender as mensagens recebidas (mesma ontologia). O estado percebido pelo agente é criado a partir da informação de tamanho de fila nas vias de entrada. Os planos semafóricos são as ações dos agentes. Foi utilizada uma discretização de três ou cinco estados possíveis (dependendo do número de vias de entrada), baseado nas filas relativas de cada via, como visto anteriormente na Seção 5.1.4. Para cinco estados possíveis temos as 4 direções ou todas as direções tem aproximadamente o mesmo tamanho de fila ($\pm 10\%$ de diferença é considerado igual). No caso de três estados possíveis temos as 2 direções ou todas com aproximadamente o mesmo tamanho de fila ($\pm 10\%$ de diferença). Por exemplo, na Figura 5.6 o agente "1" tem apenas 3 estados possíveis já que ele possui apenas 2 vias de entrada, já o agente do cruzamento "5" tem 5 estados possíveis.

A vizinhança inicial do agente i ($\mathcal{N}_i$) para todos os agentes dos cenários regulares é composta dos agentes que controlam as vias de entrada de cada semáforo. Lembrando que $\mathcal{N}_i$ é o conjunto dos identificadores dos agentes vizinhos do agente i. Por exemplo, na Figura 5.6 temos as seguintes vizinhanças iniciais: $\mathcal{N}_1 = \{\emptyset\}$, $\mathcal{N}_2 = \{1, 5\}$, $\mathcal{N}_3 = \{2\}$, $\mathcal{N}_4 = \{1, 5\}$, $\mathcal{N}_5 = \{2, 4, 6\}$, $\mathcal{N}_6 = \{3, 5\}$. Se um agente i recebe uma mensagem do agente j, e j não pertence a sua vizinhança inicial, ele adiciona j ao conjunto de vizinhança $\mathcal{N}_i$.

Os parâmetros de aprendizado utilizados no OPPORTUNE para todos os experimentos foram $\alpha = 0.5$, $\gamma = 0.9$, $\varepsilon = 0.05$ para a seleção da ação (ε-gulosa) e $E_{min} = 0.05$.

5.3 Experimentos e Resultados

Nesta seção iremos apresentar os experimentos para validação da abordagem em cenários de controle de trafego com o OPPORTUNE em cenários regulares com 25 e 81 agentes e em um cenário real com 8 agentes (região da Av. Assis Brasil). O simulador microscópico ITSUMO foi utilizado para a realização de todos os experimentos.

Todos os experimentos foram executados em uma máquina com processadores Intel Core 2 Quad CPU modelo Q6600@2.40GHz, cache de 4096 KB e 4GB de memória RAM, com sistema operacional Ubuntu Linux, *kernel* 2.6.24-24 e compilador gcc versão 4.2.4.

Em todos os métodos de aprendizado a recompensa nos cenários de tráfego é relativa ao tamanho da fila média (por ciclo) do cruzamento. Foram avaliados 4 funções de re-

compensa possíveis nos cenários de tráfego, todas com a fila média como parâmetro. A fila média é calculada de acordo com o tamanho da memória do agentes. A cada segundo o agente guarda em sua memória o tamanho de fila observado em suas vias de entrada. Em todos os experimentos o equivalente a 1 ciclo de semáforo.

Tipo 1 - inversamente proporcional a recompensa é inversamente proporcional à fila média em relação a fila máxima, por exemplo se a fila máxima é 10 e há uma fila de 8 veículos, a recompensa é 0.2.

Tipo 2 - relativa a recompensa é +1 se diminuiu em 10% a fila em relação à fila média anterior, -1 se aumentou, -0.01 se se manteve estável e +10 para ausência de fila.

Tipo 3 - exponencial a recompensa decresce rapidamente (com valores entre 0 e 1) em relação a fila, sendo 1 o caso de fila 0

Tipo 4 - proporcional de 3 valores se a fila for maior que um limite máximo pré estabelecido a recompensa é -1, se for menor que 10% do máximo relativo é +1 e caso fique em um valor intermediário é -0.1.

Todas as funções foram testadas com o *Q-Learning* padrão e verificou-se que a função de "Proporcional de 3 valores" teve melhores resultados em relação ao tempo de convergência, sendo assim esta função foi aplicada em todos os algoritmos, inclusive nos centralizados. A fila máxima foi definida como 10 veículos por ser o número máximo de veículos que podem cruzar um semáforo em 15 segundos (tamanho mínimo de um tempo de verde para um estágio).

5.3.1 Cenários Regulares

Foram utilizados como cenário algumas malhas viárias com configuração de grade. Esse formato de rede permite uma fácil manipulação das variáveis que determinam os fluxos e fornece uma simetria na dinâmica de tráfego em todos os nodos. Ou seja, qualquer nodo possui o mesmo número de vias de entrada e vias de saída. Nesse formato de grade se pode analisar as possibilidades de controle em um determinado ponto sabendo e controlando diretamente os fatores que determinam o tráfego naquele ponto. Sendo assim o foco passa a ser o controle em si, e não a rede de relacionamentos que pode levar a uma determinada configuração de tráfego. Os injetores e sumidouros estão dispostos simetricamente e a dinâmica da rede é configurada alterando apenas as taxas de inserção de veículos nos injetores.

Foram utilizadas duas configurações, de acordo com o número de cruzamentos semaforizados e controlados: 25 e 81 semáforos. Entre os nodos do tipo injetor ou sumidouro e os nodos principais de entrada foi configurada uma distância de 300 metros, para que a vazão de entrada da rede (no caso de um injetor) possa ser absorvida sem que os veículos fiquem retidos antes do primeiro cruzamento. Entre os demais nodos (cruzamentos) foi configurada a mesma distância de 300 metros, constituindo uma distância razoável para que haja possibilidade de movimentação, sem que se force congestionamento por motivo de distâncias demasiadamente pequenas. De acordo com o modelo de autômato celular usamos células de 5 metros, ou seja, cada via é formada por 60 células. Com essa configuração podem haver no máximo 60 veículos parados por via. Além disso, a velocidade máxima é de 3 células por iteração, o que corresponde a 15 m/s ou aproximadamente 54 Km/h, valor adequado para simulação de tráfego urbano.

Cada via possui apenas um sentido de tráfego, que pode ter duas direções: horizontal e vertical. Cada cruzamento possui um semáforo controlando com dois planos semafóricos predeterminados, que são iguais em tempo de ciclo e tempo de fase, porém possuem defasagens diferentes. As defasagens foram calculadas de modo manual e o "Plano 1" possui defasagem relativa ao cruzamento ao Norte ou ao Sul e o "Plano 2" possui defasagem relativa ao cruzamento a Leste ou a Oeste.

Foi definido um tempo de ciclo de 60 segundos, e as fases foram divididas em tempos 30 segundos. Sendo assim, a diferença entre os planos está na defasagem (*offset*) de aplicação das fases e, consequentemente, na direção da onda verde.

Com a definição de uma velocidade máxima 3 sendo células/segundo temos um tempo de 22 segundos para que um motorista percorra uma via de 60 células (tamanho de todas as vias neste cenários regulares), partindo da velocidade zero , aqui se considera o tempo de aceleração até a velocidade máxima, e um tempo de 20 segundos partindo da velocidade máxima e mantendo-se essa. Além disso esses tempos consideram fluxo livre para o motorista em questão.

Com base nesses tempos a sincronização proposta prevê em cada via a sequência de fases prevendo as defasagens (*offsets*) supracitadas.

Por exemplo, considerando a via "B" (mostrada na Figura 5.7), com direção horizontal e sentido Oeste para Leste temos o seguinte plano de sincronização horizontal:

Semáforo	1B	2B	3B	4B	5B
Fase 1	NS - 0 a 21 e NS - 52 a 59	LO - 0 a 11 e LO - 42 a 59	NS - 0 a 1 e NS - 32 a 59	NS - 0 a 21 e NS - 52 a 59	LO - 0 a 11 e LO - 42 a 59
Fase 2	LO - 22 a 51	NS - 12 a 41	LO - 2 a 31	LO - 22 a 51	NS - 12 a 41

A cada 120 segundos (dois ciclos), o simulador requisita os novos planos para os agentes controladores. Os experimentos aqui mostrados tem um total de 100600 passos, sendo que os primeiros 600 passos servem para iniciar a rede e não há atuação dos agentes. Os agentes podem realizar uma ação a cada 120 passos de simulação. Foram realizadas 20 repetições de cada experimento e todos os gráficos mostram os valores médios das 20 repetições.

A Tabela 5.2 mostra um resumo sobre os experimentos realizados nas redes regulares. No "Experimento I", foram inseridos 468 veículos/hora nos injetores localizados no início das vias horizontais (A a E) e 26 veículos/hora nos injetores das vias verticais (1 a 5). Nos Experimentos I" e "II" foram aplicados três métodos de aprendizado.

No "Experimento II", foram utilizadas as mesmas taxas de inserção do Experimento I taxas até o meio da simulação (passo 50300) e neste ponto as taxas de inserção foram invertidas, ou seja: 468 veículos/hora nos injetores localizados no início das vias verticais (1 a 5) e 26 veículos/hora nos injetores das vias horizontais (A a E).

O desempenho médio foi avaliado para toda a rede em relação ao tamanho da fila média em cada cruzamento. Nos Experimentos "I" e "II" foram adicionados agentes com comportamento diferente nos cruzamentos diretamente ligados aos injetores e sumidouros da rede (círculos brancos na Figura 5.7). Nestes cruzamentos os agentes possuem um comportamento de otimização local. O comportamento é chamado de *guloso*, neste comportamento os agentes mudam os planos semafóricos de modo a aumentar o tempo de verde dos estágios referentes às vias com maior fila. A utilização dos agentes *gulosos* nesta rede tem dois objetivos:

- verificar o comportamento dos agentes controlados em uma rede heterogênea, ou seja, uma rede onde somente parte é controlada por agentes de um mesmo tipo;

- evitar que o comportamento dos agentes fosse prejudicado devido a sua localização periférica na rede, já que nestes pontos as filas tendem a ser maiores pois os veículos estão vindo diretamente dos insersores.

Tabela 5.2: Experimentos com redes regulares

Experimento	Taxa de Inserção	Métodos	Semáforos Avaliados	Volume (v/h) Hor.	Vert.
I	fixa	OPPORTUNE,	9	468	26
II	variável	Q(λ) e JAL		468/26	26/468
III	fixa	OPPORTUNE e Q(λ)	81	1008	504

Nos experimentos I e II, por se tratarem de um cenário pequeno (com 9 semáforos controlados) foi possível executar o *Q-Learning* centralizado para comparação. O *Q-Learning* centralizado modela todos os estados e ações conjuntas, sendo que os estados e ações são sempre conhecidos. O espaço em memória para este tipo de aprendizado é grande, como visto na Subseção 5.1.4, por isso podemos aplicar somente neste cenário pequeno.

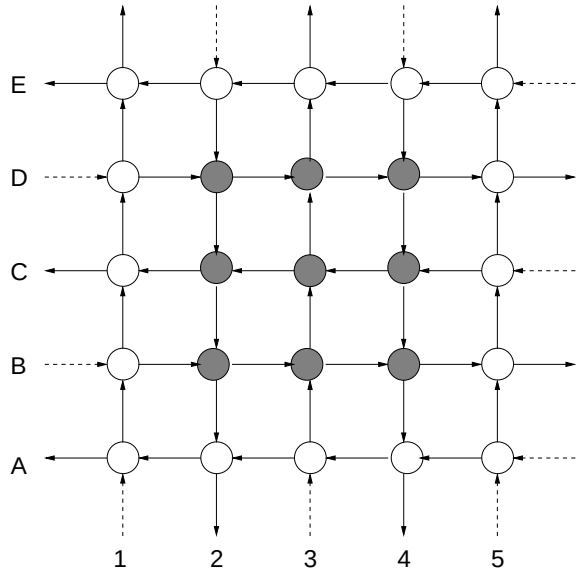

Figura 5.7: Cenário dos Experimentos I e II

A Figura 5.8 mostra os resultados comparativos dos três métodos utilizados em relação ao número de veículos parados (em fila) por cruzamento. O OPPORTUNE apresenta um resultado melhor do que os métodos de comparação ($Q(\lambda)$ e *Q-Learning* centralizado), ambos centralizados e com representação mais completa. Isto ocorre possivelmente porque o espaço de busca dos outros métodos é muito maior e o número de passos foi insuficiente para atingir um resultado melhor.

Para verificar se os métodos $Q(\lambda)$ e *Q-Learning* centralizado atingiriam resultados melhores com mais passos, todos foram simulados com 250mil passos a mais (350mil no total) e não apresentaram alteração nos resultados. O gráfico da Figura 5.8 foi cortado a partir do passo 30000 por não apresentar mudanças após este período.

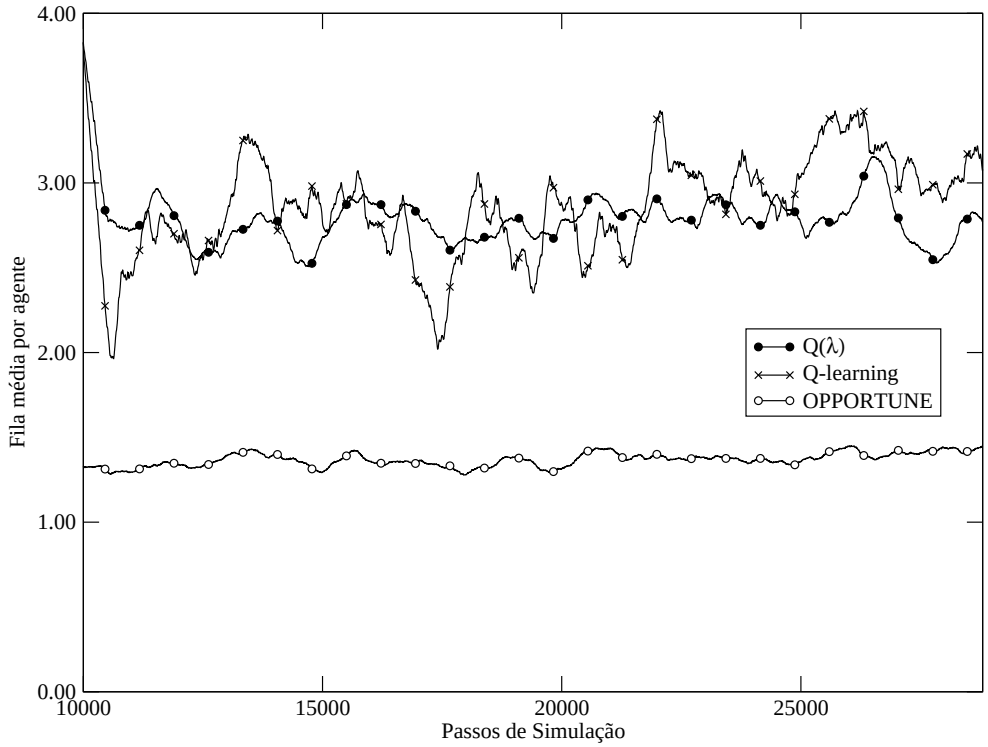

Figura 5.8: Comparação do Experimento I

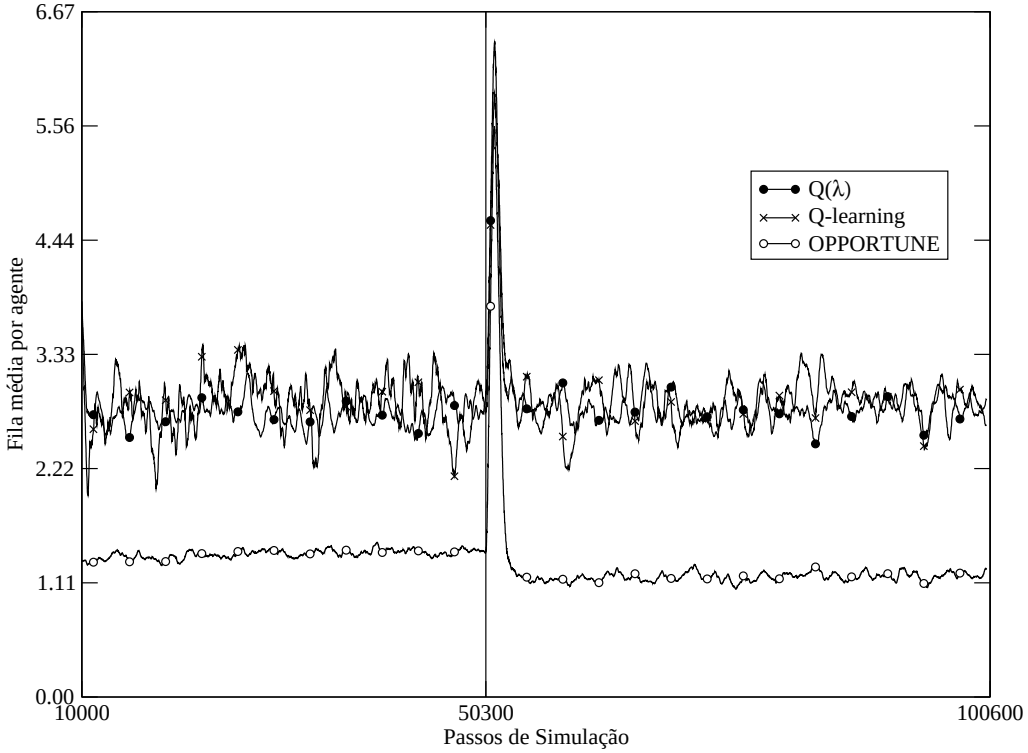

Figura 5.9: Comparação do Experimento II

No segundo experimento (II), foi testado como os métodos de aprendizagem podem se adaptar a uma mudança na taxa de inserção de veículos na rede. Este cenário foi iniciado com uma taxa de inserção fixa que foi alterada no meio do tempo de simulação de modo

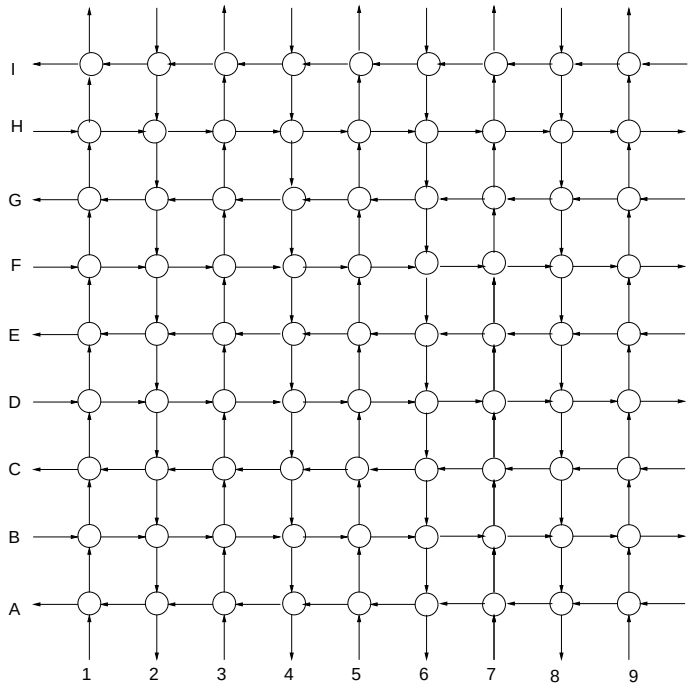

Figura 5.10: Mapa com 81 semáforos controlados.

Tabela 5.3: Comparação quanto ao uso de memória.

Método	Memória Residente (MB)		Memória Virtual (MB)	
	média	máxima	média	máxima
OPPORTUNE	7,91 $\pm$ 0,3	11,59 $\pm$ 0,24	19,46 $\pm$ 0,32	23,08$\pm$ 0,47
$Q(\lambda)$	52,03 $\pm$ 0,75	76,73 $\pm$ 1,37	955,03 $\pm$ 0,96	958.05$\pm$ 1,64

a inverter o volume de tráfego na rede.

A Figura 5.9 mostra que OPPORTUNE também alcança resultados melhores do que os outros métodos, sendo que o tempo de adaptação foi similar ao tempo dos demais métodos. Neste cenário Q(λ) e o *Q-Learning* centralizado também apresentam um comportamento muito semelhante. Todos os métodos apresentados se adaptaram com sucesso à nova situação do fluxo de tráfego, embora os métodos *Q-Learning* e $Q(\lambda)$ apresentem um comportamento pior que o OPPORTUNE.

O "Experimento III" foi conduzido em uma rede maior, com 81 semáforos, sendo todos controlados pelos métodos de aprendizagem avaliados. O mapa deste cenário também é em forma de grade, como mostra a Figura 5.10. A taxa de inserção neste cenário permaneceu fixa ao longo da simulação sendo 1008 veículos/hora nas vias horizontais (A a I) e 504 veículos/hora nos injetores das vias restantes. Isto equivale 1512 veículos/hora em cada cruzamento considerando um fluxo livre (sem controle e sem colisões).

A Tabela 5.3 mostra uma comparação do uso de memória total dos métodos no Experimento III. Nota-se que o OPPORTUNE apresenta um uso muito menor de memória tanto residente quanto virtual. Sendo aproximadamente *42 vezes menor* o uso de memória virtual comparado com o $Q(\lambda)$ e 6 vezes menor em uso de memória residente. O OPPORTUNE foi implementado em C++ e o $Q(\lambda)$ em Java, que pode ter também um impacto quanto ao uso de memória.

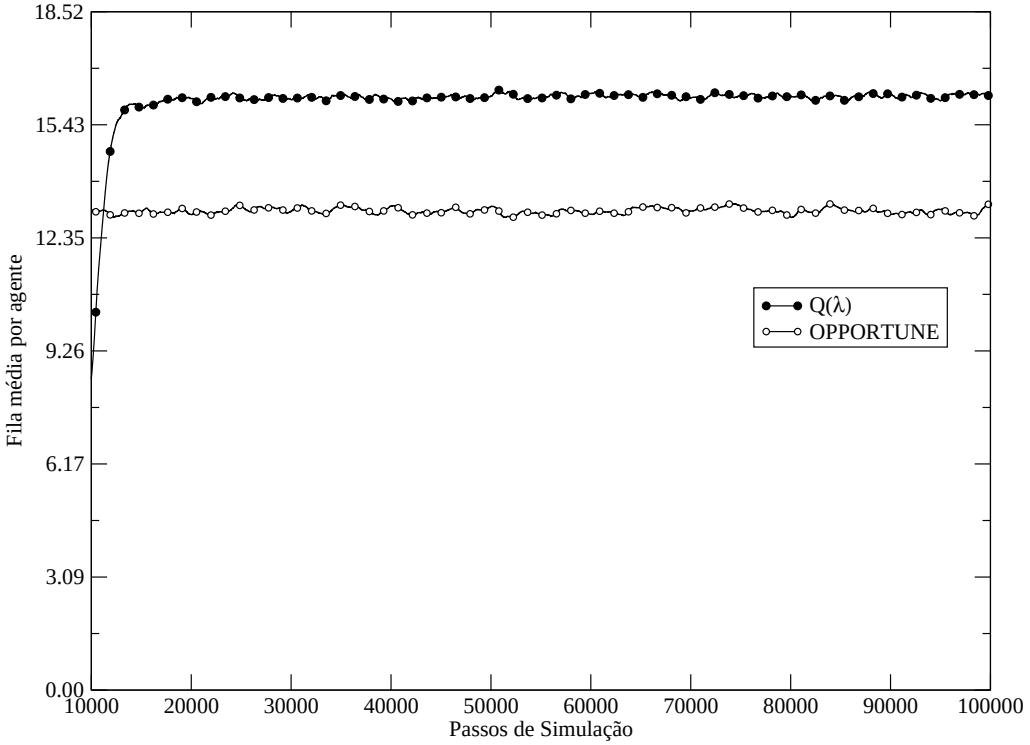

Figura 5.11: Comparação no Cenário com 81 semáforos.

O tempo de processamento também foi analisado. Como o OPPORTUNE usa comunicação entre os agentes ele usou um tempo maior de processamento total, em média 915,9 ($\pm$3,45) segundos e o $Q(\lambda)$ usou em média 413,9 ($\pm$1,44) segundos. O tempo do OPPORTUNE fica muito prejudicado comparando-se com um método centralizado porque o custo do mecanismo de comunicação interna é muito alto. Mesmo com o custo de comunicação computado o OPPORTUNE demora um pouco menos que o dobro do tempo, o que é um indicativo bom de desempenho.

Este último experimento em redes regulares (Cenário III) também foi utilizado para analisar mais detalhadamente o comportamento dos agentes OPPORTUNE em um cenário com vários agentes. Por haver uma grande variação entre o número de mensagens, estados utilizados e tamanho final da tabela Q de cada agente, estes valores serão apresentados de modo visual, nos gráficos das Figuras 5.12 e 5.13. Os resultados foram ordenados para melhor visualização, sempre do menor ao maior valor de média do parâmetro avaliado.

A Figura 5.12 mostra o tamanho final (número de entradas) da tabela Q dos agentes OPPORTUNE. É importante ressaltar que o tamanho máximo da tabela dos agentes com 2 vizinhos é 216 e não há nenhum agente que tenha aumentado a sua tabela a pouco mais de 35 entradas (mais precisamente, com média máxima 34,6 e desvio padrão 2,81). O agente com a maior tabela média está localizado no cruzamento "G5" na Figura 5.10.

O gráfico apresentado na Figura 5.13 foi construído de modo similar ao gráfico anterior (Figura 5.12), ordenando-se os agentes em relação número de mensagens enviadas ao longo da simulação Neste, o gráfico interno mostra um detalhe dos agentes que enviaram menos mensagens (para outros agentes) ao longo da simulação, menos de 100 mensagens no total. Neste caso apenas 21^{o}agentes enviaram em média menos de cem (100) mensagens por simulação. Isto mostra que o número de mensagens não está diretamente relacionando com o número de estados conjuntos utilizados, isto porque o agente pode ter

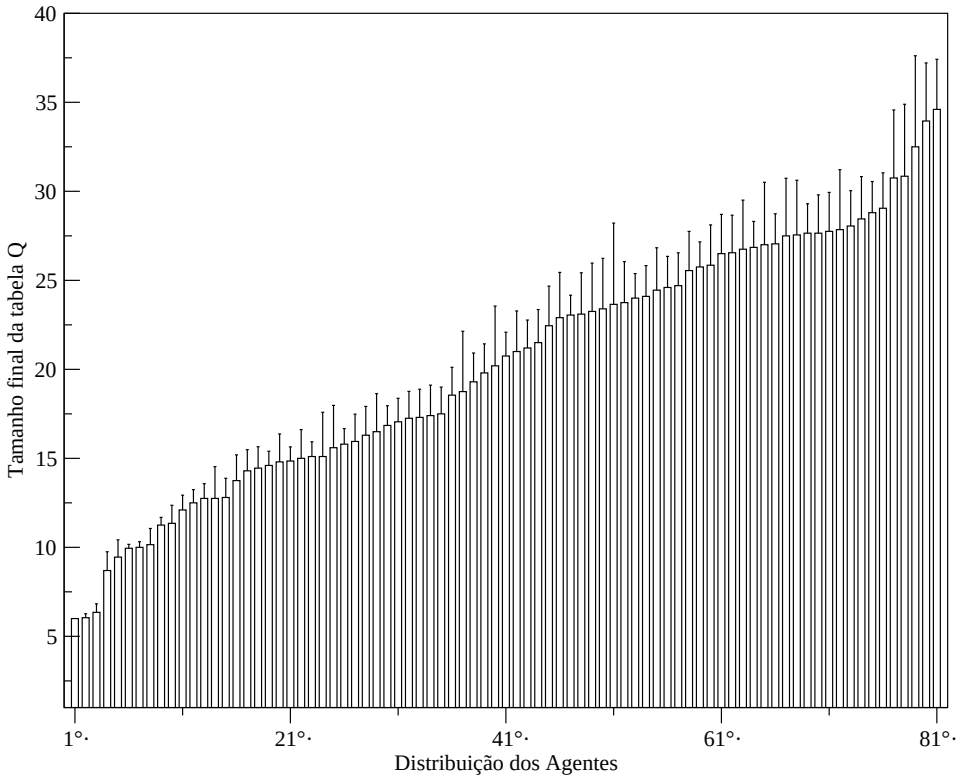

Figura 5.12: Número de entradas da tabela Q dos agentes ao final do Experimento III.

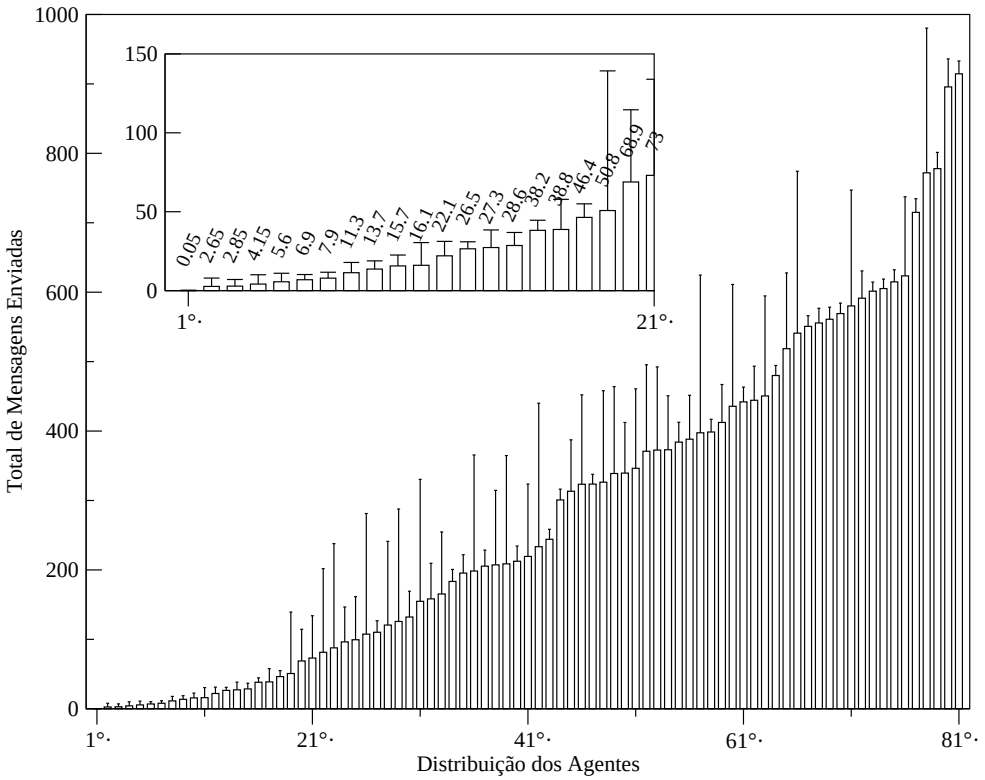

Figura 5.13: Número total de mensagens enviadas por agente.

enviado informações para outros agentes mesmo tendo executado ações individuais e não utilizado informação de outros agentes para construir estados conjuntos.

Foi analisado que número total de vezes (considerando-se toda a simulação) que cada agente utilizou ou tentou utilizar um estado composto, ou seja, um estado contendo mais do que sua própria percepção local é relativo ao número de mensagens. Os agentes que utilizam mais estados conjuntos são os que sofreram mais influências dos demais agentes da rede.

5.3.2 Cenário real: região da Av. Assis Brasil

A região da Av. Assis Brasil (mapa mostrado na Figura 5.14), em Porto Alegre, foi escolhida por ser uma região de tráfego intenso. O tráfego se mantem alto durante grande parte do dia, visto que é uma região de grande movimento comercial e de acesso à cidades vizinhas. Neste cenário serão utilizadas as informações de fluxo de tráfego e temporização de semáforos fornecidas pela Empresa Pública de Transporte e Circulação (EPTC). Cada cruzamento possui 4 planos semafóricos (4 ações), sendo que o tempo de ciclo é 40 segundos é há planos com ciclo duplo (80 segundos).

Tabela 5.4: Volume (veículos/hora).

Horário	Assis Brasil	Obirici	Bogotá	Av. do Forte	Bernardi	Domingos Rubbo	Antônio J. Mesquita	Carneiro da Fontoura
06:00 - 07:00	972	972	972	972	972	972	972	972
07:00 - 09:00	2160	1080	720	900	720	1332	684	828
09:00 - 11:30	1764	1224	576	792	756	1404	576	828
11:30 - 13:30	1620	1260	756	756	864	1404	756	936
13:30 - 17:00	3132	2268	720	1512	1512	3096	720	1800
17:00 - 20:00	1620	1548	1728	900	1008	2988	1728	1188

Os agentes controladores estão representados por círculos (Figura 5.14), onde o número do círculo mostra o identificador de cada agente. as vias com nome em destaque são as vias de fluxo variável mostradas na Tabela 5.4. Cada agente possui 5 estados possíveis, de acordo com os planos mostrados na Tabela 5.5

A cada 160 segundos (máximo de dois ciclos), o simulador requisita os novos planos para os agentes controladores. Os experimentos aqui mostrados tem um total de 100600 passos, sendo que os primeiros 600 passos servem para iniciar a rede e não há atuação dos agentes. Os agentes podem realizar uma ação a cada 160 passos de simulação. Foram realizadas 20 repetições de cada experimento e todos os gráficos mostram os valores médios das 20 repetições.

Os fluxos de entrada da rede variam ao longo do tempo de simulação. Foram simulados o número de passos equivalente a 2 dias de 16 horas, sendo que foram descontadas as faixas de horário que não possuímos dados coletados e adicionamos somente uma faixa inicial (6 as 7h da manhã). A simulação os fluxos de entrada variáveis segundo as pro-

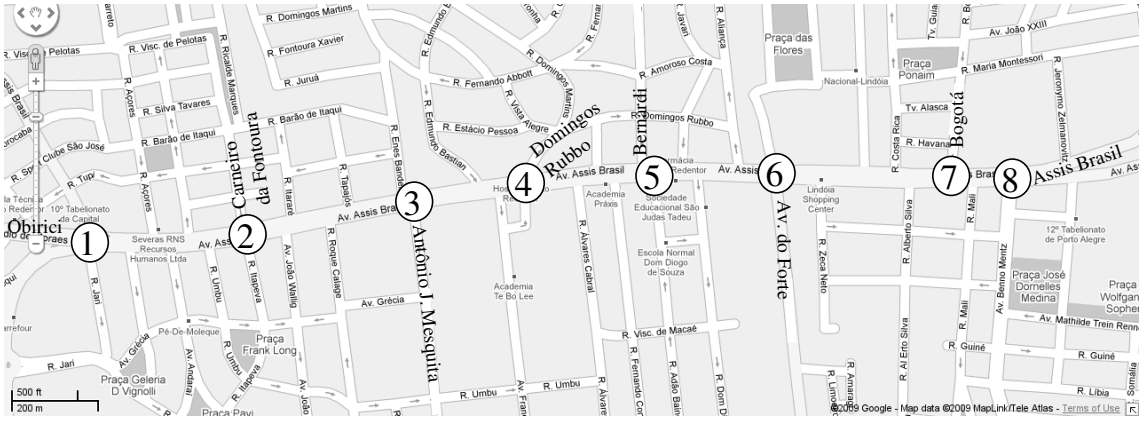

Figura 5.14: Mapa da Região da Av. Assis Brasil com a localização dos semáforos controlados (Google Maps, `http://maps.google.com`)

Tabela 5.5: Planos semafóricos do cenário Assis Brasil

	Semáforo							
	1	2	3	4	5	6	7	8
Plano 1								
1	60	60	60	60	51	60	60	60
2	20	20	20	20	29	20	20	20
Plano 2								
1	25	25	25	25	25	25	25	20
2	15	15	15	15	15	15	15	15
Plano 3								
1	60	08	09	07	07	09	10	03
2	20	60	60	60	51	60	60	60
3	-	12	11	13	22	12	10	17
Plano 4								
1	10	10	07	07	08	10	04	60
2	60	60	60	60	51	60	60	20
3	10	10	13	13	21	10	16	-

babilidades e taxas de inserção da Tabela 5.4, sendo que a Av. Beno Mentz (ligada ao cruzamento 8) possui taxa de inserção fixa no valor de 360 veículos/hora e as demais entradas atuam com taxas de inserção fixa de inserção 180 veículos/hora.

A vizinhança inicial de cada agente i é composta pelos agentes controladores mais próximos não necessariamente diretamente ligados. Neste cenário temos as seguintes vizinhanças iniciais: $\mathcal{N}_1 = \{2\}$, $\mathcal{N}_2 = \{1, 3\}$, $\mathcal{N}_3 = \{2, 4\}$, $\mathcal{N}_4 = \{3, 5\}$, $\mathcal{N}_5 = \{4, 6\}$, $\mathcal{N}_6 = \{5, 7\}$, $\mathcal{N}_7 = \{6, 8\}$ e $\mathcal{N}_8 = \{7\}$. Assim como nos cenários regulares, se um agente i recebe uma mensagem do agente j, e j não pertence a sua vizinhança inicial, ele adiciona j à vizinhança $\mathcal{N}_i$.

O cenário é altamente dinâmico e os agentes podem atuar somente a cada 160 segundos (esquivante ao tempo de 2 ou 4 ciclos, dependendo do plano). Há intervalos (de 1 h) com apenas 22 atuações, sendo assim, o agente possui um número limitado de passos que podem ser utilizados na adaptação. Não foi possível implementar o *Q-Learning* centralizado neste cenário já que há 390625 estados conjuntos e 65536 ações conjuntas, a

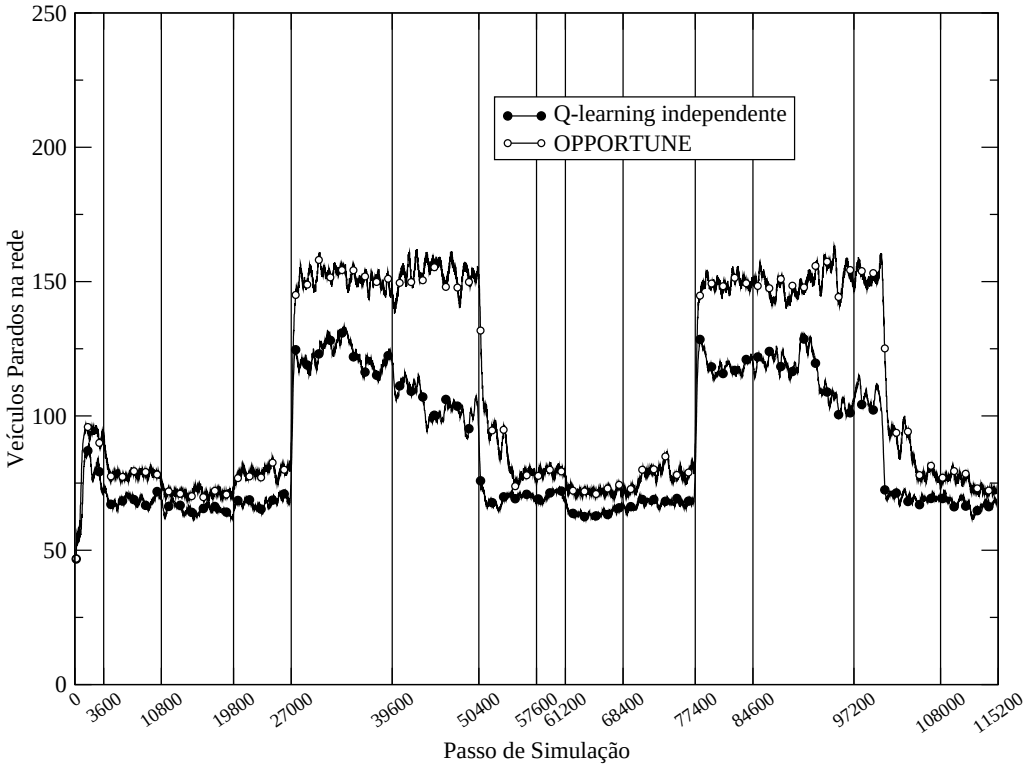

Figura 5.15: Comparação de Desempenho no Cenário Assis Brasil.

tabela Q teria de ser muito grande (65536×390625), resultando em falha de alocação de memória.

A Figura 5.15 compara o método em cenário muito variável. Ambos os métodos apresentaram comportamento bastante similar, sendo que o OPPORTUNE ainda apresenta uma pequena desvantagem. Uma solução para este tipo de cenário seria realizar um período de treinamento com diversos fluxos possíveis com intervalos grandes de aprendizado. Assim os agentes estariam aptos a executar em tempo real um método previamente aprendido.

Foram realizados testes com diversos valores de E_{min} no cenário e o máximo que se atingiu foi um comportamento idêntico ao *Q-Learning*,sem expansão das tabelas Q. A expansão aumenta o espaço de busca da solução e com isso aumenta também o tempo para encontrar um solução ótima.

Foi realizado um experimento sem mudanças nos fluxos, simulando somente um momento de pico (13:30 - 17:00), com vários ciclos para verificar se o OPPORTUNE mostraria um bom funcionamento neste cenário de fluxo e que os problemas anteriormente descritos eram realmente apenas em função das mudanças em intervalos curtos de tempo.

A Figura 5.16 mostra que o OPPORTUNE teve um desempenho melhor que a abordagem *Q-Learning* tradicional quando foi simulado somente o período de fluxo intenso do cenário real. Esta simulação também comprova de que o algoritmo teria como se adaptar a cada tipo de fluxo de modo adequado caso tivesse o número suficientemente grande de passos.

Também foi realizado este experimento de cenário fixo no simulador TRANSYT. O único resultado final dos apresentados pelo TRANSYT que conseguimos encontrar um equivalente no ITSUMO é a velocidade média. Porém no TRANSYT a velocidade média encontrada após a otimização foi é 8,6Km/h e em ambos os métodos de aprendizado o

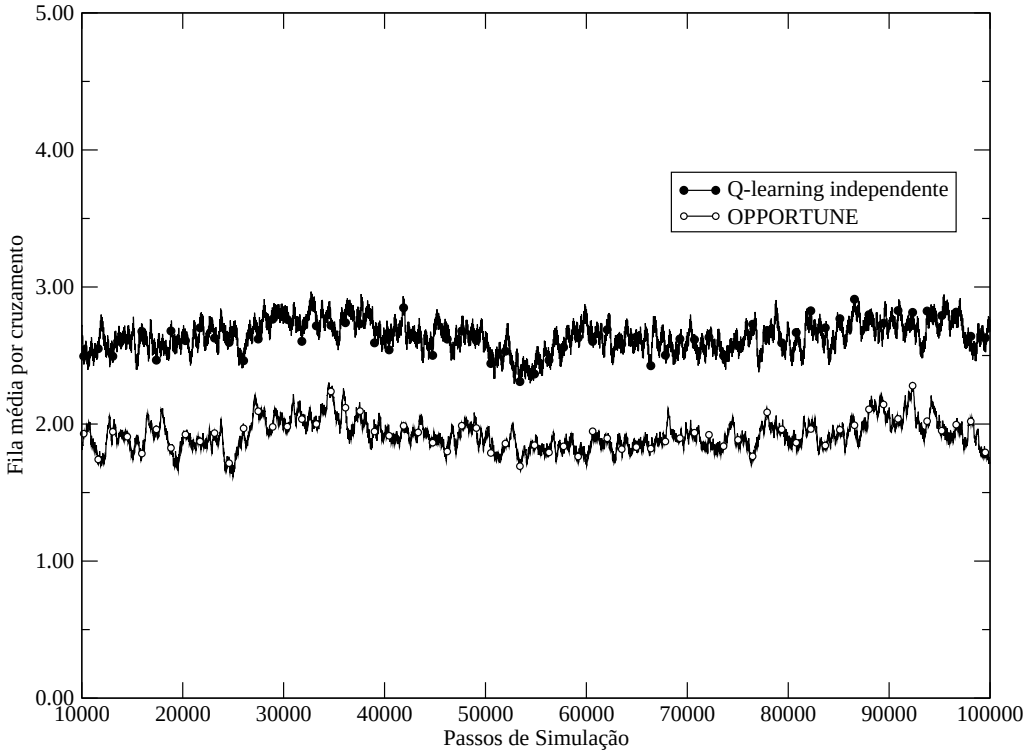

Figura 5.16: Comparação no experimento com fluxo fixo

valor de velocidade média na parte controlada da rede ficou em torno de 30 Km/h (1,6 células por segundo).

Não foi possível realizar uma comparação direta e justa entre os resultados por diversos motivos, destacamos os principais motivos o modelo de movimentação e a restrição na modelagem da rede. Não teríamos como comparar diretamente o resultado obtido no ITSUMO com o TRANSYT por serem modelos diferentes de movimentação. A velocidade no TRANSYT é contínua e no ITSUMO os veículos apresentam velocidades discretas. Além disso a rede foi simplificada de modo a ser possível sua simulação na versão do software a qual tivemos acesso, já que este é restrito a simulação de 49 nodos (controlados ou não). A rede, os resultados e parâmetros obtidos no TRANSYT encontram-se no Anexo A.

5.4 Conclusão

Este Capítulo apresentou o problema do controle de tráfego veicular urbano, desde seus conceitos básicos, análise como um MDP, abordagens de controle padrão e trabalhos relacionados. Foi apresentado o OPPORTUNE aplicado a este cenário de modo a verificar o seu funcionamento em um cenário com maior número de agentes e mais restrições nas ações cooperativas. Os resultados mostram que o OPPORTUNE em algumas situações apresenta desempenho superior à métodos tradicionais de aprendizado por reforço.

Uma grande vantagem de utilizar o OPPORTUNE neste tipo de ambiente é que não é necessário o conhecimento prévio dos requisitos de coordenação da rede, já que os agentes conseguem se adaptar aos cenário através de suas iterações.

A desvantagem apresentada nos últimos experimentos é uma desvantagem em geral de métodos de aprendizado não totalmente independentes e não invalida as demais con-

clusões apenas reforça que redes altamente dinâmicas são um desafio para mecanismo de aprendizado por reforço não supervisionado.

6 CONCLUSÃO

Existem abordagens que levam à solução ótima em aprendizado multiagente, embora geralmente não possam ser aplicadas em problemas com muitos agentes, devido principalmente ao tamanho do espaço de ações estados conjuntos. As áreas apresentadas nos Capítulos 4 e 5 possuem representações de estados e ações conjuntos muito grandes, cuja resolução centralizada se torna difícil mesmo para um número de agentes reduzido. Existe uma clara necessidade de uma abordagem capaz de lidar com cenários complexos e dinâmicos, onde os agentes interagem e aprendem com mais eficiência.

Na próxima seção são revistas e detalhadas as contribuições deste trabalho e, na Seção 6.2, são apresentadas limitações do método as questões em aberto que podem ser cobertas em trabalhos futuros.

6.1 Contribuições

Este trabalho traz contribuições para o avanço do estado-da-arte no estudo e desenvolvimento de SMA, quanto na modelagem e aplicação de técnicas de aprendizagem multiagente em problemas computacionais. As contribuições são descritas a seguir:

- A abordagem proposta no Capítulo 3 é nova e eficaz. Pode ser aplicada em SMAs simples ou complexos. Assume-se, neste trabalho, que a tomada de decisão deve ser realizada de maneira distribuída e o ambiente possui as seguintes características: dinâmico, não-deterministico, discreto, parcialmente observável, cooperativo ou parcialmente cooperativo. Diversos cenários reais, principalmente em robótica e automação, em que não se pode centralizar totalmente o processo de controle por diversos motivos: restrição de comunicação, privacidade das informações individuais, nível de tolerância a falhas, etc. Todos os mecanismos desta abordagem foram experimentados e validados, como mostram os Capítulos 4 e 5, de modo distribuído. Inclusive utilizando-se canais de comunicação reais (via *sockets*) e não acesso direto a objetos. Os resultados empíricos destes experimentos mostraram que o OPPORTUNE, pode ser utilizado com sucesso para a realização de tarefas distribuídas, fazendo com que os agentes tomem suas decisões de forma racional e com baixo custo de comunicação.

- Quanto à característica de ambientes de grande porte, a abordagem se mostrou suficientemente seletiva de modo a utilizar de modo econômico o canal de comunicação. Além disto foi visto que as tabelas Q utilizadas não cresceram de modo exponencial, mantendo um crescimento com incrementos pequenos até o ponto de estabilidade. O interessante do método é que chegando neste ponto de estabilidade, caso haja alguma variação no sistema, o agente é capaz de atualizar sua tabela Q,

desde que haja informação disponível ou ações possíveis a serem adicionadas a sua base de conhecimento.

- Foram propostas extensões e modificações nos modelos teóricos de aprendizado por reforço para resolver problemas de forma distribuída, que foram avaliadas e validadas. Buscou-se manter as características originais destes modelos, de modo a adaptar minimamente o método proposto aos cenários, provando que o método é flexível o suficiente e não exige que o projetista tenha muito conhecimento prévio dos ambientes a serem simulados, como partes do ambiente onde as ações devem ser coordenadas.

6.2 Limitações e Trabalhos Futuros

Este trabalho apresenta limitações e questões em aberto que apontam direções para trabalhos futuros, as quais podemos destacar:

- Primeiramente os cenários aqui apresentados embora já sejam considerados complexos do ponto de vista do agente, ainda podem ser estendidos ou modificados. No cenário de controle de tráfego, poderíamos, por exemplo, fazer com que as ações fossem mudanças nos tempos de fase, e não somente escolha do plano. No cenário do jogo de captura cooperativo, poderíamos variar o cenário de acordo com formato do mapa, criação de obstáculos, tipos de agentes (variando-se movimentação, tamanho, velocidade, capacidades perceptivas e memória), tipo de cooperação. Sendo a questão da variação número de predadores necessários para capturar a presa, diretamente ligada à questão da cooperação entre os agentes, por exemplo: se são necessários dois predadores para capturar uma presa e há três predadores no cenário quais predadores vão conseguir participar da captura?

- Recapitulando as linhas de pesquisa mostradas na Introdução, seria interessante a pesquisa se estender para times heterogêneos. Os times de agentes neste trabalho sempre são homogêneos em relação as suas capacidades básicas. Seria interessante verificar a cooperação entre agentes com capacidade diferentes, por exemplo em um cenário de busca e resgate onde é preciso que agentes com diferentes capacidades cooperem com o objetivo comum de resgatar civis.

- O número de agentes também pode ser expandido ainda mais. O número máximo de agentes simulados nesta tese foi de 81 agentes, muito em parte das limitações da criação de um cenário de tráfego de grande porte. Talvez com a escolha de cenários maiores ainda (na ordem de centenas de agentes) seja possível obter ainda mais vantagens da representação enxuta de ações e estados, característica do OPPORTUNE.

REFERÊNCIAS

ABDALLAH, S.; LESSER, V. Learning the Task Allocation Game. In: FIFTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS (AAMAS06), 2006, Hakodate, Japan. **Proceedings...** New York: ACM Press, 2006. p.850–857.

BAZZAN, A. L. C. A Distributed Approach for Coordination of Traffic Signal Agents. **Autonomous Agents and Multiagent Systems**, [S.l.], v.10, n.1, p.131–164, March 2005.

BAZZAN, A. L. C. Opportunities for Multiagent Systems and Multiagent Reinforcement Learning in Traffic Control. **Autonomous Agents and Multiagent Systems**, [S.l.], v.18, n.3, p.342–375, June 2009.

BAZZAN, A. L. C.; KLÜGL, F. Sistemas Inteligentes de Transporte e Tráfego: uma abordagem de tecnologia da informação. In: KOWALTOWSKI, T.; BREITMAN, K. K. (Ed.). **Anais das Jornadas de Atualização em Informática**. [S.l.]: SBC, 2007.

BOWLING, M.; VELOSO, M. Multiagent learning using a variable learning rate. **Artificial Intelligence**, Essex, UK, v.136, n.2, p.215–250, 2002.

CLAUS, C.; BOUTILIER, C. The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems. In: FIFTEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, 1998. **Proceedings...** [S.l.: s.n.], 1998. p.746–752.

DENATRAN. **Serviços de Engenharia**: manual de semáforos. Brasília: Concelho Nacional de Trânsito - Ministério da Justiça, 1979. 169p.

DENZINGER, J.; FUCHS, M. Experiments in Learning Prototypical Situations for Variants of the Pursuit Game. In: IN PROCEEDINGS OF THE SECOND INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS), 1996. **Anais...** Menlo Park: AAAI Press, 1996. p.48–55.

DIAKAKI, C.; DINOPOULOU, V.; ABOUDOLAS, K.; PAPAGEORGIOU, M.; BEN-SHABAT, E.; SEIDER, E.; LEIBOV, A. Extensions and New Applications of the Traffic Signal Control Strategy TUC. In: ANNUAL MEETING OF THE TRANSPORTATION RESEARCH BOARD, 82., 2003. **Proceedings...** [S.l.: s.n.], 2003. p.12–16.

DIAKAKI, C.; PAPAGEORGIOU, M.; ABOUDOLAS, K. A Multivariable Regulator Approach to Traffic-Responsive Network-Wide Signal Control. **Control Engineering Practice**, [S.l.], v.10, n.2, p.183–195, February 2002.

DOYA, K.; SAMEJIMA, K.; KATAGIRI, K.; KAWATO, M. Multiple Model-Based Reinforcement Learning. **Neural Computation**, Cambridge, MA, USA, v.14, n.6, p.1347–1369, 2002.

GORDON, G. J. Agendas for multi-agent learning. **Artificial Intelligence**, Essex, UK, v.171, n.7, p.392–401, May 2007.

GUESTRIN, C.; LAGOUDAKIS, M. G.; PARR, R. Coordinated Reinforcement Learning. In: NINETEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML), 2002, San Francisco, CA, USA. **Proceedings...** Morgan Kaufmann, 2002. p.227–234.

HOEN, P. J. 't; TUYLS, K.; PANAIT, L.; LUKE, S.; POUTRE, J. L. An Overview of Cooperative and Competitive Multiagent Learning. In: TUYLS, K.; HOEN, P. 't; VERBEECK, K.; SEN, S. (Ed.). **Learning and Adaptation in Multi-Agent Systems**. Berlin: Springer, 2006. p.1–49. (Lecture Notes in Artificial Intelligence, v.3898).

KAELBLING, L. P.; LITTMAN, M.; MOORE, A. Reinforcement Learning: a survey. **Journal of Artificial Intelligence Research**, [S.l.], v.4, p.237–285, 1996.

KOK, J. R. **Coordination and Learning in Cooperative Multiagent Systems**. 2006. Tese (Doutorado em Ciência da Computação) — Faculty of Science, University of Amsterdam.

KOK, J. R.; VLASSIS, N. Sparse cooperative Q-learning. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML), 21., 2004, New York, USA. **Proceedings...** ACM Press, 2004. p.481–488.

KOK, J.; VLASSIS, N. Collaborative Multiagent Reinforcement Learning by Payoff Propagation. **Journal of Machine Learning Research**, [S.l.], v.7, p.1789–1828, 2006.

LEITE, J. G. M. **Engenharia de Tráfego**: métodos de pesquisa, características de tráfego, intersecções e sinais luminosos. São Paulo: Companhia de Engenharia de Tráfego - CET, 1980.

LI, L.; WALSH, T. J.; LITTMAN, M. L. Towards a Unified Theory of State Abstraction for MDPs. In: NINTH INTERNATIONAL SYMPOSIUM ON ARTIFICIAL INTELLIGENCE AND MATHEMATICS, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.531–539.

LITTMAN, M. L. Friend-or-Foe Q-learning in General-Sum Games. In: EIGHTEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML01), 2001, San Francisco, CA, USA. **Proceedings...** Morgan Kaufmann, 2001. p.322–328.

LOWRIE, P. The Sydney Coordinate Adaptive Traffic System - Principles, Methodology, Algorithms. In: INTERNATIONAL CONFERENCE ON ROAD TRAFFIC SIGNALLING, 1982, Sydney, Australia. **Proceedings...** [S.l.: s.n.], 1982.

MAILLER, R.; LESSER, V. Solving Distributed Constraint Optimization Problems Using Cooperative Mediation. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 3., 2004, New York. **Proceedings...** New York: IEEE Computer Society, 2004. p.438–445.

MERKE, A.; RIEDMILLER, M. A. Karlsruhe Brainstormers - A Reinforcement Learning Approach to Robotic Soccer. In: ROBOT SOCCER WORLD CUP (ROBOCUP 2001), 2001. **Proceedings...** Berlin: Springer, 2001. p.435–440. (Lecture Notes in Computer Science, v.2377).

NAGEL, K.; SCHRECKENBERG, M. A Cellular Automaton Model for Freeway Traffic. **Journal de Physique I**, [S.l.], v.2, p.2221, 1992.

OLIVEIRA, D.; BAZZAN, A. L. C. Traffic Lights Control with Adaptive Group Formation Based on Swarm Intelligence. In: INTERNATIONAL WORKSHOP ON ANT COLONY OPTIMIZATION AND SWARM INTELLIGENCE, ANTS 2006, 5., 2006, Berlin. **Proceedings...** Springer, 2006. p.520–521. (Lecture Notes in Computer Science).

OLIVEIRA, D. d.; BAZZAN, A. L. C. Swarm Intelligence Applied to Traffic Lights Group Formation. In: VI ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL (ENIA), 2007. **Anais...** SBC, 2007. p.1003–1112.

OLIVEIRA, D. d.; BAZZAN, A. L. C.; LESSER, V. Using Cooperative Mediation to Coordinate Traffic Lights: a case study. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTI AGENT SYSTEMS (AAMAS), 4., 2005. **Proceedings...** New York: IEEE Computer Society, 2005. p.463–470.

OLIVEIRA, D. d.; FERREIRA JR., P. R.; BAZZAN, A. L. C.; KLÜGL, F. A Swarm-based Approach for Selection of Signal Plans in Urban Scenarios. In: FOURTH INTERNATIONAL WORKSHOP ON ANT COLONY OPTIMIZATION AND SWARM INTELLIGENCE - ANTS 2004, 2004, Berlin, Germany. **Proceedings...** [S.l.: s.n.], 2004. p.416–417. (Lecture Notes in Computer Science, v.3172).

PANAIT, L.; LUKE, S. Cooperative Multi-Agent Learning: the state of the art. **Autonomous Agents and Multi-Agent Systems**, Hingham, MA, USA, v.11, n.3, p.387–434, 2005.

PAPAGEORGIOU, M.; DIAKAKI, C.; DINOPOULOU, V.; KOTSIALOS, A.; WANG, Y. Review of Road Traffic Control Strategies. **Proceedings of the IEEE**, [S.l.], v.91, n.12, p.2043–2067, December 2003.

ROBERTSON, D. I.; BRETHERTON, R. D. Optimizing Networks of Traffic Signals in Real Time - The SCOOT Method. **IEEE Transactions on Vehicular Technology**, [S.l.], v.40, n.1, p.11–15, February 1991.

ROESS, R. P.; PRASSAS, E. S.; MCSHANE, W. R. **Traffic Engineering**. 3.ed. [S.l.]: Prentice Hall, 2004. 816p.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial**. Tradução da segunda edição.ed. Rio de Janeiro, RJ: Campus, 2004. 1021p.

SANDHOLM, T. Perspectives on Multiagent Learning. **Artificial Intelligence**, Essex, UK, v.171, n.7, p.382–391, May 2007.

SHOHAM, Y.; LEYTON-BROWN, K. **Multiagent Systems**: algorithmic, game-theoretic, and logical foundations. [S.l.]: Cambridge University Press, 2009.

SHOHAM, Y.; POWERS, R.; GRENAGER, T. If multi-agent learning is the answer, what is the question? **Artificial Intelligence**, Essex, UK, v.171, n.7, p.365–377, May 2007.

SILVA, B. C. d.; JUNGES, R.; OLIVEIRA, D.; BAZZAN, A. L. C. ITSUMO: an intelligent transportation system for urban mobility. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, AAMAS, 5., 2006. **Proceedings...** ACM Press, 2006. p.1471–1472.

STONE, P. Multiagent learning is not the answer. It is the question. **Artificial Intelligence**, Essex, UK, v.171, n.7, p.402–405, May 2007.

SUTTON, R.; BARTO, A. **Reinforcement Learning**: an introduction. Cambridge, MA: MIT Press, 1998.

TAN, M. Multi-Agent Reinforcement Learning: independent vs. cooperative agents. In: TENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML 1993), 1993. **Proceedings...** Morgan Kaufmann, 1993. p.330–337.

VIDAL, J. M.; DURFEE, E. H. Agents Learning about Agents: a framework and analysis. In: **Collected papers from AAAI–97 workshop on Multiagent Learning**. [S.l.]: Menlo Park: AAAI Press, 1997. p.71–76.

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. **Machine Learning**, Hingham, MA, USA, v.8, n.3, p.279–292, 1992.

WOOLDRIDGE, M. J. **An Introduction to MultiAgent Systems**. Chichester: John Wiley & Sons, 2002.

YOUNG, H. P. The possible and the impossible in multi-agent learning. **Artificial Intelligence**, Essex, UK, v.171, n.7, p.429–433, May 2007.

ZHANG, C.; ABDALLAH, S.; LESSER, V. Integrating Organizational Control into Multi-Agent Learning. In: INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS (AAMAS), 8., 2009, Budapest, Hungary. **Proceedings...** [S.l.: s.n.], 2009.

ANEXO A TRANSYT

1 T R A N S Y T

TRAffic Network StudY Tool

0Implementation for IBM-PC or compatible, running under MS-DOS
0Program TRANSYT, version 10, modification 0
0Run with file:- "TMPTRAN.DAT" at 15:37 on 08/11/2009
0

PARAMETERS CONTROLLING DIMENSIONS OF PROBLEM :

NUMBER OF NODES	= 8
NUMBER OF LINKS	= 132
NUMBER OF OPTIMISED NODES	= 8
MAXIMUM NUMBER OF GRAPHIC PLOTS	= 8
NUMBER OF STEPS IN CYCLE	= 60
MAXIMUM NUMBER OF SHARED STOPLINES	= 0
MAXIMUM NUMBER OF TIMING POINTS	= 2
MAXIMUM LINKS AT ANY NODE	= 49

CORE REQUESTED = 21248 WORDS
CORE AVAILABLE = 36000 WORDS

1Program TRANSYT Page 2

0 DATA INPUT :-

0CARD NO. | CARD TYPE | CYCLE TIME (SEC) | NO. OF STEPS PER CYCLE | TIME PERIOD 1-1200 MINS. | EFFECTIVE-GREEN DISPLACEMENTS START (SEC) | EFFECTIVE-GREEN DISPLACEMENTS END (SEC) | EQUISAT 0=NO 1=YES | 0=UNEQUAL FLOW CYCLE 1=EQUAL CYCLE | SCALE 10-200 % | CRUISE-SPEEDS SCALE 50-200 % | CRUISE-SPEEDS CARD32 0=TIMES 1=SPEEDS | OPTIMISE 0=NONE 1=O/SET 2=FULL | EXTRA COPIES FINAL OUTPUT | HILL-CLIMB 1=FULL OUTPUT | DELAY VALUE P PER PCU-H | STOP VALUE P PER 100

CARD NO.	CARD TYPE															
(1)=	TITLE:-															
2)=	1	80	60	60	2	3	1	1	0	0	0	0	0	0	930	170
0CARD NO.	CARD TYPE	LIST OF NODES TO BE OPTIMISED														
3)=	2	1	2	3	4	5	6	7	22	0	0	0	0	0	0	0

0 NODE CARDS: STAGE CHANGE TIMES AND MINIMUM STAGE TIMES

CARD	CARD	NODE	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5	STAGE 6	STAGE 7

NO.	TYPE	NO.	CHANGE	MIN	CHANGE	MIN	CHANGE	MIN	CHANGE	MIN	CHANGE	MIN	CHANGE	MIN	CHANGE	MIN
4)=	12	1	0	12	68	12	0	0	0	0	0	0	0	0	0	0
5)=	12	2	0	12	37	12	0	0	0	0	0	0	0	0	0	0
6)=	12	3	0	12	56	12	0	0	0	0	0	0	0	0	0	0
7)=	12	4	0	12	68	12	0	0	0	0	0	0	0	0	0	0
8)=	12	5	0	12	68	12	0	0	0	0	0	0	0	0	0	0
9)=	12	6	0	12	48	12	0	0	0	0	0	0	0	0	0	0
10)=	12	7	0	12	29	12	0	0	0	0	0	0	0	0	0	0
11)=	12	22	0	12	68	12	0	0	0	0	0	0	0	0	0	0

0 LINK CARDS: GIVEWAY DATA

1Program TRANSYT Page 8

274)=	32	131	352	0	87	176	5	86	176	5	0	0	0	0	0	0
275)=	32	132	180	180	0	0	5	0	0	0	0	0	0	0	0	0

0 LINK CARDS : FLARE SATURATION FLOW DATA

			..LANE 1..		..LANE 2..		..LANE 3..	
	CARD TYPE	LINK NO.	SAT. FLOW	CAPAC VEH.	SAT. FLOW	CAPAC VEH.	SAT. FLOW	CAPAC VEH.
276)=	33	1	2700	1	0	0	0	0
277)=	33	5	2700	1	0	0	0	0
278)=	33	9	2700	1	0	0	0	0
279)=	33	13	2700	1	0	0	0	0
280)=	33	17	2700	1	0	0	0	0
281)=	33	21	2700	1	0	0	0	0
282)=	33	27	2700	1	0	0	0	0
283)=	33	31	2700	1	0	0	0	0
284)=	33	40	2700	1	0	0	0	0
285)=	33	46	2700	1	0	0	0	0
286)=	33	50	2700	1	0	0	0	0
287)=	33	54	2700	1	0	0	0	0
288)=	33	58	2700	1	0	0	0	0
289)=	33	62	2700	1	0	0	0	0
290)=	33	66	2700	1	0	0	0	0
291)=	33	89	2700	1	0	0	0	0

1Program TRANSYT Page 31

0 80 SECOND CYCLE 60 STEPS

0INITIAL SETTINGS

- (SECONDS)

0 NODE NO	NUMBER OF STAGES	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5	STAGE 6	STAGE 7
1	2	0	68					
2	2	0	37					
3	2	0	56					

4	2	0	68												
5	2	0	68												
6	2	0	48												
7	2	0	29												
22	2	0	68												
0 LINK NUMBER	FLOW INTO LINK (PCU/H)	SAT FLOW (PCU/H)	DEGREE OF SAT (%)	MEAN TIMES PER PCU CRUISE (SEC)	DELAY (SEC)	-------DELAY-------- UNIFORM RANDOM+ OVERSAT (U+R+O=MEAN Q) (PCU-H/H)	COST OF DELAY ($/H)	----STOPS---- MEAN STOPS /PCU (%)	COST OF STOPS ($/H)	----QUEUE---- MEAN MAX. (PCU)	AVERAGE EXCESS (PCU)	PERFORMANCE INDEX. WEIGHTED SUM OF ( ) VALUES ($/H)	EXIT NODE	GREEN TIMES START 1ST END 1ST (SECONDS)	START 2ND END 2ND
1	1566	2752f	66	4	4	0.6 + 1.0	(14.8)	23	(102.9)	11		117.7	22	0	68
2	1566	2700	67	4	4	0.8 + 1.0	(16.8)	26	(119.9)	12		136.7	22	0	68
3	1656	800	207	4	999	30.5 +429.0	(999.9)	248	(764.4)	502	+	5037.4			
4	1656	800	207	4	998	30.3 +429.0	(999.9)	248	(764.4)	502	+	5035.9			
5	800<	2752f	34	2	2	0.2 + 0.3	(4.2)	7	(126.7)	3		130.9	1	0	68
6	800<	2700	34	2	2	0.2 + 0.3	(4.6)	7	(142.9)	3		147.5	1	0	68
7	848<	800	106	2	144	3.2 + 30.7	(315.1)	101	(*****)	68	+	2328.5			
8	848<	800	106	2	144	3.2 + 30.7	(314.7)	100	(*****)	68	+	2325.1			
9	650<	2794f	49	5	16	2.5 + 0.5	(27.5)	33	(78.5)	10		105.9	2	0	37
10	650<	2700	51	5	17	2.6 + 0.5	(29.1)	34	(81.6)	10		110.7	2	0	37
11	1090<	800	136	5	524	12.1 +146.6	(999.9)	166	(535.4)	194	+	2011.9			
12	1090<	800	136	5	524	12.0 +146.6	(999.9)	166	(535.5)	194	+	2010.6			
13	973<	2763f	49	4	6	1.3 + 0.5	(16.2)	21	(108.5)	9		124.7	3	0	56
14	973<	2700	51	4	7	1.4 + 0.5	(17.7)	23	(115.9)	10		133.6	3	0	56
15	1016<	800	127	4	431	11.2 +110.4	(999.9)	147	(775.5)	156	+	1907.3			
16	1016<	800	127	4	431	11.1 +110.5	(999.9)	147	(775.6)	155	+	1906.1			
17	593<	2752f	25	3	2	0.1 + 0.2	(2.8)	5	(37.4)	2		40.3	4	0	68
18	593<	2700	25	3	2	0.2 + 0.2	(3.0)	6	(42.0)	2		45.0	4	0	68
19	586<	800	73	3	10	0.2 + 1.4	(14.5)	16	(37.4)	7		51.9			
20	586<	800	73	3	9	0.2 + 1.4	(14.3)	15	(35.8)	6		50.2			
21	587<	2752f	25	3	2	0.1 + 0.2	(2.7)	5	(33.1)	2		35.8	5	0	68
22	587<	2700	25	3	2	0.1 + 0.2	(2.8)	5	(34.0)	2		36.8	5	0	68
23	886<	800	111	3	209	3.8 + 47.7	(478.7)	129	(376.5)	72	+	855.2			
24	886<	800	111	3	209	3.8 + 47.7	(478.4)	129	(376.5)	72	+	854.9			
25	670<	800	84	3	13	0.0 + 2.5	(23.0)	8	(55.6)	2		78.6			
26	670<	800	84	3	13	0.0 + 2.5	(23.0)	8	(55.6)	2		78.6			
27	469<	2773f	28	3	8	0.9 + 0.2	(10.0)	22	(102.1)	5		112.1	6	0	48
28	469<	2700	28	3	9	0.9 + 0.2	(10.6)	23	(107.3)	5		118.0	6	0	48
29	714<	800	89	3	25	1.1 + 3.8	(45.3)	53	(310.7)	14		356.0			
30	714<	800	89	3	24	1.0 + 3.8	(44.7)	52	(305.1)	14		349.8			
31	590<	2820f	56	5	24	3.2 + 0.6	(36.0)	51	(82.9)	10		118.9	7	0	29
32	590<	2700	58	5	25	3.4 + 0.7	(38.0)	52	(85.5)	11		123.5	7	0	29
33	594<	800	74	5	25	2.7 + 1.4	(37.9)	67	(108.4)	14		146.3			

34	594<	800	74	5	24	2.5 +	1.4	(36.9)	66	(106.6)	14		143.5				
35	1566	2700	89	4	21	5.1 +	4.0	(84.2)	85	(386.9)	33		471.0	7	29	0	
36	1566	2700	89	4	21	5.1 +	4.0	(84.2)	85	(386.9)	33		471.0	7	29	0	
37	774	800	97	4	55	3.0 +	8.8	(110.0)	132	(297.6)	26	+	407.6				
38	774	800	97	4	55	3.0 +	8.8	(110.0)	132	(297.6)	26	+	407.6				
39	438	2700	26	4	9	0.9 +	0.2	(10.0)	45	(36.4)	5		46.4	6	0	48	
40	438	2773f	26	4	8	0.8 +	0.2	(9.4)	42	(34.4)	4		43.8	6	0	48	

1Program TRANSYT Page 32

0 80 SECOND CYCLE 60 STEPS

0 LINK NUMBER	FLOW INTO LINK (PCU/H)	SAT FLOW (PCU/H)	DEGREE OF SAT (%)	MEAN TIMES PER PCU CRUISE (SEC)	DELAY (SEC)	DELAY UNIFORM (U+R+O=MEAN PCU-H/H)	RANDOM+ OVERSAT Q)	COST OF DELAY ($/H)	STOPS MEAN STOPS /PCU (%)	STOPS COST OF STOPS ($/H)	QUEUE MEAN MAX. (PCU)	QUEUE AVERAGE EXCESS (PCU)	PERFORMANCE INDEX. WEIGHTED SUM OF ( ) VALUES ($/H)	EXIT NODE	GREEN TIMES START 1ST (SECONDS)	GREEN TIMES END 1ST (SECONDS)	GREEN TIMES START 2ND (SECONDS)	GREEN TIMES END 2ND (SECONDS)
41	604	800	76	4	12	0.5 +	1.5	(19.0)	55	(96.2)	8		115.2					
42	604	800	76	4	13	0.6 +	1.5	(19.5)	57	(99.7)	8		119.2					
43	694	800	87	3	17	0.2 +	3.1	(30.6)	47	(169.8)	11		200.3					
44	695	800	87	3	18	0.3 +	3.1	(31.5)	54	(194.4)	11		225.9					
45	484	2700	21	3	2	0.1 +	0.1	(2.5)	14	(36.0)	2		38.5	5	0	68		
46	485	2752f	20	3	2	0.1 +	0.1	(2.4)	14	(34.2)	2		36.6	5	0	68		
47	542	800	68	3	7	0.1 +	1.0	(10.2)	17	(47.2)	3		57.4					
48	542	800	68	3	7	0.1 +	1.0	(10.3)	18	(51.0)	3		61.3					
49	379	2700	16	3	2	0.1 +	0.1	(1.8)	12	(24.5)	1		26.2	4	0	68		
50	380	2752f	16	3	2	0.1 +	0.1	(1.7)	12	(23.4)	1		25.1	4	0	68		
51	396	800	49	3	5	0.0 +	0.5	(4.7)	11	(22.2)	2		26.9					
52	396	800	49	3	5	0.0 +	0.5	(4.8)	13	(26.3)	2		31.0					
53	366	2700	19	3	4	0.3 +	0.1	(3.9)	23	(43.4)	2		47.3	3	0	56		
54	366	2763f	19	3	4	0.3 +	0.1	(3.8)	22	(42.6)	2		46.4	3	0	56		
55	610<	800	76	4	11	0.4 +	1.6	(18.1)	46	(59.4)	11		77.5					
56	611<	800	76	4	12	0.4 +	1.6	(18.3)	47	(168.7)	11		187.0					
57	517<	2700	40	4	13	1.5 +	0.3	(17.2)	52	(55.7)	7		72.9	2	0	37		
58	517<	2794f	39	4	12	1.4 +	0.3	(16.1)	50	(53.0)	7		69.0	2	0	37		
59	639<	800	80	4	15	0.6 +	1.9	(24.0)	52	(101.8)	10		125.8					
60	639<	800	80	4	15	0.7 +	1.9	(24.6)	53	(105.2)	10		129.8					
61	710<	2700	31	5	2	0.2 +	0.2	(3.7)	12	(16.9)	3		20.6	1	0	68		
62	711<	2752f	30	5	2	0.2 +	0.2	(3.4)	11	(15.1)	2		18.5	1	0	68		
63	1020<	800	128	5	421	7.1 +112.3		(999.9)	253	(495.3)	158	+	1605.9					
64	1020<	800	128	5	422	7.1 +112.3		(999.9)	253	(495.4)	158	+	1606.3					
65	560<	2700	24	2	2	0.1 +	0.2	(2.8)	10	(89.7)	2		92.6	22	0	68		
66	560<	2752f	24	2	2	0.1 +	0.2	(2.7)	9	(81.0)	2		83.7	22	0	68		
67	481<	800	60	5	6	0.0 +	0.7	(7.4)	15	(16.7)	3		24.1					
68	481<	800	60	2	6	0.1 +	0.7	(7.6)	17	(117.5)	3		125.1					
69	240	2700	55	5	40	2.1 +	0.6	(24.7)	99	(44.1)	5		68.7	1	68	0		

70	240	2700	55	5	40	2.1 +	0.6	(24.7)	99	(44.1)	5		68.7	1	68	0	
71	240	2700	55	5	40	2.1 +	0.6	(24.7)	99	(44.1)	5		68.7	1	68	0	
72	480<	2700	18	5	1	0.0 +	0.1	(1.0)	0	(0.9)	0		1.9				
73	180	2700	7	5	1	0.0 +	0.0	(0.3)	1	(0.3)	0		0.6				
74	480<	2700	18	5	1	0.0 +	0.1	(1.0)	0	(0.9)	0		1.9				
75	825<	2700	31	5	1	0.0 +	0.2	(2.0)	1	(1.8)	0		3.9				
76	576<	2700	68	5	31	3.8 +	1.1	(45.7)	69	(97.0)	12		142.7	3	56	0	
77	480<	2700	18	5	1	0.0 +	0.1	(1.0)	0	(0.9)	0		1.9				
78	66<	2700	2	5	1	0.0 +	0.0	(0.1)	1	(0.1)	0		0.2				
79	513<	2700	19	4	1	0.0 +	0.1	(1.1)	1	(1.5)	0		2.6				
80	202<	2700	7	4	1	0.0 +	0.0	(0.4)	1	(0.5)	0		0.9				
81	1512	2700	56	5	2	0.0 +	0.6	(5.9)	2	(5.3)	1		11.2				
82	876<	2700	32	5	1	0.0 +	0.2	(2.2)	1	(2.0)	0		4.2				
83	3096	2700	115	5	250	13.3 +	201.8	(999.9)	239	(*****)	339	+	3381.7				
84	131<	2700	5	5	1	0.0 +	0.0	(0.2)	1	(0.2)	0		0.4				
85	180	2700	7	5	1	0.0 +	0.0	(0.3)	1	(0.3)	0		0.6				
86	252	2700	9	5	1	0.0 +	0.1	(0.5)	1	(0.4)	0		0.9				
87	252	2700	9	5	1	0.0 +	0.1	(0.5)	1	(0.4)	0		0.9				
88	75	2700	17	5	35	0.6 +	0.1	(6.7)	89	(12.4)	2		19.1	4	68	0	
89	75	2976f	16	5	34	0.6 +	0.1	(6.5)	88	(12.3)	2		18.8	4	68	0	
90	562<	2700	21	4	1	0.0 +	0.1	(1.2)	1	(1.7)	0		2.9				
91	296<	2700	11	5	1	0.0 +	0.1	(0.6)	1	(0.5)	0		1.1				
92	299<	2700	11	5	1	0.0 +	0.1	(0.6)	0	(0.5)	0		1.1				
93	180	2700	7	5	1	0.0 +	0.0	(0.3)	1	(0.3)	0		0.6				
94	402<	2700	15	5	1	0.0 +	0.1	(0.8)	1	(0.7)	0		1.5				

1Program TRANSYT Page 33

0 80 SECOND CYCLE 60 STEPS

0 LINK NUMBER	FLOW INTO LINK (PCU/H)	SAT FLOW (PCU/H)	DEGREE OF SAT (%)	MEAN TIMES PER PCU CRUISE (SEC)	DELAY (SEC)	DELAY UNIFORM (U+R+O=MEAN Q) (PCU-H/H)	RANDOM+ OVERSAT (U+R+O=MEAN Q) (PCU-H/H)	COST OF DELAY ($/H)	STOPS MEAN STOPS /PCU (%)	STOPS COST OF STOPS ($/H)	QUEUE MEAN MAX. (PCU)	QUEUE AVERAGE EXCESS (PCU)	PERFORMANCE INDEX. WEIGHTED SUM OF ( ) VALUES ($/H)	EXIT NODE	GREEN TIMES START 1ST (SECONDS)	GREEN TIMES END 1ST (SECONDS)	GREEN TIMES START 2ND (SECONDS)	GREEN TIMES END 2ND (SECONDS)
95	88<	2700	3	5	1	0.0 +	0.0 (0.2)		1	(0.1)	0		0.3					
96	460<	2700	17	5	1	0.0 +	0.1 (1.0)		1	(0.9)	0		1.8					
97	147<	2700	5	5	1	0.0 +	0.0 (0.3)		1	(0.2)	0		0.5					
98	600	2700	54	5	21	3.0 +	0.6 (32.9)		75	(83.4)	11		116.4	6	48	0		
99	600	2700	54	5	21	3.0 +	0.6 (32.9)		75	(83.4)	11		116.4	6	48	0		
100	600	2700	54	5	21	3.0 +	0.6 (32.9)		75	(83.4)	11		116.4	6	48	0		
101	428<	2700	16	4	1	0.0 +	0.1 (0.9)		1	(1.2)	0		2.1					
102	180	2700	7	5	1	0.0 +	0.0 (0.3)		1	(0.3)	0		0.6					
103	428<	2700	16	5	1	0.0 +	0.1 (0.9)		1	(0.8)	0		1.7					
104	428<	2700	16	5	1	0.0 +	0.1 (0.9)		1	(0.8)	0		1.7					
105	428<	2700	16	5	1	0.0 +	0.1 (0.9)		1	(0.8)	0		1.7					

```
   106     282   2700     10     5     1    0.0 +  0.1  (  0.5)      1  (  0.5)       0               1.0
   107     851   2700     32     5     1    0.0 +  0.2  (  2.1)      1  (  1.9)       0               4.1
   108     180   2700      7     4     1    0.0 +  0.0  (  0.3)      1  (  0.5)       0               0.8
   109     488   2700     18     5     1    0.0 +  0.1  (  1.0)      1  (  0.9)       0               1.9
   110     488   2700     18     5     1    0.0 +  0.1  (  1.0)      1  (  0.9)       0               1.9
   111     180   2700      7     5     1    0.0 +  0.0  (  0.3)      1  (  0.3)       0               0.6
   112     417   2700     15     5     1    0.0 +  0.1  (  0.8)      1  (  0.8)       0               1.6
   113     240   2700     55     5    40    2.1 +  0.6  ( 24.7)     99  ( 44.1)       5              68.7      5     68    0
   114     240   2700     55     5    40    2.1 +  0.6  ( 24.7)     99  ( 44.1)       5              68.7      5     68    0
   115     240   2700     55     5    40    2.1 +  0.6  ( 24.7)     99  ( 44.1)       5              68.7      5     68    0
   116     236   2700      9     5     1    0.0 +  0.0  (  0.4)      1  (  0.4)       0               0.8
   117     180   2700      7     5     1    0.0 +  0.0  (  0.3)      1  (  0.3)       0               0.6
   118     180   2700      7     5     1    0.0 +  0.0  (  0.3)      1  (  0.3)       0               0.6
   119     364<  2700     13     5     1    0.0 +  0.1  (  0.7)      1  (  0.7)       0               1.4
   120     362<  2700     13     4     1    0.0 +  0.1  (  0.7)      1  (  1.0)       0               1.7
   121    1512   2700    102     5    85    8.3 + 27.3  (331.4)    160  (450.3)      62     +        781.7      2     37    0
   122     256<  2700      9     5     1    0.0 +  0.1  (  0.5)      1  (  0.4)       0               0.9
   123     180   2700      7     5     1    0.0 +  0.0  (  0.3)      1  (  0.3)       0               0.6
   124     480<  2700     18     5     1    0.0 +  0.1  (  1.0)      1  (  0.9)       0               1.9
   125     168<  2700      6     4     1    0.0 +  0.0  (  0.3)      1  (  0.4)       0               0.7
   126     168<  2700      6     4     1    0.0 +  0.0  (  0.3)      1  (  0.4)       0               0.7
   127     180   2700     41     5    37    1.5 +  0.3  ( 17.2)     94  ( 31.5)       4              48.7     22     68    0
   128     180   2700     41     5    37    1.5 +  0.3  ( 17.2)     94  ( 31.5)       4              48.7     22     68    0
   129    2135<  2700     79     5     3    0.0 +  1.9  ( 17.4)      3  ( 15.6)       2              33.0
   130     576<  2700     21     5     1    0.0 +  0.1  (  1.3)      1  (  1.1)       0               2.4
   131     351   2700     13     5     1    0.0 +  0.1  (  0.7)      1  (  0.6)       0               1.3
   132     180   2700      7     5     1    0.0 +  0.0  (  0.3)      1  (  0.3)       0               0.6
               *** f - average saturation flow for flared link ***
0  TOTAL          TOTAL       MEAN          TOTAL    TOTAL    TOTAL       TOTAL        PENALTY       TOTAL
   DISTANCE         TIME     JOURNEY        UNIFORM   RANDOM+    COST        COST          FOR       PERFORMANCE
 TRAVELLED         SPENT      SPEED          DELAY   OVERSAT     OF          OF          EXCESS       INDEX
                                                      DELAY    DELAY       STOPS        QUEUES
  (PCU-KM/H)      (PCU-H/H)     (KM/H)      (PCU-H/H)(PCU-H/H)   ($/H)       ($/H)         ($/H)        ($/H)
0 20995.2         2428.8         8.6         239.0   2069.8    (******) + (18149.9)  +  (   0.0)   =   39622.0      TOTALS
0*****************************************************************************************************************************
0                                  CRUISE              DELAY               STOPS              TOTALS
                                 LITRES PER HOUR     LITRES PER HOUR     LITRES PER HOUR     LITRES PER HOUR
0FUEL CONSUMPTION PREDICTIONS      25236.8         +    3232.4         +   15337.4         =   43806.5
0
 NO. OF ENTRIES TO SUBPT  =    1
 NO. OF LINKS RECALCULATED=  167
1Program TRANSYT                                                                              Page  34

 PROGRAM TRANSYT FINISHED
```

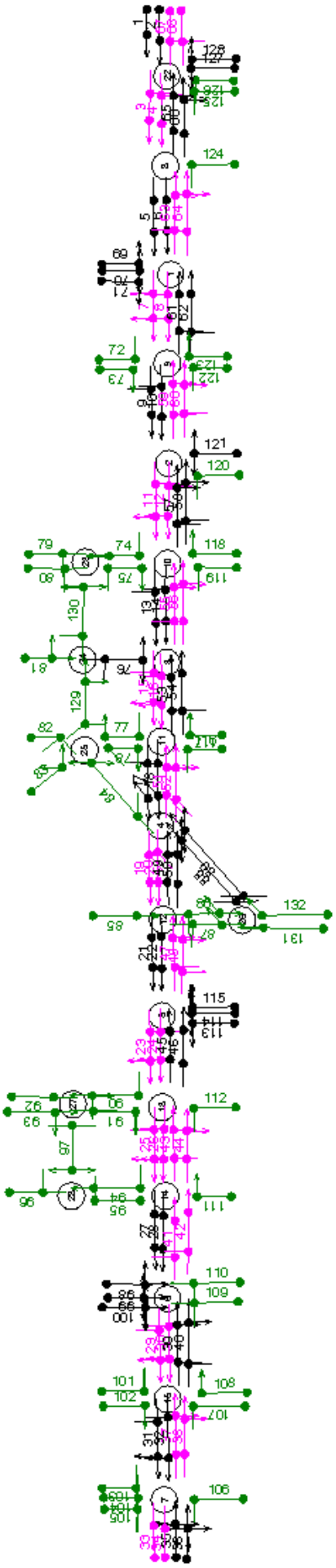

Figura A.1: Mapa da região da Av. Assis Brasil reduzido (imagem do TRANSYT)